Approximation Schemes for Dense Variants of Feedback Arc Set, Correlation Clustering, and Other Fragile Min Constraint Satisfaction Problems

Thesis proposal

Warren Schudy
ws@cs.brown.edu
Brown University Computer Science

Committee:
Amy Greenwald
Marek Karpinski (University of Bonn)
Phil Klein
Claire Mathieu (Advisor)

Abstract

We study approximation schemes for various minimum (or maximum) constraint satisfaction problems (CSPs). We give the first polynomial time approximation schemes (PTASs) for the minimization problems of feedback arc set on tournament graphs, fully dense betweenness, and correlation clustering with noisy input. We give a PTAS for dense MAX CSPs which is simpler than previous PTASs for these problems. We give a more time-efficient PTAS for correlation clustering with a fixed number of clusters. We generalize and unify the above results using a new concept of *fragile* constraints.

1 Introduction

We are surrounded by NP-complete problems. Delivering packages efficiently requires solution to variants of the traveling salesman problem (TSP). Problems in the design and verification of microprocessors are often reduced to the NP-complete problem of satisfiability of conjunctive normal form propositional formulae, also known as SAT. For hundreds of additional NP-complete problems see Garey and Johnson [1979].

It is well known that if there is an efficient (i.e. polynomial time) algorithm for *any* NP-complete problem then there is an efficient algorithm for *all*. Unfortunately despite decades of effort no polynomial time algorithm has been found. One way around this difficulty is *approximation algorithms*, which do not generally find the optimum solution but provably find a solution that is not much worse. An algorithm for a minimization problem is an α-approximation if its output has cost at most α times the cost of the optimum solution. A polynomial-time approx scheme (PTAS) is a family of $1+\epsilon$-approximation algorithms for all $\epsilon > 0$ with runtime polynomial in the input size m. The runtime of a PTAS need not be polynomial in $1/\epsilon$; for example $m^{O(1/\epsilon^2)}$ and $m + 2^{O(1/\epsilon^2)}$ count as polynomial.

In this thesis we study approximation algorithms for two problems, namely *feedback arc set tournament* and *correlation clustering*, plus generalizations.

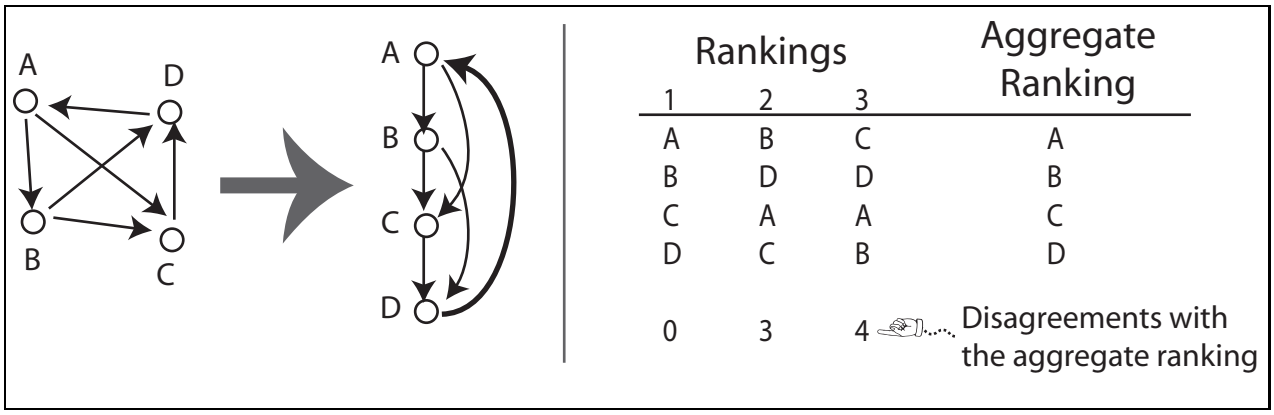

Figure 1: Feedback arc set tournament (left) and Kemeny rank aggregation (right) problems.

Feedback arc set tournament

Suppose you ran a chess tournament, everybody played everybody (a.k.a. round robin) and you wanted to use the results to rank everybody. Unless you were really lucky, the results would not be acyclic, so you could not just sort the players by who beat whom. A natural objective is to find a ranking that minimizes the number of upsets, where an upset is a pair of players where the player ranked lower in the ranking beat the player ranked higher. Minimizing the number of upsets is called *feedback arc set problem* on tournaments (FAST). The complementary problem of *maximizing* the number of pairs that are *not upsets* is called the *maximum acyclic subgraph problem* on tournaments. These problems are NP-hard [Ailon et al., 2008, Alon, 2006, Charbit et al., 2007] (see also [Conitzer, 2006]). We give the first PTAS [Kenyon-Mathieu and Schudy, 2007, Mathieu and Schudy, 2009] for FAST.

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

A second application of (weighted) FAS tournaments is *rank aggregation*. Frequently, one has access to several rankings of objects of some sort, such as search engine outputs [Dwork et al., 2001], and desires to aggregate the input rankings into a single output ranking that is similar to all of the input rankings: it should have minimum average distance from the input rankings, for some notion of distance. This ancient problem was already studied in the context of voting by [Borda, 1781] and [Condorcet, 1785] in the 18^{th} century, and has aroused renewed interest recently [Dwork et al., 2001, Conitzer et al., 2006]. A natural notion of distance is the number of pairs of vertices that are in different orders: this defines the *Kemeny rank aggregation* problem [Kemeny, 1959, Kemeny and Snell, 1962] illustrated in Figure 1. This choice yields a maximum likelihood estimator for a certain naïve Bayes model [Young, 1995]. This problem is NP-hard [Bartholdi et al., 1989], even with only 4 voters [Dwork et al., 2001]. There is a randomized 4/3 approximation algorithm [Ailon et al., 2008]. We improve on these results by giving a polynomial time approximation scheme (PTAS) via reduction to (a weighted version of) FAST.

The constants in our PTASs for FAST and Kemeny rank aggregation make them impractical but the role of local search in our analysis helps shed light on the excellent performance of local search for Kemeny rank aggregation observed in practice [Schalekamp and van Zuylen, 2009].

As an additional illustration of the power of our techniques we will show [Karpinski and Schudy, 2009b] that the *optimal* feedback arc set of a tournament graph can be computed in time $2^{O(\sqrt{OPT})}$, a moderate improvement on the previously best known runtime of $2^{\tilde{O}(\sqrt{OPT})}$.

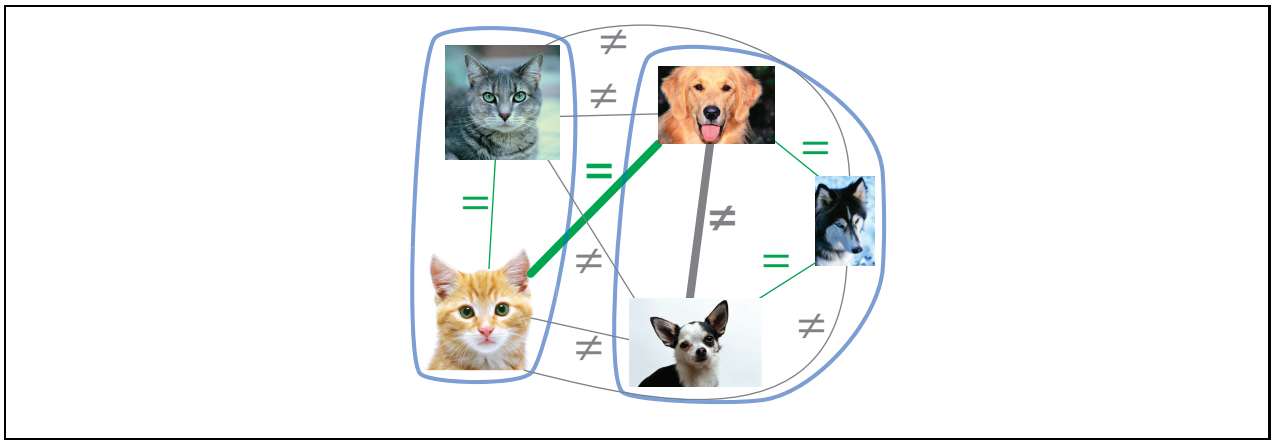

Figure 2: An example correlation clustering problem and optimal clustering. The optimal clustering (dogs and cats) has two disagreements, shown in bold.

Correlation Clustering

Clustering is an important tool for analyzing large data sets. Correlation clustering [Ben-Dor et al., 1999, Bansal et al., 2004] is a type of clustering that uses a very basic form of input data: indications that certain pairs of vertices (pairs of data items) belong in the same cluster, and certain other pairs belong in different clusters [Bansal et al., 2004]. Unfortunately the information is not necessarily consistent, possibly claiming for example that "cat" is similar to "dog" and "dog" is similar to "bog" but "cat" is not similar to "bog". We assume that we have information about every pair of objects. The goal is to find a clustering, that is, a partition of the vertices, that agrees with as many pieces of information as possible (maximization) or disagrees with as few as possible (minimization). Correlation clustering is illustrated in Figure 2. This has applications in data mining and natural language processing [Cohen and Richman, 2002, Finkel and Manning, 2008]. Correlation clustering has no PTAS unless P=NP [Charikar et al., 2005], so to get good approximation algorithms we must restrict the problem somehow to separate real-world instances from those produced by the reductions of Charikar et al. [2005].

One possible restriction is limiting the number of clusters to some small number d. With this assumption Giotis and Guruswami found a PTAS [Giotis and Guruswami, 2006], but their runtime of $n^{O(9^d/\epsilon^2)}$ leaves something to be desired. We state a PTAS with runtime $n^2 2^{O((\log d)d^6/\epsilon^2)}$, which improves on Giotis and Guruswami [2006] in two ways. Firstly it is simply exponential in d rather than doubly exponential and secondly the power of n is a constant 2, independent of d and ϵ. Our result extends to the (previously known) hierarchical generalization of correlation clustering, which is the first PTAS known for that problem.

A different assumption is that there may be lots of clusters but the input is generated by chance rather than an adversary. We study the following semirandom noisy model which is compromise between random and adversarial: start from an arbitrary partition of the vertices into clusters. Then, for each pair of vertices, the similarity information is corrupted (noisy) independently with probability p. Finally, an adversary generates the input by choosing similarity/dissimilarity information arbitrarily for each corrupted pair of vertices. In this model we give a $(1 + O(n^{-1/6}))$-approximation [Mathieu and Schudy, 2010], which is even better than a PTAS. We can also perfectly recover the planted clusters that are relatively large.

Our noisy input algorithm is relatively implementable, with the time-consuming part being the solution of a semi-definite program. Micha Elsner and I tested a variety of algorithms on correlation clustering data from natural language processing [Elsner and Schudy, 2009], including a variant of the algorithm discussed in the previous paragraph. The semi-definite programming relaxation used in both previous work [Charikar et al., 2005] and ours [Mathieu and Schudy, 2010] produced much better lower bounds than the simple bounds previously used. On the other hand the clusterings found by our algorithm [Mathieu and Schudy, 2010]

Choose $x_1, x_2, x_3, x_4 \in \{0, 1, 2\}$ satisfying as many of the following as possible:

$$2x_1 + 4x_2 = 3 \pmod 7$$
$$|x_1 - x_3| \in \{1, 3\}$$
$$x_1 \text{ is the } (x_4)^{th} \text{ prime}$$
$$\cos(x_2 x_3) \geq (1/2)$$
$$2x_3 - 42x_4 = 20 \pmod{100}$$

Figure 3: An example MIN-2CSP. The optimal assignment $x_1 = 2$, $x_2 = 1$, $x_3 = 0$, $x_4 = 1$ satisfies all constraints except the last and hence has cost 1.

were no better than clusterings found by much faster greedy and local search techniques. We conclude that the semi-definite program is useful for lower-bounds but does not appear to be competitive for finding clusterings.

Generalizations

The reader may be wondering what our results have in common and whether there are any other problems amenable to our techniques. We now consider generalizations, first of the correlation clustering with a fixed number of clusters result and second the feedbasck arc set tournament result.

A k-CSP consists of a constant arity k ($k = 2$ for d-correlation clustering), a set of variables V (one variable per object being clustered) which take values from a constant-sized domain D (the clusters), and a set of constraints on the variables. Each constraint depends on a set of k of the variables and is *satisfied* by some of the possible configurations of those variables. We assume that every set of k variables is constrained by a number of constraints bounded above by a constant. One can express d-correlation clustering as a 2-CSP with a constraint for every pair of objects which is satisfied if the two objects are placed in the same cluster whenever the input asks them to be. Satisfying all the constraints simultaneously is generally impossible so the goal of a MIN-k-CSP (resp. MAX-k-CSP) is to minimize (maximize) the number of unsatisfied (satisfied) constraints.

The k-CSP class of problems is quite broad, including for example the famous MAX CUT problem of finding a cut of an undirected graph with the maximum number of edges crossing it and its corresponding minimization version MIN-UNCUT. Unfortunately MAX CUT has no PTAS unless P = NP, so we cannot generalize our results to all kCSPs. So what makes correlation clustering with a fixed number of clusters different? We identify two properties: *density* and *fragility*.

We discuss three types of density. We say that an instance is *average-dense* if there are $\Omega(n^k)$ constraints; i.e. within a constant factor of the maximum possible number of constraints. We say that an instance is *everywhere-dense* if each variable participates in $\Omega(n^{k-1})$ constraints, which is again within a constant factor of the maximum possible number. We say that an instance is *fully dense* if there is a constraint for every set of k variables. Correlation clustering is fully dense. A random assignment satisfies a constant fraction of the constraints (excluding useless constraints that are never satisfied) so average-dense MAX-kCSPs have optimum value $\Theta(n^k)$. So-called *additive error algorithms* find an assignment with cost at most ϵn^k *plus* the optimum cost, which implies a PTAS for all MAX-kCSPs [Frieze and Kannan, 1999, Alon et al., 2002]. One of our contributions is an additive error algorithm which is much simpler than previous ones.

Additive error algorithms are insufficient for MIN-k-CSPs since optimum values of minimization problems can be very small, even 0. Indeed there are many MIN-k-CSPs without PTASs (unless P=NP) even on everywhere-dense instances, for example MIN-3-UNCUT. There are also fully dense (albiet unatural) MIN-kCSPs with no PTAS. Our PTAS therefore cannot rely on density alone but must also rely on a specific property of correlation clustering constraints.

The key property of d-correlation clustering is that *most of the vertices in a low cost clustering cannot be moved to different clusters without increasing the cost by* $\Omega(n)$. We now give a wide class of problems that also satisfy this property. A constraint is *fragile* if modifying any variable in a satisfied constraint makes the constraint unsatisfied. A CSP is *fragile* if all of its constraints are. Several interesting CSPs feature fragile constraints, including MIN-2UNCUT, unique games, and nearest codeword problem. We introduce the class of fragile MIN-CSPs and show that they all satisfy this key property and hence have a PTAS. Our runtime is essentially the best possible up to constants. The d-correlation clustering problem is not fragile but does satisfy this key property, which as noted before allows us to give a PTAS.

We generalize FAS to ranking CSP [Ailon and Alon, 2007], which is similar to an ordinary CSP except that the goal is to choose an ordering of the objects rather than an assignment to variables. In general a ranking CSP consists of a constant arity k (2 for FAS), a set of vertices V and a set of constraints on the vertices. Each constraint depends on the ordering of a set k of vertices and is satisfied by some of the possible orderings of those vertices. In FAS each directed edge (game) corresponds to a constraint which is satisfied whenever the winner of the game is ordered before the loser. A second MIN-RANK-CSP is betweeness: rank objects given information of the form v is between u and w, i.e. either $u < v < w$ or $w < v < u$. We have discovered (but not yet written up) [Karpinski and Schudy, 2009b] a PTAS for fully dense betweenness. We are working on determining an appropriate generalization of fragility to rank CSPs encompassing our FAST and betweeness results.

Summary of techniques

Techniques include:

1. **Exhaustive sampling (Additive error, Fragile MIN-CSP)**: We make extensive use of this technique, introduced in Arora et al. [1995], for effectively taking a random sample of optimal assignments even though the optimal assignment is unknown.

2. **Local search (Fragile MIN-RANK-CSP)**: Local search techniques are extensively used to solve NP-complete optimization problems in practice. Local search is difficult to analyze and therefore is rarely used in approximation algorithms. For example a standard approximation algorithms textbook Vazirani [2001] mentions local search only in chapter notes and exercises. Local search is the best known technique for only one problem (to our knowledge): the k-median problem Arya et al. [2004]. Local search plays a key role in several of our PTASs.

3. **Martingales and Azuma-Hoeffding inequality (Additive Error)**: We use martingales to, roughly speaking, take a random sample of a moving target.

4. **Semi-definite programming duality (Noisy correlation clustering)**: Semi-definite programming is an extension of linear programming that has attracted increasing interest in the approximation algorithms community since the seminal work of Goemans and Williamson [1995]. We analyze a semi-definite program by presenting matching primal and dual feasible solutions and using complementary slackness conditions.

5. **Spectra of random matrices (Noisy correlation clustering)**: We extend the techniques in Füredi and Komlós [1981] to analyze the eigenvalue spectrum of a certain class of random symmetric matrices.

Summary of major tasks remaining

I.e. some of the things I plan to work on between proposal and defense.

1. Finish extending our FAST PTAS to other ranking CSPs [Karpinski and Schudy, 2009b]. Also fixed parameter tractability result.

2. Some loose ends (i.e. prepare journal version) of additive error result [Mathieu and Schudy, 2008b].

Open Problems

One open problem related to this thesis stands out both in potential for applicability and in apparent difficulty: is there a PTAS for partial rank aggregation? *Partial rank aggregation* is a generalization of Kemeny rank aggregation where rankings may give several candidates the same rank, i.e. ties. Partial rank aggregation is especially interesting since voters frequently do not rank the entire universe; for example search engines give only the top results. A 1.5-approximation of partial rank aggregation exists [Ailon, 2007] which reduces to a form of weighted feedback arc set where the weights satisfy the triangle inequality (i.e. directed metric). The additive error algorithms we use as subroutines also generalize. Unlike these previous results it is not evident how to generalize our PTASs to partial rank aggregation.

Other Results

The following results of mine are outside the scope of the proposed thesis but are mentioned here for reference. See the papers [Greenwald et al., 2008, Schudy, 2008, Mathieu et al., 2009, Schudy, 2009] for more information.

An online decision problem (ODP) consists of a series of rounds, during each of which an agent chooses one of n pure actions and receives a reward corresponding to its choice. For example the agent may be playing a matrix game such as rock-paper-scissors or chicken repeatedly. The agent's objective is to maximize its cumulative rewards. It can work towards this goal by abiding by an online learning algorithm, which prescribes a possibly mixed action (i.e., a probability distribution over the set of pure actions) to play each round, based on past actions and their corresponding rewards. No-internal-regret (NIR) learners converge to the set of correlated equilibria in repeated matrix games. However, standard NIR learning algorithms involve a fixed point calculation during each round of learning, which is time-consuming when the number of pure actions available to the player is large. Amy Greenwald, Zheng Li and I [Greenwald et al., 2008] give a natural NIR learner that only requires a matrix-vector multiplication, improving the runtime.

I show [Schudy, 2008] how to use $O(\log^2 n)$ reachability queries suffice to compute strongly connected components and topological sort of directed graphs. This yields faster parallel algorithms for these problems, reducing the number of processors needed to achieve a speedup of s from $O(s^3)$ to $O(s^2)$. Parallel computation of strongly connected components has applications in scientific computing [McLendon et al., 2005, 2001, Plimpton et al., 2000]. Our algorithm is simple with reasonable hidden constants and hence may be practical.

Ocan Sankur, Claire Mathieu and I have some preliminary results on online correlation clustering. In this problem the objects to be clustered arrive one by one. The algorithm may add new objects to existing clusters and possibly merge clusters, but is not allowed to split existing clusters. The goal is to maintain a clustering that is competitive with the offline optimum clustering.

I have some preliminary results [Schudy, 2009] on theoretical results related to restart strategies for branch-and-bound tree search. Details will be added here once they stabilize.

Organization

In section 2 we present known results on additive error for CSPs and our contributions [Mathieu and Schudy, 2008b] to this area. With this foundation in hand we move on to fragile MIN-CSPs [Karpinski and Schudy, 2009a] in section 3 and correlation clustering with a constant number of clusters [Karpinski and Schudy, 2009a] in section 5. We then describe our results for fragile MIN-RANK-CSPs [Kenyon-Mathieu and Schudy, 2007, Mathieu and Schudy, 2009, Karpinski and Schudy, 2009b] such as feedback arc set tournament in section 4. Finally we give our results for correlation clustering in the noisy model [Mathieu and Schudy, 2010] in section 6.

For brevity we omit most of the proofs. See the corresponding papers [Kenyon-Mathieu and Schudy, 2007, Mathieu and Schudy, 2009, 2010, Karpinski and Schudy, 2009a,b] for proofs.

Papers

Papers are available from `http://www.cs.brown.edu/~ws/` or by clicking on author names below.

Feedback arc set and generalizations:

- C. Kenyon-Mathieu and W. Schudy. How to rank with few errors: a PTAS for weighted feedback arc set on tournaments. In *Procs. 39^{th} ACM STOC*, pages 95–103, 2007.
- C. Mathieu and W. Schudy. How to rank with fewer errors – a PTAS for feedback arc set in tournaments. In Submission, 2009.
- M. Karpinski and W. Schudy. Approximation of the Betweeness-Tournament Problem and Some Ranking Generalizations. Soon to be in preparation.

Correlation clustering and generalizations:

- M. Karpinski and W. Schudy. Linear time approximation schemes for the Gale-Berlekamp game and related minimization problems. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 313–322, 2009.
- M. Elsner and W. Schudy Bounding and Comparing Methods for Correlation Clustering Beyond ILP. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 19–27. Association for Computational Linguistics, June 2009.
- C. Mathieu and W. Schudy. Correlation clustering with noisy input. In submission, 2009.

Additive error:

- C. Mathieu and W. Schudy. Yet Another Algorithm for Dense Max Cut: Go Greedy. In *Proc. 19th ACM-SIAM SODA*, pages 176–182, 2008.

Results not in thesis:

- A. Greenwald, Z. Li, and W. Schudy. More efficient internal-regret-minimizing algorithms. In *21st Annual Conference on Learning Theory - COLT 2008*, pages 239–250, 2008.
- W. Schudy. Finding strongly connected components in parallel using $O(log^2 n)$ reachability queries. In *SPAA '08: Proceedings of the twentieth annual symposium on Parallelism in algorithms and architectures*, pages 146–151, 2008.
- C. Mathieu, O. Sankur, and W. Schudy. Online correlation clustering. In preparation, 2009.
- W. Schudy. Optimal restart strategies for tree search. In preparation, 2009.

2 Additive error

Over a decade ago two groups [**?**Fernandez de la Vega, 1996] independently discovered polynomial-time approximation algorithms for MAX-CUT achieving additive error of ϵn^2, which implies a PTAS for average-dense MAX-CUT instances. The fastest algorithms [Alon et al., 2002, Mathieu and Schudy, 2008b] have constant runtime $2^{O(1/\epsilon^2)}$ for approximating the value of any MAX-kCSP over a binary domain D. This can be generalized to an arbitrary domain D. To see this, note that we can code D in binary and correspondingly enlarge the arity of the constraints to $k\lceil \log |D| \rceil$. A random sample of $\tilde{O}(1/\epsilon^4)$ variables suffices to achieve an additive approximation [Alon et al., 2002, Rudelson and Vershynin, 2007]. These results extend

Take a sample S of $t_0 = 1/\epsilon^2$ variables chosen uniformly at random without replacement.
for each of the d^{t_0} possible assignments of variables S **do**
  for each variable v of $V \setminus S$ in random order, **do**
    Assign v to the value that maximizes the number of resulting satisfied constraints.
  end for
end for
Output the best assignment found.

Figure 4: Simplest-to-analyze greedy algorithm

Take a sample S of $t_1 = O(1/\epsilon^4)$ variables chosen uniformly at random.
Find a near-optimal assignment to the problem induced by the input on S, using Algorithm 4.
for each variable v of $V \setminus S$ in random order **do**
  Greedily assign a value to v, maximizing the number of resulting crossing edges.
end for

Figure 5: Fastest greedy algorithm

to MAX-BISECTION [Fernandez de la Vega et al., 2006]. Ailon [2007] show that if the general version of a CSP is NP-hard to solve exactly then it is NP-hard to approximation to within an additive $n^{r-\epsilon}$ for all $\epsilon > 0$.

These additive error algorithms are crucial subroutines in most of our later PTASs. We have also contributed our own novel additive error algorithms, which are substantially simpler than previous algorithms. Our new techniques also allow for a slight improvement of the sample complexity of MAX-k-CSPs. Previously the sample complexity of MAX-k-CSPs was known to be $O(1/\epsilon^4)$ for $k = 2$ [Rudelson and Vershynin, 2007] and $\tilde{O}(1/\epsilon^4)$ for $k \geq 3$ [Alon et al., 2002]; we show that the former bound holds for $k \geq 3$ as well.

We describe our algorithms in the context of MaxCut. The greedy algorithm for MaxCut considers vertices one by one in arbitrary order and places each of them on the left or right side of the cut, depending on the number of neighbors that are already placed on each side (breaking ties arbitrarily). It is well known that the greedy algorithm is a 2-approximation, and that this bound is tight if an adversary controls the input graph and the order in which the vertices are considered.

Here, we take advantage of the power of randomness by considering vertices *in random order*. We define three variants of the greedy algorithm for MaxCut. The first algorithm is closest to the "exhaustive search" techniques of previous papers, hence easiest to analyze.

Theorem 1. *For any $\epsilon > 0$, Algorithm 4 has running time $O(n^2)2^{O(1/\epsilon^2)}$ and the expected value of the output is at least $OPT - O(\epsilon n^2)$.*

Thus, for average dense instances, Algorithm 4 is a polynomial-time approximation scheme.

The second algorithm is basically the greedy algorithm with random order, except that we start with a good "hint" of a near-optimal solution, in the form of a near optimal solution in a constant size sample. Note that the running time separates into one term depending on ϵ but not on n (for solving the problem in the sample) and the other term depending on n but not on ϵ (for running the greedy algorithm on the rest of the graph.) Thus it is the fastest of our variants.

Theorem 2. *For any $\epsilon > 0$, Algorithm 5 has running time $n^2 + 2^{O(1/\epsilon^2)}$ and the expected value of the output is at least $OPT - O(\epsilon n^2)$.*

Note that the use of Algorithm 4 in Algorithm 5 rather than an arbitrary approximation algorithm is crucial for our proof to work with a sample of size $O(1/\epsilon^4)$.

The third algorithm, which we have analyzed for max cut only, eliminates the preliminary random sample phase altogether; it is just the naïve greedy algorithm with random order, but it is repeated enough times to guarantee near-optimality. It is slowest but simplest in design and does not even really require prior

Repeat $2^{2^{\tilde{O}(poly(\epsilon))}}$ times
for each vertex v of V in random order **do**
  Place v on the side that maximizes the number of resulting crossing edges.
end for
Output the best cut found

Figure 6: Most naïve greedy algorithm, for MAX CUT only

knowledge of ϵ: one could just run it until satisfied with the quality of the current cut, with the reassuring knowledge that it will converge to a near-optimal value in polynomial time.

Theorem 3. *For any $\epsilon > 0$, the expected value of the output of Algorithm 6 is at least $OPT - \epsilon n^2$.*

Interestingly one of the best greedy algorithms in our correlation clustering experiments was greedy with random order, i.e. one iteration of Algorithm 6. Our results do not apply in this setting because the number of clusters was not limited to a constant.

3 Fragile MIN-k-CSPs

MIN-k-CSPs can be much harder to approximate than MAX-k-CSPs since the error must be charged against a potentially much smaller optimum value. For this reasons many MIN-k-CSPs provably have no PTAS even for everywhere-dense or fully dense instances. One such problem is MIN-3-UNCUT, which is the problem of partitioning the vertices of an undirected graph into 3 pieces minimizing the number of uncut edges. This hardness follows from a simple reduction from sparse MIN-2-UNCUT.

Arora, Karger and Karpinski [**?**] introduced the first PTASs for dense *minimum* constraint satisfaction problems. They give PTASs with runtime $n^{O(1/\epsilon^2)}$ [**?**] for min bisection and multiway cut (MIN-d-CUT). Bazgan, Fernandez de la Vega and Karpinski [Bazgan et al., 2003] designed PTASs for MIN-SAT and the nearest codeword problem with runtime $n^{O(1/\epsilon^2)}$. Giotis and Guruswami [Giotis and Guruswami, 2006] give a PTAS for correlation clustering with d clusters with runtime $n^{O(9^d/\epsilon^2)}$. We give linear-time approximation schemes for all of the problems mentioned in this paragraph except for the MIN-BISECTION problem.

Theorem 4. *For every $\epsilon > 0$ there is a randomized $1+\epsilon$-approximation algorithm for everywhere-dense fragile MIN-kCSPs with runtime $O(n^k) + 2^{O(1/\epsilon^2)}$.*

3.1 Intuition

We use the Gale-Berlekamp Switching Game (GB Game) as a concrete context to describe our techniques. The GB Game was introduced independently by Elwyn Berlekamp [Carlson and Stolarski, 2004, Spencer, 1994] and David Gale [Spencer, 1994] in the context of coding theory. This game is played using of a m by m grid of lightbulbs. The adversary chooses an arbitrary subset of the lightbulbs to be initially "on." Next to every row (resp. column) of lightbulbs is a switch, which can be used to invert the state of every lightbulb in that row (resp. column). The protagonist's task is to minimize the number of lit lightbulbs (by flipping switches). This problem was proven very recently to be NP-hard [Roth and Viswanathan, 2008]. Let $\Phi = \{-1, 1\} \subset \mathbb{R}$. For matrices M, N let $d(M, N)$ denote the number of entries where M and N differ. It is fairly easy to see that the GB Game is equivalent to the following natural problems: [Roth and Viswanathan, 2008]

- Given matrix $M \in \Phi^{m \times m}$ find column vectors $x, y \in \Phi^m$ minimizing $d(M, xy^T)$.
- Given matrix $M \in \mathbb{F}_2^{m \times m}$ find $x, y \in \mathbb{F}_2^m$ minimizing $\sum_{ij} \mathbb{1}\left(M_{ij} \neq x_i \oplus y_j\right)$ where $\mathbb{F}_2$ is the finite field over two elements with addition operator $\oplus$.

- Given matrix $M \in \Phi^{m \times m}$ find column vectors $x, y \in \Phi^m$ *maximizing* $x^T M y$.

Consider the following scenario. Suppose that our nemesis, who knows the optimal solution to the Gale-Berlekamp problem shown in Figure 7, gives us a constant size random sample of it to tease us. How can we use this information to construct a good solution? One reasonable strategy is to set each variable greedily based on the random sample. Throughout this section we will focus on the row variables; the column variables are analogous. For simplicity our example has the optimal solution consisting of all of the switches in one position, which we denote by α. For row v, the greedy strategy, resulting in assignment $x^{(1)}$, is to set switch v to α iff $\hat{b}(v, \alpha) < \hat{b}(v, \beta)$, where $\hat{b}(v, \alpha)$ (resp. $\hat{b}(v, \beta)$) denotes the number of light bulbs in the intersection of row v and the sampled columns that would be lit if we set the switch to position α (resp. β).

With a constant size sample we can expect to set most of the switches correctly but a constant fraction of them will elude us. Can we do better? Yes, we simply do greedy again. The greedy prices analogous to $\hat{b}$ are shown in the columns labeled with b in the middle of Figure 7. For the example at hand, this strategy works wonderfully, resulting in us reconstructing the optimal solution exactly, as evidenced by the $b(x^{(1)}, v, \alpha) < b(x^{(1)}, v, \beta)$ for all v. In general this does not reconstruct the optimal solution but provably gives something close.

Some of the rows, e.g. the last one, have $b(x^{(1)}, v, \alpha)$ much less than $b(x^{(1)}, v, \beta)$ while other rows, such as the first, have $b(x^{(1)}, v, \alpha)$ and $b(x^{(1)}, v, \beta)$ closer together. We call variables with $|b(x^{(1)}, v, \alpha) - b(x^{(1)}, v, \beta)| > \Theta(n)$ *clearcut*. Intuitively, one would expect the clearcut rows to be more likely correct than the nearly tied ones. In fact, we can show that we get all of the clearcut ones correct, so the remaining problem is to choose values for the rows that are close to tied. However, those rows have a lot of lightbulbs lit, suggesting that the optimal value is large, so it is reasonable to run an additive approximation algorithm and use that to set the remaining variables.

Finally observe that we can simulate the random sample given by the nemesis by simply taking a random sample of the variables and then doing exhaustive search of all possibly assignments of those variables. We have just sketched our algorithm.

Our techniques differ from previous work [Bazgan et al., 2003, **?**, Giotis and Guruswami, 2006] in two key ways:

1. Previous work used a sample size of $O((\log n)/\epsilon^2)$, which allowed the clearcut variables to be set correctly after a single greedy step. We instead use a constant-sized sample and run a second greedy step before identifying the clearcut variables.

2. Our algorithm is the first one that runs the additive error algorithm after identifying clearcut variables. Previous work ran the additive error algorithm at the beginning.

The same ideas apply to all dense fragile CSPs.

3.2 Algorithm

Let $b(x, v, i)$ denote the number of unsatisfied constraints v would be in if x_v were set to i. Note that $b(x^*, v, i)$ can be written as a sum, over $S \in \binom{V}{k-1}$, of indicator function that there is a constraint over $S \cup \{v\}$ that would be unsatisfied if v were set to i. Each term is a function of x^*_u for $u \in S$. We estimate $b(x^*, v, i)$ with a random sample of $s = \frac{18 \log(480|D|k/\delta)}{\delta^2}$ of its terms. Let $S_1, S_2, \ldots, S_s$ be independent random samples of $k-1$ variables from V. We show (see Lemma 13 in [Karpinski and Schudy, 2009a]) that

$$\hat{b}(v, i) = \frac{\binom{n}{k-1}}{s} \sum_{j=1}^{s} \mathbb{1} \text{ (Exists unsatisfied constraint over } S \cup \{v\} \text{ if } x_v \leftarrow i)$$

is an unbiased estimator of $b(x^*, v, i)$.

We now describe our linear-time Algorithm 8 for fragile-dense MIN-k-CSPs. We first (lines 1–3) dispose of the easy case OPT=$\Omega(\binom{n}{k})$ by running an additive error algorithm and returning the output if it is

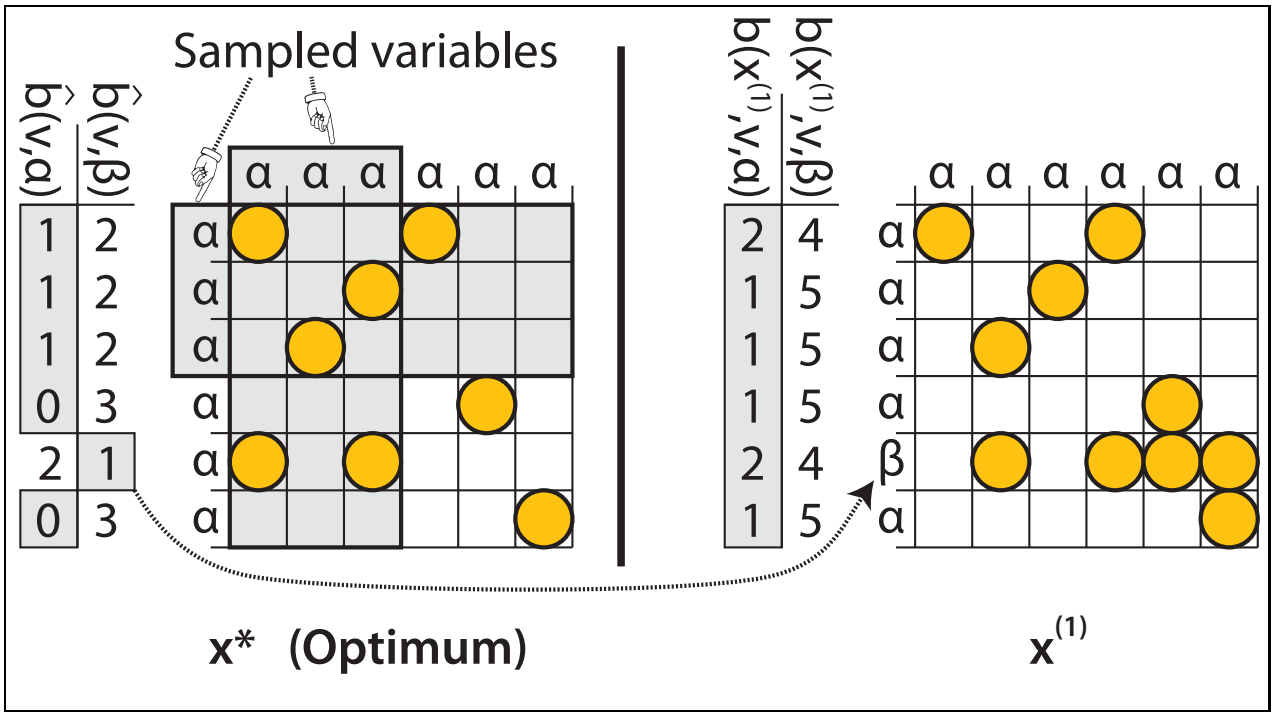

Figure 7: An illustration of our algorithmic ideas on the Gale-Berlekamp Game.

sufficiently costly. Second (lines 5–7) we take a random sample of variables. The optimal assignment x^* is of course unknown to us so we simply try every possible assignment of the sampled variables (line 7). We then compute $\hat{b}$ (line 8) and then make a preliminary assignment $x^{(1)}$ by setting each variable greedily relative to $\hat{b}$ (line 9). To reduce noise we do a second greedy step, yielding assignment $x^{(2)}$ (line 10). While constructing $x^{(2)}$ we make a note when the best value for a variable is far better than the second best; we fix such *clear-cut* variables to their values in assignment $x^{(2)}$. We finally run an additive error algorithm (line 12) to set the remaining variables.

4 Ranking MIN-CSPs

For general directed graphs, the FAS problem consists of removing the fewest number of edges so as to make the graph acyclic, and has applications such as scheduling [Flood, 1990] and graph layout [Sugiyama et al., 1981, Demetrescu and Finocchi, 2000] (see also [Lempel and Cederbaum, 1966, Baker and Hubert, 1977, Jünger, 1985]). This problem has been much studied, both in the mathematical programming community [Grötschel et al., 1985a,b, Jünger, 1985, Korte, 1979, Newman and Vempala, 2001, Newman, 2004], the approximation algorithms community [Leighton and Rao, 1999, Seymour, 1995, Even et al., 2000, 1998] and various applied communities [Demetrescu and Finocchi, 2000, 2001, Eades et al., 1993, Dwork et al., 2001]. The best known approximation ratio is $O(\log n \log\log n)$. Unfortunately the problem is NP-hard to approximate better than 1.36 [Karp, 1972, Dinur and Safra, 2002]. The equivalent problem of maximizing the number of edges not removed, called *max acyclic subgraph*, has also been extensively studied [Hassin and Rubinstein, 1994, Berger and Shor, 1997, Charikar et al., 2007a, Newman, 2001].

A *tournament* is the special case of directed graphs where every pair of vertices is connected by exactly one of the two possible directed edges. For tournaments, the FAS problem also has a long history, starting in the early 1960s in combinatorics [Seshu and Reed, 1961, Younger, 1963] and statistics [Slater, 1961]. In combinatorics and discrete probability, much early work [Erdös and Moon, 1965, Reid, 1969, Reid and Parker,

1: Run a $\frac{\epsilon}{1+\epsilon} \cdot \frac{\delta^2}{72k}\binom{n}{k}$ additive approximation algorithm.
2: **if** $Obj(answer) \geq \binom{n}{k}\delta^2/(72k)$ **then**
3: Return *answer*.
4: **else**
5: Let $s = \frac{18\log(480|D|k/\delta)}{\delta^2}$
6: Draw $S_1, S_2, \ldots, S_s$ randomly from $\binom{V}{k-1}$ with replacement.
7: **for** Each assignment $\hat{x^*}$ of the variables in $\bigcup_{j=1}^{s} S_j$ **do**
8: For all $v \in V$ and $i \in D$ let $\hat{b}(v,i) = \frac{\binom{n}{k-1}}{s}\sum_{j=1}^{s} \mathbb{1}$ (Exists unsat. constraint over $S \cup \{v\}$ if $x_v \leftarrow i$)
9: For all $v \in V$ let $x_v^{(1)} = \arg\min_i \hat{b}(v,i)$
10: For all $v \in V$ let $x_v^{(2)} = \arg\min_i b(x^{(1)}, v, i)$
11: Let $C = \{\, v \in V \;:\; b(x^{(1)}, v, x_v^{(2)}) < b(x^{(1)}, v, j) - \delta\binom{n}{k-1}/6 \text{ for all } j \neq x_v^{(2)} \,\}$.
12: Find $x^{(3)}$ of cost at most $\frac{\epsilon|V\setminus C|\delta}{3n}\binom{n}{k} + \min[Obj(x)]$ using an additive approximation algorithm, where the minimum ranges over x such that $x_v = x_v^{(2)}\ \forall v \in C$. See Karpinski and Schudy [2009a] for details.
13: **end for**
14: Return the best assignment $x^{(3)}$ found.
15: **end if**

Figure 8: Our $1+\epsilon$ approximation for δ-fragile-dense MIN-kCSP with variables taking values from domain D.

1970, Jung, 1970, Spencer, 1971, 1980, Fernandez de la Vega, 1983] focused on worst-case tournaments. In statistics and psychology, one motivation is *ranking by paired comparisons* [Slater, 1961]: here, you wish to sort some set by some objective but you do not have access to the objective, only a way to compare a pair and see which is greater; for example, determining people's preferences for types of food. Feedback arc set in tournament graphs and closely related problems have also been used in machine learning [Cohen et al., 1999, Ailon and Mohri, 2008]. Unfortunately the FAS problem is NP-hard even in tournaments [Ailon et al., 2008, Alon, 2006, Charbit et al., 2007] (see also [Conitzer, 2006]).

Researchers have been designing algorithms for the FAS problem in tournament graphs since the dawn of computer science. Slater and Alway describe simple heuristics for the FAS tournament problem in Slater [1961] (for comparison Hoare published quicksort the same year). Coleman and Wirth [2008] compare various algorithms empirically, including an algorithm by Ailon et al. [2008] and many others with no approximation guarantees. The best previously known approximation algorithms achieve constant factor approximations: 2.5 in the randomized setting [Ailon et al., 2008] and 3 in the deterministic setting [Van Zuylen et al. 2007b; Van Zuylen and Williamson 2007; Ailon et al. 2008] (see also Coppersmith et al. [2006]). Our main result is a polynomial time approximation scheme (PTAS) for this problem: thus the problem really is provably easier to approximate in tournaments than on general graphs.

Here is a weighted generalization of feedback arc set in tournaments.

Problem 5 (Weighted FAS Tournament)**.**
Input: complete directed graph with vertex set V, $n \equiv |V|$ and non-negative edge weights w_{uv} with $b \leq w_{uv} + w_{vu} \leq 1$ for some fixed positive constant b.
Output: An ranking π minimizing $C(\pi) = \sum_{\{u,v\}\subset V:\pi(v)>\pi(u)} w_{vu}$, where a ranking *is a bijective mapping from V to $\{1, 2, \ldots |V|\}$.*

(The unweighted problem is the special case where $w_{uv} = 1$ if (u,v) is an edge and $w_{uv} = 0$ otherwise.)

Theorem 6 (PTAS)**.** *There is a randomized algorithm for minimum Feedback Arc Set on weighted tournaments. Given $\epsilon > 0$, it outputs a ranking with expected cost at most $(1+\epsilon)OPT$. The expected running time*

is:

$$O\left(n^3 \log n(\log(1/b) + 1/\epsilon)\right) + n2^{\tilde{O}(1/(\epsilon b)^6)}.$$

The algorithm can be derandomized at the cost of increasing the running time to $n^{\tilde{O}(1/(\epsilon b)^{12})}$.

We remark that our PTAS is singly exponential in $1/\epsilon$, whereas the PTAS in the conference version of this work [Kenyon-Mathieu and Schudy, 2007] was doubly exponential in $1/\epsilon$.

Ailon et al. [2008] study the special-case $b = 1$, which they call *weighted FAS tournament with probability constraints*. Indeed, sampling a population naturally leads to defining w_{ij} as the probability that type i is preferred to type j. We note that all known approximation algorithms [Ailon et al. 2008; Coppersmith et al. 2006; Van Zuylen et al. 2007b; Van Zuylen and Williamson 2007] extend to weighted tournaments for $b = 1$.

An important application of weighted FAS tournaments is *rank aggregation*. Frequently, one has access to several rankings of objects of some sort, such as search engine outputs [Dwork et al., 2001], and desires to aggregate the input rankings into a single output ranking that is similar to all of the input rankings: it should have minimum average distance from the input rankings, for some notion of distance. This ancient problem was already studied in the context of voting by Borda [1781] and Condorcet [1785] in the 18^{th} century, and has aroused renewed interest recently [Dwork et al., 2001, Conitzer et al., 2006]. A natural notion of distance is the number of pairs of vertices that are in different orders: this defines the *Kemeny rank aggregation* problem [Kemeny, 1959, Kemeny and Snell, 1962]. This choice yields a maximum likelihood estimator for a certain naïve Bayes model [Young, 1995]. This problem is NP-hard [Bartholdi et al., 1989], even with only 4 voters [Dwork et al., 2001]. Constant factor approximation algorithms are known: choosing a random (or best) input ranking as the output gives a 2-approximation; finding the optimal aggregate ranking using the footrule distance as the metric instead of the Kendall-Tau distance also gives a 2-approximation [Dwork et al., 2001, Diaconis and Graham, 1977]. There is also a randomized 4/3 approximation algorithm [Ailon et al., 2008]. We improve on these results by giving a polynomial time approximation scheme (PTAS).

Corollary 7. *There is a randomized algorithm for Kemeny rank aggregation that. Given* $\epsilon > 0$*, it outputs a ranking with expected cost at most* $(1+\epsilon)OPT$*. The expected running time for* n *candidates is:*

$$O\left(\frac{n^3 \log n}{\epsilon}\right) + n2^{\tilde{O}(1/\epsilon^6)} + O\left(n^2 \cdot (\textit{number of voters})\right).$$

The algorithm can be derandomized at the cost of increasing the running time to $n^{\tilde{O}(1/\epsilon^{12})}$.

An important open question is whether there is a PTAS for the generalization of Kemeny rank aggregation to *partial* rankings such as search engine outputs. There is a 1.5-approximation algorithm [Ailon, 2007].

It is surprising that the minimum Feedback Arc Set problem on tournaments has an approximation scheme. The related problems of correlation clustering on complete graphs[1] [Charikar et al., 2005] and of feedback *vertex* set on tournaments [Cai et al., 2001] do not have PTASs unless P=NP.

Certain special cases can be solved exactly in polynomial time. Braverman and Mossel [2008] give an exact algorithm for feedback arc set when the input is generated by adding noise to a base ranking. There are also good algorithms for low-cost instances of FAS tournament and Kemeny rank aggregation (i.e. fixed-parameter tractability) [Alon et al., 2009, Dom et al., 2006, Betzler et al., 2008]. We will show an improvement in the runtime of FAST tournament from $2^{\tilde{O}(\sqrt{OPT})}$ to $2^{O(\sqrt{OPT})}$. We will investigate fixed parameter tractability of Kemeny rank aggregation and fully dense betweenness and expect to improve those runtimes as well.

Other related problems include d-dimensional arrangement [Charikar et al., 2007b].

We note that Amit Agarwal has informed us that he has obtained similar results.

[1] This problem is identical to FAS except it deals with symmetric transitive relations instead of antisymmetric ones.

Betweenness and generalizations

The betweenness problem has input consisting of information of the form object b must be between objects a and c in the ordering. It is NP hard in general to determine if there is an ordering consistent with all of the input [Chor and Sudan, 1998, Opatrny, 1979]. This implies that it is NP-hard to approximate the problem of minimizing disagreements to any factor. The problem of maximizing agreements has a constant factor approximation algorithm and no PTAS unless P=NP [Chor and Sudan, 1998].

We study the problem of minimizing disagreements in the special case of instances with information available for every set of three objects (fully dense). An easy reduction from FAST shows this problem is still NP-hard. We give a PTAS for this problem, which to our knowledge is the first non-trivial approximation algorithm for it.

4.1 Main Results

4.1.1 Algorithmic tools

Our first algorithmic tool is local search. A *single vertex move*, given an ranking π, a vertex x and a position i, consists of taking x out of π and putting it back in position i. We say a ranking is *locally optimal* if no single vertex move can improve its cost. Single vertex moves were used for FAS tournament as early as 1961 [Slater, 1961]. Single vertex moves and variants were also used in [Hassin and Rubinstein, 1994, Younger, 1963, Dwork et al., 2001, Coleman and Wirth, 2008]. Coleman and Wirth [2008] show that single vertex moves alone do not yield a constant factor approximation by giving a graph with global optimum $\Theta(n)$ and a local optimum of value $\Theta(n^2)$.

Our second algorithmic tool is the KwikSort algorithm by Ailon Charikar and Newman Ailon et al. [2008]. Recall from Problem 5 that b is a lower bound on $w_{uv} + w_{vu}$ for every pair $\{u, v\}$. We show that Ailon et al. [2008]'s KwikSort algorithm is a $5/b$-approximation for any $b > 0$. This follows from a trivial modification of their proof for the $b = 1$ case.

Our third and last algorithmic tool is the sampling-based approximation algorithms due to Arora, Frieze and Kaplan 2002 and to Frieze and Kannan 1999 described in Section 2. For completeness we use a (much simpler) algorithm based on our work reported in Section 2 [Mathieu and Schudy, 2008b].

Theorem 8. *Let $\delta > 0$ be fixed. There is a randomized algorithm* AddApprox *for minimum Feedback Arc Set on weighted tournaments with expected additive error δn^2 and runtime $O\left(n^2\right) + 2^{\tilde{O}(1/\delta^2)}$.*

4.1.2 Algorithm

Our main algorithm FAST-Scheme (short for Feedback Arc Set Tournament Approximation Scheme) is presented in Figure 9. To obtain the dependence on b claimed in Theorem 6, we actually need a slightly modified version of FAST-Scheme that executes the last three lines of FAST-Scheme repeatedly.

4.1.3 Why FAST-Scheme needs single vertex moves

As a warm-up to build intuition we demonstrate that FAST-Scheme needs single vertex moves. Consider a variant of FAST-Scheme that calls ImproveRec directly on the output of KwikSort, without doing any single vertex moves. We now show that this variant is not a PTAS.

Consider a chess tournament (instance) where all the results are consistent except that the worst player beat the best. If KwikSort picks any player other than the worst or best as the first pivot it finds a ranking with cost 1, which is optimal. On the other hand, if KwikSort picks the best player as the first pivot it produces ranking π^{bad}, which puts the worst player first and has cost $n - 1$.

Now consider what happens when ImproveRec is called on π^{bad}. As long as n is not too small ImproveRec divides and conquers, irrevocably committing to ranking the worst player among the k best for some $n/3 \leq k \leq 2n/3$. No matter what happens in the later recursions, the output clearly has cost at least $n/3$.

FAST-SCHEME:

Run the KWIKSORT algorithm on V to define an ranking π
Set $\eta \leftarrow \epsilon b/5$
Set $\beta \leftarrow \frac{\eta C(\pi)}{4n \log_{3/2} n}$ and $\kappa \leftarrow \frac{\eta^2 b^3}{350 \cdot 400^2}$
Perform single vertex moves on π until none can improve the cost by more than β.
Return IMPROVEREC(V, π)

IMPROVEREC(vertices S, ranking π on S):
if $|S| = 1$ **then**
Return π
else
if $C(\pi) \geq \kappa |S|^2$ **then**
Return the ranking from ADDAPPROX with $\delta = \frac{\eta}{4}\kappa$
else
Choose an integer k uniformly at random from $[|S|/3, 2|S|/3]$
Let L be the set of vertices v such that $\pi(v) \leq k$ and $R = S \setminus L$
Return concatenation of IMPROVEREC(L, π_L) and IMPROVEREC(R, π_R) where
π_L is the ranking of L induced by π, i.e. $\pi_L(v) = \pi(v)$, and
π_R is the ranking of R induced by π, i.e. $\pi_R(v) = \pi(v) - k$.
end if
end if

Figure 9: Approximation scheme for minimum Feedback Arc Set on tournaments

The overall expected cost of the output of this variant of FAST-SCHEME is therefore at least

$$\frac{n-2}{n}1 + \frac{2}{n}\frac{n}{3} = \frac{5}{3} + o(1)$$

hence this variant is not a PTAS.

5 Correlation Clustering with a fixed number of clusters

As noted in the introduction, the input to the correlation clustering problem consists of a graph where each edge is labeled either "+" or "-". The objective is to color the vertices with d colors minimizing the number of bichromatic "+" edges plus the number of monochromatic "-" edges. This problem was introduced by Ben-Dor et al. [1999] to cluster gene expression patterns. Since then it has been applied to problems in data mining and natural language processing [McCallum and Wellner, 2004, Finkel and Manning, 2008, Elsner and Charniak, 2008]. The correlation clustering approach has several strengths. It does not require users to specify a parametric form for the clusters, nor to pick the number of clusters. Unlike fully unsupervised clustering methods, it can use training data to optimize the pairwise classifier, but unlike classification, it does not require samples from the specific clusters found in the test data. For instance, it can use messages about cars to learn a similarity function that can then be applied to messages about whales.

As correlation clustering is hard to solve exactly, many previous papers focused on approximation algorithms [Bansal et al., 2004, Charikar et al., 2005, Demaine et al., 2006, Ailon et al., 2005, Swamy, 2004]. For the purpose of approximation, note that the maximization and minimization goals are not equivalent: minimizing is harder. How hard is it to minimize disagreements? A constant approximation ratio (currently 2.5) is achievable in the restricted *complete information* setting that assumes that information is available for every pair of vertices [Ailon et al., 2005, van Zuylen et al., 2007a, Bansal et al., 2004]; this result is essentially best possible (except for lowering the value of the constant) since the complete information problem

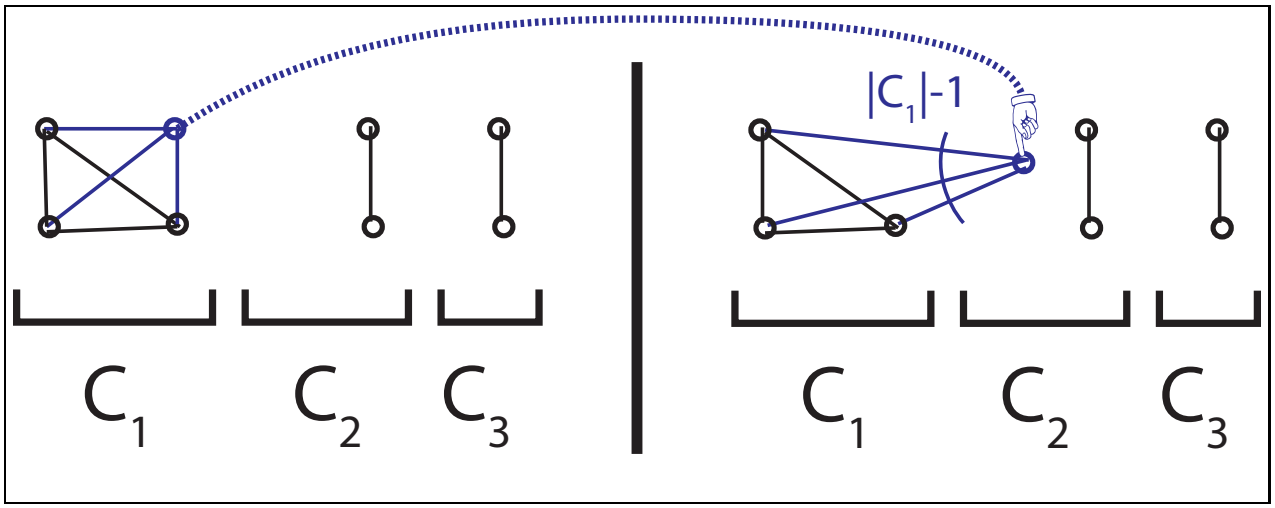

Figure 10: An illustration of correlation clustering and the rigidity property. Only "+" edges are shown.

is APX-hard [Charikar et al., 2005]. The generalization where no information is available for some pairs of objects admits an $O(\log n)$ approximation [Demaine et al., 2006], where n is the number of items (vertices) being clustered. In this thesis we focus on the complete information version.

How can one get around this APX-hardness result to provably get a clustering with value within $1+\epsilon$ of optimal? Additional assumptions are needed. In this section we assume that the number of clusters is a fixed constant. (In section 6 we study an alternate assumption, namely a noisy input model.) Correlation clustering with 2 clusters is fragile and Theorem 4 gives a linear-time approximation scheme. For $d > 2$ correlation clustering is not fragile but has properties allowing for a PTAS anyway. We also solve a generalization of correlation clustering called hierarchical clustering [Ailon and Charikar, 2005]. We prove the following theorem.

Theorem 9. *For every $\epsilon > 0$ there is a randomized $1+\epsilon$-approximation algorithm for correlation clustering and hierarchical clustering with fixed number of clusters d with running time $n^2 2^{O((\log d)d^6/\epsilon^2)}$.*

The hierarchical generalization has no new ideas so we discuss only correlation clustering in this proposal. See [Karpinski and Schudy, 2009a] for the generalization to hierarchical clustering.

The above results improves on the running time $n^{O(9^d/\epsilon^2)} \cdot \log n = n^{O(9^d/\epsilon^2)}$ of the previous PTAS for correlation clustering by Giotis and Guruswami [Giotis and Guruswami, 2006] in two ways: first the polynomial is linear in the size of the input and second the exponent is polynomial in d rather than exponential. Our result for hierarchical clustering with a fixed number of clusters is the first PTAS for this problem.

5.1 Intuition

As we noted previously in Section 1, correlation clustering constraints are not fragile for $d > 2$. Indeed, the constraint corresponding to a pair of vertices that are connected by a "−" edge can be satisfied by any coloring of the endpoints as long as the endpoints are colored differently. Fortunately there is a key observation in [Giotis and Guruswami, 2006] that allows for the construction of a PTAS. Consider the cost-zero clustering shown on the left of Figure 10. Note that moving a vertex from a small cluster to another small one increases the cost very little, but moving a vertex from a large cluster to anywhere else increases the cost a lot. Fortunately most vertices are in big clusters so, as in [Giotis and Guruswami, 2006], we can postpone processing the vertices in small clusters. We use the above ideas, which are due to [Giotis and Guruswami, 2006], the fragile-dense ideas sketched above, plus some additional ideas, to analyze our correlation clustering algorithm.

5.2 Reduction to Rigid MIN-2CSP

For expositional simplicity we focus here on the non-hierarchical $M = 1$ case. See Karpinski and Schudy [2009a] for $M > 1$ case.

We now show how to reduce correlation clustering with a constant number of clusters to the solution of a constant number of min-2CSPs with cardinality of D equal to d. We use a variant of our MIN-kCSP notation that is specialized for MIN-2CSPs. For vertices u, v and values i, j, let $p_{u,v}(i, j)$ be the cost of putting u in cluster i and v in cluster j. This is the same concept as p_I for the fragile case, but this notation is more convenient here. Define $b(x, v, i) = \sum_{u \in V\,:\,u \neq v} p_{u,v}(x_u, i)$, which is identical to b of the fragile-dense analysis but expressed using different notation.

Definition 10. *A MIN-2CSP is* rigid *if for some $\delta > 0$, all $v \in V$ and all $j \neq x^*_v$*

$$b(x^*, v, x^*_v) + b(x^*, v, j) \geq \delta |\{\, u \in V \setminus \{v\} \;:\; x^*_u = x^*_v \,\}|$$

Observe that $|\{\, u \in V \setminus \{v\} \;:\; x^*_u = x^*_v \,\}| \leq |V| = \binom{|V|}{2-1}$ hence any fragile-dense CSP is also rigid.

It is easy to see that correlation clustering is a 1-rigid MIN-2CSP with $|D| = d$. Our algorithm for solving rigid MIN-2CSPs is similar to our algorithm for fragile MIN-CSPs but adds a few new ideas; see Karpinski and Schudy [2009a] for details.

6 Correlation clustering with noisy input

As noted in Section 5 correlation clustering has no PTAS unless P=NP. In Section 5 we worked around this hardness result by assuming that the number of clustes is fixed. In this section, motivated by the data mining context, we take a different approach and assume that the input comes from a noisy model defined as follows. To generate the input, start from an arbitrary partition $\mathcal{B}$ of the vertices into clusters (*base clustering*). Then, each pair of vertices is perturbed independently with probability p. In the *fully-random* model, the input is generated from $\mathcal{B}$ simply by switching every perturbed pair. In the *semi-random* variant, an adversary controls the perturbed pairs and decides whether or not to switch them. Such noisy models are hardly new: they have been studied for many graph problems such as complete information feedback arc set [Braverman and Mossel, 2008], maximum bisection [Boppana, 1987], k-coloring, unique games[Kolla and Tulsiani, 2008], maximum clique [Jerrum, 1992, Kucera, 1995, Alon et al., 1998, Feige and Krauthgamer, 2000], and even for correlation clustering itself [Ben-Dor et al., 1999, Bansal et al., 2004, McSherry, 2001, Shamir and Tsur, 2007, Elsner and Schudy, 2009]. Indeed, studying the noisy model theoretically led [Ben-Dor et al., 1999] to a heuristic algorithm that they used to successfully cluster *real* gene expression data.

We note that, prior to our work, noisy correlation clustering had only been studied in the fully-random model [Ben-Dor et al., 1999, Bansal et al., 2004, McSherry, 2001, Shamir and Tsur, 2007]. Typically, results assume that all clusters have size bounded below, by $\Omega(n)$ [Ben-Dor et al., 1999] or by $\Omega(\sqrt{n \log n}/(1/2 - p)^{1+\epsilon})$ [Shamir and Tsur, 2007]. From a slightly different angle, [Bansal et al., 2004] makes no assumption on minimum input cluster size but finds all clusters of the base clustering that have size $\Omega(p\sqrt{n \log n}/(1/2 - p)^2)$.[2,3] Their algorithm can be used to yield a $1 + o(1)$ approximation for $p = \omega((\log n)/n)$. (More precisely, the additive error is $O(pn^{3/2}\sqrt{\log n}/(1/2 - p)^2)$ for $\sqrt{\log n/n} < p < 1/2$ and $O(n \log n)$ for smaller p.)

In contrast to our work, none of those prior results on noisy correlation clustering *certify* optimality of the output clustering. Certification is a highly desirable feature, as explained by Feige and Krauthgamer [Feige and Krauthgamer, 2000]: *"since average case algorithms do not have an a priori guarantee on their performance, it is important to certify that the algorithm is indeed successful on the particular instance at hand."* Our analysis relies on

[2] A straightforward variation works in the semi-random model but only finds clusters of size $\Omega(np^2\sqrt{\log n}/(1/2 - p)^2)$.

[3] McSherry [McSherry, 2001] also solved this problem, however their results depend on the number of clusters and are worse than [Shamir and Tsur, 2007] if the number of clusters is large.

the validity of the noisy model,[4] that is also important for us. All of our results address that issue and come with accompanying certificates.

Our results. We design a simple approximation algorithm. It uses a semi-definite programming relaxation and, for rounding, a constant factor approximation algorithm [van Zuylen et al., 2007a, Ailon et al., 2005]. Note that this semidefinite program was already used by [Swamy, 2004], but our rounding algorithm is completely different from theirs. We then proceed to prove that it has several desirable features.

Our first result bounds the cost of the output (Theorem 11). In the worst-case model, that cost is a constant-factor approximation. In the noisy model,[5] let OPT (resp. OPT_{full}) denote the optimum cost in the semi-random model (resp. fully-random model). In the fully random model, our algorithm is a PTAS. In the semi-random variant, our algorithm yields cost at most $OPT + O(n^{-1/6})OPT_{full}$ with high probability (over the random perturbation). Our algorithm also produces a lower-bound, proving that the random perturbation was "random enough" for this high probability event to occur.

Our second result (Theorem 12) shows one circumstance in which the base clustering can be reconstructed *exactly*. We show that if $p \leq 1/3$ and all clusters have size $\Omega(\sqrt{n})$ then the base clustering is the unique optimum solution of the natural SDP relaxation of correlation clustering, hence is the output of the algorithm.

Our third result (Theorem 13) analyzes the small noise regime, $p = O(n^{-\delta})$ for some δ. (Such a regime makes sense in applications in which perturbations are rare, for example if they correspond to mutations). Then, we have an algorithm which finds all clusters of size $\Omega(1/\delta)$, even in the semi-random model. It also produces a certificate (witness) proving that the clusters found are part of all optimal clusterings. For example, consider a base clustering consisting of clusters of (large enough) constant size, and let $p = n^{-1/10}$. In the input graph, every vertex belongs to, on average, $\Theta(1)$ edges inherited from the base clustering, and $\Theta(n^{9/10})$ edges created by the noisy process. Yet we can reconstruct the base clustering *exactly*, hence, for every vertex, determine which $\Theta(1)$ edges are correct among the sea of $\Theta(n^{9/10})$ noisy edges.

Our analysis uses probabilistic and combinatorial arguments, spectral properties of random matrices, and semidefinite programming duality.

Comparison between our results and previous results.

1. We give the first algorithm that achieves a $1 + o(1)$ approximation even when there are arbitrarily few noisy edges (so that OPT is small) and clusters are arbitrarily small (so that reconstructing them exactly is impossible.) (Theorem 11)

2. In the semi-random model, as it turns out, for constant p our additive error is $O(n^{3/2})$; the previous best was $O(\epsilon n^2)$ [Giotis and Guruswami, 2006].

3. In the fully random model, compared to previous work [Bansal et al., 2004, Shamir and Tsur, 2007] we find clusters a factor $\sqrt{\log n}$ smaller for constant p. (For example, when $p = 1/3$ we reconstruct the base clustering when all clusters are size $O(\sqrt{n})$ rather than $O(\sqrt{n \log n})$.) We improve additive error by the same factor.

4. Not only does the output have cost that, with high probability, is within $1 + o(1)$ of optimal, but our algorithm also produces a deterministic lower-bound witnessing that fact. In other words our algorithm knows whether or not its input is "sufficiently random."

5. Let $\delta > 0$. If $p = O(n^{-\delta})$ we give the first algorithm that exactly reconstructs every cluster of size $\Omega(1/\delta)$.

Our work is worse than previous work in two ways: First, the algorithm for item 5 has impractical runtime even for modest $\delta = 1/3$. Second, when p is a constant, we, like Shamir and Tsur [2007], can only find the base clusters exactly if *all* are size $\Omega(\sqrt{n})$. Bansal Blum Chawla [Bansal et al., 2004]'s result is incomparable, since they find the clusters of size $\Omega(\sqrt{n \log n})$, even if the base clustering is a mixture of large and small clusters.

[4] The algorithm is well-defined in general, but its quality relies on the noisy model assumption.

[5] Assuming $p \leq 1/2 - n^{-1/3}$

6.1 Algorithms

The input is an undirected graph $G = (V, E)$, where $\{u, v\} \in E$ iff we have information that u and v are similar, and $\{u, v\} \notin E$ iff we have information that u and v are dissimilar. A *clustering* is a partition of the vertices into *clusters*. To any clustering, we associate the induced graph, which has an edge between every pair of intra-cluster vertices, and an associated matrix, the adjacency matrix of the induced graph. We use the terminology *vertex pair* (VP) to avoid ambiguity with *edge*, which refers to vertex pairs that have an edge between them in some graph.

The *correlation clustering problem* consists of finding a clustering $\mathcal{A}$ of V minimizing $d(E, \mathcal{A})$, where $d(\cdot, \cdot)$ denotes the Hamming distance between two graphs (viewed as sets of edges), or equivalently, half of the ℓ_1 distance between the associated adjacency matrices: for symmetric matrices M and N, $d(M, N) = \frac{1}{2}\sum_{u,v} |M_{uv} - N_{uv}|$.

Fix the error parameter p. Our noisy model assumes that the input graph $G = (V, E)$ is generated by perturbing some unknown base clustering $\mathcal{B}$ as follows: identify $\mathcal{B}$ with the associated graph. Then, in the *fully random model*, for each pair of vertices, the information is flipped independently with probability p (in other words, to go from $\mathcal{B}$ to the input graph we flip every edge and every non-edge independently with probability p). In the *semi-random model*, for each pair of vertices, the information is corrupted (noisy) independently with probability p; then, an adversary generates the input graph by choosing similarity/dissimilarity information arbitrarily for each corrupted pair of vertices (when the adversary always flips the information of every corrupted pair, the resulting input is exactly the input generated in the fully random model).

We let $\mathcal{B}$ denote the unknown base clustering, $G = (V, E)$ the input graph generated from $\mathcal{B}$ in the semi-random model, and $G_{full} = (V, E_{full})$ the corresponding graph generated from $\mathcal{B}$ in the fully-random model. OPT denotes the cost of the optimal solution for input G, and, in the semi-random model, OPT_{full} denotes the cost of the optimal solution for the associated fully-random input.

Throughout the paper, when we write "w.h.p.", we mean that the event holds with probability at least $1 - n^{-\alpha}$ for some $\alpha > 0$.

We use the semi-definite programming relaxation previously used in Charikar et al. [2005], Swamy [2004]. We now describe the intuition behind this relaxation. One can picture a clustering geometrically by associating cluster c with the standard basis vector $e_c = (\underbrace{0, 0, \ldots, 0}_{c-1}, 1, \underbrace{0, \ldots, 0}_{n-c}) \in \mathbb{R}^n$. If object i is in cluster c then it is natural to associate i with the vector $r_i = e_c$. This gives a nice geometric picture of a clustering, with objects i and j in the same cluster if and only if $r_i = r_j$. Note that the dot product $r_i \bullet r_j$ is 1 if i and j are in the same cluster and 0 otherwise. These ideas yield a simple reformulation of the correlation clustering problem:

$$\begin{array}{c} \min_r \sum_{i,j:i<j} (r_i \bullet r_j) w_{ij}^- + (1 - r_j \bullet r_j) w_{ij}^+ \\ \text{s.t. } \forall i \quad \exists c : r_i = e_c \end{array}$$

To get an efficiently computable lower-bound we relax the constraints that the r_is are standard basis vectors, replacing them with two sets of constraints: $r_i \bullet r_i = 1$ for all i and $r_i \bullet r_j \geq 0$ for all i, j.

Since the r_i only appear as dot products, we can rewrite in terms of $x_{ij} = r_i \bullet r_j$. However, we must now constrain the x_{ij} to be the dot products of some set of vectors in $\mathbb{R}^n$. This is true if and only if the symmetric matrix $X = \{x_{ij}\}_{ij}$ is *positive semi-definite*. We now have the standard semi-definite programming (SDP) relaxation of correlation clustering (e.g. Charikar et al. [2005], Mathieu and Schudy [2008a]):

$$\begin{array}{c} \min_x \sum_{i,j:i<j} x_{ij} w_{ij}^- + (1 - x_{ij}) w_{ij}^+ \\ \text{s.t. } \left\{ \begin{array}{ll} x_{ii} = 1 & \forall i \\ x_{ij} \geq 0 & \forall i, j \\ X = \{x_{ij}\}_{ij} \text{ PSD} & \end{array} \right. . \end{array}$$

Theorem 11 (Main Theorem)**.** *Algorithm* MainCluster *runs in polynomial time and is such that:*

Input: graph $G = (V, E)$
Output: clustering $Output$

Let $\widehat{E}$ be E with edges that are not part of triangles removed.
Call algorithm SDPCluster on input $(V, \widehat{E})$, yielding clustering $\mathcal{A}$.
Let U denote the vertices in singleton clusters of $\mathcal{A}$.
Use maximum matching to compute an optimal clustering of U into clusters of size at most 2.

Figure 11: MainCluster algorithm

Input: graph $G = (V, F)$
Output: a clustering $\mathcal{A}$

Compute an optimal solution X^* to the following semi-definite program:

$$\min\ d(X,F)\ \text{ s.t. } \begin{cases} X_{ii} = 1 \\ X_{ij} \geq 0 \\ X_{ij} + X_{jk} - X_{ik} \leq 1 \\ X \text{ positive semi-definite} \end{cases}$$

Let $U \leftarrow V$.
while U is non-empty **do**
  Pick a pivot vertex v uniformly at random from U
  $A \leftarrow \{v\}$
  For each vertex u of $U \setminus \{v\}$, add u to A independently with probability X^*_{uv}.
  Output the resulting cluster A, and let $U \leftarrow U \setminus A$.
end while

Figure 12: SDPCluster algorithm

1. *For any input graph G, $\mathbf{E}_{alg}[Cost(OUT)] \leq 3.5\ OPT$, where $\mathbf{E}_{alg}[\cdot]$ denotes expectation over the random choices made by* MainCluster.
2. *In the semi-random model, if $p \leq 1/2 - n^{-1/3}$ then, with high probability over the noisy model, $\mathbf{E}_{alg}[Cost(OUT)] \leq OPT + O(n^{-1/6})OPT_{full}$.*
3. *One can compute a lower bound L on OPT. In the fully random model, if $p \leq 1/2 - n^{-1/3}$ then, with high probability over the noisy model, $\mathbf{E}_{alg}[Cost(OUT)] \leq L + o(L)$.*

If all the clusters are large then SDPCluster finds the base clustering exactly.

Theorem 12. *If $p \leq 1/3$ and all clusters have size $\Omega(\sqrt{n})$ then $\mathcal{B}$ is with high probability the unique optimum of the SDP.*

In addition, when the noise is small it is possible to reconstruct, not merely a clustering whose value is close to optimal, but a clustering such that all clusters of super-constant size are exactly the clusters of the base clustering.

Theorem 13 (Large Cluster Theorem)**.** *Assume that $p \leq n^{-\delta}/60$ for some δ. Then there exists an algorithm that outputs a set of clusters $\mathcal{A}$ such that:*

1. *W.h.p. the output clusters $\mathcal{A}$ are exactly the clusters of the base clustering that have size $\geq 3150/\delta$.*
2. *The algorithm certifies that for any optimal clustering, its clusters of size $\geq 3150/\delta$ are exactly the output clusters.*

7 Correlation clustering experiments

7.1 Introduction

Practical work has adopted one of three strategies for solving correlation clustering problmes. For a few specific tasks, one can restrict the problem so that it is efficiently solvable [Malioutov and Barzilay, 2006]. In most cases, however, this is impossible. Integer linear programming (ILP) can be used to solve the general problem optimally, but only when the number of data points is small. Beyond a few hundred points, the only available solutions are heuristic or approximate.

In this paper, we evaluate a variety of solutions for correlation clustering on two realistic NLP tasks, text topic clustering and chat disentanglement, where typical datasets are too large for ILP to find a solution. We investigate the relationship between the clustering objective and external evaluation metrics such as F-score and one-to-one overlap, showing that optimizing the objective is usually a reasonable aim, but that other measurements like number of clusters found should sometimes be used to reject pathological solutions. We prove that the best heuristics are quite close to optimal, using the first implementation of the semi-definite programming (SDP) relaxation to provide tighter bounds.

7.2 Algorithms

We begin with some notation and a formal definition of the problem. Our input is a complete, undirected graph G with n nodes; each edge in the graph has a probability p_{ij} reflecting our belief as to whether nodes i and j come from the same cluster. Our goal is to find a clustering, defined as a new graph G' with edges $x_{ij} \in \{0, 1\}$, where if $x_{ij} = 1$, nodes i and j are assigned to the same cluster. To make this consistent, the edges must define an equivalence relationship: $x_{ii} = 1$ and $x_{ij} = x_{jk} = 1$ implies $x_{ij} = x_{ik}$.

Our objective is to find a clustering as consistent as possible with our beliefs—edges with high probability should not cross cluster boundaries, and edges with low probability should. We define w_{ij}^+ as the cost of cutting an edge whose probability is p_{ij} and w_{ij}^- as the cost of keeping it. Mathematically, this objective can be written Ailon and Mohri [2008], Finkel and Manning [2008] as:

$$\min \sum_{ij:i<j} x_{ij} w_{ij}^- + (1 - x_{ij}) w_{ij}^+. \tag{1}$$

There are two plausible definitions for the costs w^+ and w^-, both of which have gained some support in the literature. We can take $w_{ij}^+ = p_{ij}$ and $w_{ij}^- = 1 - p_{ij}$ (*additive* weights) as in Ailon and Mohri [2008] and others, or $w_{ij}^+ = \log(p_{ij}), w_{ij}^- = \log(1 - p_{ij})$ (*logarithmic* weights) as in Finkel and Manning [2008]. The logarithmic scheme has a tenuous mathematical justification, since it selects a maximum-likelihood clustering under the assumption that the p_{ij} are independent and identically distributed given the status of the edge ij in the true clustering. If we obtain the p_{ij} using a classifier, however, this assumption is obviously untrue—some nodes will be easy to link, while others will be hard—so we evaluate the different weighting schemes empirically.

7.2.1 Greedy Methods

We use four greedy methods drawn from the literature; they are both fast and easy to implement. All of them make decisions based on the *net weight* $w_{ij}^\pm = w_{ij}^+ - w_{ij}^-$.

These algorithms step through the nodes of the graph according to a permutation π. We try 100 random permutations for each algorithm and report the run which attains the best objective value (typically this is slightly better than the average run; we discuss this more in the experimental sections). To simplify the pseudocode we label the vertices $1, 2, \ldots n$ in the order specified by π. After this relabeling $\pi(i) = i$ so π need not appear explicitly in the algorithms.

Three of the algorithms are given in Figure 13. All three algorithms start with the empty clustering and add the vertices one by one. The BEST algorithm adds each vertex i to the cluster with the strongest $w^\pm$

```
k ← 0 {number of clusters created so far}
for i = 1 ... n do
  for c = 1 ... k do
    if BEST then
      Quality_c ← max_{j∈C[c]} w_ij^±
    else if FIRST then
      Quality_c ← max_{j∈C[c]:w_ij^±>0} j
    else if VOTE then
      Quality_c ← Σ_{j∈C[c]} w_ij^±
    end if
  end for
  c* ← arg max_{1≤c≤k} Quality_c
  if Quality_{c*} > 0 then
    C[c*] ← C[c*] ∪ {i}
  else
    C[k++] ← {i} {form a new cluster}
  end if
end for
```

Figure 13: BEST/FIRST/VOTE algorithms

connecting to i, or to a new singleton if none of the $w^{\pm}$ are positive. The FIRST algorithm adds each vertex i to the cluster containing the most recently considered vertex j with $w_{ij}^{\pm} > 0$. The VOTE algorithm adds each vertex to the cluster that minimizes the correlation clustering objective, i.e. to the cluster maximizing the total net weight or to a singleton if no total is positive.

7.2.2 Local Search

We use the straightforward local search previously used by Gionis et al. [2007] and Goder and Filkov [2008]. The allowed *one element moves* consist of removing one vertex from a cluster and either moving it to another cluster or to a new singleton cluster. The best one element move (BOEM) algorithm repeatedly makes the most profitable best one element move until a local optimum is reached. *Simulated Annealing* (SA) makes a random single-element move, with probability related to the difference in objective it causes and the current temperature. Our annealing schedule is exponential and designed to attempt $2000n$ moves for n nodes. We initialize the local search either with all nodes clustered together, or at the clustering produced by one of our greedy algorithms (in our tables, the latter is written, eg. PIVOT/BOEM, if the greedy algorithm is PIVOT).

7.3 Bounding with SDP

Although comparing different algorithms to one another gives a good picture of relative performance, it is natural to wonder how well they do in an absolute sense—how they compare to the optimal solution. For very small instances, we can actually find the optimum using ILP, but since this does not scale beyond a few hundred points (see Section 7.4.1), for realistic instances we must instead bound the optimal value. Bounds are usually obtained by solving a *relaxation* of the original problem: a simpler problem with the same objective but fewer constraints.

The bound used in previous work Goder and Filkov [2008], Gionis et al. [2007], Bertolacci and Wirth [2007], which we call the *trivial bound*, is obtained by ignoring the transitivity constraints entirely. To optimize, we link ($x_{ij} = 1$) all the pairs where w_{ij}^{+} is larger than w_{ij}^{-}; since this solution is quite far from being a clustering, the bound tends not to be very tight.

To get a better idea of how good a real clustering can be, we use a semi-definite programming (SDP) relaxation to provide a better bound. Here we motivate and define this relaxation.

7.4 Experiments

7.4.1 Scalability

Using synthetic data, we investigate the scalability of the linear programming solver and SDP bound. To find optimal solutions, we pass the complete ILP[6] to CPLEX. This is reasonable for 100 points and solvable for 200; beyond this point it cannot be solved due to memory exhaustion. As noted below, despite our inability to compute the LP bound on large instances, we can sometimes prove that they must be worse than SDP bounds, so we do not investigate LP-solving techniques further.

The SDP has fewer constraints than the ILP ($O(n^2)$ vs $O(n^3)$), but this is still more than many SDP solvers can handle. For our experiments we used one of the few SDP solvers that can handle such a large number of constraints: Christoph Helmberg's ConicBundle library Helmberg [2009, 2000]. This solver can handle several thousand datapoints. It produces loose lower-bounds (off by a few percent) quickly but converges to optimality quite slowly; we err on the side of inefficiency by running for up to 60 hours. Of course, the SDP solver is only necessary to bound algorithm performance; our solvers themselves scale much better.

7.4.2 Twenty Newsgroups

In this section, we test our approach on a typical benchmark clustering dataset, 20 Newsgroups, which contains posts from a variety of Usenet newsgroups such as `rec.motorcycles` and `alt.atheism`. Since our bounding technique does not scale to the full dataset, we restrict our attention to a subsample of 100 messages[7] from each newsgroup for a total of 2000—still a realistically large-scale problem. Our goal is to cluster messages by their newsgroup of origin. We conduct experiments by holding out four newsgroups as a training set, learning a pairwise classifier, and applying it to the remaining 16 newsgroups to form our affinity matrix.[8]

Our pairwise classifier uses three types of features previously found useful in document clustering. First, we bucket all words[9] by their log document frequency (for an overview of TF-IDF see Joachims [1997]). For a pair of messages, we create a feature for each bucket whose value is the proportion of shared words in that bucket. Secondly, we run LSA Deerwester et al. [1990] on the TF-IDF matrix for the dataset, and use the cosine distance between each message pair as a feature. Finally, we use the same type of shared words features for terms in message subjects. We make a training instance for each pair of documents in the training set and learn via logistic regression.

The classifier has an average F-score of 29% and an accuracy of 88%—not particularly good. We should emphasize that the clustering task for 20 newsgroups is much harder than the more common classification task—since our training set is entirely disjoint with the testing set, we can only learn weights on feature categories, not term weights. Our aim is to create realistic-looking data on which to test our clustering methods, not to motivate correlation clustering as a solution to this specific problem. In fact, Zhong and Ghosh [2003] report better results using generative models.

We evaluate our clusterings using three different metrics (see Meila [2007] for an overview of clustering metrics). The *Rand* measure counts the number of pairs of points for which the proposed clustering agrees with ground truth. This is the metric which is mathematically closest to the objective. However, since most points are in different clusters, any solution with small clusters tends to get a high score. Therefore we also report the more sensitive *F-score* with respect to the minority ("same cluster") class. We also report the

[6]Consisting of the objective plus constraints $0 \leq x_{ij} \leq 1$ and triangle inequality Ailon and Mohri [2008].
[7]Available as `mini_newsgroups.tar.gz` from the UCI machine learning repository.
[8]The experiments below are averaged over four disjoint training sets.
[9]We omit the message header, except the subject line, and also discard word types with fewer than 3 occurrences.

Logarithmic Weights				
	Obj	Rand	F	1-1
SDP bound	51.1%	-	-	-
VOTE/BOEM	55.8%	93.80	33	41
SA	56.3%	93.56	31	36
PIVOT/BOEM	56.6%	93.63	32	39
BEST/BOEM	57.6%	93.57	31	38
FIRST/BOEM	57.9%	93.65	30	36
VOTE	59.0%	93.41	29	35
BOEM	60.1%	93.51	30	35
PIVOT	100%	90.85	17	27
BEST	138%	87.11	20	29
FIRST	619%	40.97	11	8

Table 1: Score of the solution with best objective for each solver, averaged over newsgroups training sets, sorted by objective.

one-to-one score, which measures accuracy over single points. For this metric, we calculate a maximum-weight matching between proposed clusters and ground-truth clusters, then report the overlap between the two.

When presenting objective values, we locate them within the range between the trivial lower bound discussed in Section 7.3 and the objective value of the singletons clustering ($x_{ij} = 0, i \neq j$). On this scale, lower is better; 0% corresponds to the trivial bound and 100% corresponds to the singletons clustering. It is possible to find values greater than 100%, since some particularly bad clusterings have objectives worse than the singletons clustering. Plainly, however, real clusterings will not have values as low as 0%, since the trivial bound is so unrealistic.

Our results are shown in Table 1. The best results are obtained using logarithmic weights with VOTE followed by BOEM; reasonable results are also found using additive weights, and annealing, VOTE or PIVOT followed by BOEM. On its own, the best greedy scheme is VOTE, but all of them are substantially improved by BOEM. First-link is by far the worst. Our use of the SDP lower bound rather than the trivial lower-bound of 0% reduces the gap between the best clustering and the lower bound by over a factor of ten. It is easy to show that the LP relaxation can obtain a bound of at most 50%[10]—the SDP beats the LP in both runtime and quality!

We analyze the correlation between objective values and metric values, averaging Kendall's tau[11] over the four datasets (Table 2). Over the entire dataset, correlations are generally good (large and negative), showing that optimizing the objective is indeed a useful way to find good results. We also examine correlations for the solutions with objective values within the top 10%. Here the correlation is much poorer; selecting the solution with the best objective value will not necessarily optimize the metric, although the correspondence is slightly better for the log-weights scheme. The correlations do exist, however, and so the solution with the best objective value is typically slightly better than the median.

In Figure 14, we show the distribution of one-to-one scores obtained (for one specific dataset) by the best solvers. From this diagram, it is clear that log-weights and VOTE/BOEM usually obtain the best scores for this metric, since the median is higher than other solvers' upper quartile scores. All solvers have quite high variance, with a range of about 2% between quartiles and 4% overall. We omit the F-score plot, which is similar, for space reasons.

[10]The solution $x_{ij} = \frac{1}{2}\mathbb{1}\left(w_{ij}^- > w_{ij}^+\right)$ for $i < j$ is feasible in the LP.

[11]The standard Pearson correlation coefficient is less robust to outliers, which causes problems for this data.

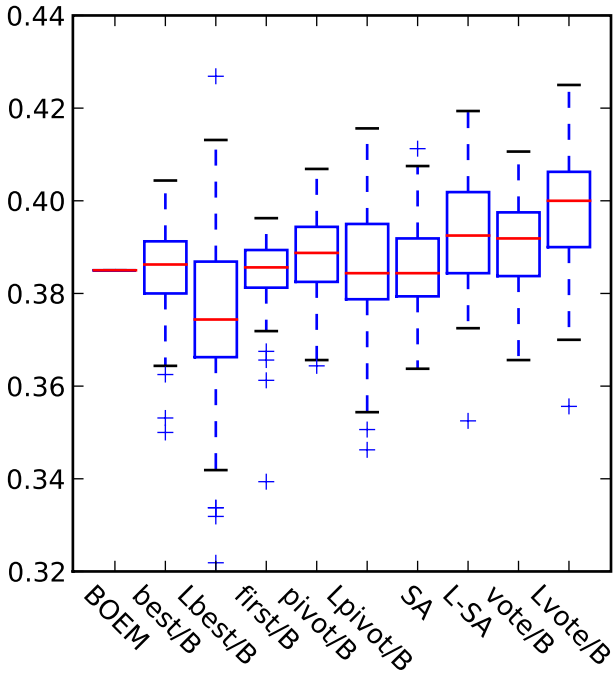

Figure 14: Box-and-whisker diagram (outliers as +) for one-to-one scores obtained by the best few solvers on a particular newsgroup dataset. L means using log weights. B means improved with BOEM.

	Rand	F	1-1
Log-wt	-.60	-.73	-.71
Top 10 %	-.14	-.22	-.24
Add-wt	-.60	-.67	-.65
Top 10 %	-.13	-.15	-.14

Table 2: Kendall's tau correlation between objective and metric values, averaged over newsgroup datasets, for all solutions and top 10% of solutions.

7.4.3 Chat Disentanglement

In the disentanglement task, we examine data from a shared discussion group where many conversations are occurring simultaneously. The task is to partition the utterances into a set of conversations. This task differs from newsgroup clustering in that data points (utterances) have an inherent linear order. Ordering is typical in discourse tasks including topic segmentation and coreference resolution.

We use the annotated dataset and pairwise classifier made available by Elsner and Charniak [2008];[12] this study represents a competitive baseline, although more recently Wang and Oard [2009] have improved it. Since this classifier is ineffective at linking utterances more than 129 seconds apart, we treat all decisions for such utterances as abstentions, $p = .5$. For utterance pairs on which it does make a decision, the classifier has a reported accuracy of 75% with an F-score of 71%.

[12]Downloaded from `cs.brown.edu/~melsner`

7.5 Conclusions

It is clear from these results that heuristic methods can provide good correlation clustering solutions on datasets far too large for ILP to scale. The particular solver chosen[13] has a substantial impact on the quality of results obtained, in terms of external metrics as well as objective value.

For general problems, our recommendation is to use log weights and run VOTE/BOEM. This algorithm is fast, achieves good objective values, and yields good metric scores on our datasets. Although objective values are usually only weakly correlated with metrics, our results suggest that slightly better scores can be obtained by running the algorithm many times and returning the solution with the best objective. This may be worth trying even when the datapoints are inherently ordered, as in chat.

Whatever algorithm is used to provide an initial solution, we advise the use of local search as a post-process. BOEM always improves both objective and metric values over its starting point.

The objective value is not always sufficient to select a good solution (as in the chat dataset). If possible, experimenters should check statistics like the number of clusters found to make sure they conform roughly to expectations. Algorithms that find far too many or too few clusters, regardless of objective, are unlikely to be useful. This type of problem can be especially dangerous if the pairwise classifier abstains for many pairs of points.

SDP provides much tighter bounds than the trivial bound used in previous work, although how much tighter varies with dataset (about 12 times smaller for newsgroups, 3 times for chat). This bound can be used to evaluate the absolute performance of our solvers; the VOTE/BOEM solver whose use we recommend is within about 5% of optimality. Some of this 5% represents the difference between the bound and optimality; the rest is the difference between the optimum and the solution found. If the bound were exactly optimal, we could expect a significant improvement on our best results, but not a very large one—especially since correlation between objective and metric values grows weaker for the best solutions. While it might be useful to investigate more sophisticated local searches in an attempt to close the gap, we do not view this as a priority.

References

N. Ailon. Aggregation of partial rankings, p-ratings and top-m lists. In *Procs. 18th ACM-SIAM SODA*, pages 415–424, 2007. Journal version to appear in *Algorithmica*. 6, 8, 13

N. Ailon and N. Alon. Hardness of fully dense problems. *Inf. Comput.*, 205(8):1117–1129, 2007. 5

N. Ailon and M. Charikar. Fitting tree metrics: Hierarchical clustering and phylogeny. In *Proc. 46th IEEE FOCS*, pages 73–82, 2005. 16

N. Ailon and M. Mohri. An efficient reduction of ranking to classification. In *Procs. 21st COLT*, pages 87–97, 2008. 2, 12, 21, 23

N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: ranking and clustering. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 684–693, 2005. ISBN 1-58113-960-8. 15, 18

N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: ranking and clustering. *J. ACM*, 55(5):Article 23, 2008. 2, 12, 13, 14

N. Alon. Ranking tournaments. *SIAM J. Discrete Math.*, 20(1):137–142, 2006. 2, 12

[13] Our C++ correlation clustering software and SDP bounding package are available for download from `cs.brown.edu/~melsner`.

N. Alon, M. Krivelevich, and B. Sudakov. Finding a large hidden clique in a random graph. In *SODA '98: Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, pages 594–598, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics. 17

N. Alon, W. Fernandez de la Vega, R. Kannan, and M. Karpinski. Random Sampling and Approximation of MAX-CSP Problems. In *Proc. 34th ACM STOC*, pages 232–239, 2002. Journal version in *J. Comput. System Sciences*, 67(2):212–243, 2003. 4, 7, 8

N. Alon, D. Lokshtanov, and S. Saurabh. Fast FAST. In *Procs. 36th ICALP, Part I, LNCS*, volume 5555, pages 49–58, 2009. 13

S. Arora, D. Karger, and M. Karpinski. Polynomial Time Approximation Schemes for Dense Instances of NP-Hard Problems. In *Proc. 27th ACM STOC*, pages 284–293, 1995. Journal version in *J. Comput. System Sciences*, 58(1):193–210, 1999. 5

S. Arora, A. M. Frieze, and H. Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. *Mathematical Programming*, 92(1):1–36, 2002. 14

V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004. 5

F. Baker and L. Hubert. Applications of combinatorial programming to data analysis: seriation using symmetric proximity measures. *British J. Math. Statis. Psych.*, 30:154–164, 1977. 11

N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Mach. Learn.*, 56(1-3):89–113, 2004. 3, 15, 17, 18

J. Bartholdi, III, C. Tovey, and M. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6:157–165, 1989. 2, 13

C. Bazgan, W. Fernandez de la Vega, and M. Karpinski. Polynomial Time Approximation Schemes for Dense Instances of Minimum Constraint Satisfaction. *Random Structures and Algorithms*, 23(1):73–91, 2003. 9, 10

A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3-4):281–297, 1999. 3, 15, 17

B. Berger and P. Shor. Tight bounds for the maximum acyclic subgraph problem. *J. Algorithms*, 25(1): 1–18, 1997. 11

M. Bertolacci and A. Wirth. Are approximation algorithms for consensus clustering worthwhile? In *SDM '07: Procs. 7th SIAM International Conference on Data Mining*, 2007. 22

N. Betzler, M. R. Fellows, J. Guo, R. Niedermeier, and F. A. Rosamond. Fixed-parameter algorithms for Kemeny scores. In *AAIM '08: Procs. 4th int'l conf. on Algorithmic Aspects in Information and Management*, pages 60–71, 2008. ISBN 978-3-540-68865-5. 13

R. Boppana. Eigenvalues and graph bisection: An average-case analysis. In *28th Annual Symposium on Foundations of Computer Science*, pages 280–285, Los Angeles, October 1987. IEEE. 17

J. Borda. Mémoire sur les élections au scrutin. *Histoire de l'Académie Royale des Sciences*, 1781. 2, 13

M. Braverman and E. Mossel. Noisy sorting without resampling. In *Procs. 19th ACM-SIAM SODA*, pages 268–276, 2008. 13, 17

M. Cai, X. Deng, and W. Zang. An approximation algorithm for feedback vertex sets in tournaments. *SIAM J. Computing*, 30(6):1993–2007, 2001. 13

J. Carlson and D. Stolarski. The Correct Solution to Berlekamp's Switching Game. *Discrete Mathematics*, 287(1–3):145–150, 2004. 9

P. Charbit, S. Thomasse, and A. Yeo. The minimum feedback arc set problem is NP-hard for tournaments. *Combinatorics, Probability and Computing*, 16:1–4, 2007. 2, 12

M. Charikar, V. Guruswami, and A. Wirth. Clustering with qualitative information. *J. Computer and System Sciences*, 71(3):360 – 383, 2005. 3, 13, 15, 16, 19

M. Charikar, K. Makarychev, and Y. Makarychev. On the advantage over random for maximum acyclic subgraph. In *Procs. 48th IEEE FOCS*, pages 625–633, 2007a. 11

M. Charikar, K. Makarychev, and Y. Makarychev. A divide and conquer algorithm for d-dimensional arrangement. In *Procs. 18th ACM-SIAM SODA*, pages 541–546, 2007b. 13

B. Chor and M. Sudan. A geometric approach to betweenness. *SIAM J. Discrete Math.*, 11(4):511–523, 1998. 14

W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 475–480, New York, NY, USA, 2002. ACM. ISBN 1-58113-567-X. 3

W. W. Cohen, R. E. Schapire, and Y. Singer. Learning to order things. *J. Artificial Intelligence Research*, 10:243–270, 1999. 2, 12

T. Coleman and A. Wirth. Ranking tournaments: local search and a new algorithm. In *Procs. 10th workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 133–141, 2008. 12, 14

M. J. Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. 1785. Reprinted by AMS Bookstore in 1972. 2, 13

V. Conitzer. Computer Slater rankings using similarities among candidates. In *Procs. 21st AAAI*, pages 613–619, 2006. 2, 12

V. Conitzer, A. Davenport, and J. Kalagnanam. Improved bounds for computing Kemeny rankings. In *Proc. 21st AAAI*, pages 620–626, 2006. 2, 13

D. Coppersmith, L. Fleischer, and A. Rudra. Ordering by weighted number of wins gives a good ranking for weighted tournaments. In *Procs. 17th ACM-SIAM SODA*, pages 776–782, 2006. 12, 13

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990. 23

E. D. Demaine, D. Emanuel, A. Fiat, and N. Immorlica. Correlation clustering in general weighted graphs. *Theor. Comput. Sci.*, 361(2):172–187, 2006. ISSN 0304-3975. 15, 16

C. Demetrescu and I. Finocchi. Break the "right" cycles and get the "best" drawing. In *Procs. 2nd Int'l workshop on Algorithmic Engineering and Experiments (ALENEX)*, pages 171–182, 2000. 11

C. Demetrescu and I. Finocchi. Breaking cycles for minimizing crossings. *J. Experimental Algorithmics*, 6: Article 2, 2001. 11

P. Diaconis and R. Graham. Spearman's footrule as a measure of disarray. *J. Royal Statistical Society*, 39 (2):262–268, 1977. 13

I. Dinur and S. Safra. The importance of being biased. In *Procs. 34th ACM STOC*, pages 33–42, 2002. 11

M. Dom, J. Guo, F. Hüffner, R. Niedermeier, and A. Truß. Fixed-parameter tractability results for feedback set problems in tournaments. In *LNCS*, volume 3998, pages 320–331. Springer, 2006. 13

C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggegation methods for the web. In *Procs. 10^{th} WWW*, pages 613–622, 2001. The NP-hardness proof is in the online-only appendix available from `http://www10.org/cdrom/papers/577/`. 2, 11, 13, 14

P. Eades, X. Lin, and W. Smyth. A fast and effective heuristic for the feedback arc set problem. *Information Processing Letters*, 47(6):319–323, 1993. 11

M. Elsner and E. Charniak. You talking to me? a corpus and algorithm for conversation disentanglement. In *Proceedings of ACL-08: HLT*, pages 834–842, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P08/P08-1095`. 15, 25

M. Elsner and W. Schudy. Bounding and comparing methods for correlation clustering beyond ILP. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 19–27. Association for Computational Linguistics, June 2009. 3, 17

P. Erdös and J. Moon. On sets of consistent arcs in tournaments. *Canadian Math. Bulliten*, 8(3):269–271, 1965. 11

G. Even, J. Naor, B. Schieber, and M. Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998. 11

G. Even, J. S. Naor, S. Rao, and B. Schieber. Divide-and-conquer approximation algorithms via spreading metrics. *J. ACM*, 47(4):585–616, 2000. 11

U. Feige and R. Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Struct. Algorithms*, 16(2):195–208, 2000. 17

W. Fernandez de la Vega. MAX-CUT has a Randomized Approximation Scheme in Dense Graphs. *Random Struct. Algorithms*, 8(3):187–198, 1996. 7

W. Fernandez de la Vega. On the maximal cardinality of a consistent set of arcs in a random tournament. *J. Combinatorial Theory, Ser. B*, 35(3):328–332, 1983. 12

W. Fernandez de la Vega, R. Kannan, and M. Karpinski. Approximation of Global MAX–CSP Problems. Technical Report TR06-124, Electronic Colloquim on Computation Complexity, 2006. 8

J. R. Finkel and C. D. Manning. Enforcing transitivity in coreference resolution. In *Proceedings of ACL-08: HLT, Short Papers*, pages 45–48, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P08/P08-2012`. 3, 15, 21

M. Flood. Exact and heuristic algorithms for the weighted feedback arc set problem: a special case of the skew-symmetric quadratic assignment problem. *Networks*, 20(1):1–23, 1990. 11

A. M. Frieze and R. Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2): 175–220, 1999. 4, 14

Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981. 5

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979. 1

A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. *ACM Trans. on Knowledge Discovery from Data*, 1(1):Article 4, 2007. 22

I. Giotis and V. Guruswami. Correlation clustering with a fixed number of clusters. *Theory of Computing*, 2(1):249–266, 2006. 3, 9, 10, 16, 18

A. Goder and V. Filkov. Consensus clustering algorithms: Comparison and refinement. In *ALENEX '08: Procs. 10th Workshop on Algorithm Enginering and Experiments*, pages 109–117. SIAM, 2008. 22

M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/227683.227684. 5

A. Greenwald, Z. Li, and W. Schudy. More efficient internal-regret-minimizing algorithms. In *21st Annual Conference on Learning Theory - COLT 2008*, pages 239–250, 2008. 6

M. Grötschel, M. Jünger, and G. Reinelt. Facets of the linear ordering polytope. *Math Programming*, 33: 43–60, 1985a. 11

M. Grötschel, M. Jünger, and G. Reinelt. On the acyclic subgraph polytope. *Math Programming*, 33:28–42, 1985b. 11

R. Hassin and S. Rubinstein. Approximations for the maximum acyclic subgraph problem. *Information Processing Letters*, 51(3):133–140, 1994. 11, 14

C. Helmberg. Semidefinite programming for combinatorial optimization. Technical Report ZR-00-34, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2000. 23

C. Helmberg. *The ConicBundle Library for Convex Optimization*, 2009. Ver. 0.2i from `http://www-user.tu-chemnitz.de /~helmberg/ConicBundle/`. 23

M. Jerrum. Large cliques elude the metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992. 17

T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In *International Conference on Machine Learning (ICML)*, pages 143–151, 1997. 23

H. A. Jung. On subgraphs without cycles in tournaments. In P. Erdös, A. Rényi, and V. Sós, editors, *Combinatorial theory and its applications II*, pages 675–677. North Holland, 1970. 12

M. Jünger. *Ployhedral Combinators and the Acyclic Subgraph Problem*. Heldermann, Berlin, 1985. 11

R. Karp. Reducibility among combinatorial problems. In *Procs. Complexity of Computer Computations*, pages 85–103, 1972. 11

M. Karpinski and W. Schudy. Linear time approximation schemes for the Gale-Berlekamp game and related minimization problems. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 313–322, 2009a. 6, 10, 12, 16, 17

M. Karpinski and W. Schudy. Approximation of the betweeness-tournament problem and some ranking generalizations. Soon to be in preparation, 2009b. 2, 5, 6

J. Kemeny. Mathematics without numbers. *Daedalus*, 88:571–591, 1959. 2, 13

J. Kemeny and J. Snell. *Mathematical models in the social sciences*. Blaisdell, New York, 1962. Reprinted by MIT press, Cambridge, 1972. 2, 13

C. Kenyon-Mathieu and W. Schudy. How to rank with few errors: a PTAS for weighted feedback arc set on tournaments. In *Procs. 39th ACM STOC*, pages 95–103, 2007. 2, 6, 13

A. Kolla and M. Tulsiani. Playing random and expanding unique games. Unpublished manuscript, 2008. 17

B. Korte. Approximation algorithms for discrete optimization problems. *Ann. Discrete Math*, 4:85–120, 1979. 11

L. Kucera. Expected complexity of graph partitioning problems. *Discrete Appl. Math.*, 57(2–3):193–212, 1995. 17

T. Leighton and S. Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999. 11

A. Lempel and I. Cederbaum. Minimum feedback arc and vertex set of a directed graph. *IEEE Trans. Circuit Theory*, 13(4):399–403, 1966. 11

I. Malioutov and R. Barzilay. Minimum cut model for spoken lecture segmentation. In *ACL*. The Association for Computer Linguistics, 2006. 21

C. Mathieu and W. Schudy. Correlation clustering with noisy input. Unpublished manuscript available from http://www.cs.brown.edu/∼ws/papers/clustering.pdf, 2008a. 19

C. Mathieu and W. Schudy. Yet another algorithm for dense max cut: Go greedy. In *Procs. 19th ACM-SIAM SODA*, pages 176–182, 2008b. 6, 7, 14

C. Mathieu and W. Schudy. Correlation clustering with noisy input. In *To appear in Procs. 21st SODA, preprint: http://www.cs.brown.edu/~ws/papers/cluster.pdf*, 2010. 3, 6

C. Mathieu and W. Schudy. How to rank with fewer errors – a PTAS for feedback arc set in tournaments. In Submission `http://www.cs.brown.edu/~ws/papers/fast_journal.pdf`, 2009. 2, 6

C. Mathieu, O. Sankur, and W. Schudy. Online correlation clustering. In preparation, 2009. 6

A. McCallum and B. Wellner. Conditional models of identity uncertainty with application to noun coreference. In *Proceedings of the 18th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 905–912. MIT Press, 2004. URL `http://www.cs.umass.edu/~mccallum/papers/condid-nips2004.pdf`. 15

W. McLendon, III, B. Hendrickson, S. Plimpton, and L. Rauchwerger. Finding strongly connected components in parallel in particle transport sweeps. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 328–329, 2001. 6

W. McLendon, III, B. Hendrickson, S. J. Plimpton, and L. Rauchwerger. Finding strongly connected components in distributed graphs. *J. Parallel Distrib. Comput.*, 65(8):901–910, 2005. 6

F. McSherry. Spectral partitioning of random graphs. In *FOCS '01: Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*, page 529, Washington, DC, USA, 2001. IEEE Computer Society. 17

M. Meila. Comparing clusterings–an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895, May 2007. doi: http://dx.doi.org/10.1016/j.jmva.2006.11.013. URL `http://dx.doi.org/10.1016/j.jmva.2006.11.013`. 23

A. Newman. The maximum acyclic subgraph problem and degree-3 graphs. In *Procs. APPROX*, pages 147–158, 2001. 11

A. Newman. Cuts and orderings: On semidefinite relaxations for the linear ordering problem. In *Procs. APPROX*, pages 195–206, 2004. 11

A. Newman and S. Vempala. Fences are futile: on relaxations for the linear ordering problem. In *Procs. Integer Programming and Combinatorial Optimization (IPCO)*, pages 333–347, 2001. 11

J. Opatrny. Total ordering problems. *SIAM J. Comput.*, 8(1):111–114, 1979. 14

S. Plimpton, B. Hendrickson, S. Burns, and W. McLendon, III. Parallel algorithms for radiation transport on unstructured grids. In *Supercomputing '00: Proceedings of the 2000 ACM/IEEE conference on Supercomputing (CDROM)*, page 25, 2000. 6

K. Reid. On sets of arcs containing no cycles in tournaments. *Canad. Math Bulletin*, 12:261–264, 1969. 11

K. Reid and E. Parker. Disproof of a conjecture of Erdös and Moser on tournaments. *J. Combin. Theory*, 9:225–238, 1970. 11

R. Roth and K. Viswanathan. On the Hardness of Decoding the Gale-Berlekamp Code. *IEEE Transactions on Information Theory*, 54(3):1050–1060, March 2008. 9

M. Rudelson and R. Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *J. ACM*, 54(4):21, 2007. 7, 8

F. Schalekamp and A. van Zuylen. Rank aggregation: Together we're strong. In *ALENEX*, pages 38–51, 2009. 2

W. Schudy. Optimal restart strategies for tree search. In preparation, 2009. 6

W. Schudy. Finding strongly connected components in parallel using $O(log^2 n)$ reachability queries. In *SPAA '08: Proceedings of the twentieth annual symposium on Parallelism in algorithms and architectures*, pages 146–151, 2008. 6

S. Seshu and M. B. Reed. *Linear Graphs and Electrical Networks*. Addison-Wesley, Reading, MA, 1961. 11

P. D. Seymour. Packing directed circuits fractionally. *Combinatorica*, 15:281–288, 1995. 11

R. Shamir and D. Tsur. Improved algorithms for the random cluster graph model. *Random Structures and Algorithms*, 31(4):418–449, 2007. 17, 18

P. Slater. Inconsistencies in a schedule of paired comparisons. *Biometrika*, 48:303–312, 1961. 2, 11, 12, 14

J. Spencer. *Ten Lectures on the Probabilistic Method*. SIAM, Regional Conference Series, second edition, 1994. 9

J. Spencer. Optimal ranking of tournaments. *Networks*, 1:135–138, 1971. 12

J. Spencer. Optimal ranking of unrankable tournaments. *Period. Math. Hungar.*, 11(2):131–144, 1980. 12

K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierachical system structures. *IEEE Trans. System Man Cybern.*, 11(2):109–125, 1981. 11

C. Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In *SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 526–527, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics. 15, 18, 19

A. van Zuylen and D. P. Williamson. Deterministic algorithms for rank aggregation and other ranking and clustering problems. In *Procs. Workshop on Approximation and Online Algorithms*, pages 260–273, 2007. 12, 13

A. van Zuylen, R. Hegde, K. Jain, and D. P. Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 405–414, 2007a. 15, 18

A. van Zuylen, R. Hegde, K. Jain, and D. P. Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. In *Procs. ACM-SIAM SODA*, pages 405–414, 2007b. 12, 13

V. V. Vazirani. *Approximation Algorithms*. Springer, 2001. 5

L. Wang and D. W. Oard. Context-based message expansion for disentanglement of interleaved text conversations. In *Proceedings of NAACL-09 (to appear)*, 2009. 25

P. Young. Optimal voting rules. *The Journal of Economic Perspectives*, 9(1):51–64, 1995. 2, 13

D. H. Younger. Minimum feedback arc sets for a directed graph. *IEEE Trans. Circuit Theory*, 10:238–245, 1963. 11, 14

S. Zhong and J. Ghosh. Model-based clustering with soft balancing. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 459, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1978-4. 23