MATHEMATICS OF OPERATIONS RESEARCH
Vol. 00, No. 0, Xxxxxx 20xx, pp. xxx–xxx
ISSN 0364-765X | EISSN 1526-5471 |xx|0000|0xxx

DOI 10.1287/moor.xxxx.xxxx

Deterministic pivoting algorithms for constrained ranking and clustering problems[1]

Anke van Zuylen[2]
Institute for Theoretical Computer Science, Tsinghua University, Beijing, P.R. China.
email: anke@tsinghua.edu.cn

David P. Williamson
School of Operations Research and Information Engineering, Cornell University, Ithaca, NY 14853.
email: dpw@cs.cornell.edu

We consider ranking and clustering problems related to the aggregation of inconsistent information, in particular, rank aggregation, (weighted) feedback arc set in tournaments, consensus and correlation clustering, and hierarchical clustering. Ailon, Charikar, and Newman [4], Ailon and Charikar [3], and Ailon [2] proposed randomized constant factor approximation algorithms for these problems, which recursively generate a solution by choosing a random vertex as "pivot" and dividing the remaining vertices into two groups based on the pivot vertex.

In this paper, we answer an open question in these works by giving deterministic approximation algorithms for these problems. The analysis of our algorithms is simpler than the analysis of the randomized algorithms in [4], [3] and [2]. In addition, we consider the problem of finding minimum-cost rankings and clusterings which must obey certain constraints (e.g. an input partial order in the case of ranking problems), which were introduced by Hegde and Jain [25] (see also [34]). We show that the first type of algorithms we propose can also handle these constrained problems. In addition, we show that in the case of a rank aggregation or consensus clustering problem, if the input rankings or clusterings obey the constraints, then we can always ensure that the output of any algorithm obeys the constraints without increasing the objective value of the solution.

Key words: Rank Aggregation; Minimum Feedback Arc Set in Tournaments; Consensus Clustering; Correlation Clustering; Hierarchical Clustering; Approximation Algorithm; Derandomization

MSC2000 Subject Classification: Primary: 68W25, 68W40; Secondary: 91B12, 05C20, 91C20

OR/MS subject classification: Primary: Analysis of algorithms, Secondary: Games/group decisions: Voting/committees, Networks/graphs

History: Received: Xxxx xx, xxxx; Revised: Yyyyyy yy, yyyy and Zzzzzz zz, zzzz.

1. Introduction. We consider the problem of ranking or clustering a set of elements, based on input information for each pair of elements. The objective is to find a solution that minimizes the deviation from the input information.

Consider, for example, the problem of ranking the teams of a round-robin sports tournament: each pair of teams has played against each other exactly once, and a desirable property of a ranking of the teams is that it ranks team i before team j if team i won the game against team j. Clearly, this is not always possible, since there may be teams i, j, k where team i beat team j, team j beat team k and team k beat team i. An example of clustering that has a similar flavor to the round-robin sports tournament is clustering webpages based on similarity scores. For each pair of pages we have a score between 0 and 1, and we would like to cluster the pages so that pages with a high similarity score are in the same cluster, and pages with a low similarity score are in different clusters. We will also consider hierarchical clustering. Given a set of elements and integer values D_{ij} between 0 and M for each pair of elements i, j, we want to find a hierarchical clustering of M levels, i.e. M nested partitions of the elements, such

[1]Preliminary versions of these results appeared in the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007) [34], and in the Fifth Workshop on Approximation and Online Algorithms (WAOA 2007) [35].

[1]Supported by NSF grant CCF-0514628.

[2]Research performed while the author was at the School of Operations Research and Information Engineering, Cornell University.

that elements i, j are in different clusters in the lowest D_{ij} levels. As in the sports tournament example, not every input admits a solution that exactly satisfies these desired properties. We will therefore seek a solution that minimizes the sum of the pairwise deviations from the input information.

Other examples arise in aggregation of existing rankings or clusterings. For example, Dwork, Kumar, Naor and Sivakumar [16] propose meta-search engines for Web search, where we want to get more robust rankings that are not sensitive to the various shortcomings and biases of individual search engines by combining the rankings of individual search engines. Another example comes from gene expression analysis. The goal is to find classifications of genes based on the results from different microarray experiments. In this context, Filkov and Skiena [20] introduce the problem called consensus clustering, in which we want to find a clustering of a set of elements that is as close as possible to a collection of clusterings of the same set of elements.

These examples are related to the topic of voting systems, in which each voter gives a preference on a set of alternatives, and the system outputs a single preference order on the set of alternatives based on the voters' preferences. Arrow's impossibility theorem [8] states that no voting system can simultaneously achieve non-dictatorship, independence of irrelevant alternatives, and Pareto efficiency. Non-dictatorship means that the voting system does not simply output the preference order of a particular voter, independence of irrelevant alternatives means that if the output order ranks i before j, then this should also be the case if one or more voters change the position of an alternative $k \neq i, j$, and finally, Pareto efficiency or unanimity ensures that if all voters prefer alternative i over alternative j, then so does the output ordering. Kemeny [27] proposed the following objective for determining the best voting system: Given permutations $\pi_1, \ldots, \pi_\ell$ of V, we want to find π that minimizes $\sum_{k=1}^{\ell} d(\pi, \pi_k)$, where $d(\pi, \pi_k)$ is the Kendall tau distance, which is defined as the number of pairs i, j such that $\pi_k(i) < \pi_k(j)$ and $\pi(i) > \pi(j)$. In other words, in Kemeny aggregation we want to find a permutation that minimizes the number of pairwise disagreements with the ℓ input permutations.

Fagin, Kumar, Mahdian, Sivakumar and Vee [19] extend the problem to *partial* rank aggregation (see also Ailon [2]). The input rankings do not have to be permutations of the same set of elements, but are instead allowed to have ties. More precisely, a partial ranking of a set of elements V is a function $\pi : V \to \{1, \ldots, |V|\}$. Examples of partial rankings are top-m rankings, i.e. permutations of only a subset of the elements (in which case we make the assumption that the unranked elements all share the position after the last ranked element), or the rankings may be p-ratings, i.e. mappings from V to $\{1, \ldots, p\}$, as is the case in movie rankings. Fagin et al. propose several metrics to compare partial rankings, and show that they are within constant multiples of each other. Following Ailon, we will say the distance $d(\pi_1, \pi_2)$ between two partial rankings π_1 and π_2 is again the number of pairs i, j such that $\pi_1(i) < \pi_1(j)$, and $\pi_2(i) > \pi_2(j)$. The goal of partial rank aggregation is given partial rankings $\pi_1, \ldots, \pi_\ell$ to find a *permutation* that minimizes $\sum_{k=1}^{\ell} d(\pi, \pi_k)$. Note that the output is required to be a permutation, and cannot be a partial ranking. We will refer to Kemeny aggregation and partial rank aggregation simply as rank aggregation. If the input rankings are permutations rather than partial rankings we will use the term full rank aggregation.

Consensus clustering, as defined by Filkov and Skiena [20], is similar to rank aggregation. Given ℓ clusterings (partitions) $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$ of a set V, we want to find a clustering $\mathcal{C}$ that minimizes $\sum_{k=1}^{\ell} d(\mathcal{C}_k, \mathcal{C})$, where $d(\mathcal{C}_k, \mathcal{C})$ is the number of pairs i, j such that i, j are in the same cluster in one of the clusterings $\mathcal{C}_k, \mathcal{C}$, and in different clusters in the other clustering. We introduce here the problem of *partial* consensus clustering, which is similar to partial rank aggregation. The input is $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$ where each $\mathcal{C}_k$ is a partition of V. However, each set in $\mathcal{C}_k$ now comes with a label "final" or "not final". The label "not final" indicates that the set may be broken down into smaller clusters. We will call such a partition $\mathcal{C}_k$ a partial clustering of V. We define the distance $d(\mathcal{C}, \mathcal{D})$ between two (partial) clusterings $\mathcal{C}$ and $\mathcal{D}$ as the number of pairs i, j such that one of the two partitions $\mathcal{C}, \mathcal{D}$ contains a final set that contains both i, j, and the other collection has i and j in different sets. For example, if $V = \{1, 2, 3, 4\}$, and $\mathcal{C} = \big\{\{1, 2\}_{\text{final}}, \{3, 4\}_{\text{not final}}\big\}, \mathcal{D} = \big\{\{1, 3\}_{\text{final}}, \{2, 4\}_{\text{final}}\big\}$, then $d(\mathcal{C}, \mathcal{D}) = 3$, since the pairs contributing to the distance are $\{1, 2\}, \{1, 3\}$ and $\{2, 4\}$; $\{3, 4\}$ does not not contribute because it is not final in $\mathcal{C}$. If $\mathcal{C}, \mathcal{D}$ are full clusterings or partitions of V then the distance function is the same as the one defined previously. As before the goal of partial consensus clustering is to find a clustering (partition) $\mathcal{C}$ that minimizes $\sum_{k=1}^{\ell} d(\mathcal{C}_k, \mathcal{C})$. From now on, we will refer to the problem when the input clusterings contain only final clusters as full consensus clustering.

We believe that partial consensus clustering is a useful extension of consensus clustering. As an example we mention a problem studied by Modha and Spangler [31, 30] of clustering web search results based on three criteria: words contained in the document, out-links from the document, and in-links to the document. Documents are assumed to be similar if they share one or more words, and if they share one or more in-links or out-links. Although this is not the approach proposed in [31, 30], a possible approach to this is to find a clustering based on each of the three criteria, and aggregate the three resulting clusterings using consensus clustering. But what if we have some pages that have no out-links? Such a document is not similar to any other document based on the out-link criterion, since it does not share an out-link with any document. Hence we could say it is dissimilar to any document that does have out-links, but it really is neither similar nor dissimilar to another document that does not have out-links. A partial clustering allows us to represent this information by putting the documents that have no out-links into a single cluster that we mark "not final".

Partial consensus clusterings also provide a natural way to deal with missing values. For example, a data set that has been often used to test consensus clustering algorithms is the Mushrooms data set from the UCI repository [9, 22, 23]. The data set contains 8124 mushrooms and has values for 22 attributes for each mushroom, such as cap color, stalk shape, habitat, etc. Each attribute thus defines a clustering of the mushrooms, where mushrooms that have the same value are clustered together. However, there are some missing values in the data set. Different researchers have dealt with this in different ways. In [22], a coin is tossed independently for each pair of mushrooms where at least one of them is missing a value for the attribute, which determines whether these mushrooms should be in the same cluster or not. Note that the coin tosses in this approach may not yield a valid clustering, for example if we have three mushrooms, m_1, m_2, m_3 where m_1, m_2 have the same value for the attribute and m_3 does not have value. Then the coin tosses may require that m_3 should be in the same cluster as m_1 and not in the same cluster as m_2, which is not possible since m_1, m_2 are in the same cluster. In [23] all mushrooms with missing values are clustered together. If we allow partial clusterings, we can mark the cluster of mushrooms with missing values as "not final", so that we do not incur any penalty if our consensus clustering separates two mushrooms with missing values.

We now give a general framework for the ranking problems and the clustering problems that we consider in this paper. In the *weighted minimum feedback arc set problem*, we are given a set of vertices V, nonnegative weights $w = \{w_{(i,j)}, w_{(j,i)} : \{i,j\} \subseteq V\}$, where we assume without loss of generality that the weights are scaled so that $w_{(i,j)} + w_{(j,i)} \leq 1$. We want to find a permutation π that minimizes the weight of pairs of vertices out of order with respect to the permutation, i.e.

$$\sum_{\pi(i)<\pi(j)} w_{(j,i)}.$$

The (unweighted) feedback arc set problem has weights that are either 0 or 1. The best known approximation algorithm for this problem has performance guarantee $O(\log n \log\log n)$ [32, 18], and the problem is at least as hard as vertex cover [26]. In this paper, we will consider weighted feedback arc set problems where the weights satisfy one of the following: *probability constraints*: for any pair i, j, $w_{(i,j)} + w_{(j,i)} = 1$, or the *triangle inequality*: for any triple i, j, k, $w_{(i,j)} + w_{(j,k)} \geq w_{(i,k)}$. We will sometimes refer to these problems as the *ranking* problem. The feedback arc set problem in tournaments is the special case when the weights satisfy the probability constraints and are either 0 or 1. Note that we can represent the problem of ranking the teams of a round-robin sports tournament as a feedback arc set in tournaments by setting $w_{(i,j)} = 1$ if team i beat team j and 0 otherwise. Rank aggregation is also a special case of weighted minimum feedback arc set, since we can set $w_{(i,j)} = \frac{1}{\ell}\sum_{k=1}^{\ell} \mathbf{1}\{\pi_k(i) < \pi_k(j)\}$ where $\mathbf{1}\{\cdot\}$ is the indicator function. Note that these weights satisfy the triangle inequality, and in the case of full rank aggregation also the probability constraints.

In the *weighted clustering problem*, we are given a set of vertices V, and nonnegative weights $w^+ = \{w^+_{\{i,j\}} : \{i,j\} \subseteq V\}, w^- = \{w^-_{\{i,j\}} : \{i,j\} \subseteq V\}$ satisfying $w^+_{\{i,j\}} + w^-_{\{i,j\}} \leq 1$ for every i, j. For ease of notation, we will use the notation ij instead of $\{i,j\}$ in subscripts, hence from now on we will write $w^+_{ij} = w^+_{\{i,j\}}, w^-_{ij} = w^-_{\{i,j\}}$. We want to find a clustering $\mathcal{C}$ minimizing

$$\sum_{\{i,j\}:\nexists C\in\mathcal{C}\text{s.t.}\{i,j\}\subseteq C} w^+_{ij} + \sum_{\{i,j\}:\exists C\in\mathcal{C}\text{s.t.}\{i,j\}\subseteq C} w^-_{ij}.$$

The unweighted clustering problem in which each weight is either 0 or 1 is the correlation clustering problem (in general graphs). The best known approximation algorithm for this problem has performance

guarantee $O(\log n)$ [13]. In this paper, we consider two special cases. We say the weights satisfy *probability constraints* if for every $i, j \in V$, $w^+_{ij} + w^-_{ij} = 1$. We will say the weights satisfy the *triangle inequality* if for every triple i, j, k, $w^-_{ij} + w^-_{jk} \geq w^-_{ik}$ and $w^+_{ij} + w^-_{jk} \geq w^+_{ik}$. We note that the triangle inequality assumption in [4] only assumes the first set of inequalities. The problem where exactly one of w^+_{ij} and w^-_{ij} is 1 (and the other 0) is known as *correlation clustering* (in complete graphs). We will use the term correlation clustering to refer to correlation clustering in complete graphs. Given an instance $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$ of consensus clustering, we can define $w^+_{ij} = \frac{1}{\ell}\sum_{k=1}^{\ell} \mathbf{1}\{\exists C \in \mathcal{C}_k \text{ s.t. } \{i,j\} \subseteq C, C \text{ is final}\}$, and $w^-_{ij} = \frac{1}{\ell}\sum_{k=1}^{\ell} \mathbf{1}\{\exists C \in \mathcal{C}_k \text{ s.t. exactly one of } i,j \text{ is in } C\}$. It is easily verified that the weights satisfy the triangle inequality. If the input clusterings consist only of final clusters, the weights also satisfy the probability constraints.

Hierarchical clustering can be seen as a generalization of correlation clustering. An M-level hierarchical clustering of a set V is a nested clustering of the vertices in V, where the clustering at level m is a refinement of the clustering at level $m+1$. Given a set V and a matrix D with $D_{ij} \in \{0, \ldots, M\}$ for any distinct $i, j \in V$, we want to find an M-level hierarchical clustering of V minimizing $\sum_{i,j \in V} |D_{ij} - \lambda_{ij}|$, where λ_{ij} is the number of levels in which i and j are in different clusters, or equivalently, since the clusterings are nested, i and j are in different clusters at levels $1, \ldots, \lambda_{ij}$, and in the same cluster at levels $\lambda_{ij}+1, \ldots, M$. It was shown in [24] that this is equivalent to finding an *ultrametric* that minimizes the ℓ_1 distance to D. An ultrametric is a tree metric in which all vertices are at the leaves of the tree, and the distance from each leaf to the root is the same.

We also consider *constrained* versions of these problems, which were introduced by Hegde and Jain [25] (see also [34]). The constraints we consider are constraints on pairs of vertices. In the case of clustering, constraints can be given as sets $P^+, P^- \subseteq V \times V$ and any feasible clustering must have a pair i, j in the same cluster if $\{i, j\} \in P^+$, and in different clusters if $\{i, j\} \in P^-$. In hierarchical clustering, in addition to matrix D, we are given matrices L, U such that $L_{ij} \leq D_{ij} \leq U_{ij}$ for every $i, j \in V$. Any feasible hierarchical clustering must have $L_{ij} \leq \lambda_{ij} \leq U_{ij}$, where λ_{ij} is again the number of levels in which i and j are in different clusters. In the ranking case, the constraints are a partial order P, and any feasible solution must be a linear extension of P. These constraints can reflect prior beliefs about the output ranking or clustering; for example, when aggregating the ranked results of different search engines, we may want to ensure that the top-level page of a website is ordered before subpages of the site, or when clustering webpages we can ensure all pages of a website are in the same cluster.

1.1 Related work. The feedback arc set problem in tournaments is NP-hard [4, 5, 12, 14]. Rank aggregation is also NP-hard [11], even if the number of input rankings is only 4 [16]. Correlation clustering is MAX-SNP hard [13], and consensus clustering is NP-hard [33, 29], although it is not known to be NP-hard if the number of input clusterings is constant.

The maximization versions of these problems, where instead of minimizing the pairwise deviations from the input information we try to maximize the pairwise agreements, are "easy" to approximate: there exist polynomial time approximation schemes (PTAS) for the corresponding maximization problems of feedback arc set in tournaments [7, 21] and correlation clustering [10].

Ailon, Charikar and Newman [4] give the first constant-factor approximation algorithms for the unconstrained ranking and clustering problems with weights that satisfy either triangle inequality constraints, probability constraints, or both. Their algorithms are randomized and based on Quicksort: the algorithms recursively generate a solution by choosing a random vertex as a "pivot" and ordering all other vertices with respect to the pivot vertex according to some criterion. In the first type of algorithm they give for the ranking problem, a vertex j is ordered before the pivot k if $w_{(j,k)} \geq w_{(k,j)}$ or ordered after k otherwise. Next, the algorithm recurses on the two instances induced by the vertices before and after the pivot. In the case of a clustering problem, a vertex j is placed in the same cluster as the pivot vertex k if $w^+_{jk} \geq w^-_{jk}$. The algorithm recurses on the instance induced by the vertices that are not placed in the same cluster as the pivot vertex. The second type of algorithm first solves a linear programming relaxation of the problem under consideration, which has variables $x_{(i,j)}$ and $x_{(j,i)} = 1 - x_{(i,j)}$ or x^+_{ij} and $x^-_{ij} = 1 - x^+_{ij}$ for every pair $i, j \in V$ for the ranking and clustering problems respectively. The pivoting scheme is then used to randomly round the fractional solution, i.e. if k is the pivot vertex, then j is ordered before k (clustered together with k) with probability $x_{(j,k)}$ (x^+_{jk}) and ordered after k (not clustered in the same cluster as k) with probability $x_{(k,j)}$ (x^-_{jk}).

In the case of rank aggregation and consensus clustering, a folklore result is that returning the best of the input rankings or clusterings is a 2-approximation algorithm. Ailon, Charikar and Newman also show that one can obtain better approximation factors for rank aggregation and consensus clustering by returning the best of their algorithm's solution and the best input ranking/clustering. For instance, for rank aggregation, they obtain a randomized $\frac{11}{7}$-approximation algorithm using their first type of algorithm, and a randomized $\frac{4}{3}$-approximation algorithm using their second, LP-based, algorithm.

There has been a good deal of follow-up work since the Ailon et al. paper. Ailon and Charikar [3] consider hierarchical clustering and fitting ultrametrics. An ultrametric satisfies the property that for any three distinct vertices, the two largest pairwise distances are equal. An M-level hierarchical clustering is a special case of an ultrametric in which the distance between each pair of vertices is an integer between 0 and M. Their algorithm for hierarchical clustering takes a random pivot vertex k, and adjusts the distance of a pair i, j if the triple i, j, k does not satisfy the ultrametric property. They show that this gives an expected $(M+2)$-approximation algorithm. They also give an LP based $O(\log n \log\log n)$ approximation algorithm for fitting ultrametrics.

Coppersmith, Fleischer, and Rudra [15] give a simple, greedy 5-approximation algorithm for the ranking problem when weights obey the probability constraints. Hegde and Jain [25] give a 4-approximation algorithm for the same problem, and a 6-approximation algorithm for the constrained version of the ranking problem with weights that satisfy probability constraints. Kenyon-Mathieu and Schudy [28] give a polynomial-time approximation scheme for unconstrained weighted feedback arc set in tournaments with weights satisfying $b \leq w_{(i,j)} + w_{(j,i)} \leq 1$ for all $i, j \in V$ for some $b > 0$. Note that this includes problems satisfying the probability constraints and hence includes the full rank aggregation problem as a special case. Their approximation scheme assumes the availability of a solution with cost that is not more than a constant factor α from optimal. To get a $(1+\epsilon)$-approximate solution, the running time of their algorithm is doubly exponential in $\frac{1}{\epsilon}$, $\frac{1}{b}$ and α.

Ailon [2] considers the partial rank aggregation problem. He generalizes and improves some of the results from Ailon et al. to partial rank aggregation. He shows that perturbing the solution to the linear programming relaxation and using these perturbed values as probabilities gives a randomized $\frac{3}{2}$-approximation algorithm for partial rank aggregation. Since his analysis only uses the fact that the weights satisfy the triangle inequality, this also yields $\frac{3}{2}$-approximation algorithm for ranking with triangle inequality constraints on the weights.

1.2 Our results. We show how to give deterministic algorithms matching the best randomized algorithms of [4], [2], and [3]. This answers an open question in these works. It is a fundamental question whether everything computable in randomized polynomial time is computable in deterministic polynomial time (something that the recent PRIMES in P result by Agrawal, Kayal, and Saxena [1] provided some additional evidence for). The techniques from Ailon et al. are not amenable to standard techniques of derandomization, but we show that we can amortize in place of the expectation and make the randomized algorithm deterministic.

In Section 2 and Section 3 respectively, we give simple deterministic algorithms as derandomizations of the combinatorial algorithms in [4]. The analysis of these algorithms is simpler than the analysis of the original algorithms, and actually implies the performance guarantees for the original algorithms as well.

In Section 4 we consider constrained problems and show how to adapt our algorithms from Sections 2 and 3 to deal with constrained problems. In addition we show that, if we assume that the weights are "consistent" with the constraints, then any approximation algorithm for an unconstrained problem with triangle inequality also implies the same guarantee for constrained versions of the problem.

In Section 5, we extend the ideas from Sections 2 and 3 to the randomized rounding algorithms in [4] and [2], and show how to derandomize these algorithms. Although the analysis of these algorithms gets more involved, our analysis here is again simpler than the analysis of the original randomized algorithms, and again implies the approximation guarantees of the original randomized algorithms.

Finally, in Section 6, we show how to use the algorithm in Section 3 to obtain a deterministic $(M+2)$-approximation algorithm for M-level hierarchical clustering. The algorithm and its analysis follow easily from Section 3, in contrast to the (expected) $(M+2)$-approximation algorithm in [3], which is rather

involved.

2. A simple ranking algorithm. Ailon, Charikar and Newman [4] propose a simple algorithm to obtain a permutation that costs at most 2 times the optimum if the weights satisfy the triangle inequality, at most 5 times the optimum in the case of probability constraints, and at most 3 times the optimum in the case of feedback arc set in tournaments. Given an instance of the weighted feedback arc set problem, they form a tournament $G = (V, A)$ by including arc (i, j) only if $w_{(i,j)} \geq w_{(j,i)}$ (breaking ties arbitrarily). This is called the majority tournament [4]. Note that if the majority tournament is acyclic, it corresponds to an optimal permutation: the cost for pair i, j in any solution is at least $\min\{w_{(i,j)}, w_{(j,i)}\}$, and this lower bound is met for every pair. We give a general framework, FAS-Pivot, which generalizes both our algorithm, and the algorithm in [4]. We use the following notation: We denote by $G(V') = (V', A(V'))$ the subgraph of G induced by $V' \subseteq V$. If π_1 and π_2 are permutations of disjoint sets V_1, V_2, we let $\pi = \pi_1, \pi_2$ denote the concatenation of the two permutations: if $i \in V_1$ then $\pi(i) = \pi_1(i)$ and if $i \in V_2$ then $\pi(i) = |V_1| + \pi_2(i)$.

FAS-Pivot$(G = (V, A))$

Pick a pivot $k \in V$.
$V_L = \{i \in V : (i, k) \in A\}$,
$V_R = \{i \in V : (k, i) \in A\}$.
Return FAS-Pivot$(G(V_L))$, k, FAS-Pivot$(G(V_R))$.

In Ailon, Charikar and Newman's algorithm, $G = (V, A)$ is the majority tournament and a pivot is chosen randomly. In the algorithms we consider, G is either the majority tournament or a tournament obtained by rounding the solution to a linear programming relaxation. The main difference between our algorithm and that of Ailon, Charikar and Newman is that in our deterministic version of the algorithm, we will use a lower bound on the weight of an optimal solution to give a way of always choosing a "good" pivot. We will first state our algorithm, and in particular our choice of pivot, in general terms, and we then show how this implies a 2-approximation algorithm for weighted feedback arc set with triangle inequality, a $\frac{8}{5}$-approximation algorithm for partial rank aggregation, and a 3-approximation algorithm for weighted feedback arc set with probability constraints.

Our algorithms will have a "budget" c_{ij} for every pair of distinct vertices $\{i, j\}$. We will use the notation $\{\{i, j\} \subseteq V\}$ to denote the set of all unordered pairs of distinct vertices. Note that we dropped the curly brackets in the subscript; we will always use the subscript ij to denote the *unordered* pair $\{i, j\}$. The budgets will be chosen in such a way that the total budget is a lower bound on the value of the optimal solution. Theorem 2.1 below gives conditions under which we can show that on average the cost incurred by a vertex pair is not more than α times its budget.

The first condition of Theorem 2.1 states that the cost of ordering i before j if $(i, j) \in A$ is at most α times its budget. Note that the only way a pair of vertices i, j is *not* ordered according to A (i.e. j is ordered before i even though $(i, j) \in A$) is if in some recursive call j ends up in V_L, and i in V_R. For a pivot k, let $T_k(G)$ be the set of arcs for which this occurs, if k is the pivot and the input to the recursive call is $G = (V, A)$, i.e. $T_k(G) = \{(i, j) \in A : (j, k) \in A, (k, i) \in A\}$. Our algorithm chooses the pivot that minimizes the ratio of the cost for these arcs to their budget.

We now prove the following key theorem, which states conditions under which FAS-Pivot can be used to obtain a solution to a given input of a weighted feedback arc set problem, that costs at most α times a given budget $\sum_{\{i,j\} \subseteq V} c_{ij}$.

THEOREM 2.1 *Given an input (V, w) of weighted feedback arc set, a set of budgets $\{c_{ij} : \{i, j\} \subseteq V\}$, and a tournament $G = (V, A)$ such that*

(i) $w_{(j,i)} \leq \alpha c_{ij}$ for all $(i, j) \in A$,

(ii) $w_{(i,j)} + w_{(j,k)} + w_{(k,i)} \leq \alpha\big(c_{ij} + c_{jk} + c_{ki}\big)$ for every directed triangle $(i, j), (j, k), (k, i)$ in A,

then FAS-Pivot returns a solution that costs at most $\alpha \sum_{\{i,j\} \subseteq V} c_{ij}$ if we choose a pivot k that minimizes[3]

$$\frac{\sum_{(i,j) \in T_k(G)} w_{(i,j)}}{\sum_{(i,j) \in T_k(G)} c_{ij}}. \tag{1}$$

PROOF. Let $G = (V, A)$ be a tournament that satisfies the conditions in the theorem. For a pair of vertices i, j with $(i, j) \in A$, we will say i, j incurs a "forward" cost if i is ordered to the left of j, and we say i, j incurs a "backward" cost if i is ordered to the right of j. In the latter case, we will also say that (i, j) becomes a backward arc.

Let an "iteration" be the work done by FAS-Pivot(G) before the recursive call. We bound the forward and backward costs that are incurred in one iteration by α times the respective budgets of the vertex pairs involved. Since the conditions of the theorem will also hold in subsequent iterations, and since each vertex pair incurs either a forward or a backward cost in exactly one iteration, this then guarantees that we find a solution of at most α times the total budget given by $\sum_{\{i,j\}\subseteq V} c_{ij}$.

Note that the first condition implies that a forward cost incurred by a pair i, j is at most αc_{ij}. We now show that the second condition implies that we always choose a pivot so that the total backward cost incurred by pivoting on this vertex is at most α times the budget for these vertex pairs.

For a given pivot k, $T_k(G) \subset A$ is the set of arcs that become backward by pivoting on k. The backward cost incurred if k is the pivot is equal to $\sum_{(i,j)\in T_k(G)} w_{(i,j)}$, and we have a budget for these vertex pairs of $\sum_{(i,j)\in T_k(G)} c_{ij}$. Hence we need to show that we always choose a pivot such that the ratio in (1) is at most α. Note that this can be done by by showing that

$$\sum_{k\in V} \sum_{(i,j)\in T_k(G)} w_{(i,j)} \leq \alpha \sum_{k\in V} \sum_{(i,j)\in T_k(G)} c_{ij}. \tag{2}$$

We observe that (i, j) becomes a backward arc if (k, i) and (j, k) in A, in other words, exactly when (i, j) is in a directed triangle with the pivot k. Therefore $T_k(G)$ contains exactly the arcs that are in a directed triangle with k in G. Let T be the set of directed triangles in G, and for a triangle $t = \{(i, j), (j, k), (k, i)\}$, let $w(t) = \sum_{(g,h)\in t} w_{(g,h)}$ and let $c(t) = \sum_{(g,h)\in t} c_{gh}$. If we sum $\sum_{(i,j)\in T_k(G)} w_{(i,j)}$ over all $k \in V$, i.e. $\sum_{k\in V} \sum_{(i,j)\in T_k(G)} w_{(i,j)}$, then we count $w_{(i,j)}$ exactly once for every pivot k such that $(i, j), (j, k), (k, i)$ is a directed triangle, hence $\sum_{k\in V} \sum_{(i,j)\in T_k(G)} w_{(i,j)} = \sum_{t\in T} w(t)$. Similarly, $\sum_{(i,j)\in T_k(G)} c_{ij} = \sum_{t\in T} c(t)$.

By the second condition of the theorem, $w(t) \leq \alpha c(t)$, therefore (2) holds, and hence there must exist some pivot k such that $\sum_{(i,j)\in T_k(G)} w_{(i,j)} \leq \alpha \sum_{(i,j)\in T_k(G)} c_{ij}$. □

REMARK 2.1 *The proof of Theorem 2.1 also implies that under the conditions of the theorem, choosing a pivot at random gives a solution with expected cost at most* $\alpha \sum_{\{i,j\}\subseteq V} c_{ij}$.

LEMMA 2.1 *The algorithm in Theorem 2.1 can be implemented in* $O(n^3)$ *time.*

PROOF. We maintain a list of the directed triangles in G for which all three vertices are currently contained in a single recursive call, and for each vertex we maintain the total cost for the arcs that become backward if pivoting on that vertex and the total budget for these pairs (i.e. the numerator and denominator of (1)). We bound the time needed to maintain this information, and the time required by the algorithm, given that this information is available, separately.

Given the numerator and denominator of (1) for each vertex, we can implement a single recursive call in $O(n)$ time, and there are at most $O(n)$ iterations, giving a total of $O(n^2)$.

Initializing the list of triangles and the numerator and denominator of (1) for each vertex takes $O(n^3)$ time. Over all recursive calls combined, the time needed to update the list of directed triangles, and the numerator and denominator of (1) is $O(n^3)$: after each pivot, we need to remove all triangles that either contain the pivot vertex, or contain (i, j) where i and j are separated into different recursive calls, and for each triangle removed from the list, we need to update the numerator and denominator of (1) for the three vertices in the triangle. Assuming the list of triangles is linked to the vertices and arcs contained in it and vice versa, finding a triangle that contains a certain vertex or arc, removing it, and updating the numerator and denominator for the vertices contained in it, can be done in constant time. Finally,

[3] Throughout this work, we define a ratio to be 0 if both numerator and denominator are 0. If only the denominator is 0, we define it to be ∞.

note that each triangle is removed from the list exactly once, hence the overall time to update the list of directed triangles, and the numerator and denominator of (1) is $O(n^3)$. □

2.1 Weighted minimum feedback arc set with triangle inequality and rank aggregation. We now show that Theorem 2.1 implies a 2-approximation algorithm for weighted feedback arc set with triangle inequality. We then use the theorem plus a 2-approximation algorithm from Ailon [2] to obtain an improved performance guarantee for rank aggregation problems, where in addition to the weights we have the original input rankings.

THEOREM 2.2 *There exists a deterministic combinatorial 2-approximation algorithm for weighted feedback arc set in tournaments with triangle inequality which runs in $O(n^3)$ time.*

PROOF. Let $G = (V, A)$ be the majority tournament, i.e. $(i, j) \in A$ only if $w_{(i,j)} \geq w_{(j,i)}$ (breaking ties arbitrarily). Let $c_{ij} = \min\{w_{(i,j)}, w_{(j,i)}\}$ for every $i, j \in V$. Note that $\sum_{\{i,j\}\subseteq V} c_{ij}$ is a lower bound on the weight of any feasible solution. We will show that the conditions of Theorem 2.1 are satisfied with $\alpha = 2$, hence Theorem 2.1 plus Lemma 2.1 imply the result.

The first condition is met for every $\alpha \geq 1$, since if $(i, j) \in A$, then $w_{(j,i)} \leq w_{(i,j)}$, hence $w_{(j,i)} = \min\{w_{(i,j)}, w_{(j,i)}\} = c_{ij}$.

Let $t = \{(i, j), (j, k), (k, i)\}$ be a directed triangle in G, then by the triangle inequality on the weights, $w_{(i,j)} + w_{(j,k)} + w_{(k,i)} \leq (w_{(i,k)} + w_{(k,j)}) + (w_{(j,i)} + w_{(i,k)}) + (w_{(k,j)} + w_{(j,i)}) = 2(w_{(j,i)} + w_{(k,j)} + w_{(i,k)})$. Now note that $(i, j) \in A$ implies that $c_{ij} = w_{(j,i)}$ and similarly, $c_{jk} = w_{(k,j)}, c_{ki} = w_{(i,k)}$. Hence $w_{(i,j)} + w_{(j,k)} + w_{(k,i)} \leq 2(c_{ij} + c_{jk} + c_{ki})$, and the second condition is satisfied for $\alpha = 2$. □

As in Ailon, Charikar and Newman [4], we can do better in the case of rank aggregation. In fact, we will extend the ideas from Ailon, Charikar and Newman [4], and Ailon [2] to give a combinatorial $\frac{8}{5}$-approximation algorithm for partial rank aggregation.

Recall that in the partial rank aggregation problem, we have ℓ partial rankings $\pi_1, \ldots, \pi_\ell$ such that $w_{(i,j)} = \frac{1}{\ell}\sum_{\ell'=1}^{\ell} \mathbf{1}\{(\pi_{\ell'}(i) < \pi_{\ell'}(j))\}$. In the (full) rank aggregation problem, $\pi_1, \ldots, \pi_\ell$ are full rankings. A well-known 2-approximation for full rank aggregation outputs one of the input permutations at random: the expected cost for pair i, j is $2w_{(i,j)}w_{(j,i)}$ which is not more than $2\min\{w_{(i,j)}, w_{(j,i)}\}$. It follows that returning the best input permutation is also a 2-approximation algorithm.

In the partial rank aggregation problem, we will denote by $i \equiv j$ if $w_{(i,j)} = w_{(j,i)} = 0$, i.e. if none of the input rankings prefer i to j or vice versa. Ailon [2] proposes the algorithm RepeatChoice for partial rank aggregation. Let $\pi_1, \ldots, \pi_\ell$ be the input rankings; π will be our final output ranking. We start by setting $\pi(i) = 1$ for all $i \in V$. Then we repeatedly choose an input ranking π_k uniformly at random without replacement; we check each $i, j \in V$ and if $\pi(i) = \pi(j)$ but $\pi_k(i) < \pi_k(j)$, we modify π so that now $\pi'(i) < \pi'(j)$. We can do this by setting $\pi'(h) = \pi(h)$ if $\pi(h) \leq \pi(i)$ and $\pi'(h) = \pi(h) + 1$ if $h = j$ or $\pi(h) > \pi(i)$. At the end of the algorithm, the only pairs that have $\pi(i) = \pi(j)$ are the pairs such that $i \equiv j$. We break these ties arbitrarily, to obtain a full ranking.

Note that for $i \not\equiv j$ the probability that i is ranked before j is $\frac{w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}}$ which incurs a cost of $w_{(j,i)}$. Since i is either ranked before j, or j before i, the expected cost for pair i, j such that $i \not\equiv j$ is $\frac{2w_{(j,i)}w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}}$, which is clearly at most $2\min\{w_{(i,j)}, w_{(j,i)}\}$. Ailon shows that this algorithm can be derandomized.

Ailon, Charikar and Newman [4] show that the best of their algorithm's solution and the best input permutation is a $\frac{11}{7}$-approximation algorithm for (full) rank aggregation. We show that a similar guarantee can be given for our deterministic algorithm, and moreover that this guarantee also holds for partial rank aggregation, i.e. the best of our algorithm's solution, and the solution given by RepeatChoice gives a combinatorial $\frac{8}{5}$-approximation algorithm for partial rank aggregation. We remark that $\frac{11}{7} \approx 1.57$, and $\frac{8}{5} = 1.6$ so our guarantee is slightly worse than the one given by Ailon et al.

THEOREM 2.3 *There exists a deterministic combinatorial $\frac{8}{5}$-approximation algorithm for partial rank aggregation which runs in $O(n^3)$ time.*

PROOF. We can exploit the fact that the (deterministic) RepeatChoice algorithm gives us a permutation π such that $\sum_{\pi(i)<\pi(j)} w_{(j,i)} \leq \sum_{\{i,j\}\subseteq V} \frac{2w_{(j,i)}w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}}$ as follows. Let $0 \leq \gamma \leq 1$. Given an

instance, we consider a modified instance with weights $\tilde{w}_{(i,j)} = \gamma w_{(i,j)} + (1-\gamma)\frac{2w_{(j,i)}w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}}$ if $i \not\equiv j$, and $\tilde{w}_{(i,j)} = 0$ otherwise (since if $i \equiv j$ then $w_{(i,j)} = w_{(j,i)} = 0$). Suppose we have an algorithm for the modified instance that is guaranteed to return a solution σ with $\sum_{\sigma(i)<\sigma(j)} \tilde{w}_{(j,i)} \leq C$. Since $\gamma \in [0,1]$, this implies that either $\sum_{\sigma(i)<\sigma(j)} w_{(j,i)} \leq C$, or $\sum_{\sigma(i)<\sigma(j)} \frac{2w_{(j,i)}w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}} \leq C$. Now, note that $\sum_{\sigma(i)<\sigma(j)} \frac{2w_{(j,i)}w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}} = \sum_{\{i,j\}\subseteq V} \frac{2w_{(j,i)}w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}}$ so the latter case implies that the output of RepeatChoice satisfies $\sum_{\pi(i)<\pi(j)} w_{(j,i)} \leq C$. Hence given an algorithm for the modified instance which outputs a solution of modified cost at most C, and permutation π from the RepeatChoice algorithm, we can give an algorithm for the original instance which outputs a solution of (original) cost at most C.

To prove the theorem, we will show that Theorem 2.1 implies that we can define a modified instance for which Fas-Pivot is guaranteed to return a permutation σ of modified cost at most $\frac{8}{5}OPT$, where OPT is the optimum of the original instance.

Let $\gamma = \frac{2}{5}$, so $\tilde{w}_{(i,j)} = \frac{2}{5}w_{(i,j)} + \frac{3}{5}\frac{2w_{(j,i)}w_{(i,j)}}{w_{(i,j)}+w_{(j,i)}}$. Let $G = (V, A)$ again be the majority tournament (where we note that the majority tournament with respect to the original weights w is the same as the majority tournament with respect to the modified weights $\tilde{w}$). Let $c_{ij} = \min\{w_{(i,j)}, w_{(j,i)}\}$. Since $\sum_{\{i,j\}\subseteq V} c_{ij} \leq OPT$, we just need to show that the conditions in Theorem 2.1 hold for $\alpha = \frac{8}{5}$.

The first condition of Theorem 2.1 is met for $\alpha \geq \frac{8}{5}$, since if $(i,j) \in A$, then $w_{(j,i)} = c_{ij}$, so if $i \not\equiv j$ then

$$\tilde{w}_{(j,i)} = \frac{2}{5}w_{(j,i)} + \frac{3}{5}2\frac{w_{(i,j)}w_{(j,i)}}{w_{(i,j)}+w_{(j,i)}} = \frac{2}{5}c_{ij} + \frac{6}{5}\frac{w_{(i,j)}c_{ij}}{w_{(i,j)}+c_{ij}} \leq \frac{8}{5}c_{ij},$$

and of course if $i \equiv j$ then $\tilde{w}_{(j,i)} = c_{ij} = 0$.

To show the second condition is met, we let $a_{ij} = w_{(i,j)} + w_{(j,i)}$, and first rewrite $\tilde{w}_{(i,j)}$ for $(i,j) \in A$ with $i \not\equiv j$:

$$\tilde{w}_{(i,j)} = \frac{2}{5}w_{(i,j)} + \frac{3}{5}2\frac{w_{(i,j)}w_{(j,i)}}{w_{(i,j)}+w_{(j,i)}} = \frac{2}{5}w_{(i,j)} + \frac{6}{5}\frac{w_{(i,j)}(a_{ij}-w_{(i,j)})}{a_{ij}} = \frac{8}{5}w_{(i,j)} - \frac{6}{5}\frac{w_{(i,j)}^2}{a_{ij}}.$$

Let $\{(i,j),(j,k),(k,i)\}$ be a directed triangle in G. We want to show that $\tilde{w}_{(i,j)} + \tilde{w}_{(j,k)} + \tilde{w}_{(k,i)} \leq \alpha(c_{ij}+c_{jk}+c_{ki})$. Note that if $i \equiv j$, then $\tilde{w}_{(i,j)} = c_{ij} = 0$, hence i,j contributes nothing to either side of the inequality. Hence we need to show that $\sum_{(g,h)\in t: g\not\equiv h}(\frac{8}{5}w_{(g,h)} - \frac{6}{5}\frac{w_{(g,h)}^2}{a_{gh}}) \leq \frac{8}{5}\sum_{(g,h)\in t: g\not\equiv h} c_{gh}$.

By the triangle inequality on the weights, $w_{(i,j)} + w_{(j,k)} + w_{(k,i)} \leq w_{(i,j)} + w_{(j,k)} + (w_{(k,j)} + w_{(j,i)}) = a_{ij} + a_{jk}$. Similarly, we get that $w_{(i,j)} + w_{(j,k)} + w_{(k,i)} \leq a_{ij} + a_{ki}$ and $w_{(i,j)} + w_{(j,k)} + w_{(k,i)} \leq a_{jk} + a_{ki}$. Adding these inequalities, we get that $\sum_{(g,h)\in t} w_{(g,h)} \leq \frac{2}{3}\sum_{(g,h)\in t} a_{gh}$.

By these observations and since $g \not\equiv h$ if and only if $a_{gh} > 0$, we can apply Claim 2.1 below, and conclude that $\sum_{(g,h)\in t: g\not\equiv h}\left(16w_{(g,h)} - 6\frac{w_{(g,h)}^2}{a_{gh}} - 8a_{gh}\right) \leq 0$, or $\sum_{(g,h)\in t: g\not\equiv h}\left(8w_{(g,h)} - 6\frac{w_{(g,h)}^2}{a_{gh}}\right) \leq \sum_{(g,h)\in t: g\not\equiv h} 8(a_{gh} - w_{(g,h)}) = \sum_{(g,h)\in t: g\not\equiv h} 8c_{gh}$, as required. □

Claim 2.1 *For $w = (w_1, w_2, w_3)$, and $a = (a_1, a_2, a_3)$ such that $0 \leq w_i \leq a_i \leq 1$ for $i = 1,2,3$, and $\sum_{i=1}^3 w_i \leq \frac{2}{3}\sum_{i=1}^3 a_i$:*

$$\sum_{i:a_i>0}\left(16w_i - 6\frac{w_i^2}{a_i} - 8a_i\right) \leq 0.$$

Proof. Let $f(w,a) = \sum_{i:a_i>0}\left(16w_i - 6\frac{w_i^2}{a_i} - 8a_i\right)$, and let $P = \left\{(w,a) : 0 \leq w_i \leq a_i \leq 1, \text{ for } i = 1,2,3, \text{ and } \sum_{i=1}^3 w_i \leq \frac{2}{3}\sum_{i=1}^3 a_i\right\}$. We want to show that $\sup\{f(w,a)|(w,a) \in P\} \leq 0$.

Note that P is a closed and bounded set, and f is continuous on P, hence f attains a maximum on P. We make the following observations:

(i) The maximum has $\sum_{i=1}^3 w_i = \frac{2}{3}\sum_{i=1}^3 a_i$: Take any solution with $\sum_{i=1}^3 w_i < \frac{2}{3}\sum_{i=1}^3 a_i$. Then there exists some $w_j < a_j$. Let $\delta > 0$ such that $\delta \leq a_j - w_j$, $\delta \leq \frac{2}{3}\sum_{i=1}^3 a_i - \sum_{i=1}^3 w_i$. Note that since $0 < w_j < a_j$, $2w_j < 2a_j < \frac{8}{3}a_j$, therefore we can choose $\gamma > 0$ with $\gamma < \min(\delta, \frac{8}{3}a_j - 2w_j)$.

Increasing w_j by γ changes f by $16\gamma - 6\frac{\gamma^2+2\gamma w_j}{a_j} = \gamma(16 - 6\frac{\gamma+2w_j}{a_j}) > 0$, where the final inequality follows since $\gamma + 2w_j < \frac{8}{3}a_j$.

(ii) We now consider a fixed $\bar{a}$, with $\bar{a}_i \geq 0$ for $i = 1, 2, 3$. Then maximizing $f(w, \bar{a})$ subject to $\sum_{i=1}^3 w_i = \frac{2}{3}\sum_{i=1}^3 \bar{a}_i$ reduces to minimizing $\sum_{i:\bar{a}_i>0} \frac{w_i^2}{\bar{a}_i}$. Since $\sum_{i:\bar{a}_i>0} \frac{w_i^2}{\bar{a}_i}$ is a convex function of w, and $\sum_{i=1}^3 w_i = \frac{2}{3}\sum_{i=1}^3 \bar{a}_i$ is linear, a necessary and sufficient condition for w to be a global minimum is (i) $\sum_{i=1}^3 w_i = \frac{2}{3}\sum_{i=1}^3 \bar{a}_i$ and (ii) there exists some c such that $\frac{2w_i}{\bar{a}_i} = c$ for every i such that $\bar{a}_i > 0$. This implies that the global minimum is achieved at $w_i = \frac{2}{3}\bar{a}_i$ for every i.

It remains to note that $f(w, a) = 0$ if $w_i = \frac{2}{3}a_i$ for $i = 1, 2, 3$. □

2.2 Weighted feedback arc set with probability constraints. In the previous subsection we used $\sum_{\{i,j\}\subseteq V} \min\{w_{(i,j)}, w_{(j,i)}\}$ as the lower bound or budget in Theorem 2.1. If we only know that the weights satisfy the probability constraints, this lower bound has the problem that it is possible that $\min\{w_{(i,j)}, w_{(j,i)}\} = 0$ for all i, j, even if there is no feasible solution with cost 0. We will therefore now turn to a linear programming relaxation, which can provide a better lower bound. If we let $x_{(i,j)} = 1$ denote that i is ranked before j, then x should satisfy probability constraints and the triangle inequality: any feasible ranking satisfies $x_{(i,j)}+x_{(j,i)} = 1$ and $x_{(i,j)}+x_{(j,k)}+x_{(k,i)} \geq 1$ (since if $x_{(i,j)}+x_{(j,k)}+x_{(k,i)} = 0$, then j is ranked before i, k is ranked before j but i is ranked before k, which is not possible). Hence the following linear program gives a lower bound on the minimum weight feedback arc set:

$$
\begin{array}{lll}
\min & \displaystyle\sum_{\{i,j\}\subseteq V} \Big(x_{(i,j)} w_{(j,i)} + x_{(j,i)} w_{(i,j)} \Big) & \\
\text{s.t.} & x_{(i,j)} + x_{(j,k)} + x_{(k,i)} \geq 1 & \text{for all distinct } i, j, k \\
(LP_{FAS}) & x_{(i,j)} + x_{(j,i)} = 1 & \text{for all } i \neq j \\
& x_{(i,j)} \geq 0 & \text{for all } i \neq j.
\end{array}
$$

We use an optimal solution x to (LP_{FAS}), to give the budgets in Theorem 2.1, and in addition we use it to form the tournament we need in Theorem 2.1.

THEOREM 2.4 *There exists a deterministic 3-approximation algorithm for weighted minimum feedback arc set with probability constraints.*

PROOF. Let x be an optimal solution to (LP_{FAS}). We let $c_{ij} = x_{(i,j)}w_{(j,i)} + x_{(j,i)}w_{(i,j)}$. We form a tournament $G = (V, A)$, where $(i, j) \in A$ only if $x_{(i,j)} \geq \frac{1}{2}$. We show that the conditions in Theorem 2.1 hold with $\alpha = 3$. Since $\sum_{\{i,j\}\subseteq V} c_{ij}$ is a lower bound on the weight of any feasible solution, this will imply the result.

The first condition in Theorem 2.1 holds with $\alpha = 2$, since for $(i, j) \in A$, $w_{(j,i)} \leq 2x_{(i,j)}w_{(j,i)} \leq 2c_{ij}$.

Let $t = \{(i, j), (j, k), (k, i)\}$ be a directed triangle in G. We need to show that $\sum_{(g,h)\in t} w_{(g,h)} \leq 3\sum_{(g,h)\in t} c_{gh}$. We denote the lefthand side by $w(t)$ and the righthand side by $3c(t)$. We first rewrite the right hand side:

$$
\begin{aligned}
c(t) &= \sum_{(g,h)\in t} \big(w_{(h,g)} x_{(g,h)} + w_{(g,h)}(1 - x_{(g,h)}) \big) \\
&= \sum_{(g,h)\in t} w_{(g,h)} + \sum_{(g,h)\in t} (w_{(h,g)} - w_{(g,h)}) x_{(g,h)} \\
&= w(t) + \sum_{(g,h)\in t} (w_{(h,g)} - w_{(g,h)}) x_{(g,h)}.
\end{aligned}
$$

Suppose without loss of generality that $w_{(j,i)} - w_{(i,j)} = \min_{(g,h)\in t}\{w_{(h,g)} - w_{(g,h)}\}$. To give a lower bound on $c(t)$, we consider the case that $w_{(j,i)} - w_{(i,j)} \geq 0$ and the case that $w_{(j,i)} - w_{(i,j)} < 0$.

In the first case, $w_{(h,g)} - w_{(g,h)} \geq 0$ for all $(g, h) \in t$. Hence $c(t) = w(t) + \sum_{(g,h)\in t}(w_{(h,g)} - w_{(g,h)})x_{(g,h)} \geq w(t)$. In the second case, we rewrite $c(t)$ further as

$$
c(t) = w(t) + \sum_{(g,h)\in t} \frac{1}{2} \big(w_{(h,g)} - w_{(g,h)} \big) + \sum_{(g,h)\in t} \left(x_{(g,h)} - \frac{1}{2} \right) (w_{(h,g)} - w_{(g,h)}).
$$

By the definition of A, $x_{(g,h)} - \frac{1}{2} \geq 0$ for every $(g,h) \in t$, and we know from feasibility of x that $\sum_{(g,h)\in t} x_{(g,h)} \leq 2$. This means that if $\min_{(g,h)\in t}\{w_{(h,g)} - w_{(g,h)}\} = w_{(j,i)} - w_{(i,j)} < 0$, then $c(t)$ obtains its lowest possible value if $x_{(i,j)} = 1$ and $x_{(j,k)} = x_{(k,i)} = \frac{1}{2}$. Hence in this case

$$\begin{aligned} c(t) &\geq w(t) + (w_{(j,i)} - w_{(i,j)}) + \frac{1}{2}(w_{(k,j)} - w_{(j,k)}) + \frac{1}{2}(w_{(i,k)} - w_{(k,i)}) \\ &= w_{(j,i)} + \frac{1}{2}(w_{(k,j)} + w_{(j,k)}) + \frac{1}{2}(w_{(i,k)} + w_{(k,i)}). \end{aligned} \tag{3}$$

Since the weights satisfy the probability constraints, $w_{(g,h)} + w_{(h,g)} = 1$, and hence (3) is equal to $1+w_{(j,i)} \geq 1$. By the probability constraints we also know that $w(t) \leq 3$, hence it follows that $w(t) \leq 3c(t)$. □

REMARK 2.2 *The LP based method from Theorem 2.4 can also be applied to give a 2-approximation algorithm for instances when the weights satisfy the triangle inequality. In the case when the weights satisfy the triangle inequality, the part of the proof of Theorem 2.4 that follows (3) should be changed to: "When the weights satisfy the triangle inequality, $w_{(i,k)} + w_{(k,j)} \geq w_{(i,j)}$, so the quantity in (3) is not less than $w_{(j,i)} + \frac{1}{2}w(t) \geq \frac{1}{2}w(t)$."*

3. A simple clustering algorithm. We will now show how to turn the algorithms from the previous sections into algorithms for correlation and consensus clustering. We use similar ideas as in Ailon, Charikar and Newman [4].

Instead of the majority tournament, they consider the complete undirected graph $G = (V, E)$, and partition the edge set into two sets E^+, E^- so that E^+ contains the edges $\{i,j\}$ such that $w^+_{ij} > w^-_{ij}$ and E^- contains the edges $\{i,j\}$ with $w^-_{ij} > w^+_{ij}$. Edges $\{i,j\}$ for which $w^+_{ij} = w^-_{ij}$ can be in either E^+ or E^-, as long as E^+, E^- form a partition of E. As in [4] we will say this graph has a "bad triplet" if there exist vertices i,j,k such that $\{i,j\} \in E^+, \{j,k\} \in E^+, \{k,i\} \in E^-$. It is not hard to show that if the graph has no bad triplets, then there is a clustering that has i,j in the same cluster if $\{i,j\} \in E^+$ and in separate clusters if $\{i,j\} \in E^-$. Clearly this is an optimal clustering, since for any vertex pair, the cost incurred in this clustering is $\min\{w^+_{ij}, w^-_{ij}\}$.

We give the algorithm CC-Pivot from [4] below (where in the version in [4] the pivot is chosen randomly from V). We use the following notation: Given $G = (V, E^+, E^-)$, we denote by $G(V')$ the subgraph of G induced by $V' \subseteq V$.

CC-Pivot$(G = (V, E^+, E^-))$

Pick a pivot $k \in V$.
$C = \{k\} \cup \{i \in V : \{i,k\} \in E^+\}$,
$R = \{i \in V : \{i,k\} \in E^-\}$.
Return $\{C$, CC-Pivot$(G(R))\}$.

Ailon, Charikar and Newman [4] show that if the pivot is chosen uniformly at random from V, then CC-Pivot gives a 5-approximation if the weights satisfy probability constraints and a 3-approximation algorithm for correlation clustering. It can also be shown using their analysis that CC-Pivot with a random pivot gives a 2-approximation if the weights satisfy the triangle inequality. As in the case of ranking, we show that we can give a deterministic version of this algorithm with the same performance guarantees, by using a lower bound on the optimal value to guide our choice of pivot vertex.

Suppose a pair of vertices i,j is *not* clustered according to G. If j is clustered with i even though $\{i,j\} \in E^-$, then there is some recursive call in which both i and j end up in cluster C. If j is not clustered with i even though $\{i,j\} \in E^+$, then there is a recursive call in which i ends up in cluster C, and j is added to R. Neither i nor j can be the pivot in these cases. For a pivot k, let $T^+_k(G)$ and $T^-_k(G)$ be the sets of pairs for which these two events occur, if k is the pivot and the input to the recursive call is G, i.e. $T^+_k(G) = \{\{i,j\} \in E^+ : \{j,k\} \in E^-, \{i,k\} \in E^+\}$ and $T^-_k(G) = \{\{i,j\} \in E^- : \{j,k\} \in E^+, \{i,k\} \in E^+\}$.

We have a theorem similar to Theorem 2.1 in the previous section:

THEOREM 3.1 *Given an input (V, w^+, w^-) of weighted clustering, a set of budgets $\{c_{ij} : \{i,j\} \subseteq V\}$, and a graph $G = (V, E^+, E^-)$, where E^+, E^- is a partition of $\{\{i,j\} \subseteq V\}$ such that*

(i) $w^-_{ij} \le \alpha c_{ij}$ *for all* $\{i,j\} \in E^+$, *and*
$w^+_{ij} \le \alpha c_{ij}$ *for all* $\{i,j\} \in E^-$,

(ii) $w^+_{ij} + w^+_{jk} + w^-_{ki} \le \alpha(c_{ij} + c_{jk} + c_{ki})$ *for every bad triplet* $\{i,j\} \in E^+, \{j,k\} \in E^+, \{k,i\} \in E^-$.

Then CC-Pivot returns a solution that costs at most $\alpha \sum_{\{i,j\} \subseteq V} c_{ij}$ *if we choose a pivot* k *that minimizes*

$$\frac{\sum_{\{i,j\} \in T^+_k(G)} w^+_{ij} + \sum_{\{i,j\} \in T^-_k(G)} w^-_{ij}}{\sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} c_{ij}}. \tag{4}$$

PROOF. Let $G = (V, E^+, E^-)$ be a graph that satisfies the conditions in the theorem. For a pair $\{i,j\}$ with $\{i,j\} \in E^+$, we will let $w_{ij} = w^+_{ij}$ and $\bar{w}_{ij} = w^-_{ij}$. For a pair $\{i,j\}$ with $\{i,j\} \in E^-$, we let $w_{ij} = w^-_{ij}$ and $\bar{w}_{ij} = w^+_{ij}$.

If a pair i,j is clustered according to G, the cost incurred is $\bar{w}_{ij}$, and for each pair not clustered according to G, the cost is w_{ij}. In order to stress the similarities with the ranking algorithm, we will call the first type of cost "forward cost" and the second "backward cost".

The first condition implies that the forward cost is at most αc_{ij}. As in the ranking case, we will show that the second condition implies that we always choose a pivot so that the backward cost incurred by pivoting on this vertex is at most α times the budget for the vertex pairs involved.

For a given pivot k, $T^+_k(G) \cup T^-_k(G)$ is the set of pairs that incur a backward cost by pivoting on k when the set of vertices in the recursive call is V. Observe that if $\{i,j\} \in T^+_k(G)$, then i,j,k is a bad triplet, since $\{i,j\} \in E^+$, $\{i,k\} \in E^+$ and $\{j,k\} \in E^-$, and if $\{i,j\} \in T^-_k(G)$, then i,j,k is also a bad triplet, since $\{i,j\} \in E^-$, $\{i,k\} \in E^+$ and $\{j,k\} \in E^+$.

Therefore $T^+_k(G) \cup T^-_k(G)$ contains exactly the pairs that are in a bad triplet with k in G. The cost incurred for the pairs in $T^+_k(G) \cup T^-_k(G)$ if k is the pivot is equal to $\sum_{\{i,j\} \in T^+_k(G)} w^+_{ij} + \sum_{\{i,j\} \in T^-_k(G)} w^-_{ij} = \sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} w_{ij}$, and we have a budget for these vertex pairs of $\sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} c_{ij}$. The pivot chosen minimizes the ratio of the backward cost that is incurred to the budget for these pairs. We now show that the second condition implies that there exists a pivot for which this ratio is at most α.

Let T be the set of bad triplets in G, and for a bad triplet with $\{i,j\} \in E^+, \{j,k\} \in E^+, \{k,i\} \in E^-$, let $w(t) = w^+_{ij} + w^+_{jk} + w^-_{ki} = w_{ij} + w_{jk} + w_{ki}$ and let $c(t) = c_{ij} + c_{jk} + c_{ki}$. If we sum $\sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} w_{ij}$ over all $k \in V$, i.e. $\sum_{k \in V} \sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} w_{ij}$, then we count w_{ij} exactly once for every pivot k such that the vertex pairs $\{i,j\}, \{j,k\}, \{k,i\}$ are in a bad triplet, hence $\sum_{k \in V} \sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} w_{ij} = \sum_{t \in T} w(t)$. Similarly, $\sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} c_{ij} = \sum_{t \in T} c(t)$.

By the second condition of the theorem, $w(t) \le \alpha c(t)$. Therefore, there must exist some pivot k such that $\sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} w_{ij} \le \alpha \sum_{\{i,j\} \in T^+_k(G) \cup T^-_k(G)} c_{ij}$, and the backward cost incurred when pivoting on k is not more than α times the lower bound on the cost for the vertex pairs involved. □

REMARK 3.1 *The proof of Theorem 3.1 also implies that under the conditions of the theorem, choosing a pivot uniformly at random gives a solution with expected cost at most* $\alpha \sum_{\{i,j\} \subseteq V} c_{ij}$.

LEMMA 3.1 *The algorithm in Theorem 3.1 can be implemented in* $O(n^3)$ *time.*

PROOF. The implementation can be done exactly as in the case of the ranking algorithm, except that we now maintain a list of the bad triplets for which all three vertices are in a single recursive call, and for each vertex we maintain the total backward cost incurred if pivoting on that vertex and the total "budget" for the vertex pairs involved. □

3.1 Weighted clustering with triangle inequality and consensus clustering. We now show how to use Theorem 3.1 to obtain a 2-approximation for clustering with triangle inequality. As in the case of rank aggregation, we then show that we can use the theorem plus the input clusterings to obtain a $\frac{8}{5}$-approximation algorithm for consensus clustering.

THEOREM 3.2 *There exists a deterministic combinatorial 2-approximation algorithm for weighted clustering with triangle inequality which runs in $O(n^3)$ time.*

PROOF. Partition the edges of the complete graph on V into E^+, E^- so that if $\{i,j\} \in E^+$ then $w_{ij}^+ \geq w_{ij}^-$ and if $\{i,j\} \in E^-$ then $w_{ij}^- \geq w_{ij}^+$. Let $c_{ij} = \min\{w_{ij}^+, w_{ij}^-\}$ for every i,j. Clearly $\sum_{\{i,j\}\subseteq V} c_{ij}$ is a lower bound on the cost of any clustering of V.

The first condition of Theorem 3.1 is met for $\alpha \geq 1$. Let $\{i,j\} \in E^+, \{j,k\} \in E^+, \{k,i\} \in E^-$ be a bad triplet. By the triangle inequality on the weights,

$$w_{ij}^+ \leq w_{jk}^- + w_{ki}^+ = c_{jk} + c_{ki}, \tag{5}$$

$$w_{jk}^+ \leq w_{ki}^+ + w_{ij}^- = c_{ki} + c_{ij}, \tag{6}$$

$$w_{ki}^- \leq w_{ij}^- + w_{jk}^- = c_{ij} + c_{jk}. \tag{7}$$

Adding these three inequalities implies the second condition of Theorem 3.1 holds for $\alpha = 2$. Hence Theorem 3.1 gives a deterministic combinatorial 2-approximation algorithm, which by Lemma 3.1 can be implemented in $O(n^3)$ time. □

In the case of consensus clustering, we can again do better. Recall that we are given ℓ collections $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$, where each $\mathcal{C}_k$ is a partition of V in which each set has a label "final" or "not final". We defined $w_{ij}^+ = \frac{1}{\ell}\sum_{k=1}^{\ell} \mathbf{1}\{\exists C \in \mathcal{C}_k \text{ s.t. } \{i,j\} \subseteq C, C \text{ is final}\}$, and $w_{ij}^- = \frac{1}{\ell}\sum_{k=1}^{\ell} \mathbf{1}\{\exists C \in \mathcal{C}_k \text{ s.t. } |\{i,j\} \cap C| = 1\}$ where $\mathbf{1}\{.\}$ is the indicator function. We will first need a 2-approximation algorithm like RepeatChoice for partial rank aggregation from [2].

We will maintain two collections $\mathcal{C}_{\text{final}}$ and $\mathcal{C}_{\text{not final}}$ such that their union is a partitioning of V. We start by setting $\mathcal{C}_{\text{final}} = \emptyset, \mathcal{C}_{\text{not final}} = \{V\}$. We repeatedly choose an input clustering $\mathcal{C}_k$ uniformly at random without replacement; we consider the sets $C \in \mathcal{C}_k$ one by one, and use C to further divide the clusters in $\mathcal{C}_{\text{not final}}$. If C itself is final, the resulting sets become final clusters. To be precise, we find the collection $\mathcal{D} = \{C \cap \mathcal{C}_{\text{not final}}\}$. We update $\mathcal{C}_{\text{not final}}$ to be $\{\mathcal{C}_{\text{not final}} \backslash C\}$. If C is final, we add the sets in $\mathcal{D}$ to $\mathcal{C}_{\text{final}}$, otherwise, we add $\mathcal{D}$ to $\mathcal{C}_{\text{not final}}$.

Once all input clusterings have been considered, we add the clusters in $\mathcal{C}_{\text{not final}}$ to $\mathcal{C}_{\text{final}}$ and we output $\mathcal{C}_{\text{final}}$. Suppose that there still was some set C in $\mathcal{C}_{\text{not final}}$ at the end of the procces. If one of the input clusterings $\mathcal{C}_1, \ldots, \mathcal{C}_\ell$ has a final cluster containing a vertex in C, then this vertex would be in some set in $\mathcal{C}_{\text{final}}$. Hence the sets in $\mathcal{C}_{\text{not final}}$ contain vertices that are not in any final cluster in the input clusterings. Also, $\mathcal{C}_1, \ldots, \mathcal{C}_k$ must each contain some superset of C as a not final cluster, since otherwise C would not be a set in $\mathcal{C}_{\text{not final}}$.

We therefore get that for two distinct vertices i,j in some cluster $C \in \mathcal{C}_{\text{not final}}$ once all input clusterings have been considered then $w_{ij}^+ = 0$ and $w_{ij}^- = 0$. In other words, the input clusterings give us no information about the similarity or dissimilarity of i,j. We will denote this by $i \parallel j$.

CC-RepeatChoice$(V, \mathcal{C}_1, \ldots, \mathcal{C}_\ell)$

Initialize $\mathcal{C}_{\text{final}} \leftarrow \emptyset, \mathcal{C}_{\text{not final}} \leftarrow \{V\}$, $K \leftarrow \{1, \ldots, \ell\}$.
While $|K| > 0$
    Choose $k \in K$ uniformly at random.
    For $C \in C_k$ do
        $\mathcal{D} \leftarrow \{C \cap C' : C' \in \mathcal{C}_{\text{not final}}\}$.
        $\mathcal{C}_{\text{not final}} \leftarrow \{C' \backslash C : C' \in \mathcal{C}_{\text{not final}}\}$.
        If C is final, then
            $\mathcal{C}_{\text{final}} \leftarrow \mathcal{C}_{\text{final}} \cup \mathcal{D}$,
        else
            $\mathcal{C}_{\text{not final}} \leftarrow \mathcal{C}_{\text{not final}} \cup \mathcal{D}$.
    $K \leftarrow K \backslash \{k\}$.
$\mathcal{C}_{\text{final}} \leftarrow \mathcal{C}_{\text{not final}} \cup \mathcal{C}_{\text{final}}$.
Return $\mathcal{C}_{\text{final}}$.

LEMMA 3.2 *Let $\mathcal{C}$ be the output of CC-RepeatChoice($V, \mathcal{C}_1, \ldots, \mathcal{C}_\ell$). Then the expected cost of $\mathcal{C}$ is $\sum_{\{i,j\}: i \not\parallel j} 2\frac{w_{ij}^+ w_{ij}^-}{w_{ij}^+ + w_{ij}^-}$.*

PROOF. Note that two vertices i, j which are in some set in $\mathcal{C}_{\text{not final}}$ are separated into different clusters if $\mathcal{C}_k$ contains a cluster C (which may be final or not final) which contains only one of i and j. If, on the other hand, $\mathcal{C}_k$ contains a final cluster C that contains both i and j, then i and j will be in the same cluster in $\mathcal{C}_{\text{final}}$ and hence in the output $\mathcal{C}$. Hence the probability that two vertices i, j, with $i \not\parallel j$ are in the same cluster of $\mathcal{C}$ is equal to $\frac{w_{ij}^+}{w_{ij}^+ + w_{ij}^-}$, and the probability that they are not in the same cluster of $\mathcal{C}$ is $\frac{w_{ij}^-}{w_{ij}^+ + w_{ij}^-}$. It follows that the expected cost in $\mathcal{C}$ for pair i, j is $2\frac{w_{ij}^+ w_{ij}^-}{w_{ij}^+ + w_{ij}^-}$. □

Since $2\frac{w_{ij}^+ w_{ij}^-}{w_{ij}^+ + w_{ij}^-} \leq 2\min\{w_{ij}^+, w_{ij}^-\}$, CC-RepeatChoice is a 2-approximation algorithm. This algorithm can be derandomized using the method of conditional expectation [17].

THEOREM 3.3 *There exists a deterministic combinatorial $\frac{8}{5}$-approximation algorithm for partial consensus clustering that runs in $O(n^3)$ time.*

PROOF. We can exploit the fact that the (deterministic) CC-RepeatChoice algorithm gives us a clustering $\mathcal{C}$ of cost at most $\sum_{\{i,j\}: i \not\parallel j} 2\frac{w_{ij}^+ w_{ij}^-}{w_{ij}^+ + w_{ij}^-}$ in a similar fashion as in the rank aggregation case.

As in the proof for Theorem 2.3, we can consider a modified instance with weights

$$\begin{aligned} \tilde{w}_{ij}^+ &= \frac{2}{5} w_{ij}^+ + \frac{3}{5} 2\frac{w_{ij}^+ w_{ij}^-}{w_{ij}^+ + w_{ij}^-}, \\ \tilde{w}_{ij}^- &= \frac{2}{5} w_{ij}^- + \frac{3}{5} 2\frac{w_{ij}^+ w_{ij}^-}{w_{ij}^+ + w_{ij}^-} \end{aligned}$$

for $i \not\parallel j$. If $i \parallel j$, then we let $\tilde{w}_{ij}^+ = \tilde{w}_{ij}^- = 0$, since in that case both $w_{ij}^+ = 0$ and $w_{ij}^- = 0$, and no cost is ever incurred for i, j.

We again let $c_{ij} = \min\{w_{ij}^+, w_{ij}^-\}$. We will show that Theorem 3.1 implies that we can find a clustering with modified cost at most $\frac{8}{5}\sum_{\{i,j\} \subseteq V} c_{ij} \leq \frac{8}{5} OPT$, where OPT is the optimal value of the original instance. By the definition of the modified costs, it follows that either this clustering has (original) cost at most $\frac{8}{5} OPT$, or the CC-RepeatChoice solution has cost at most $\frac{8}{5} OPT$.

We let $G = (V, E^+, E^-)$ be the same as in the proof of Theorem 3.2, i.e. $\{i, j\} \in E^+$ only if $w_{ij}^+ \geq w_{ij}^-$ and $\{i, j\} \in E^-$ only if $w_{ij}^- \geq w_{ij}^+$ (breaking ties arbitrarily).

Let $w_{ij} = \max\{w_{ij}^+, w_{\{j,i\}}^-\}$, and let $a_{ij} = w_{(i,j)}^+ + w_{(j,i)}^- = w_{ij} + c_{ij}$. We rewrite the weights in terms of this notation: If $\{i, j\} \in E^+$, then $w_{ij}^+ = w_{ij}, w_{ij}^- = c_{ij}$, hence if $i \not\parallel j$

$$\begin{aligned} \tilde{w}_{ij}^+ &= \frac{2}{5} w_{ij} + \frac{6}{5} \frac{w_{ij} c_{ij}}{w_{ij} + c_{ij}} \\ \tilde{w}_{ij}^- &= \frac{2}{5} c_{ij} + \frac{6}{5} \frac{w_{ij} c_{ij}}{w_{ij} + c_{ij}}. \end{aligned}$$

Similarly, $\{i, j\} \in E^-$, then $w_{ij}^+ = c_{ij}, w_{ij}^- = w_{ij}$, and for $i \not\parallel j$,

$$\begin{aligned} \tilde{w}_{ij}^+ &= \frac{2}{5} c_{ij} + \frac{6}{5} \frac{w_{ij} c_{ij}}{w_{ij} + c_{ij}} \\ \tilde{w}_{ij}^- &= \frac{2}{5} w_{ij} + \frac{6}{5} \frac{w_{ij} c_{ij}}{w_{ij} + c_{ij}}. \end{aligned}$$

We see that in this notation, if $i \not\parallel j$ then irrespective of whether the edge $\{i, j\}$ is in E^+ or E^-, the backward cost is $\frac{2}{5} w_{ij} + \frac{6}{5} \frac{w_{ij} c_{ij}}{w_{ij} + c_{ij}}$ and the forward cost is $\frac{2}{5} c_{ij} + \frac{6}{5} \frac{w_{ij} c_{ij}}{w_{ij} + c_{ij}}$.

The first condition of Theorem 3.1 is met for $\alpha \geq \frac{8}{5}$, since for $i \not\parallel j$, $\frac{2}{5} c_{ij} + \frac{6}{5} \frac{w_{ij} c_{ij}}{w_{ij} + c_{ij}} \leq \frac{8}{5} c_{ij}$ and for $i \parallel j$, $\tilde{w}_{ij}^+ = \tilde{w}_{ij}^- = c_{ij} = 0$.

To verify the second condition, let i, j, k be a bad triplet in G with $\{i,j\} \in E^+, \{j,k\} \in E^+, \{k,i\} \in E^-$, and let $t = \{\{i,j\},\{j,k\},\{k,i\}\}$. We need to show that

$$\tilde{w}^+_{ij} + \tilde{w}^+_{jk} + \tilde{w}^-_{ki} \leq \frac{8}{5}(c_{ij} + c_{jk} + c_{ki}). \tag{8}$$

We rewrite the expression for the backward cost of $g \not\parallel h$ as

$$\frac{2}{5}w_{gh} + \frac{6}{5}\frac{w_{gh}c_{gh}}{w_{gh}+c_{gh}} = \frac{2}{5}w_{gh} + \frac{6}{5}\frac{w_{gh}(a_{gh}-w_{gh})}{a_{gh}} = \frac{8}{5}w_{gh} - \frac{6}{5}\frac{w^2_{gh}}{a_{gh}}.$$

Since pairs $\{g,h\} \in t$ such that $g \parallel h$ contribute 0 to both sides of (8), we need to show that $\sum_{\{g,h\}\in t: g\not\parallel h}\left(\frac{8}{5}w_{gh} - \frac{6}{5}\frac{w^2_{gh}}{a_{gh}}\right) \leq \frac{8}{5}\sum_{\{g,h\}\in t: g\not\parallel h} c_{gh}$.

By adding inequalities (5), (6) and (7), we see that $w_{ij}+w_{jk}+w_{ki} = w^+_{ij}+w^+_{jk}+w^-_{ki} \leq 2(c_{ij}+c_{jk}+c_{ki})$. Hence $3(w_{ij}+w_{jk}+w_{ki}) \leq 2(c_{ij}+c_{jk}+c_{ki})+2(w_{ij}+w_{jk}+w_{ki}) = 2(a_{ij}+a_{jk}+a_{ki})$. Hence the conditions of Claim 2.1 are satisfied, and therefore $\sum_{\{g,h\}\in t: g\not\parallel h}\left(\frac{8}{5}w_{gh} - \frac{6}{5}\frac{w^2_{gh}}{a_{gh}}\right) \leq \sum_{\{g,h\}\in t: g\not\parallel h}\frac{8}{5}(a_{gh}-w_{gh})$ as required. □

3.2 Correlation clustering and weighted clustering with probability constraints. In the case of clustering with probability constraints, we can use the optimal solution to a linear programming relaxation to provide the graph $G = (V, E^+, E^-)$ and the budgets c_{ij} in Theorem 3.1.

Let $x^+_{ij} = 1$ denote that i and j are in the same cluster, $x^+_{ij} = 0$ that i are j are not in the same cluster, and let $x^-_{ij} = 1 - x^+_{ij}$. The variables must satisfy the triangle inequality, which combined with the fact that they satisfy the probability constraints, reduces to $x^-_{ij} + x^-_{jk} + x^+_{ik} \geq 1$ for every i, j, k. These constraints capture the fact that it is impossible that i and j are in the same cluster ($x^-_{ij} = 0$), j and k are in the same cluster ($x^-_{jk} = 0$), but i and k are not in the same cluster ($x^+_{ik} = 0$). The following linear program thus gives a lower bound on the value of an optimal clustering:

$$(LP_{CC}) \qquad \begin{array}{lll} \min & \displaystyle\sum_{\{i,j\}\subseteq V}(x^+_{ij}w^-_{ij} + x^-_{ij}w^+_{ij}) & \\ \text{s.t.} & x^-_{ij} + x^-_{jk} + x^+_{ik} \geq 1 & \text{for all distinct } i,j,k \\ & x^+_{ij} + x^-_{ij} = 1 & \text{for all } i \neq j \\ & x^+_{ij}, x^-_{ij} \geq 0 & \text{for all } i \neq j. \end{array}$$

This linear program was also used by Ailon et al. [4] for the randomized rounding algorithm we will discuss in Section 5. We show that it can be used together with Theorem 3.1 to give a simple deterministic rounding that guarantees a 3-approximate solution.

THEOREM 3.4 *There exists a deterministic 3-approximation algorithm for weighted clustering with probability constraints.*

PROOF. Let x be an optimal solution to (LP_{CC}). We will let $c_{ij} = x^+_{ij}w^-_{ij} + x^-_{ij}w^+_{ij}$, and we partition the edges of the complete graph on V into E^+, E^- so that if $\{i,j\} \in E^+$ then $x^+_{ij} \geq x^-_{ij}$, and if $\{i,j\} \in E^-$ then $x^-_{ij} \geq x^+_{ij}$.

The first condition in Theorem 3.1 holds with $\alpha = 2$, since if $\{i,j\} \in E^+$, then $x^+_{ij} \geq \frac{1}{2}$, hence $c_{ij} = x^+_{ij}w^-_{ij} + x^-_{ij}w^+_{ij} \geq \frac{1}{2}w^-_{ij}$, and similarly if $\{i,j\} \in E^-$, then $x^-_{ij} \geq \frac{1}{2}$, hence $c_{ij} \geq \frac{1}{2}w^+_{ij}$.

To see that the second condition holds, let i, j, k be a bad triplet with $\{i,j\} \in E^+, \{j,k\} \in E^+, \{k,i\} \in E^-$ and let $t = \{\{i,j\},\{j,k\},\{k,i\}\}$. Let $w_{ij} = w^+_{ij}, w_{jk} = w^+_{jk}, w_{ki} = w^-_{ki}$, and let $\bar{w}_{ij} = w^-_{ij}, \bar{w}_{jk} = w^-_{jk}, \bar{w}_{ki} = w^+_{ki}$; in other words w_{gh} is the backward cost for pair $\{g,h\}$ and $\bar{w}_{gh}$ is the forward cost. We need to show that $w_{ij}+w_{jk}+w_{ki} \leq 3(c_{ij}+c_{jk}+c_{ki})$. Let $x_{gh} = x^+_{(g,h)}$ if $\{g,h\} \in E^+$, and let $x_{gh} = x^-_{(g,h)}$ otherwise. Then

$$c_{gh} = w_{gh}(1 - x_{gh}) + \bar{w}_{gh}x_{gh} = w_{gh} + (\bar{w}_{gh} - w_{gh})x_{gh}. \tag{9}$$

We also note that the feasibility of solution to (LP_{CC}) implies that $(1 - x_{ij}) + (1 - x_{jk}) + (1 - x_{ki}) \geq 1$, or $x_{ij} + x_{jk} + x_{ki} \leq 2$.

Suppose $\bar{w}_{ij} - w_{ij} = \min\{\bar{w}_{ij} - w_{ij}, \bar{w}_{jk} - w_{jk}, \bar{w}_{ki} - w_{ki}\}$. This assumption is without loss of generality, since we will not be using any property to distinguish $\{i,j\}$ from $\{j,k\}$ and $\{k,i\}$ except for the fact that $\bar{w}_{ij} - w_{ij} = \min\{\bar{w}_{ij} - w_{ij}, \bar{w}_{jk} - w_{jk}, \bar{w}_{ki} - w_{ki}\}$.

To give a lower bound on $c(t)$, we consider the case that $\bar{w}_{ij} - w_{ij} \geq 0$ and the case that $\bar{w}_{ij} - w_{ij} < 0$. In the first case, also $\bar{w}_{jk} - w_{jk} \geq 0$ and $\bar{w}_{ki} - w_{ki} \geq 0$, and hence we see from (9) that $c_{ij} + c_{jk} + c_{ki} \geq w_{ij} + w_{jk} + w_{ki}$.

On the other hand, if $\bar{w}_{ij} - w_{ij} < 0$, then we use the fact that $x_{ij} + x_{jk} + x_{ki} \leq 2$, and that each x-value is at least $\frac{1}{2}$ by definition. If we look at

$$c_{ij} + c_{jk} + c_{ki} = w_{ij} + w_{jk} + w_{ki} + (\bar{w}_{ij} - w_{ij})x_{ij} + (\bar{w}_{jk} - w_{jk})x_{jk} + (\bar{w}_{ki} - w_{ki})x_{ki},$$

then we see that by our assumption that $\bar{w}_{ij} - w_{ij}$ is the most negative value among the three, the smallest possible value this can take is if $x_{ij} = 1, x_{jk} = \frac{1}{2}, x_{ki} = \frac{1}{2}$, hence

$$\begin{aligned} c_{ij} + c_{jk} + c_{ki} &\geq w_{ij} + w_{jk} + w_{ki} + (\bar{w}_{ij} - w_{ij}) + \frac{1}{2}(\bar{w}_{jk} - w_{jk}) + \frac{1}{2}(\bar{w}_{ki} - w_{ki}) \\ &= \bar{w}_{ij} + \frac{1}{2}(w_{jk} + \bar{w}_{jk}) + \frac{1}{2}(w_{ki} + \bar{w}_{ki}). \end{aligned} \tag{10}$$

Since the weights satisfy the probability constraints, $w_{gh} + \bar{w}_{gh} = 1$, and hence the above is at least 1. But by the probability constraints we also know that $w_{ij} + w_{jk} + w_{ki} \leq 3$. □

REMARK 3.2 *The LP based method from Theorem 3.4 can also be applied to give a 2-approximation algorithm for instances when the weights satisfy the triangle inequality. In the case when the weights satisfy the triangle inequality, we note from equations (5), (6), (7) that* $\bar{w}_{jk} + \bar{w}_{ki} \geq w_{ij}$*, hence the quantity in (10) is not less than* $\frac{1}{2}(w_{ij} + w_{jk} + w_{ki})$.

4. Constrained problems. We now turn to constrained problems. In a constrained feedback arc set problem, we are given a partial order (V, P) in addition to (V, w), and our output needs to be a linear extension of P.

In a constrained clustering problem we are given sets P^+, P^- such that any feasible clustering should have i, j in the same cluster if $\{i,j\} \in P^+$ and in different clusters if $\{i,j\} \in P^-$. We also assume that P^+, P^- are consistent and transitive, i.e. that $P^+ \cap P^- = \emptyset$ and if $\{i,j\}, \{j,k\} \in P^+$ then $\{k,i\} \in P^+$ and if $\{i,j\} \in P^+, \{j,k\} \in P^-$ then $\{k,i\} \in P^-$.

For a constrained problem with general weights, we can give an extra set of conditions on the tournament $G = (V, A)$ or the graph $G = (V, E^+, E^-)$ in Theorems 2.1 and 3.1, that ensure that the solution returned by the FAS-Pivot and CC-Pivot algorithms satisfy a given set of constraints.

LEMMA 4.1 *Given a partial order P, if the tournament $G = (V, A)$ in Theorem 2.1 satisfies that if $(i,j) \in P$ then*

(*i*) $(i,j) \in A$, *and*

(*ii*) *there exists no k such that both $(j,k) \in A, (k,i) \in A$,*

then FAS-Pivot outputs a linear extension of P.

PROOF. It is clear from the proof of Theorem 2.1 that a pair i, j is not ordered according to the arc connecting them in the tournament G only if there is some directed triangle containing this arc. Hence the conditions of the lemma ensure that the solution is a linear extension of P. □

LEMMA 4.2 *Given sets $P^+, P^- \subseteq \{\{i,j\} : \{i,j\} \subseteq V\}$, if $G = (V, E^+, E^-)$ in Theorem 2.1 satisfies:*

(*i*) $\{i,j\} \in P^+$ *implies that* $\{i,j\} \in E^+$ *and* $\{i,j\} \in P^-$ *implies that* $\{i,j\} \in E^-$.

(*ii*) *If $\{i,j\} \in P^+$, then there exists no k such that both $\{j,k\} \in E^+, \{k,i\} \in E^-$, and if $\{i,j\} \in P^-$, then there exists no k such that both $\{j,k\} \in E^+, \{k,i\} \in E^+$.*

then CC-Pivot outputs a clustering that has i, j in the same cluster if $\{i,j\} \in P^+$ and i, j in different clusters if $\{i,j\} \in P^-$.

PROOF. If $\{i,j\} \in E^+$, then the only case in which the clustering found by CC-Pivot will have i and j in different clusters is if there is some bad triplet containing i,j and the third vertex of the bad triplet is chosen as pivot. Similarly, if $\{i,j\} \in E^-$ then i and j can only end up in the same cluster if there is a bad triplet containing i,j. Hence the conditions of the lemma ensure that the clustering found by CC-Pivot satisfies the requirements given by P^+, P^-. □

4.1 Ranking and clustering with probability constraints. Using Lemmas 4.1 and 4.2, we can now show that the results in Theorem 2.4 and Theorem 3.4 for weighted problems with probability constraints also hold for constrained versions of these problems.

THEOREM 4.1 *There exists a deterministic 3-approximation algorithm for constrained weighted minimum feedback arc set with probability constraints.*

PROOF. We only need to show that we can ensure that the tournament described in the proof of Theorem 2.4 satisfies the two conditions of Lemma 4.1.

Let P be the input partial order. We add the following constraints to (LP_{FAS}):

$$x_{(i,j)} = 1 \text{ for all } (i,j) \in P.$$

Let x be an optimal solution to (LP_{FAS}). Note that $\sum_{\{i,j\}\subseteq V}(w_{(i,j)}x_{(j,i)} + w_{(j,i)}x_{(i,j)})$ still provides a lower bound on the optimal value. As in the proof of Theorem 2.4, we form a tournament $G = (V, A)$ by including arc (i,j) only if $x_{(i,j)} \geq \frac{1}{2}$. Such a tournament satisfies the first condition of Lemma 4.1. We can ensure that we satisfy the second condition by being careful about breaking ties, as shown below.

Suppose $(i,j) \in P$ and $(i,j),(j,k),(k,i)$ is a directed triangle in G. Since $(i,j) \in P$, $x_{(i,j)} = 1$, and if $(j,k),(k,i)$ are in A, then we must have $x_{(j,k)} \geq \frac{1}{2}, x_{(k,i)} \geq \frac{1}{2}$. But by the first set of constraints of (LP_{FAS}), $x_{(i,k)} + x_{(k,j)} \geq 1 - x_{(j,i)} = 1$, or $1 - x_{(k,i)} + 1 - x_{(j,k)} \geq 1$, hence $x_{(k,i)}$ and $x_{(j,k)}$ must be equal to $\frac{1}{2}$. Hence we can ensure that there are no directed triangles containing arcs in P by first labeling the vertices such that if $(i,j) \in P$, then $label(i) < label(j)$. In the case of a tie $x_{(g,h)} = x_{(h,g)} = \frac{1}{2}$, we add arc (g,h) to A if $label(g) < label(h)$. Now an arc $(i,j) \in P$ cannot be in a directed triangle $(i,j),(j,k),(k,i)$ in A, since if it were then $label(j) < label(k)$ and $label(k) < label(i)$, which contradicts the property of the labeling that $(i,j) \in P$ implies $label(i) < label(j)$. □

THEOREM 4.2 *There exists a deterministic 3-approximation algorithm for constrained clustering with probability constraints.*

PROOF. We only need to show that we can ensure that the partition of the edges into E^+, E^- described in the proof of Theorem 3.4 satisfies the two conditions of Lemma 4.2. Let P^+, P^- be the sets of vertex pairs that need to be in the same cluster respectively in different clusters. We add the following set of constraints to (LP_{CC}):

$$x^+_{ij} = 1 \text{ for all } \{i,j\} \in P^+$$
$$x^-_{ij} = 1 \text{ for all } \{i,j\} \in P^-.$$

Note that we can do this while maintaining that the optimal value of (LP_{CC}) gives a lower bound on the cost of the optimal clustering.

We label the vertices in V such that $\{i,j\} \in P^+ \Rightarrow label(i) = label(j)$ and $\{i,j\} \in P^- \Rightarrow label(i) \neq label(j)$. Note that this can be done by having one distinct label for each connected component of the graph (V, P^+), and giving this label to all the vertices in the component. We now partition the edges into E^+, E^- as in the proof of Theorem 3.4 except that in the case when $x^+_{ij} = x^-_{ij}$, we no longer break ties arbitrarily, but add $\{i,j\}$ to E^+ if i and j have the same label, and otherwise we add $\{i,j\}$ to E^-. The edge sets E^+, E^- satisfy the first condition of Lemma 4.2. We now verify that they satisfy the second condition.

Suppose $\{i,j\} \in P^+$ and i,j and k form a bad triplet, i.e. $\{i,j\} \in E^+, \{j,k\} \in E^+, \{k,i\} \in E^-$. Then $x^+_{jk} \geq \frac{1}{2}, x^-_{ki} \geq \frac{1}{2}$, but by the triangle inequality constraints of (LP_{CC}) and the fact that $x^+_{ij} = 1$ if $\{i,j\} \in P^+$, we have $x^-_{jk} + x^+_{ki} \geq 1$, or $1 - x^+_{jk} + 1 - x^-_{ki} \geq 1$, so $x^+_{jk} = \frac{1}{2}$ and $x^-_{ki} = \frac{1}{2}$. But then $label(j) = label(k)$ and $label(k) \neq label(i)$, so $label(j) \neq label(i)$ which contradicts the properties of our labeling, since $\{i,j\} \in P^+$. A similar contradiction can be derived if we assume there is a bad triplet containing $\{i,j\} \in P^-$. □

4.2 Ranking and clustering with triangle inequality. Note that the arguments in Section 4.1 combined with Remarks 2.2 and 3.2 imply deterministic 2-approximation algorithms for ranking and clustering with triangle inequality based on their respective linear programming relaxations.

To get combinatorial algorithms for the constrained problems with triangle inequality, we need to make an additional assumption. We'll say that the input is *consistent* with the constraints if $w_{(j,i)} = 0$ for $(i,j) \in P$ in the case of a ranking problem. For a constrained clustering problem, consistency implies that if $\{i,j\} \in P^+$ then $w^-_{ij} = 0$ and if $\{i,j\} \in P^-$ then $w^+_{ij} = 0$. If we assume that the input is *consistent* with the constraints, we can also show that the algorithms in Sections 2.1 and 3.1 (which use the weights to determine the tournament $G = (V, A)$ or the graph $G = (V, E^+, E^-)$ rather than an optimal solution to the respective LP relaxation) return solutions that obey the constraints. The proof is similar to the proofs of Theorems 4.1 and 4.2.

However, in the case of a consistent input with triangle inequality, we can also just do a "clean up" of any given solution to ensure that the constraints are satisfied, without increasing the cost of the solution. We thank Frans Schalekamp for suggesting that this might be the case.

LEMMA 4.3 *Given an input (V, w) of weighted feedback arc set, a partial order P, with weights that satisfy the triangle inequality and are consistent with P, and a permutation π, we can find a permutation π' that is a linear extension of P and costs not more than π in time $O(n|P|)$.*

PROOF. Let $(i,j) \in P$ and suppose $\pi(j) < \pi(i)$. We call such (i,j) violated. Let $K(i,j)$ be the set of vertices k such that $\pi(j) < \pi(k) < \pi(i)$, and let (i^*, j^*) be a violated pair such that for any vertex $k \in K(i^*, j^*)$ it is the case that $(j^*, k) \notin P$ and $(k, i^*) \notin P$. We will show that such a pair always exists if a violated pair exists and comment on how to find such a pair when we discuss the running time of the algorithm.

Consider the permutation π' we obtain by moving j^* to the position just after i^* with probability $p = \frac{1}{2}$ or otherwise moving i^* to the position just before j^*. Note that (i^*, j^*) is not violated in π' and no new violations are created.

The expected difference in the cost of permutations π' and π is given by

$$\begin{aligned} &w_{(j^*,i^*)} - w_{(i^*,j^*)} + \frac{1}{2} \sum_{k \in K(i^*,j^*)} (w_{(j^*,k)} - w_{(k,j^*)} + w_{(k,i^*)} - w_{(i^*,k)}) \\ &\leq w_{(j^*,i^*)} - w_{(i^*,j^*)} + \frac{1}{2} \sum_{k \in K(i^*,j^*)} (2w_{(j^*,i^*)}) = -w_{(i^*,j^*)} \leq 0, \end{aligned}$$

where the first inequality follows from the triangle inequality, since $w_{(j^*,k)} \leq w_{(j^*,i^*)} + w_{(i^*,k)}$ and $w_{(k,i^*)} \leq w_{(k,j^*)} + w_{(j^*,i^*)}$, and the last equality follows since $w_{(j^*,i^*)} = 0$. Hence either moving j^* to the position just after i^* or moving i^* to the position just before j^* does not increase the cost of the permutation, and has fewer violations. Repeating this until all violations have been removed gives the result.

The total time required to implement this algorithm is $O(n|P|)$: We begin by constructing a list of violated pairs (i,j) sorted by increasing $(\pi(i) - \pi(j))$. Consider any violated pair (i,j) in the list, and suppose there exists some $k \in K(i,j)$ such that $(j,k) \in P$ or $(k,i) \in P$. In the first case, transitivity of P implies that $(i,k) \in P$, hence (i,k) is a violated pair and $\pi(i) - \pi(k) < \pi(i) - \pi(j)$, so (i,k) appears above (i,j) in the list of violated pairs. Similarly, in the second case (k,j) is a violated pair that appears above (i,j) in the list. We will therefore go through the list in order, and "uncross" violated pairs using the above procedure. Note that it may be the case that, after "uncrossing" a violated pair, the list of the remaining violated pairs may no longer be ordered by $(\pi(i) - \pi(j))$. However, since no new violated pairs are created, the only new vertices k in $K(i,j)$ for a violated pair (i,j) must satisfy that $(j,k) \notin P$ and $(k,i) \notin P$. Therefore, if we go through the original list in order, we know that once we consider pair (i,j), then we have already removed all violations (i,k) or (k,j) with $k \in K(i,j)$, and hence we know that for all $k \in K(i,j)$ we have that $(j,k) \notin P$ and $(k,i) \notin P$.

For each violated pair (i,j) that we consider, it takes at most $O(n)$ time to "uncross" the pair using the above procedure, because we need to compare $w_{(j,i)} - w_{(i,j)} + \sum_{k \in K(i,j)} (w_{(j,k)} - w_{(k,j)})$ and $w_{(j,i)} -$

$w_{(i,j)} + \sum_{k \in K(i,j)} w_{(k,i)} - w_{(i,k)})$, thus giving a total running time of $O(|P| \log(|P|) + |P|n) = O(n|P|)$. □

LEMMA 4.4 *Given an input (V, w^+, w^-) of weighted clustering, constraints given by P^+, P^-, with weights that satisfy the triangle inequality and are consistent with P^+, P^-, and a clustering $\mathcal{C}$ that does not satisfy P^+, P^-, then we can find a clustering $\mathcal{C}'$ that satisfies P^+, P^- and does not cost more than $\mathcal{C}$ in time $O(n(|P^+| + |P^-|))$.*

PROOF. We will say a clustering $\mathcal{C}$ violates $\{i,j\} \in P^+$ if it has i and j in different clusters, and we will say it violates $\{i,j\} \in P^-$ if it has i and j in the same cluster. Consider a clustering $\mathcal{C}$ that violates $\{i,j\} \in P^-$. Let C be the cluster containing i and j. Let $C_i = \{i\} \cup \{i' \in C : \{i,i'\} \in P^+\}$, and similarly define C_j. Note that for $i' \in C_i, j' \in C_j$, it must be the case that $w^+_{\{i',j'\}} = 0$, since $\{i',j'\} \in P^-$ by transitivity of the constraints.

Let $R = C \backslash (C_i \cup C_j)$. We randomly construct a clustering $\mathcal{C}'$ from $\mathcal{C}$ by making C_i, C_j and R separate clusters, and then merging R with C_i with probability $\frac{|C_i|}{|C_i|+|C_j|}$ or with C_j with probability $\frac{|C_j|}{|C_i|+|C_j|}$. Then the expected difference in cost between $\mathcal{C}$ and $\mathcal{C}'$ is

$$\sum_{i' \in C_i, j' \in C_j} (w^+_{\{i',j'\}} - w^-_{\{i',j'\}}) + \frac{1}{|C_i|+|C_j|} \sum_{k \in R} \Big(\sum_{j' \in C_j} |C_i|(w^+_{\{k,j'\}} - w^-_{\{k,j'\}}) + \sum_{i' \in C_i} |C_j|(w^+_{\{k,i'\}} - w^-_{\{k,i'\}}) \Big).$$

The term $\sum_{i' \in C_i, j' \in C_j} (w^+_{\{i',j'\}} - w^-_{\{i',j'\}})$ is nonpositive, since $w^+_{\{i',j'\}} = 0$ for all $i' \in C_i, j' \in C_j$.

We now fix some $k \in R$ and consider

$$\frac{1}{|C_i|+|C_j|} \sum_{j' \in C_j} |C_i|(w^+_{\{k,j'\}} - w^-_{\{k,j'\}}) + \sum_{i' \in C_i} |C_j|(w^+_{\{k,i'\}} - w^-_{\{k,i'\}}).$$

We rewrite this as

$$\frac{1}{|C_i|+|C_j|} \sum_{i' \in C_i} \sum_{j' \in C_j} (w^+_{\{k,j'\}} - w^-_{\{k,j'\}} + w^+_{\{k,i'\}} - w^-_{\{k,i'\}}).$$

Now, note that $w^+_{\{k,j'\}} \le w^-_{\{k,i'\}} + w^+_{\{i',j'\}} = w^-_{\{k,i'\}}$ and similary $w^+_{\{k,i'\}} \le w^-_{\{k,j'\}}$. Hence the second term is nonpositive also.

We can evaluate the cost of the two possible clusterings in time $O(n|C_i||C_j|)$. Letting $\mathcal{C}'$ be the cheaper of the two clusterings does not increase the cost of the clustering. Moreover, no new violations are created, and $|C_i||C_j| \ge 1$ violations are removed since for any $i' \in C_i, j' \in C_j$ we have that $\{i',j'\} \in P^-$.

Repeatedly applying this procedure ensures that $\mathcal{C}'$ has no violations of P^- in time $O(n|P^-|)$.

Now suppose $\mathcal{C}'$ still violates $\{i,j\} \in P^+$. Let C be the cluster containing i and D the cluster containing j. Let $C_i = \{i\} \cup \{i' \in C : \{i,i'\} \in P^+\}$ and let $C_j = \{j\} \cup \{j' \in D : \{j,j'\} \in P^+\}$. By transitivity, $\{i',j'\} \in P^+$ for every $i' \in C_i, j' \in C_j$, and hence by consistency, $w^-_{\{i',j'\}} = 0$.

We randomly construct a clustering $\mathcal{C}''$ obtained from $\mathcal{C}'$ by either adding C_i to D with probability $\frac{|C_j|}{|C_i|+|C_j|}$ or adding C_j to C with probability $\frac{|C_i|}{|C_i|+|C_j|}$. Note that this does not create any new violations by transitivity of the constraints, and the assumption that no violations of P^- exist.

The expected difference in cost between $\mathcal{C}'$ and $\mathcal{C}''$ is

$$\sum_{i' \in C_i, j' \in C_j} (w^-_{\{i',j'\}} - w^+_{\{i',j'\}}) \tag{11}$$

$$+ \sum_{k \in C \backslash C_i} \frac{|C_j|}{|C_i|+|C_j|} \sum_{i' \in C_i} (w^+_{\{k,i'\}} - w^-_{\{k,i'\}}) + \sum_{k \in C \backslash C_i} \frac{|C_i|}{|C_i|+|C_j|} \sum_{j' \in C_j} (w^-_{\{k,j'\}} - w^+_{\{k,j'\}}) \tag{12}$$

$$+ \sum_{k \in D \backslash C_j} \frac{|C_i|}{|C_i|+|C_j|} \sum_{j' \in C_j} (w^+_{\{k,j'\}} - w^-_{\{k,j'\}}) + \sum_{k \in D \backslash C_j} \frac{|C_j|}{|C_i|+|C_j|} \sum_{i' \in C_i} (w^-_{\{k,i'\}} - w^+_{\{k,i'\}}). \tag{13}$$

The term in (11) is nonpositive, since $\{i',j'\} \in P^+$, so $w^-_{\{i',j'\}} = 0$ for every $i' \in C_i, j' \in C_j$.

Now fix $k \in C \backslash C_i$, and note that

$$\begin{aligned}
&\frac{|C_j|}{|C_i|+|C_j|}\sum_{i'\in C_i}(w^+_{\{k,i'\}} - w^-_{\{k,i'\}}) + \frac{|C_i|}{|C_i|+|C_j|}\sum_{j'\in C_j}(w^-_{\{k,j'\}} - w^+_{\{k,j'\}}) \\
&= \frac{1}{|C_i|+|C_j|}\sum_{i'\in C_i}\sum_{j'\in C_j}(w^+_{\{k,i'\}} - w^-_{\{k,i'\}} + w^-_{\{k,j'\}} - w^+_{\{k,j'\}}) \\
&\leq \frac{1}{|C_i|+|C_j|}\sum_{i'\in C_i}\sum_{j'\in C_j}\Big((w^+_{\{k,j'\}} + w^-_{\{i',j'\}}) - w^-_{\{k,i'\}} + (w^-_{\{k,i'\}} + w^-_{\{i',j'\}}) - w^+_{\{k,j'\}}\Big) \\
&= 0.
\end{aligned}$$

The inequality follows from the fact that by the triangle inequality $w^+_{\{k,i'\}} \leq w^+_{\{k,j'\}} + w^-_{\{i',j'\}}$ and $w^-_{\{k,j'\}} \leq w^-_{\{k,i'\}} + w^-_{\{i',j'\}}$. By symmetry, we thus find that (12) and (13) are both nonpositive. Hence replacing $\mathcal{C}'$ by the option with the smaller cost gives a clustering with $|C_i||C_j| \geq 1$ fewer violations without increasing the cost. We can find $\mathcal{C}'$ in $O(n|C_i||C_j|)$ time, and $\mathcal{C}'$ has $|C_i||C_j|$ fewer violations than before. Repeating this procedure gives the result. □

5. Better LP based algorithms. In Sections 2.2 and 3.2 we deterministically rounded the optimal solution to an LP relaxation to obtain the graph $G = (V, A)$ or $G = (V, E^+, E^-)$. Ailon, Charikar and Newman [4] propose randomized rounding algorithms based on the same LP relaxation: in the case of a feedback arc set problem, the solution to (LP_{FAS}) from Section 2.2 is rounded by repeatedly choosing a pivot k at random, and partitioning the vertices in sets V_L and V_R as before, but now a vertex i is put into V_L with probability $x_{(i,k)}$ and into V_R with probability $x_{(k,i)} = 1 - x_{(i,k)}$. A similar approach is proposed for weighted clustering, where the optimal solution to (LP_{CC}) from Section 3.2 is rounded by repeatedly choosing a pivot k at random and partitioning the vertices into R and C, by including i in C with probability x^+_{ik} and in R with probability $x^-_{ik} = 1 - x^+_{ik}$. Ailon [2] generalizes this randomized rounding algorithm for weighted feedback arc set with triangle inequality by perturbing the LP solution by a function $h(x)$ that satisfies $h(x)+h(1-x) = 1$, and using the perturbed LP solution as probabilities.

We now show how to extend the ideas from the previous sections to derandomize the randomized rounding algorithms in Ailon et al. [4], and the perturbed version in Ailon [2]. In particular, this allows us to obtain a deterministic $\frac{5}{2}$-approximation algorithm for ranking and clustering with probability constraints, and a $\frac{3}{2}$-approximation algorithm for ranking and clustering with triangle inequality. Combined with the ideas from Theorem 2.3 and Theorem 3.3, this also allows us to obtain a deterministic $\frac{4}{3}$-approximation algorithm for full rank aggregation and full consensus clustering, which matches the best randomized algorithms in [4].

5.1 Weighted Feedback Arc Set. We will now give a formal description of the LP rounding algorithms for weighted feedback arc set in [4] and [2]. We give the algorithm in a generalized form, where $p = \{p_{(i,j)} : \{i,j\} \subseteq V\}$ satisfying $p_{(i,j)} + p_{(j,i)} = 1$ gives the probabilities that a vertex is ordered to the left or right of the pivot vertex. In the algorithms in [4] and [2], the pivot is chosen at random from the vertices in V, and p is determined by an optimal solution x to (LP_{FAS}) from Section 2.2.

FASLP-Pivot(V, p)

Pick a pivot $k \in V$.
Set $V_L = \emptyset, V_R = \emptyset$.
For all $i \in V, i \neq k$,
    with probability $p_{(i,k)}$: add i to V_L,
    else (with probability $p_{(k,i)}$): add i to V_R.
Return FASLP-Pivot(V_L, p), k, FASLP-Pivot(V_R, p).

Suppose j is ordered before i in the output of FASLP-Pivot. Note that this means that there was some recursive call that contained both i and j in which either (i) one of them was the pivot, and j was ordered to the left of i, or (ii) some vertex $k \neq i, j$ was the pivot, and j was added to V_L and i was added to V_R. We will say that the cost $w_{(i,j)}$ of ordering j before i is a forward cost in case (i), and we will say it is a backward cost in case (ii). Note the difference from our previous definition of forward and

backward costs. We will say a pair i, j gets *decided* in a particular iteration of FasLP-Pivot if it either incurs a forward cost (i.e. one of i, j is the pivot) or a backward cost (i.e. one of them gets assigned to V_L and one to V_R).

Let $T_k(V)$ be the set of arcs (i, j) that become backward in a recursive call on V when $k \neq i, j$ is the pivot, i.e. j is in V_L and i is in V_R. Note that $T_k(V)$ is a random set, since V_L, V_R are random sets. In particular, the probability that a particular arc (i, j) is in $T_k(V)$ is $p_{(j,k)}p_{(k,i)}$. For notational convenience, we define $p^k_{(j,i)}$ as the probability that j is ordered to the left of i when k is the pivot vertex, i.e. $p^k_{(j,i)} = p_{(j,k)}p_{(k,i)}$. Then the expected backward cost in the iteration is $\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} w_{(i,j)}\Big] = \sum_{\{i,j\}\subseteq V\setminus\{k\}} \Big(p^k_{(j,i)}w_{(i,j)} + p^k_{(i,j)}w_{(j,i)}\Big)$.

The algorithm has two sources of randomness: the pivot is chosen randomly from the vertex set, and the vertices are ordered to the left or right of the pivot according to given probabilities. We will derandomize the algorithm in two steps. We will again have a notion of a budget c_{ij} for each vertex pair, and we choose a pivot k such that ratio of the expected cost for the arcs in $T_k(V)$ and $\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} c_{ij}\Big]$ is as small as possible. Then we use the method of conditional expectation [6] to assign the vertices in $V\setminus\{k\}$ to V_L or V_R. We start by analyzing the algorithm that chooses a pivot deterministically, but randomly assigns the vertices in $V\setminus\{k\}$ to V_L and V_R according to the probabilities $p_{(i,k)}, p_{(k,i)}$. The following theorem states conditions under which this gives a solution for which the expected cost is within a factor α of a given budget.

Theorem 5.1 *Given an input (V, w) of weighted feedback arc set, a probability matrix p, and budgets $\{c_{ij} : \{i, j\} \subseteq V\}$, such that*

(i) $p_{(i,j)}w_{(j,i)} + p_{(j,i)}w_{(i,j)} \leq \alpha c_{ij}$ for all distinct $i, j \in V$,

(ii) for every distinct triple $\{i, j, k\}$ in V,

$$\big(p^k_{(i,j)}w_{(j,i)} + p^k_{(j,i)}w_{(i,j)}\big) + \big(p^i_{(j,k)}w_{(k,j)} + p^i_{(k,j)}w_{(j,k)}\big) + \big(p^j_{(k,i)}w_{(i,k)} + p^j_{(i,k)}w_{(k,i)}\big) \leq$$
$$\alpha\Big(\big(p^k_{(i,j)} + p^k_{(j,i)}\big)c_{ij} + \big(p^i_{(j,k)} + p^i_{(k,j)}\big)c_{jk} + \big(p^j_{(k,i)} + p^j_{(i,k)}\big)c_{ki}\Big)$$

then FAS-LPPivot gives a solution with expected cost at most $\alpha\sum_{\{i,j\}\subseteq V} c_{ij}$ if we choose a pivot k that minimizes

$$\frac{\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} w_{(i,j)}\Big]}{\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} c_{ij}\Big]}. \tag{14}$$

Proof. For iteration ℓ of FAS-LPPivot, let X_ℓ be the cost of the pairs that get decided, and let Y_ℓ be the budget for the pairs that get decided. Since FasLP-Pivot has at most n iterations, we define these values for $\ell = 1, \ldots, n$, where $X_\ell = Y_\ell = 0$ if the FasLP-Pivot algorithm has fewer than $\ell - 1$ recursive calls. We will show that

$$\mathbb{E}\Big[X_\ell\Big] \leq \alpha\mathbb{E}\Big[Y_\ell\Big] \text{ for } \ell = 1, \ldots, n. \tag{15}$$

Note that $\mathbb{E}\Big[\sum_{\ell=1}^n X_\ell\Big]$ is the expected cost of the solution constructed by FAS-LPPivot and that $\sum_{\ell=1}^n Y_\ell = \sum_{\{i,j\}\subseteq V} c_{ij}$ deterministically, since each pair gets decided in exactly one iteration. By linearity of expectation, $\mathbb{E}\Big[\sum_{\ell=1}^n X_\ell\Big] = \sum_{\ell=1}^n \mathbb{E}\Big[X_\ell\Big]$. Hence proving (15) is sufficient to prove the theorem.

Under the first condition in the theorem, the expected cost for a pair that gets decided and incurs a forward cost in an iteration of FasLP-Pivot is at most α times the budget for this pair.

On the other hand, if we show that there always exists a pivot for which the ratio in (14) is at most α, then the expected combined cost of the pairs that get decided and incur a backward cost is at most α times the expected budget for the pairs that incur a backward cost.

Recall that

$$\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} w_{(i,j)}\Big] = \sum_{\{i,j\}\subseteq V\setminus\{k\}} \Big(p^k_{(j,i)}w_{(i,j)} + p^k_{(i,j)}w_{(j,i)}\Big).$$

Now consider $\sum_{k\in V}\sum_{\{i,j\}\subseteq V\setminus\{k\}}\left(p^k_{(j,i)}w_{(i,j)}+p^k_{(i,j)}w_{(j,i)}\right)$: we add $p^k_{(j,i)}w_{(i,j)}+p^k_{(i,j)}w_{(j,i)}$ once for every pair $\{i,j\}$ and vertex $k\notin\{i,j\}$. We can attribute this amount to the triple $\{i,j,k\}$. Then we see that to every triple of distinct vertices i,j,k we attribute a cost of

$$p^k_{(j,i)}w_{(i,j)}+p^k_{(i,j)}w_{(j,i)}+p^j_{(i,k)}w_{(k,i)}+p^j_{(k,i)}w_{(i,k)}+p^i_{(k,j)}w_{(j,k)}+p^i_{(j,k)}w_{(k,j)}. \tag{16}$$

For a triple $t=\{i,j,k\}$ of distinct vertices, we will denote the quantity in (16) by $w(t)$. So if we let T be the set of all distinct triples of vertices, then $\sum_{k\in V}\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)}w_{(i,j)}\Big]=\sum_{t\in T}w(t)$.

Similarly, in

$$\sum_{k\in V}\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)}c_{ij}\Big]=\sum_{k\in V}\sum_{\{i,j\}\subseteq V\setminus\{k\}}\left(p^k_{(j,i)}+p^k_{(i,j)}\right)c_{ij},$$

we add $(p^k_{(j,i)}+p^k_{(i,j)})c_{ij}$ for every pair $\{i,j\}$ and vertex $k\notin\{i,j\}$. If we again attribute this amount to the triple $\{i,j,k\}$, and let $c(t)$ be the total budget attributed to triple $t=\{i,j,k\}$, then we see that

$$c(t)=\big(p^k_{(j,i)}+p^k_{(i,j)}\big)c_{ij}+\big(p^j_{(i,k)}+p^j_{(k,i)}\big)c_{ki}+\big(p^i_{(k,j)}+p^i_{(j,k)}\big)c_{jk}. \tag{17}$$

and $\sum_{k\in V}\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)}c_{ij}\Big]=\sum_{t\in T}c(t)$.

Now, note that by the second condition of the theorem $w(t)\le\alpha c(t)$, and we conclude that

$$\sum_{k\in V}\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)}w_{(i,j)}\Big]\le\alpha\sum_{k\in V}\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)}c_{ij}\Big]$$

and hence a vertex k for which the ratio in (14) is at most α exists. □

Remark 5.1 *The proof of Theorem 5.1 also implies that FasLP-Pivot with a randomly chosen pivot gives a solution with expected cost at most* $\alpha\sum_{\{i,j\}\subseteq V}c_{ij}$.

We now know that we can choose a pivot deterministically, but to give a deterministic algorithm, we need to assign the vertices to V_L and V_R deterministically, instead of randomly as in FasLP-Pivot. We can achieve this by using the method of conditional expectation [6].

Theorem 5.2 *Under the conditions in Theorem 5.1, there exists a deterministic algorithm for weighted feedback arc set that returns a solution of cost at most* $\alpha\sum_{\{i,j\}\subseteq V}c_{ij}$.

Proof. We define the following notation: Let V_L,V_R,V' be a partition of $V\setminus\{k\}$, and let $\mathbb{E}\Big[W_k(V)\Big|V_L,V_R\Big]$ be the expected total cost incurred in an iteration of FasLP-Pivot for the arcs that get decided in that iteration when pivoting on k conditioned on the vertices in V_L and V_R being ordered to the left and right of k respectively and the vertices in V' are ordered left or right with probability $p_{(i,k)}$ and $p_{(k,i)}$. Let $\mathbb{E}\Big[C_k(V)\Big|V_L,V_R\Big]$ be the expected total budget c_{ij} for the vertex pairs i,j that get decided in an iteration of FasLP-Pivot when pivoting on k, again conditioned on V_L,V_R. Note that the conditional expected backward cost is

$$\begin{aligned}\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)}w_{(i,j)}\Big|V_L,V_R\Big]=\quad&\sum_{j\in V_L,i\in V_R}w_{(i,j)}+\sum_{\{i,j\}\subseteq V'}\left(p^k_{(j,i)}w_{(i,j)}+p^k_{(i,j)}w_{(j,i)}\right)\\&+\sum_{j\in V',i\in V_R}p_{(j,k)}w_{(i,j)}+\sum_{j\in V_L,i\in V'}p_{(k,i)}w_{(i,j)},\end{aligned}$$

and the conditional expected budget for these vertex pairs is

$$\begin{aligned}\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)}c_{ij}\Big|V_L,V_R\Big]=\quad&\sum_{j\in V_L,i\in V_R}c_{ij}+\sum_{\{i,j\}\subseteq V'}\left(p^k_{(j,i)}+p^k_{(i,j)}\right)c_{ij}\\&+\sum_{j\in V',i\in V_R}p_{(j,k)}c_{ij}+\sum_{j\in V_L,i\in V'}p_{(k,i)}c_{ij}.\end{aligned}$$

DerandFASLP-Pivot(V,p)

Pick $k \in V$ minimizing $\dfrac{\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} w_{(i,j)}\Big]}{\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} c_{ij}\Big]}$.
Set $V_L = \emptyset, V_R = \emptyset, V' = V\backslash\{k\}$.
While there exists $i \in V'$
    If $\dfrac{\mathbb{E}\Big[W_k(V)\Big|\ V_L \cup \{i\}, V_R\Big]}{\mathbb{E}\Big[C_k(V)\Big|V_L \cup \{i\}, V_R\Big]} \le \dfrac{\mathbb{E}\Big[W_k(V)\ \Big|\ V_L, V_R \cup \{i\}\Big]}{\mathbb{E}\Big[C_k(V)\Big|V_L, V_R \cup \{i\}\Big]}$
        add i to V_L,
    else
        add i to V_R.
    $V' = V'\backslash\{i\}$.
Return DerandFASLP-Pivot(V_L,p), k, DerandFASLP-Pivot(V_R,p).

Figure 1: Derandomization of FASLP-Pivot.

Hence we can easily compute these conditional expectations and we get that the conditional expectation of the forward plus backward cost is equal to

$$\begin{aligned}\mathbb{E}\Big[W_k(V)\Big|V_L,V_R\Big] &= \sum_{i\in V_L} w_{(k,i)} + \sum_{i\in V_R} w_{(i,k)} + \sum_{i\in V'}\Big(p_{(i,k)}w_{(k,i)} + p_{(k,i)}w_{(i,k)}\Big)\\ &\quad + \mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} w_{(i,j)}\Big|V_L,V_R\Big],\end{aligned}$$

and the conditional expectation of the forward plus backward budget is equal to

$$\mathbb{E}\Big[C_k(V)\Big|V_L,V_R\Big] = \sum_{i\in V\backslash\{k\}} c_{ik} + \mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} c_{ij}\Big|V_L,V_R\Big].$$

Initializing $V_L = \emptyset, V_R = \emptyset, V' = V\backslash\{k\}$, then we have $\mathbb{E}\Big[W_k(V)\Big|V_L,V_R\Big] \le \alpha\mathbb{E}\Big[C_k(V)\Big|V_L,V_R\Big]$, since

$$\mathbb{E}\Big[W_k(V)\Big|V_L=\emptyset,V_R=\emptyset\Big] = \sum_{i\in V\backslash\{k\}}\Big(p_{(i,k)}w_{(k,i)} + p_{(k,i)}w_{(i,k)}\Big) + \mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} w_{(i,j)}\Big],$$

and

$$\mathbb{E}\Big[C_k(V)\Big|V_L=\emptyset,V_R=\emptyset\Big] = \sum_{i\in V\backslash\{k\}} c_{ik} + \mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} c_{ij}\Big],$$

and by the first condition of Theorem 5.1 $p_{(i,k)}w_{(k,i)} + p_{(k,i)}w_{(i,k)} \le \alpha c_{ik}$, and by the second condition and the proof of Theorem 5.1, $\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} w_{(i,j)}\Big] \le \alpha\mathbb{E}\Big[\sum_{(i,j)\in T_k(V)} c_{ij}\Big]$.

By standard conditional expectation arguments, we know that if we consider some vertex $i \in V'$ and $\mathbb{E}\Big[W_k(V)\Big|V_L,V_R\Big] \le \alpha\mathbb{E}\Big[C_k(V)\Big|V_L,V_R\Big]$, then we can add i to either V_L or V_R and maintain the invariant that $\mathbb{E}\Big[W_k(V)\Big|V_L,V_R\Big] \le \alpha\mathbb{E}\Big[C_k(V)\Big|V_L,V_R\Big]$.

Hence we obtain the algorithm in Figure 5.1. By the invariant, at the end of the algorithm, we have a partition V_L, V_R of $V\backslash\{k\}$ such that $\mathbb{E}\Big[W_k(V)\Big|V_L,V_R\Big] \le \alpha\mathbb{E}\Big[C_k(V)\Big|V_L,V_R\Big]$. Since V_L, V_R partition $V\backslash\{k\}$, $\mathbb{E}\Big[W_k(V)\Big|V_L,V_R\Big]$ is now equal to the (deterministic) cost of ordering the vertices in V_L and V_R to the left and right of k respectively, and $\mathbb{E}\Big[C_k(V)\Big|V_L,V_R\Big]$ is the total budget of the pairs that get decided in this iteration. □

COROLLARY 5.1 *There exists a deterministic $\frac{5}{2}$-approximation algorithm for weighted feedback arc set with probability constraints.*

PROOF. By Theorem 5.2, it suffices to show that we can give probabilities $\{p_{(i,j)}\}$ and a lower bound $\sum_{\{i,j\}\subseteq V} c_{ij}$ that satisfy the two conditions in Theorem 5.1 with $\alpha = 2.5$. Let x be an optimal solution to (LP_{FAS}) in Section 2.2. Let $p_{(i,j)} = x_{(i,j)}, p_{(j,i)} = x_{(j,i)}$ and $c_{ij} = x_{(i,j)}w_{(j,i)} + x_{(j,i)}w_{(i,j)}$ for every $i, j \in V$. Clearly, the first condition is satisfied with $\alpha = 1$. The second condition was shown to hold with $\alpha = \frac{5}{2}$ in Lemma 13 in [4]. □

COROLLARY 5.2 *There exists a deterministic $\frac{3}{2}$-approximation algorithm for weighted feedback arc set with triangle inequality.*

PROOF. Let x be an optimal solution to (LP_{FAS}) in Section 2.2, and let

$$h(y) = \begin{cases} \frac{3}{4}y, & 0 \le y \le \frac{1}{3} \\ \frac{3}{2}y - \frac{1}{4}, & \frac{1}{3} < y \le \frac{2}{3} \\ \frac{3}{4}y + \frac{1}{4}, & \frac{2}{3} < y \le 1 \end{cases}$$

Let $p_{(i,j)} = h(x_{(i,j)})$ and $c_{ij} = x_{(i,j)}w_{(j,i)} + x_{(j,i)}w_{(i,j)}$ for every $i, j \in V$. These settings satisfy the conditions in Theorem 5.1 with $\alpha = \frac{3}{2}$, which was shown by Ailon in the proof of Theorem 4.1 in [2]. □

COROLLARY 5.3 *There exists a deterministic $\frac{4}{3}$-approximation algorithm for full rank aggregation.*

PROOF. Let x be an optimal solution to the LP, and let $c_{ij} = x_{(i,j)}w_{(j,i)} + x_{(j,i)}w_{(i,j)}$, and let $p_{(i,j)} = x_{(i,j)}$. Because we are considering a full rank aggregation instance, RepeatChoice will give us a permutation π of cost at most $\sum_{\{i,j\}\subseteq V} 2w_{(i,j)}w_{(j,i)}$. By the proof of Theorem 2.3 it is therefore enough to consider a modified instance with weights $\tilde{w}_{(i,j)} = \frac{2}{3}w_{(i,j)} + \frac{1}{3}(2w_{(i,j)}w_{(j,i)})$, and show that for this modified instance DerandFASLP-Pivot will output a permutation with modified cost at most $\frac{4}{3}\sum_{\{i,j\}\subseteq V} c_{ij}$.

Note that $\tilde{w}_{(i,j)} \le \frac{4}{3}w_{(i,j)}$ for any (i,j), so

$$p_{(i,j)}\tilde{w}_{(j,i)} + p_{(j,i)}\tilde{w}_{(i,j)} \le \frac{4}{3}\big(x_{(i,j)}w_{(j,i)} + x_{(j,i)}w_{(i,j)}\big) = \frac{4}{3}c_{ij},$$

and hence the first condition of Theorem 5.1 is satisfied for $\alpha = \frac{4}{3}$.

The second condition is exactly Lemma 14 in [4]. □

5.2 Weighted Clustering. We give a brief sketch of the randomized rounding approach, and subsequent derandomization, for the clustering case. Given $p = \{p_{ij}^+, p_{ij}^- : \{i,j\} \subseteq V\}$ that satisfy $p_{ij}^+ + p_{ij}^- = 1$, the general form of the randomized rounding algorithm is

CCLP-Pivot(V, p)

Pick a pivot $k \in V$.
Set $C = \{k\}, V_R = \emptyset$.
For all $i \in V, i \ne k$,
    with probability p_{ik}^+: add i to C,
    else (with probability p_{ik}^-): add i to R.
Return $\{C$, CCLP-Pivot$(R, p)\}$.

We will say a pair i, j incurs a forward cost if the vertices i, j were in the same recursive call in which one of them was the pivot, and we will say a pair i, j incurs a backward cost if the vertices i, j were in the same recursive call, in which some vertex $k \ne i, j$ was the pivot, and i and j are not both in the next recursive call. We will say a pair i, j gets *decided* in a particular iteration of CCLP-Pivot if it either incurs a forward or a backward cost.

Note that there are two ways in which a pair can incur a backward cost: let $T_k^+(V)$ be the pairs i, j such that exactly one of i, j is not in the next recursive call (for which we incur a cost w_{ij}^+), and let $T_k^-(V)$ be the pairs i, j such that neither i nor j is in the next recursive call (incurring a cost w_{ij}^-). Let

$p^-_{\{i,j\},k} = p^-_{ik}p^+_{jk} + p^+_{ik}p^-_{jk}$, and let $p^+_{\{i,j\},k} = p^+_{ik}p^+_{jk}$. Note that

$$\begin{aligned}\mathbb{E}\Big[\sum_{\{i,j\}\in T^+_k(V)} w^+_{ij}\Big] &= \sum_{\{i,j\}\subseteq V\setminus\{k\}} p^-_{\{i,j\},k} w^+_{ij}\\ \mathbb{E}\Big[\sum_{\{i,j\}\in T^-_k(V)} w^-_{ij}\Big] &= \sum_{\{i,j\}\subseteq V\setminus\{k\}} p^+_{\{i,j\},k} w^-_{ij}.\end{aligned}$$

We analyze the algorithm that chooses a pivot deterministically, but randomly assigns the vertices in $V\setminus\{k\}$ to C and R according to the probabilities given by p. The following theorem states conditions under which this gives a solution within a factor α of a given budget.

THEOREM 5.3 *Given an input (V, w^+, w^-) of weighted clustering, probabilities p, budgets $\{c_{ij} : \{i,j\} \subseteq V\}$ such that*

(i) $p^+_{ij}w^-_{ij} + p^-_{ij}w^+_{ij} \le \alpha c_{ij}$ for all distinct $i, j \in V$,

(ii) for every distinct triple $\{i,j,k\}$ in V,

$$\begin{aligned}&\big(p^+_{\{i,j\},k}w^-_{ij} + p^-_{\{i,j\},k}w^+_{ij}\big) + \big(p^+_{\{j,k\},i}w^-_{jk} + p^-_{\{j,k\},i}w^+_{jk}\big) + \big(p^+_{\{k,i\},j}w^-_{ki} + p^-_{\{k,i\},j}w^+_{ki}\big) \le\\ &\quad\alpha\Big(\big(p^+_{\{i,j\},k} + p^-_{\{i,j\},k}\big)c_{ij} + \big(p^+_{\{j,k\},i} + p^-_{\{j,k\},i}\big)c_{jk} + \big(p^+_{\{k,i\},j} + p^-_{\{k,i\},j}\big)c_{ki}\Big).\end{aligned}$$

Then CCLP-Pivot returns a solution with expected cost at most $\alpha\sum_{\{i,j\}\subseteq V} c_{ij}$ if we choose a pivot k that minimizes

$$\frac{\mathbb{E}\Big[\sum_{\{i,j\}\in T^+_k(V)} w^+_{ij} + \sum_{\{i,j\}\in T^-_k(V)} w^-_{ij}\Big]}{\mathbb{E}\Big[\sum_{\{i,j\}\in T^+_k(V)\cup T^-_k(V)} c_{ij}\Big]}. \tag{18}$$

PROOF. As in the proof of Theorem 5.1, it suffices to show that for any iteration of CCLP-Pivot, the expected cost incurred for the pairs that get decided it at most α times the expected budget for the pairs that get decided.

Under the first condition in the theorem, the expected cost for a pair that gets decided and incurs a forward cost in an iteration of CCLP-Pivot is at most α times the budget for this pair.

As in the proof of Theorem 5.1 and Theorem 3.1, one can show that under the second condition, there always exists a pivot for which the ratio in (18) is at most α. Hence the expected combined cost of the pairs that get decided and incur a backward cost is at most α times the expected budget for the pairs that get decided and incur a backward cost. □

As in the ranking algorithm, given a pivot chosen according to (18), we can use the method of conditional expectation [6] to deterministically assign the vertices in $V\setminus\{k\}$ to C and R, without increasing the ratio between the cost and the budget.

THEOREM 5.4 *Under the conditions in Theorem 5.3, there exists a deterministic algorithm for weighted clustering that returns a solution of cost at most $\alpha\sum_{\{i,j\}\subseteq V} c_{ij}$.*

PROOF. We define the following notation: Let R, C, V' be a partition of $V\setminus\{k\}$, and let $\mathbb{E}\Big[W_k(V)\Big|R,C\Big]$ be the expected total cost incurred in an iteration of CCLP-Pivot for the pairs that get decided in that iteration when pivoting on k conditioned on the vertices in C being clustered with k and the vertices in R not being clustered with k, where the vertices in V' are clustered together with k with probability p^+_{ik} and separated from k with probability p^-_{ik}. Let $\mathbb{E}\Big[C_k(V)\Big|R,C\Big]$ be the expected total budget c_{ij} for the vertex pairs i, j that get decided in an iteration of CCLP-Pivot when pivoting on k, again conditioned on R and C.

If $C = R = \emptyset$, then by Theorem 5.3 we have $\mathbb{E}\Big[W_k(V)\Big|R,C\Big] \le \alpha\mathbb{E}\Big[C_k(V)\Big|R,C\Big]$. If we can compute $\mathbb{E}\Big[W_k(V)\Big|R,C\Big]$ and $\mathbb{E}\Big[C_k(V)\Big|R,C\Big]$ for any R, C then by the definition of conditional expectation we

can iterate through the vertices in V' and add them to C or R while maintaining that $\mathbb{E}\Big[W_k(V)\Big|R,C\Big] \leq \alpha\mathbb{E}\Big[C_k(V)\Big|R,C\Big]$.

Note that the conditional expected backward cost is

$$\sum_{i\in C}\left(\sum_{j\in R} w^+_{ij} + \sum_{j\in C: i<j} w^-_{ij} + \sum_{j\in V'}\left(p^+_{\{k,j\}}w^-_{ij} + p^-_{\{k,j\}}w^+_{ij}\right)\right) + \sum_{i\in R}\sum_{j\in V'} p^+_{\{k,j\}}w^+_{ij} + \sum_{\{i,j\}\subseteq V'}\left(p^+_{\{i,j\},k}w^-_{ij} + p^-_{\{i,j\},k}w^+_{ij}\right). \tag{19}$$

The conditional expectation of the forward cost is

$$\sum_{i\in C} w^-_{ik} + \sum_{i\in R} w^+_{ik} + \sum_{i\in V'}(p^+_{ik}w^-_{ik} + p^-_{ik}w^+_{ik}). \tag{20}$$

Hence we can efficiently compute $\mathbb{E}\Big[W_k(V)\Big|R,C\Big]$ by adding up (19) and (20). The conditional expected budget $\mathbb{E}\Big[C_k(V)\Big|R,C\Big]$ is obtained by replacing w^+_{ij} and w^-_{ij} by c_{ij} and w^+_{ik}, w^-_{ik} by c_{ik} in the expression for $\mathbb{E}\Big[W_k(V)\Big|R,C\Big]$. □

COROLLARY 5.4 *There exists a deterministic $\frac{5}{2}$-approximation algorithm for weighted clustering with probability constraints.*

PROOF. By Theorem 5.4, it suffices to show that we can give probabilities p and a lower bound $\sum_{\{i,j\}\subseteq V} c_{ij}$ that satisfy the two conditions in Theorem 5.3 with $\alpha = 2.5$. Let x be an optimal solution to (LP_{CC}) in Section 3.2. Let $p^+_{ij} = x^+_{ij}, p^-_{ij} = x^-_{ij}$ and $c_{ij} = x^+_{ij}w^-_{ij} + x^-_{ij}w^+_{ij}$ for every $i, j \in V$. Clearly, the first condition is satisfied with $\alpha = 1$. The second condition was shown to hold with $\alpha = \frac{5}{2}$ in Lemma 15 in [4]. □

We note that the result in Corollary 5.4 is close to the best possible result we can get if we use the optimal value of (LP_{CC}) as a lower bound, since Charikar, Guruswami and Wirth [13] give the following example for which the integrality gap is 2: Given vertex set $V = \{1,\dots,n\}$, let $w^+_{\{n,j\}} = 1$ for $j = 1,\dots,n-1$ and let $w^+_{ij} = 0$ otherwise, and let $w^-_{ij} = 1 - w^+_{ij}$. Then the optimal fractional solution has $x^+_{\{n,j\}} = \frac{1}{2}$ for $j = 1,\dots,n-1$ and $x^+_{ij} = 0$ otherwise, with objective value $\frac{1}{2}(n-1)$. An optimal clustering for this example places one (or two) of the vertices in a cluster with vertex n, and all other vertices in singleton clusters, which gives objective value $(n-2)$.

COROLLARY 5.5 *There exists a deterministic $\frac{4}{3}$-approximation algorithm for full consensus clustering.*

PROOF. Let x be an optimal solution to (LP_{CC}) in Section 3.2. Let $p^+_{ij} = x^+_{ij}, p^-_{ij} = x^-_{ij}$ and $c_{ij} = x^+_{ij}w^-_{ij} + x^-_{ij}w^+_{ij}$ for every $i, j \in V$.

We consider modified weights $\tilde{w}$ defined as $\tilde{w}^+_{ij} = \frac{2}{3}w^+_{ij} + \frac{2}{3}w^+_{ij}w^-_{ij}$ and $\tilde{w}^-_{ij} = \frac{2}{3}w^-_{ij} + \frac{2}{3}w^+_{ij}w^-_{ij}$ and show that DerandCCLP-Pivot finds a clustering with modified cost at most $\frac{4}{3}\sum_{\{i,j\}\subseteq V} c_{ij}$. By the proof of Theorem 3.3, this means that either the output of DerandCCLP-Pivot (when ran on the modified instance), or the output of CC-RepeatChoice, has cost at most $\frac{4}{3}\sum_{\{i,j\}\subseteq V} c_{ij}$.

To invoke Theorem 5.4, we need to show that the weights $\tilde{w}$ satisfy the conditions in Theorem 5.3. It is easily verified that the first condition of Theorem 5.3 is satisfied, and the second condition is exactly Lemma 16 in [4]. □

6. Hierarchical Clustering. Recall that an M-level hierarchical clustering of a set V is a nested clustering of the elements in V, where the clustering at level m is a refinement of the clustering at level $m+1$. Given a set V and a matrix D with $D_{ij} \in \{0,\dots,M\}$ for any distinct $i, j \in V$, we want to find an M-level hierarchical clustering of V minimizing $\sum_{i,j\in V} |D_{ij} - \lambda_{ij}|$, where λ_{ij} is the number of levels in which i and j are in different clusters, or equivalently, since the clusterings are nested, i and j are in

different clusters at levels $1,\dots,\lambda_{ij}$, and in the same cluster at levels $\lambda_{ij}+1,\dots,M$. By Lemma 1(a) in [24] this is equivalent to finding an *ultrametric* that minimizes the ℓ_1 distance with D. An ultrametric is a tree metric in which all vertices are at the leaves of the tree, and the distance from each leaf to the root is the same.

A natural approach to the hierarchical clustering problem is to construct a hierarchical clustering in a top-down fashion: we first use a clustering algorithm to find the top-level clustering $\mathcal{C}_M$, and for each of the clusters in $\mathcal{C}_M$, we call the clustering algorithm to find a refinement, which together form $\mathcal{C}_{M-1}$, etcetera.

This approach is different from the approach used by Ailon and Charikar in [3]: although their algorithm is also a pivoting algorithm, it does not construct clusters top down, but instead uses the pivot to correct the distances of triples that violate the *ultrametric property*: For any ultrametric D and for any distinct i,j,k, it is the case that $D_{ij} \le \max\{D_{jk}, D_{ik}\}$. If k is the pivot, then the algorithm of Ailon and Charikar fixes the distances D_{ik} and D_{jk} to their current value, and adjusts D_{ij} to $\max\{D_{ik}, D_{jk}\}$ if $D_{ik} \neq D_{jk}$ or to $\min\{D_{ij}, D_{jk}\}$ otherwise. The algorithm then recurses on the sets $\{j \in V : D_{jk} = m\}$ for $m = 1,\dots,M$. The analysis is rather complicated and uses two different LP relaxations to give an upper bound on the cost of the algorithm's solution. Our algorithm and analysis are much simpler, and obtain the same approximation guarantee.

We start by redefining the input of the hierarchical clustering problem to make the connection to correlation clustering more obvious. For every pair of distinct vertices i,j and for every level $m \in \{1,\dots,M\}$, we let

$$w_{ij}^{+,m} = \begin{cases} 1 & \text{if } D_{ij} < m \\ 0 & \text{otherwise,} \end{cases} \qquad w_{ij}^{-,m} = 1 - w_{ij}^{+,m}. \tag{21}$$

Given an M-level hierarchical clustering and values λ_{ij} for every i,j such that i and j are in different clusters at levels $1,\dots,\lambda_{ij}$, and in the same cluster at levels $\lambda_{ij}+1,\dots,M$, then

$$\begin{aligned}
& \sum_{m=1}^{M} \Big(w_{ij}^{+,m} \mathbf{1}\{i,j \text{ in different clusters at level } m\} + w_{ij}^{-,m} \mathbf{1}\{i,j \text{ in the same cluster at level } m\} \Big) \\
= \; & \sum_{m=1}^{M} \Big(w_{ij}^{+,m} \mathbf{1}\{m \le \lambda_{ij}\} + w_{ij}^{-,m} \mathbf{1}\{m > \lambda_{ij}\} \Big) \\
= \; & \sum_{m=1}^{M} \Big(\mathbf{1}\{D_{ij} < m\} \mathbf{1}\{m \le \lambda_{ij}\} + \mathbf{1}\{D_{ij} \ge m\} \mathbf{1}\{m > \lambda_{ij}\} \Big) \\
= \; & |D_{ij} - \lambda_{ij}|.
\end{aligned}$$

Hence we can interpret the value $w_{ij}^{+,m}$ as the cost of having i and j in different clusters at level m, and $w_{ij}^{-,m}$ as the cost of having i and j in the same cluster at level m. At each level, we are solving a correlation clustering problem, with the additional requirement that the clusterings need to be nested.

We can now just think of calling the algorithm for correlation clustering from Section 3 for the top level, get a clustering $\mathcal{C}_M$ and then call CC-Pivot again for each cluster in $\mathcal{C}_M$ to get $\mathcal{C}_{M-1}$ etcetera. Hence we will need to have a partition of $\{\{i,j\} : \{i,j\} \subseteq V\}$ into E^+ and E^- for each of the M levels. Let these be given by $G^m = (V, E^{+,m}, E^{-,m})$.

We will now give our algorithm.

Hierarchical-Cluster$(G^M,\dots,G^1)$

$\mathcal{C}_M$ = CC-Pivot$(G^M(V))$
For $m = M-1$ down to 1
    $\mathcal{C}_m = \emptyset$
    for $C \in \mathcal{C}_{m+1}$
        $\mathcal{C}_m \leftarrow \mathcal{C}_m \cup$ CC-Pivot$(G^m(C))$
return $\{\mathcal{C}_1,\dots,\mathcal{C}_M\}$.

We cannot invoke Theorem 3.1 in this case, because if after the first call to CC-Pivot, i,j are in different clusters in $\mathcal{C}_M$, then clearly i,j will also be in different clusters in $\mathcal{C}_{M-1},\dots,\mathcal{C}_1$. Hence we will

need to attribute a cost of $\sum_{m=1}^{M} w_{ij}^{+,m}$ if we separate i,j at the M-th level. Note that if i and j are in the same cluster in $\mathcal{C}_M$, then we only need to attribute a cost of $w_{ij}^{-,M}$.

We assume that we are given a budget c_{ij}^m for each pair of distinct vertices i,j and each level m. Our next theorem will give conditions under which we can run Hierarchical-Cluster and find a hierarchical clustering that costs at most $(M+2)\sum_{m=1}^{M}\sum_{\{i,j\}\subseteq V} c_{ij}^m$.

Let $T_k^+(G^m)$ and $T_k^-(G^m)$ be defined similar to $T_k^+(G)$ and $T_k^-(G)$ in Section 3: $T_k^+(G^m) = \{\{i,j\} \in E^{+,m} : \{j,k\} \in E^{-,m}, \{i,k\} \in E^{+,m}\}$ and $T_k^-(G^m) = \{\{i,j\} \in E^{-,m} : \{j,k\} \in E^{+,m}, \{i,k\} \in E^{+,m}\}$. We will say that i,j,k are in a bad triplet on level m if $\{i,j\} \in E^{+,m}, \{j,k\} \in E^{+,m}, \{k,i\} \in E^{-,m}$. For notational convenience, we let $W_{ij}^{+,m} = \sum_{\ell=1}^{m} w_{ij}^{+,\ell}$, and $C_{ij}^m = \sum_{\ell=1}^{m} c_{ij}^{\ell}$.

THEOREM 6.1 *Given an input to M-level hierarchical clustering, let $(V, \{w_{ij}^{+,m}, w_{ij}^{-,m} : \{i,j\} \subseteq V\})$ for $m = 1,\ldots,M$ be inputs of weighted clustering derived according to (21). Given a set of budgets $\{c_{ij}^m : \{i,j\} \subseteq V\}$, and graphs $G^m = (V, E^{+,m}, E^{-,m})$, where $E^{+,m}, E^{-,m}$ is a partition of $\{\{i,j\} : \{i,j\} \subseteq V\}$ such that for every $m \in \{1,\ldots,M\}$:*

(i) $w_{ij}^{-,m} \le \alpha c_{ij}^m$ for all $\{i,j\} \in E^{+,m}$, and $W_{ij}^{+,m} \le \alpha C_{ij}^m$ for all $\{i,j\} \in E^{-,m}$,

(ii) $W_{ij}^{+,m} + W_{jk}^{+,m} + w_{ki}^{-,m} \le \alpha(C_{ij}^m + C_{jk}^m + c_{ki}^m)$ for every bad triplet $\{i,j\} \in E^{+,m}, \{j,k\} \in E^{+,m}, \{k,i\} \in E^{-,m}$.

Then Hierarchical-Cluster returns a solution that costs at most $\alpha\sum_{m=1}^{M}\sum_{\{i,j\}\subseteq V} c_{ij}^m$ if in CC-Pivot($G^m(C)$) we choose a pivot k that minimizes

$$\frac{\sum_{\{i,j\}\in T_k^+(G^m(C))} W_{ij}^{+,m} + \sum_{\{i,j\}\in T_k^-(G^m(C))} w_{ij}^{-,m}}{\sum_{\{i,j\}\in T_k^+(G^m(C))} C_{ij}^m + \sum_{\{i,j\}\in T_k^-(G^m(C))} c_{ij}^m}. \tag{22}$$

PROOF. In a call to CC-Pivot with graph $G^m(C)$, if we separate two vertices i,j, then they will be separated in the clusterings at levels $1,\ldots,m$. Hence we attribute to this call a cost of $\sum_{\ell=1}^{m} w_{ij}^{+,\ell} = W_{ij}^{+,m}$. If two vertices i,j are in the same cluster, then we only attribute the cost $w_{ij}^{-,m}$ of joining them in the m-th level (since we do not yet know whether they will be in the same or in different clusters in the lower levels). We can also attribute budgets to the call to CC-Pivot in the same way, attributing a budget of C_{ij}^m for a vertex pair i,j that gets separated, and a budget of c_{ij}^m for a vertex pair that gets clustered together.

We note that a piece of budget c_{ij}^m is attributed to exactly one call to CC-Pivot, and that the total cost of the solution generated by Hierarchical-Cluster is attributed.

It remains to show that the total cost attributed to a call to CC-Pivot is not more than α times the total budget attributed to it. This follows immediately from Theorem 3.1. □

We will use the following linear program to demonstrate a set of budgets c^m and graphs G^m for $m = 1,\ldots,M$ that satisfy the conditions in Theorem 6.1. We let $x_{ij}^{+,m} = 1$ denote that i,j are in the same cluster at level m, and $x_{ij}^{-,m} = 1$ that they are in different clusters. We note that the first two sets of constraints are the same as in (LP_{CC}), and that the third set of constraints ensures that the clustering at level $m-1$ is a refinement of the clustering at level m.

$$\begin{aligned}
& \min \sum_{m=1}^{M} \sum_{\{i,j\}\subseteq V} \left(w_{ij}^{+,m} x_{ij}^{-,m} + w_{ij}^{-,m} x_{ij}^{+,m} \right) \\
\text{s.t.} \quad & x_{ij}^{-,m} + x_{jk}^{-,m} + x_{ik}^{+,m} \ge 1 && \forall \text{ distinct } i,j,k, \forall m = 1,\ldots,M \\
(LP_{HC}) \quad & x_{ij}^{+,m} + x_{ij}^{-,m} = 1 && \forall \text{ distinct } i,j, \forall m = 1,\ldots,M \\
& x_{ij}^{-,m} \le x_{ij}^{-,m-1} && \forall \text{ distinct } i,j, \forall m = 2,\ldots,M \\
& x_{ij}^{+,m}, x_{ij}^{-,m} \ge 0 && \forall \text{ distinct } i,j, \forall m = 1,\ldots,M.
\end{aligned}$$

THEOREM 6.2 *There exists a deterministic $(M+2)$-approximation algorithm for M-level hierarchical clustering.*

PROOF. We form the graph $G^m = (V, E^{+,m}, E^{-,m})$ exactly as we formed $G = (V, E^+, E^-)$ in the proof of Theorem 3.4, by including $\{i,j\} \in E^{+,m}$ if $x_{ij}^{+,m} \geq \frac{1}{2}$ and including $\{i,j\} \in E^{-,m}$if $x_{ij}^{-,m} \geq \frac{1}{2}$ (breaking ties arbitrarily).

We let $c_{ij}^m = w_{ij}^{+,m} x_{ij}^{-,m} + w_{ij}^{-,m} x_{ij}^{+,m}$. Then $\sum_{m=1}^{M} \sum_{\{i,j\} \subseteq V} c_{ij}^m$ is a lower bound on the cost of the optimal hierarchical clustering. Hence if we can prove that the conditions in Theorem 6.1 hold with $\alpha = M+2$, then our algorithm Hierarchical-Cluster, with pivots chosen as in Theorem 6.1, is an $(M+2)$-approximation algorithm.

We claim that the first condition holds with $\alpha = 2$: If $\{i,j\} \in E^{+,m}$, then $x_{ij}^{+,m} \geq \frac{1}{2}$, so $c_{ij}^m \geq \frac{1}{2} w_{ij}^{-,m}$. If $\{i,j\} \in E^{+,m}$, then $x_{ij}^{-,m} \geq \frac{1}{2}$, and by the third set of constraints this implies that also $x_{ij}^{-,\ell} \geq \frac{1}{2}$ for $\ell = 1, \ldots, m$. Hence $C_{ij}^m = \sum_{\ell=1}^{m} c_{ij}^\ell \geq \sum_{\ell=1}^{m} \frac{1}{2} w_{ij}^{+,\ell} = \frac{1}{2} W_{ij}^{+,m}$.

We now show that the second condition holds with $\alpha = m+2$, if considering a bad triplet in G^m. Since $m \leq M$, this implies that the second condition always holds with $\alpha = M+2$. Let $\{i,j\} \in E^{+,m}, \{j,k\} \in E^{+,m}, \{k,i\} \in E^{-,m}$ be a bad triplet. We note that we know from the proof of Theorem 3.1 that

$$w_{ij}^{+,m} + w_{jk}^{+,m} + w_{ki}^{-,m} \leq 3(c_{ij}^m + c_{jk}^m + c_{ki}^m). \tag{23}$$

If $m = 1$, then this is exactly the second condition with $\alpha = m+2$. Otherwise, let $m \geq 2$.

The idea behind the rest of the proof is the following. Note that $W_{ij}^m + W_{jk}^m + w_{ki}^m$ may be as large as $2m+1$. On the other hand, using our previous results, we will be able to show that if $W_{ij}^m + W_{jk}^m + w_{ki}^m > 0$, then $c_{ij}^m + c_{jk}^m + c_{ki}^m$ is at least 1. Hence the budgets at level m alone pay at least a $\frac{1}{2m+1}$ fraction of the cost. In fact, it may be the case that the lower level budgets $c_{ij}^\ell + c_{jk}^\ell$ are 0. However, we will show that for each level $\ell \leq m$ if $w_{ij}^\ell + w_{jk}^\ell = 2$, then $c_{ij}^\ell + c_{jk}^\ell$ is at least $\frac{1}{4}$.

To be precise, we will now show that for every $\ell \in 1, \ldots, m-1$:

$$w_{ij}^{+,\ell} + w_{jk}^{+,\ell} \leq (c_{ij}^m + c_{jk}^m + c_{ki}^m) + 4(c_{ij}^\ell + c_{jk}^\ell). \tag{24}$$

Adding (23) plus the inequalities (24) for $\ell = 1, \ldots, m-1$, then gives

$$\begin{aligned}
W_{ij}^{+,m} + W_{jk}^{+,m} + w_{ki}^{-,m} &= \sum_{\ell=1}^{m} w_{ij}^{+,\ell} + \sum_{\ell=1}^{m} w_{jk}^{+,\ell} + w_{ki}^{-,m} \\
&\leq (m+2)(c_{ij}^m + c_{jk}^m + c_{ki}^m) + 4\left(\sum_{\ell=1}^{m-1} c_{ij}^\ell + \sum_{\ell=1}^{m-1} c_{jk}^\ell\right) \\
&\leq (m+2)\left(\sum_{\ell=1}^{m} c_{ij}^\ell + \sum_{\ell=1}^{m} c_{jk}^\ell + c_{ki}^m\right) \\
&= (m+2)(C_{ij}^m + C_{jk}^m + c_{ki}^m),
\end{aligned}$$

where the last inequality follows since $m \geq 2$.

It remains to prove that inequality (24) holds. Note that the left hand side is either 0, 1 or 2. If it is 0, then clearly the inequality holds. If the left hand side is 1, suppose without loss of generality that $w_{ij}^{+,\ell} = 1, w_{jk}^{+,\ell} = 0$. Note that from the definition $w_{ij}^{+,\ell} = 1$ means that $D_{ij} < \ell$. Then also $D_{ij} < m$, hence $w_{ij}^{+,m} = 1$. In the proof of Theorem 3.4 we showed that either $c_{ij}^m + c_{jk}^m + c_{ki}^m \geq w_{ij}^{+,m} + w_{jk}^{+,m} + w_{ki}^{-,m}$, or that $c_{ij}^m + c_{jk}^m + c_{ki}^m \geq 1$. Hence in this case $c_{ij}^m + c_{jk}^m + c_{ki}^m$ is always at least 1, which proves that (24) holds if the left hand side is 1. Finally, we consider the case when the left hand side is 2. Then

$w_{ij}^{+,\ell} = w_{jk}^{+,\ell}$, therefore

$$\begin{aligned} c_{ij}^{\ell} + c_{jk}^{\ell} &= x_{ij}^{-,\ell} + x_{jk}^{-,\ell} \\ &\geq 1 - x_{ki}^{+,\ell} && \text{(from the first set of constraints in } (LP_{HC})) \\ &= x_{ki}^{-,\ell} && \text{(from the second set of constraints in } (LP_{HC})) \\ &\geq x_{ki}^{-,m} && \text{(from the third set of constraints in } (LP_{HC})) \\ &\geq \frac{1}{2} && \text{(since } \{k,i\} \in E^{-,m}) \\ &= \frac{1}{4}(w_{ij}^{+,\ell} + w_{jk}^{+,\ell}). \end{aligned}$$

□

We now turn to constrained hierarchical clustering: in addition to matrix D, we are given matrices L, U such that $L_{ij} \leq D_{ij} \leq U_{ij}$ for every $i, j \in V$. Any feasible hierarchical clustering must have $L_{ij} \leq \lambda_{ij} \leq U_{ij}$, where λ_{ij} is again the number of levels in which i and j are in different clusters. We assume that the constraints are consistent, i.e. that for any i, j, $L_{ij} \leq U_{ij}$ and for any i, j, k $L_{ij} \leq \max\{U_{jk}, U_{ki}\}$, since it is easy to check that otherwise no feasible hierarchical clustering exists.

LEMMA 6.1 *There exists a deterministic $(M+2)$-approximation algorithm for constrained M-level hierarchical clustering.*

PROOF. Let $P^{+,m} = \{\{i,j\} : m > U_{ij}\}, P^{-,m} = \{\{i,j\} : m \leq L_{ij}\}$ for $m = 1, \ldots, M$. We add the following set of constraints to (LP_{CC})

$$x_{ij}^{+,m} = 1 \text{ for all } \{i,j\} \in P^{+,m}$$
$$x_{ij}^{-,m} = 1 \text{ for all } \{i,j\} \in P^{-,m}.$$

Note that the optimal value of (LP_{HC}) still gives a lower bound on the cost of the optimal hierarchical clustering. As in the proof of Theorem 4.2, we can then partition the edges into $E^{+,m}, E^{-,m}$ so that the conditions of Lemma 4.2 are satisfied for $m = 1, \ldots, M$. Hence by Lemma 4.2 our algorithm does not separate i, j at level $m \in \{U_{ij}+1, \ldots, M\}$ since $\{i,j\} \in P^{+,m}$ for $m > U_{ij}$. On the other hand, if i, j are not separated already at some level $m > L_{ij}$, then the fact that $\{i,j\} \in P^{-,L_{ij}}$ ensures that i and j will be separated at level L_{ij} (and thus for all levels $m \leq L_{ij}$). Hence the algorithm returns a feasible solution. □

Acknowledgments. We would like to thank the anonymous referees for helpful comments and suggestions. In particular, we thank one of the referees for suggesting a more general definition of partial consensus clustering than the one we proposed in an earlier draft of this paper.

References

[1] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, *PRIMES is in P*, Ann. of Math. (2) **160** (2004), no. 2, 781–793.

[2] Nir Ailon, *Aggregation of partial rankings, p-ratings and top-m lists*, SODA '07: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, 2007, pp. 415–424.

[3] Nir Ailon and Moses Charikar, *Fitting tree metrics: Hierarchical clustering and phylogeny*, FOCS '05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, 2005, pp. 73–82.

[4] Nir Ailon, Moses Charikar, and Alantha Newman, *Aggregating inconsistent information: ranking and clustering*, STOC '05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing, 2005, pp. 684–693.

[5] Noga Alon, *Ranking tournaments*, SIAM J. Discret. Math. **20** (2006), no. 1, 137–142.

[6] Noga Alon and Joel H. Spencer, *The probabilistic method*, second ed., Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley-Interscience [John Wiley & Sons], New York, 2000.

[7] Sanjeev Arora, Alan M. Frieze, and Haim Kaplan, *A new rounding procedure for the assignment problem with applications to dense graph arrangement problems*, FOCS '96: Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science, 1996, pp. 21–30.

[8] Kenneth J. Arrow, *Social choice and individual values*, Yale University Press, 1951.

[9] A. Asuncion and D.J. Newman, *UCI machine learning repository*, 2007.

[10] Nikhil Bansal, Avrim Blum, and Shuchi Chawla, *Correlation clustering*, Machine Learning **56** (2004), no. 1-3, 89–113.

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

[12] Pierre Charbit, Stéphan Thomassé, and Anders Yeo, *The minimum feedback arc set problem is NP-hard for tournaments*, Combin. Probab. Comput. **16** (2007), no. 1, 1–4.

[13] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth, *Clustering with qualitative information*, FOCS '03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, 2003, pp. 524–533.

[14] Vincent Conitzer, *Computing Slater rankings using similarities among candidates*, AAAI'06: Proceedings of the 21st National Conference on Artificial Intelligence, 2006.

[15] Don Coppersmith, Lisa Fleischer, and Atri Rudra, *Ordering by weighted number of wins gives a good ranking for weighted tournaments*, SODA '06: Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, 2006, pp. 776–782.

[16] Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar, *Rank aggregation methods for the web*, WWW '01: Proceedings of the 10th international conference on World Wide Web, 2001, pp. 613–622.

[17] Paul Erdős and Joel Spencer, *Probabilistic methods in combinatorics*, Academic Press [A subsidiary of Harcourt Brace Jovanovich, Publishers], New York-London, 1974, Probability and Mathematical Statistics, Vol. 17.

[18] Guy Even, Joseph Naor, Baruch Schieber, and Madhu Sudan, *Approximating minimum feedback sets and multicuts in directed graphs*, Algorithmica **20** (1998), no. 2, 151–174.

[19] Ronald Fagin, Ravi Kumar, Mohammad Mahdian, D. Sivakumar, and Erik Vee, *Comparing partial rankings*, SIAM J. Discrete Math. **20** (2006), no. 3, 628–648.

[20] Vladimir Filkov and Steven Skiena, *Integrating microarray data by consensus clustering*, ICTAI '03: Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence, 2003, pp. 418–425.

[21] Alan M. Frieze and Ravi Kannan, *Quick approximation to matrices and applications*, Combinatorica **19** (1999), no. 2, 175–220.

[22] Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas, *Clustering aggregation*, ACM Trans. Knowl. Discov. Data **1** (2007), no. 1, 4.

[23] Andrey Goder and Vladimir Filkov, *Consensus clustering algorithms: Comparison and refinement*, ALENEX'08: Proceedings of the Workshop on Algorithm Engineering and Experiments, 2008.

[24] Boulos Harb, Sampath Kannan, and Andrew McGregor, *Approximating the best-fit tree under L_p norms*, Approximation, randomization and combinatorial optimization, Lecture Notes in Comput. Sci., vol. 3624, Springer, Berlin, 2005, pp. 123–133.

[25] Rajneesh Hegde and Kamal Jain, *personal communication.*

[26] Richard M. Karp, *Reducibility among combinatorial problems*, Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972), Plenum, New York, 1972, pp. 85–103.

[27] J.G. Kemeny, *Mathematics without numbers*, Daedalus **88** (1959), 575–591.

[28] Claire Kenyon-Mathieu and Warren Schudy, *How to rank with few errors*, STOC '07: Proceedings of the 39th Annual ACM Symposium on Theory of Computing, 2007, pp. 95–103.

[29] Mirko Křivánek and Jaroslav Morávek, *NP-hard problems in hierarchical-tree clustering*, Acta Inform. **23** (1986), no. 3, 311–323.

[30] Dharmendra Modha and Scott Spangler, *Feature weighting in k-means clustering*, Machine Learning **52** (2003), no. 3, 217–237.

[31] Dharmendra S. Modha and W. Scott Spangler, *Clustering hypertext with applications to web searching*, HYPERTEXT '00: Proceedings of the 11th ACM on Hypertext and Hypermedia, 2000, pp. 143–152.

[32] Paul D. Seymour, *Packing directed circuits fractionally*, Combinatorica **15** (1995), no. 2, 281–288.

[33] Yoshiko Wakabayashi, *The complexity of computing medians of relations*, Resenhas **3** (1998), no. 3, 323–349.

[34] Anke van Zuylen, Rajneesh Hegde, Kamal Jain, and David P. Williamson, *Deterministic pivoting algorithms for constrained ranking and clustering problems*, SODA '07: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, 2007, pp. 405–414.

[35] Anke van Zuylen and David P. Williamson, *Deterministic algorithms for rank aggregation and other ranking and clustering problems*, WAOA'07: 5th International Workshop on Approximation and Online Algorithms, Lecture Notes in Computer Science, vol. 4927, Springer, 2008, pp. 260–273.