Linear Time Approximation Schemes for the Gale-Berlekamp Game and Related Minimization Problems

Marek Karpinski* and Warren Schudy†

Abstract

We design a linear time approximation scheme for the Gale-Berlekamp Switching Game and generalize it to a wider class of dense fragile minimization problems including the Nearest Codeword Problem (NCP) and Unique Games Problem. Further applications include, among other things, finding a constrained form of matrix rigidity and maximum likelihood decoding of an error correcting code. As another application of our method we give the first linear time approximation schemes for correlation clustering with a fixed number of clusters and its hierarchical generalization. Our results depend on a new technique for dealing with small objective function values of optimization problems and could be of independent interest.

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

†Dept. of Computer Science, Brown University. Part of this work was done while visiting University of Bonn. Email: ws@cs.brown.edu

§The authors thank the Hausdorff Research Institute for Mathematics for the kind support.

1 Introduction

The Gale-Berlekamp Switching Game (GB Game) was introduced independently by Elwyn Berlekamp [10, 23] and David Gale [23] 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 [21]. 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: [21]

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

We focus on the equivalent minimization versions and prove existence of linear-time approximation schemes for them.

Theorem 1. *For every $\epsilon > 0$ there is a randomized $1 + \epsilon$-approximation algorithm for the Gale-Berlekamp Switching Game (its minimization version) with runtime $O(m^2) + 2^{O(1/\epsilon^2)}$.*

In order to achieve the linear-time bound of our algorithms, we introduce two new techniques: calling the additive error approximation algorithm *at the end* of our algorithm and greedily refining the random sample used by the algorithm. These new methods could also be of independent interest.

A constraint satisfaction problem (CSP) consists of n variables over a domain of constant-size d and a collection of arity-k constraints (k constant). The objective of MIN-kCSP (MAX-kCSP) is to minimize the number of unsatisfied (maximize the number of satisfied) constraints. An (everywhere) dense instance is one where every variable is involved in at least a constant times the maximum possible number of constraints, i.e. $\Omega(n^{k-1})$. For example, the GB Game is a dense MIN-2CSP since each of the $n = 2m$ variables is involved in precisely $m = n/2$ constraints. It is natural to consider generalizing Theorem 1 to all dense MIN-CSPs, but unfortunately many such problems have no PTASs unless P=NP [7] so we must look at a restricted class of MIN-CSPs. 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. Clearly the GB Game can be modeled as a fragile dense MIN-2CSP. Our results generalize to all dense *fragile* MIN-kCSPs.

We now formulate our general theorem.

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

Any approximation algorithm for MIN-kCSP must read (by adversary argument) the entire input to distinguish between instances with optimal value of 1 and 0 and hence the $O(n^k)$ term of the runtime cannot be improved. It is fairly easy to see that improving the second term (to $2^{o(1/\epsilon^2)}$) would imply a $O(n^2) + 2^{o(1/\epsilon^2)}$-time PTAS for average-dense max cut. Over a decade worth of algorithms [5, 6, 16, 2, 20] for MAX-kCSP all have dependence on ϵ of at best $2^{O(1/\epsilon^2)}$, so any improvement to the runtime of Theorem 2 would be surprising.

We begin exploring applications of Theorem 2 by generalizing the Gale-Berlekamp game to higher dimensions k (k-ary GB) and then to arbitrary k-ary equations. Given n variables $x_i \in \mathbb{F}_2$ and m linear equations of the form $x_{i_1} \oplus x_{i_2} \oplus \ldots \oplus x_{i_k} = 0$ (or $= 1$), the k-ary Nearest Codeword Problem (NCP) consists of finding an assignment minimizing the number of unsatisfied equations. As the name suggests, the Nearest Codeword Problem can be interpreted as maximum likelihood decoding for linear error correcting codes. The Nearest Codeword Problem has fragile constraints so Theorem 2 implies a linear-time PTAS for the k-ary GB problem and the dense k-ary Nearest Codeword Problem.

The Unique Games Problem (UGP) [12, 18] consists of solving MIN-2CSPs where the constraints are permutations over a finite domain D of colors; i.e. a constraint involving variables x_u and x_v is satisfied iff $x_u = \pi_{uv}(x_v)$ for permutation π_{uv}. These constraints are clearly fragile, so Theorem 2 implies also a linear-time PTAS for the dense Unique Game Problem (with a constant number of colors).

The multiway cut problem, also known as MIN-dCUT, consists of coloring an undirected graph with d colors, such that each of d terminal nodes t_i is colored with color i, minimizing the number of bichromatic edges. The requirement that the terminal nodes must be colored particular colors does not fit in our dense fragile MIN-CSP framework, so we use a work-around: let the constraint corresponding to an edge be satisfied only if it is monochromatic *and* the endpoint(s) that are terminals (if any) are colored correctly.

As another application, consider MIN-kSAT, the problem of minimizing the number of *satisfied* clauses of a boolean expression in conjunctive normal form where each clause has k variables (some negated). We consider the equivalent problem of minimizing the number of *unsatisfied* conjunctions of a boolean expression in *disjunctive* normal form. A conjunction can be represented as a fragile constraint indicating that all of the negated variables within that constraint are false and the remainder are true, so Theorem 2 applies to MIN-kSAT as well.

Finally we consider correlation clustering and hierarchical clustering with a fixed number of clusters [17, 1]. Correlation cluster consists of coloring an undirected graph with d colors (like multiway cut so far), minimizing the sum of the number of cut edges and the number of uncut non-edges. Correlation clustering with two clusters is equivalent to the following symmetric variant of the Gale-Berlekamp game: given a symmetric matrix $M \in \Phi^{m \times m}$ find a row vector $x \in \Phi^m$ minimizing $d(M, xx^T)$. Like the GB game, correlation clustering with 2 clusters is fragile and Theorem 2 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 [1]. We prove the following theorem.

Theorem 3. *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 $n^2 2^{O(d^6/\epsilon^2)}$.*

The above results improves on the running time $O(n^{9^d/\epsilon^2}) \log n = O(n^{9^d/\epsilon^2})$ of the previous PTAS for correlation clustering by Giotis and Guruswami [17] 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 that problem.

We prove Theorem 2 in Sections 2 and 3 and Theorem 3 in Sections 4 and 5.

Related Work

Elwyn Berlekamp built a physical model of the GB game with either $m = 8$ or $m = 10$ [10, 23] at Bell Labs in the 1960s motivated by the connection with coding theory and the Nearest Codeword

Problem. Several works [15, 10] investigated the cost of worst-case instances of the GB Game; for example the worst-case instance for $m = 10$ has cost 35 [10]. Roth and Viswanathan [21] showed very recently that the GB game is in fact NP-hard. They also give a linear-time algorithm if the input is generated by adding random noise to a cost zero instance. Replacing Φ with $\mathbb{R}$ in the third formulation of the GB Game yields the problem of computing the 1-rigidity of a matrix. Lower bounds on matrix rigidity have applications to circuit and communication complexity [19].

The Nearest Codeword Problem is hard to approximate in general [4, 11] better than $n^{\Omega(1/\log\log n)}$. It is hard even if each equation has exactly 3 variables and each variable appears in exactly 3 equations [9]. There is a $O(n/\log n)$ approximation algorithm [8, 3].

Over a decade ago two groups [6, 13] independently discovered polynomial-time approximation algorithms for MAX-CUT achieving additive error of ϵn^2, implying a PTAS for average-dense MAX-CUT instances. The fastest algorithms [2, 20] 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 [2, 20, 22]. These results extend to MAX-BISECTION [14].

Arora, Karger and Karpinski [6] introduced the first PTASs for dense *minimum* constraint satisfaction problems. They give PTASs with runtime $n^{O(1/\epsilon^2)}$ [6] for min bisection and multiway cut (MIN-d-CUT). Bazgan, Fernandez de la Vega and Karpinski [7] designed PTASs for MIN-SAT and the nearest codeword problem with runtime $n^{O(1/\epsilon^2)}$. Giotis and Guruswami [17] give a PTAS for correlation clustering with d clusters with runtime $O(n^{9^d/\epsilon^2})$. We give linear-time approximation schemes for all of the problems mentioned in this paragraph except for the MIN-BISECTION problem.

2 Fragile-dense Algorithm

2.1 Intuition

Consider the following scenario. Suppose that our nemesis, who knows the optimal solution to the Gale-Berlekamp problem shown in Figure 1, 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 1. 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.

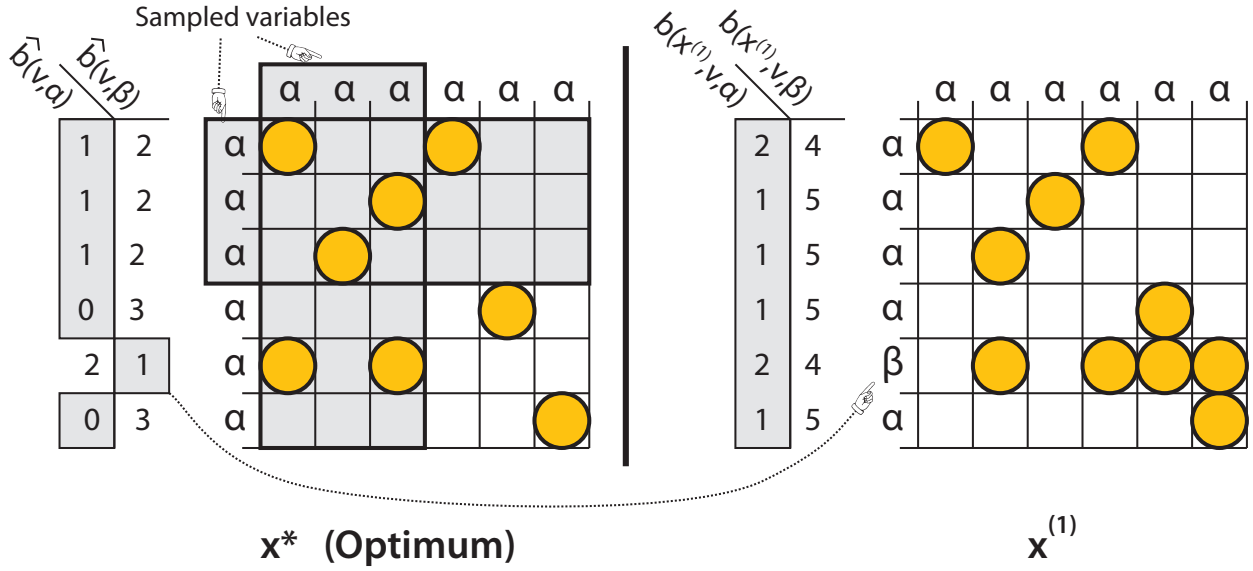

Figure 1: An illustration of our algorithmic ideas on the Gale-Berlekamp Game.

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 [7, 6, 17] 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. In the remainder of the paper we do not explicitly discuss the GB Game but present our ideas in the abstract framework of fragile-dense CSPs.

2.2 Model

We now give a formulation of MIN-kCSP that is suitable for our purposes. For non-negative integers n, k, let $\binom{n}{k} = \frac{n!}{k!(n-k)!}$, and for a given set V let $\binom{V}{k}$ denote the set of subsets of V of size k (analogous to 2^S for all subsets of S). There is a set V of n variables, each of which can take any value in constant-sized domain D. Let $x_v \in D$ denote the value of variable v in the assignment x.

Consider some $I \in \binom{V}{k}$. There may be many constraints over these variables; number them arbitrarily. Define $p(I, \ell, x)$ to be 1 if the ℓth constraint over I is unsatisfied in assignment x and zero otherwise. For $I \in \binom{V}{k}$, we define $p_I(x) = \frac{1}{\eta} \sum_\ell p(I, \ell, x)$, where η is a scaling factor to ensure $0 \leq p_I(x) \leq 1$ (e.g. $\eta = 2^k$ for MIN-kSAT). For notational simplicity we write p_I as a function

of a complete assignment, but $p_I(x)$ only depends on x_u for variables $u \in I$. For $I \notin \binom{V}{k}$ define $p_I(x) = 0$.

Definition 4. *On input V, p a* minimum constraint satisfaction problem *(MIN-kCSP) is a problem of finding an assignment x minimizing $Obj(x) = \sum_{I \in \binom{V}{k}} p_I(x)$.*

Let $R_{vi}(x)$ be an assignment over the variables V that agrees with x for all $u \in V$ except for v where it is i; i.e. $R_{vi}(x)_u = \begin{cases} i & \text{if } u = v \\ x_u & \text{otherwise} \end{cases}$. We will frequently use the identity $R_{vx_v}(x) = x$. Let $b(x, v, i) = \sum_{I \in \binom{V}{k} : v \in I} p_I(R_{vi}(x))$ be the number of unsatisfied constraints v would be in if x_v were set to i (divided by η).

We say the ℓth constraint over I is *fragile* if $p(I, \ell, R_{vi}(x)) + p(I, \ell, (R_{vj}(x)) \geq 1$ for all $v \in I$ and $i \neq j \in D$.

Definition 5. *A Min-kCSP is* fragile-dense *if $b(x, v, i) + b(x, v, j) \geq \delta\binom{n}{k-1}$ for some constant $\delta > 0$ and for all assignments x, variables v and distinct values i and j.*

Lemma 6. *An instance where every variable $v \in V$ participates in at least $\delta\eta\binom{n}{k-1}$ fragile constraints for some constant $\delta > 0$ is fragile-dense (with the same δ).*

Proof. By definitions:

$$\begin{aligned} b(x, v, i) + b(x, v, j) &= \sum_{I \in \binom{V}{k} : v \in I} (p_I(R_{vi}(x)) + p_I(R_{vj}(x))) \\ &= \sum_{I \in \binom{V}{k} : v \in I} \frac{1}{\eta} \sum_{\ell} (p(I, \ell, R_{vi}(x)) + p(I, \ell, R_{vj}(x))) \\ &\geq \sum_{I \in \binom{V}{k} : v \in I} \frac{1}{\eta} \cdot (\text{The number of fragile constraints over } I) \\ &\geq \frac{\delta\eta}{\eta}\binom{n}{k-1} = \delta\binom{n}{k-1} \end{aligned}$$

□

We will make no further mention of individual constraints, η or fragility; our algorithms and analysis use p_I and the fragile-dense property exclusively.

2.3 Algorithm

We now describe our linear-time algorithms. The main ingredients of the algorithm are new iterative applications of additive error algorithms and a special greedy technique for refining random samples of constant size.

Let $s = \frac{18 \log(480|D|k/\delta)}{\delta^2}$ and $S_1, S_2, \ldots, S_s$ be a multiset of independent random samples of $k-1$ variables from V. One can estimate $b(x^*, v, i)$ using the unbiased estimator $\hat{b}(v, i) = \frac{\binom{n}{k-1}}{s} \sum_{j=1}^{s} p_{S_j \cup \{v\}}(R_{vi}(\hat{x^*}))$ (see Lemma 13 for proof). One can determine the necessary x_v^* by exhaustively trying each possible combination.

Algorithm 1 Our algorithm for dense-fragile MIN-kCSP

Run a $\frac{\epsilon}{1+\epsilon}\delta^2/72k\binom{n}{k}$ additive approximation algorithm.
if $Obj(answer) \geq \binom{n}{k}\delta^2/72k$ **then**
  Return $answer$.
else
  Let $s = \frac{18\log(480|D|k/\delta)}{\delta^2}$
  Draw $S_1, S_2, \ldots, S_s$ randomly from $\binom{V}{k-1}$ with replacement.
  for Each assignment $\hat{x^*}$ of the variables in $\bigcup_{j=1}^{s} S_j$ **do**
    For all v and i let $\hat{b}(v,i) = \frac{\binom{n}{k-1}}{s}\sum_{j=1}^{s} p_{S_j\cup\{v\}}(R_{vi}(\hat{x^*}))$
    For all $v \in V$ let $x_v^{(1)} = \arg\min_i \hat{b}(v,i)$
    For all $v \in V$ let $x_v^{(2)} = \arg\min_i b(x^{(1)}, v, i)$
    Let $C = \{v \in V \ : \ b(x^{(1)}, v, x_v^{(2)}) < b(x^{(1)}, v, j) - \delta\binom{n}{k-1}/6 \text{ for all } j \neq x_v^{(2)}\}$.
    Find $x^{(3)}$ of cost at most $\frac{\epsilon|V\setminus C|\delta}{3n}\binom{n}{k-1} + \min[Obj(x)]$ using an additive approximation algorithm, where the minimum ranges over x such that $x_v = x_v^{(2)}\ \forall v \in C$.
  end for
  Return the best assignment $x^{(3)}$ found.
end if

3 Analysis of Algorithm 1

We use one of the known additive error approximation algorithms for MAX-kCSP problems.

Theorem 7. *[20] For any MAX-kCSP (or MIN-kCSP) and any $\epsilon' > 0$ there is a randomized algorithm which returns an assignment of cost at most $OPT + \epsilon' n^k$ in runtime $O(n^k) + 2^{O(1/\epsilon'^2)}$.*

Throughout the rest of the paper let x^* denote an optimal assignment.

First consider Algorithm 1 when the "then" branch of the "if" is taken. Choose constants appropriately so that the additive error algorithm fails with probability at most 1/10 and assume it succeeds. Let x^a denote the additive-error solution. We know $Obj(x^a) \leq Obj(x^*) + \frac{\epsilon}{1+\epsilon}P$ and $Obj(x^a) \geq P$ where $P = \binom{n}{k}\delta^2/72k$. Therefore $Obj(x^*) \geq P(1 - \frac{\epsilon}{1+\epsilon}) = \frac{P}{1+\epsilon}$ and hence $Obj(x^a) \leq Obj(x^*) + \frac{\epsilon}{1+\epsilon}(1+\epsilon)Obj(x^*) = (1+\epsilon)Obj(x^*)$. Therefore if the additive approximation is returned it is a $1+\epsilon$-approximation.

The remainder of this section considers the case when Algorithm 1 takes the "else" branch. Define γ so that $Obj(x^*) = \gamma\binom{n}{k}$. We have $Obj(x^*) \leq Obj(x^a) < \binom{n}{k}\delta^2/72k$ so $\gamma \leq \delta^2/72k$. We analyze the $\hat{x^*}$ where we guess x^*, that is when $\hat{x_v^*} = x_v^*$ for all $v \in \bigcup_{i=1}^{s} S_i$. Clearly the overall cost at most the cost of $x^{(3)}$ during the iteration when we guess correctly.

Lemma 8. *$b(x^*, v, x_v^*) \leq b(x^*, v, j)$ for all $j \in D$.*

Proof. Immediate from definition of b and optimality of x^*. □

Lemma 9. *For any assignment x,*

$$Obj(x) = \frac{1}{k}\sum_{v\in V} b(x, v, x_v)$$

Proof. By definitions,

$$b(x, v, x_v) = \sum_{I \in \binom{V}{k} : v \in I} p_I(R_{vx_v}(x)) = \sum_{I \in \binom{V}{k} : v \in I} p_I(x).$$

Write $Obj(x) = \sum_{I \in \binom{V}{k}} p_I(x) = \sum_{I \in \binom{V}{k}} p_I(x) \left[\sum_{v \in I} \frac{1}{k}\right]$ and reorder summations. □

Definition 10. *We say variable v in assignment x is* corrupted *if $x_v \neq x^*_v$.*

Definition 11. *Variable v is* clear *if $(x^*, v, x^*_v) < b(x^*, v, j) - \frac{\delta}{3}\binom{n}{k-1}$ for all $j \neq x^*_v$. A variable is* unclear *if it is not clear.*

Clearness is the analysis analog of the algorithmic notion of clear-cut vertices sketched in Section 2.1. Comparing the definition of clearness to Lemma 8 further motivates the terminology "clear."

Lemma 12. *The number of unclear variables t satisfies*

$$t \leq 3(n - k + 1)\gamma/\delta) \leq \frac{\delta n}{24k}.$$

Proof. Let v be unclear and choose $j \neq x^*_v$ minimizing $b(x^*, v, j)$. By unclearness, $b(x^*, v, x^*_v) \geq b(x^*, v, j) - (1/3)\delta\binom{n}{k-1}$. By fragile-dense, $b(x^*, v, x^*_v) + b(x^*, v, j) \geq \delta\binom{n}{k-1}$. Adding these inequalities we see

$$b(x^*, v, x^*_v) \quad \geq \quad \frac{1 - 1/3}{2}\delta\binom{n}{k-1} = \frac{1}{3}\delta\binom{n}{k-1} \tag{1}$$

By Lemma 9 and (1),

$$OPT = \gamma\binom{n}{k} = 1/k \sum_v b(x^*, v, x^*_v) \geq 1/k \sum_{v:unclear} \frac{\delta}{3}\binom{n}{k-1} = \frac{\delta}{3k}\binom{n}{k-1}t.$$

Therefore $t \leq \gamma\binom{n}{k}\frac{3k}{\delta\binom{n}{k-1}} = \frac{3\gamma}{\delta}(n - k + 1)$.

For the second bound observe $3n\gamma/\delta \leq \frac{3n}{\delta}\frac{\delta^2}{72k} = \frac{\delta n}{24k}$. □

Lemma 13. *The probability of a fixed clear variable v being corrupted in $x^{(1)}$ is bounded above by $\frac{\delta}{240k}$.*

Proof. First we show that $\hat{b}(v, i)$ is in fact an unbiased estimator of $b(x^*, v, i)$ for all i. By definitions and particular by the assumption that $p_I = 0$ when $|I| < k$, we have for any $1 \leq j \leq s$:

$$\begin{aligned}
\mathbf{E}\left[p_{S_j \cup \{v\}}(R_{vi}(x^*))\right] &= \frac{1}{\binom{n}{k-1}} \sum_{J \in \binom{V}{k-1}} p_{J \cup \{v\}}(R_{vi}(x^*)) \\
&= \frac{1}{\binom{n}{k-1}} \sum_{I \in \binom{V}{k} : v \in I} p_I(R_{vi}(x^*)) \\
&= \frac{1}{\binom{n}{k-1}} b_{vi}(x^*)
\end{aligned}$$

Therefore $\mathbf{E}\left[\hat{b}(v, i)\right] = s\frac{\binom{n}{k-1}}{s}\mathbf{E}\left[p_{S_1 \cup \{v\}}(R_{vi}(x^*))\right] = b(x^*, v, i)$.

Recall that $0 \le p_I(x) \le 1$ by definition of p, so by Azuma-Hoeffding,

$$\Pr\left[\left|\sum_{j=1}^{s} p_{S_j\cup\{v\}}(R_{vi}(x^*)) - \frac{s}{\binom{n}{k-1}} b(x^*, v, i)\right| \ge \lambda s\right] \;\le\; 2e^{-2\lambda^2 s}$$

hence

$$\Pr\left[|\hat{b}(v,i) - b(x^*, v, i)| \ge \lambda \binom{n}{k-1}\right] \;\le\; 2e^{-2\lambda^2 s}$$

Choose $\lambda = \delta/6$ and recall $s = \frac{18\log(480|D|k/\delta)}{\delta^2}$, yielding.

$$\Pr\left[|\hat{b}(v,i) - b(x^*, v, i)| \ge \frac{\delta}{6}\binom{n}{k-1}\right] \;\le\; \frac{\delta}{240|D|k}$$

By clearness we have $b(x^*, v, j) > b(x^*, v, x_v^*) + \delta\binom{n}{k-1}/3$ for all $j \ne x_v^*$. Therefore, the probability that $\hat{b}(v, x_v^*)$ is not the smallest $\hat{b}(v, j)$ is bounded by $|D|$ times the probability that a particular $\hat{b}(v, j)$ differs from its mean by at least $\delta\binom{n}{k-1}/6$. Therefore $\Pr\left[x_v^1 \ne x_v^*\right] \le |D|\frac{\delta}{240|D|k} = \frac{\delta}{240k}$. □

Let E_1 denote the event that the assignment $x^{(1)}$ has at most $\delta n/12k$ corrupted variables.

Lemma 14. *Event E_1 occurs with probability at least $1 - 1/10$.*

Proof. We consider the corrupted clear and unclear variables separately. By Lemma 12, the number of unclear variables, and hence the number of corrupted unclear variables, is bounded by $\frac{\delta n}{24k}$.

The expected number of clear corrupted variables can be bounded by $\frac{\delta n}{240k}$ using Lemma 13, so by Markov bound the number of clear corrupted variables is less than $\frac{\delta n}{24k}$ with probability at least 1 - 1/10.

Therefore the total number of corrupted variables is bounded by $\frac{\delta n}{24k} + \frac{\delta n}{24k} = \frac{\delta n}{12k}$ with probability at least 9/10. □

We henceforth assume E_1 occurs. The remainder of the analysis is deterministic.

Lemma 15. *For assignments y and y' that differ in the assignment of at most t variables, for all variables v and values i, $|b(y, v, i) - b(y', v, i)| \le t\binom{n}{k-2}$.*

Proof. Clearly $p_I(R_{vi}(y))$ is a function only of the variables in I excluding v, so if $I - \{v\}$ consists of variables u where $y_u = y'_u$, then $p_I(R_{vi}(y)) - p_I(R_{vi}(y')) = 0$. Therefore $b(y, v, i) - b(y', v, i)$ equals the sum, over $I \in \binom{V}{k}$ containing v and at least one variable u other than v where $y_u \ne y'_u$, of $[p_I(R_{vi}(y)) - p_I(R_{vi}(y'))]$. For any I, $|p_I(R_{vi}(y)) - p_I(R_{vi}(y'))| \le 1$, so by the triangle inequality a bound on the number of such sets suffices to bound $|b(y, v, i) - b(y', v, i)|$. The number of such sets can trivially be bounded above by $t\binom{n}{k-2}$. □

Lemma 16. *Let $C = \{v \in V \;:\; b(x^{(1)}, v, x_v^{(2)}) < b(x^{(1)}, v, j) - \delta\binom{n}{k-1}/6$ for all $j \ne x_v^{(2)}\}$ as defined in Algorithm 1. If E_1 then:*

- *$x_v^{(2)} = x_v^*$ for all $v \in C$.*
- *$|V \setminus C| \le \frac{3n\gamma}{\delta}$.*

Proof. Assume E_1 occurred. From the definition of corrupted, event E_1 and Lemma 15 for sufficiently large n (so that $\frac{n-k+1}{k-1} \geq \frac{n}{k}$) for any v, i:

$$|b(x^{(1)}, v, i) - b(x^*, v, i)| \leq \frac{\delta n}{12k}\binom{n}{k-2} \leq \frac{\delta}{12}\binom{n}{k-1}. \tag{2}$$

For the first, if $v \in C$ then using (2)

$$\begin{aligned} b(x^*, v, x_v^{(2)}) &\leq b(x^{(2)}, v, x_v^{(2)}) + \frac{\delta}{12}\binom{n}{k-1} < b(x^{(2)}, v, j) - \frac{\delta}{6}\binom{n}{k-1} + \frac{\delta}{12}\binom{n}{k-1} \\ &\leq b(x^*, v, j) + \left(-\frac{\delta}{6} + 2\frac{\delta}{12}\right)\binom{n}{k-1} = b(x^*, v, j). \end{aligned}$$

So by Lemma 8, $x_v^* = x_v^{(2)}$.

For any u that is clear, using (2) again:

$$\begin{aligned} b(x^{(2)}, v, x_v^*) &\leq b(x^*, v, x_v^*) + \frac{\delta}{12}\binom{n}{k-1} < b(x^*, v, j) - \frac{\delta}{3}\binom{n}{k-1} + \frac{\delta}{12}\binom{n}{k-1} \\ &\leq b(x^{(2)}, v, j) + \left(-\frac{\delta}{3} + 2\frac{\delta}{12}\right)\binom{n}{k-1} = b(x^{(2)}, v, j) - \frac{\delta}{6}\binom{n}{k-1}. \end{aligned}$$

so by definition of C, $u \in C$. Therefore the conclusion follows from Lemma 12. □

Now we give the details of the computation of $x^{(3)}$. Let $T = V \setminus C$. We call C the *clear-cut* vertices and T the *tricky* vertices. We assume that $|T| \geq k$; if not simply consider every possible assignment to the variables in T. With the variables in C fixed, those variables can be substituted into the p_I and eliminated. To restore a uniform arity of k we pad the p_I of arity less than k with irrelevant variables from T. To ensure none of the resulting p_I has excessive weight we use a uniform mixture of all possibilities for the padding vertices.

If y is an assignment to the variables in T let $R_{Ty}(x^*) = \begin{cases} y_v & \text{If } v \in T \\ x_v^* & \text{Otherwise} \end{cases}$, a natural generalization of the $R_{vi}(x)$ notation. For $K \in \binom{T}{k}$ and $y \in D^{|T|}$ define

$$q_K(y) = \sum_{j=1}^{k} \sum_{J \in \binom{K}{j}} \sum_{L \in \binom{C}{k-j}} p_{J \cup L}(R_{Ty}(x^{(2)}))\binom{|T|-j}{k-j}^{-1}$$

It is easy to see that $q_K(y)$ is a function only of y_v for $v \in K$ and is hence a cost function analogous to p_I (though not properly normalized).

Lemma 17. *For any $y \in D^{|T|}$ we have*

$$Obj(R_{Ty}(x^{(2)})) = \sum_{K \in \binom{T}{k}} q_K(y) + \sum_{I \in \binom{C}{k}} p_I(x^{(2)})$$

Proof. Let $x = R_{Ty}(x^{(2)})$. By definition

$$\sum_{K \in \binom{T}{k}} q_I(y) = \sum_{K \in \binom{T}{k}} \sum_{j=1}^{k} \sum_{J \in \binom{K}{j}} \sum_{L \in \binom{C}{k-j}} p_{J \cup L}(x)\binom{|T|-j}{k-j}^{-1} \tag{3}$$

Compare to

$$Obj(x) - \sum_{I \in \binom{C}{k}} p_I(x^{(2)}) \quad = \quad \sum_{I \in \binom{V}{k} : I \not\subseteq C} p_I(x) \tag{4}$$

Fix $I \in \binom{V}{k}$ and study the weight of $p_I(x)$ in the right-hand-sides of (3) and (4). Note there are unique $j \geq 0$, $J \in \binom{T}{j}$ and $L \in \binom{C}{k-j}$ such that $I = J \cup L$. If $j = 0$ then $p_I(x)$ has weight 0 in (3) and in (4). If $j \geq 1$ then $p_I(x)$ appears once in (3) for each $K \in \binom{T}{k}$ such that $K \supseteq J$. There are $\binom{|T|-j}{k-j}$ of those and each has weight $\binom{|T|-j}{k-j}^{-1}$ so $p_I(x)$ has an overall weight of 1 in (3). Clearly $j \geq 1$ implies $I \not\subseteq C$ hence the weight of $p_I(x)$ in (4) is 1 as well. □

Lemma 18.

$$0 \leq q_K(y) \leq O\left(\left(\frac{|C|}{|T|}\right)^{k-1}\right)$$

Proof. Recalling that $0 \leq p_I(y) \leq 1$ and $k = O(1)$:

$$q_K(y) \leq \sum_{j=1}^{k} \binom{k}{j} \binom{|C|}{k-j} \cdot 1 \cdot \binom{|T|-j}{k-j}^{-1} = \sum_{j=1}^{k} O\left(\frac{|C|^{k-j}}{|T|^{k-j}}\right) = O\left(\frac{|C|^{k-1}}{|T|^{k-1}}\right)$$

□

Lemma 18 and Theorem 7 with an error parameter of $\epsilon' = \Theta(\epsilon)$ yields an additive error of $O(\epsilon|T|^k(|C|/|T|)^{k-1}) = O(\epsilon(|T|/|C|)n^k)$ for the problem of minimizing $\sum_{K \in \binom{T}{k}} q_K(y)$. Using Lemma 16 we further bound the additive error $O(\epsilon(|T|/|C|)n^k)$ by $O(\epsilon\gamma n^k)$. By Lemma 17 this is also an additive error $O(\epsilon\gamma n^k)$ for $Obj(R_{Ty}(x^{(2)}))$. Lemma 16 implies that $x^* = R_{Ty}(x^{(2)})$ for some y, so this yields an additive error $O(\epsilon\gamma n^k) = \epsilon OPT$ for our original problem of minimizing $Obj(x)$ over all assignments x.

4 Correlation Clustering and Hierarchical Clustering Algorithm

4.1 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 not connected by an 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 [17] that allows for the construction of a PTAS. Consider the cost-zero clustering shown on the left of Figure 2. 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 [17], we can postpone processing the vertices in small clusters. We use the above ideas, which are due to [17], the fragile-dense ideas sketched above, plus some additional ideas, to analyze our correlation clustering algorithm.

To handle hierarchical clustering (c.f. [1]) we need a few more ideas. Firstly we abstract the arguments of the previous paragraph to a CSP property *rigidity*. Secondly, we note that the number of trees with d leaves is a constant and therefore we can safely try them all. We remark that all fragile-dense problems are also rigid.

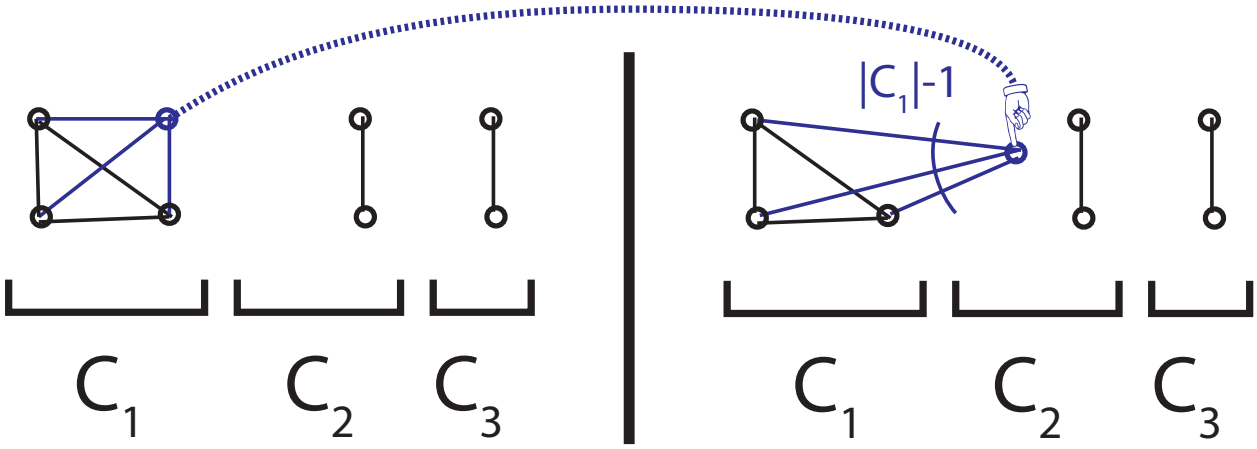

Figure 2: An illustration of correlation clustering and the rigidity property.

4.2 Reduction to Rigid MIN-2CSP

We now define hierarchical clustering formally (following [1]). For integer $M \geq 1$, an *M-level hierarchical clustering* of n objects V is a rooted tree with the elements of V as the leaves and every leaf at depth (distance to root) exactly $M+1$. For $M = 1$, a hierarchical clustering has one node at the root, some "cluster" nodes in the middle level and all of X in the bottom level. The nodes in the middle level can be identified with clusters of V. We call the subtree induced by the internal nodes of a M-level hierarchical clustering the *trunk*. We call the leaves of the trunk *clusters*. A hierarchical clustering is completely specified by its trunk and the parent cluster of each leaf.

For a fixed hierarchical clustering and clusters i and j, let $f(i,j)$ be the distance from i (or j) to the lowest common ancestor of i and j. For example when $M = 1$, $f(i,j) = \mathbb{1}(i = j)$.

We are given a function F from pairs of vertices to $\{0, 1, ...M\}$.* The objective of hierarchical clustering is to output a M-level hierarchical clustering minimizing $\sum_{u,v} \frac{1}{M} |F(u,v) - f(parent(u), parent(v))|$. Hierarchical clustering with d clusters is the same except that we restrict the number of clusters (recall that equals number of nodes whose children are leaves) to at most d. The special case of hierarchical clustering with $M = 1$ is also called *correlation clustering*.

Lemma 19. *The number of possible trunks is at most $d^{(M-1)d}$.*

Proof. The trunk can be specified by giving the parent of all non-root nodes. There are at most d nodes on each of the $M-1$ non-root levels so the lemma follows. □

We now show how to reduce hierarchical clustering with a constant number of clusters to the solution of a constant number of min-2CSPs. We use notation similar to, but not identical to, the notation used in Sections 2 and 3. 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.

*[1] chose $\{1, 2, , ...M+1\}$ instead; the difference is merely notational.

Definition 20. *A MIN-2CSP is* rigid *if for some $\delta > 0$, all $v \in V$ and all $j \neq x^*_v$*

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

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

Lemma 21. *If the trunk is fixed, hierarchical clustering can be expressed as a $1/M$-rigid MIN-2CSP with $|D| = d$.*

Proof. (C.f. Figure 2) Choose $\delta = 1/M$. Let D be the leaves of the trunk (clusters). It is easy to see that choosing

$$p_{u,v}(i,j) = \frac{1}{M}|f(i,j) - F(u,v)|$$

yields the correct objective function. To show rigidity, fix vertex v, define $i = x^*_v$ and $\mathcal{C}_i = \{u \in V \ : \ x^*_u = i\}$. Fix $j \neq i$ and $u \in \mathcal{C}_i \setminus \{v\}$. Clearly $|f(i,i) - f(i,j)| \geq 1$, hence by triangle inequality $|F(u,v) - f(i,i)| + |F(u,v) - f(i,j)| \geq 1$, hence $p_{u,v}(i,i) + p_{u,v}(i,j) \geq 1/M$. Summing over $u \in \mathcal{C}_i$ we see

$$b(x^*, v, x^*_v) + b(x^*, v, j) \geq \frac{1}{M}|\mathcal{C}_i \setminus \{v\}| \approx \frac{1}{M}|\mathcal{C}_i| = \delta |\{u \in V \ : \ x^*_u = x^*_v\}|$$

Sweeping the "$\approx$" under the rug this proves the Lemma.[†] □

Lemmas 21 and 19 suggest a technique for solving hierarchical clustering: guess the trunk and then solve the rigid MIN-2CSP. We now give our algorithm for solving rigid MIN-2CSPs.

4.3 Algorithm for Rigid MIN-2CSP

Algorithm 2 solves rigid MIN-2CSPs by identifying clear-cut variables, fixing their value, and then recursing on the remaining "tricky" variables T. The recursion terminates when the remaining subproblem is sufficiently expensive for an additive approximation to suffice.

5 Analysis of Algorithm 2

5.1 Runtime

Theorem 22. *For any T, y, an assignment of cost at most $\epsilon'|T|^2 + \min_{x:x_v=y_v \forall v \in V \setminus T} [Obj(x)]$ can be found in time $n^2 2^{O(1/\epsilon'^2)}$.*

Proof. The problem is essentially a CSP on T vertices but with an additional linear cost term for each vertex. It is fairly easy to see that Algorithm 1 from Mathieu and Schudy [20] has error proportional to the misestimation of b and hence is unaffected by arbitrarily large linear cost terms. On the other hand, the more efficient Algorithm 2 from [20] needs to estimate the objective value from a constant-sized sample as well and hence does not seem to work for this type of problem. □

In this subsection $O(\cdot)$ hides only absolute constants. Algorithm 2 has recursion depth at most $|D| + 1$ and branching factor $|D|^s$, so the number of recursive calls is at most $(|D|^s)^{|D|+1} = 2^{s(|D|+1)\log|D|} = 2^{\tilde{O}(|D|^5/\delta^4)}$. Each call spends $O(|D|n^2)$ time on miscellaneous tasks such as computing the objective value plus time required to run the additive error algorithm, which is $n^2 2^{O(|D|^6/\epsilon^2\delta^6)}$

[†]There are inelegant ways to remove this approximation. For example, assume that all d clusters of x^* are non-empty and consider one vertex from $\mathcal{C}_j$ as well.

Algorithm 2 Approximation Algorithm for Rigid MIN-2CSPs.

Return CC(V, blank assignment, 0)

CC(tricky vertices T, assignment y of $V \setminus T$, recursion depth *depth*):

1: Find assignment of cost at most $\frac{\epsilon}{1+\epsilon} \cdot \frac{\delta^3|T|^2}{6 \cdot 72^2 |D|^3} + \min_{x: x_v = y_v \forall v \in V \setminus T} [Obj(x)]$ using an additive approximation algorithm.
2: **if** $Obj(answer) \geq \frac{\delta^3|T|^2}{6 \cdot 72^2 |D|^3}$ or $depth \geq |D| + 1$ **then**
3: Return *answer*.
4: **else**
5: Let $s = \frac{432^2 |D|^4 \log(1440|D|^3/\delta)}{2\delta^4}$
6: Draw $v_1, v_2, \ldots, v_s$ randomly from T with replacement.
7: **for** Each assignment $\hat{x^*}$ of the variables $\{v_1, v_2, \ldots, v_s\}$ **do**
8: For all $v \in T$ and i let $\hat{b}(v,i) = \frac{|T|}{s} \sum_{j=1}^{s} p_{v_j,v}(\hat{x^*_{v_j}}, i) + \sum_{u \in V \setminus T} p_{u,v}(y_u, i)$
9: For all $v \in V$ let $x_v^{(1)} = \begin{cases} y_v & \text{If } v \in V \setminus T \\ \arg\min_i \hat{b}(v,i) & \text{Otherwise} \end{cases}$
10: For all $v \in T$ let $x_v^{(2)} = \arg\min_i b(x^{(1)}, v, i)$
11: Let $C = \{v \in T \ : \ b(x^{(1)}, v, x_v^{(2)}) < b(x^{(1)}, v, j) - \frac{\delta|T|}{12|D|} \text{ for all } j \neq x_v^{(2)}\}$.
12: Let $T' = T \setminus C$
13: Define assignment y' by $y'_v = \begin{cases} y_v & \text{If } v \in V \setminus T \\ x_v^{(2)} & \text{If } v \in C \\ \text{Undefined} & \text{If } v \in T \setminus C \end{cases}$.
14: If CC($T', y', depth + 1$) is the best clustering so far, update best.
15: **end for**
16: Return the best clustering found.
17: **end if**

by Theorem 22. Therefore the runtime of Algorithm 2 is $n^2 2^{O(\frac{|D|^6}{\epsilon^2\delta^6})}$, where the $2^{\tilde{O}(|D|^5/\delta^4)}$ from the size of the recursion tree got absorbed into the $2^{O(\frac{|D|^6}{\epsilon^2\delta^6})}$ from Theorem 22. For hierarchical clustering, $\delta = 1/M$ yields a runtime of $n^2 2^{O(\frac{|D|^6 M^6}{\epsilon^2})} \cdot |D|^{(M-1)|D|} = n^2 2^{O(\frac{|D|^6 M^6}{\epsilon^2})}$.

As noted in the introduction this improves on the runtime of $n^{O\left(\frac{9|D|}{\epsilon^2}\right)}$ of [17] for correlation clustering in two ways: the degree of the polynomial is independent of ϵ and $|D|$, and the dependence on $|D|$ is singly rather than doubly exponential.

5.2 Approximation

We fix optimal assignment x^*. We analyze the path through the recursion tree where we always guess $\hat{x}^*$ correctly, i.e. $\hat{x}^*_v = x^*_v$ for all $v \in \{v_1, v_2, \ldots, v_s\}$. We call this the *principal path*.

We will need the following definitions.

Definition 23. *Vertex v is m-*clear *if $b(x^*, v, x^*_v) < b(x^*, v, j) - m$ for all $j \neq x^*_v$. We say a vertex is* clear *if it is m-clear for m obvious from context. A vertex is* unclear *if it is not clear.*

Definition 24. *A vertex is* obvious *if it is in cluster $\mathcal{C}$ in OPT and it is $\delta|\mathcal{C}|/3$-clear.*

Definition 25. *A cluster $\mathcal{C}$ of OPT is* finished *w.r.t. T if $T \cap \mathcal{C}$ contains no obvious vertices.*

Lemma 26. *With probability at least 8/10, for any $(T, y, depth)$ encountered on the principle path,*

1. *$y_v = x^*_v$ for all $v \in V \setminus T$ and*
2. *The number of finished clusters w.r.t. T is at least depth.*

Before proving Lemma 26 let us see why it implies Algorithm 2 has the correct approximation factor.

Proof. Study the final call on the principal path, which returns the additive approximation clustering. The second part of Lemma 26 implies that $depth \leq |D|$, hence we must have terminated because $Obj(answer) \geq \frac{\delta^3|T|^2}{6 \cdot 72^2 |D|^3}$. By the first part of Lemma 26 the additive approximation gives error at most

$$\frac{\epsilon}{1+\epsilon} \cdot \frac{\delta^3|T|^2}{6 \cdot 72^2|D|^3} + OPT.$$

so the approximation factor follows from an easy calculation. □

Now we prove Lemma 26 by induction. Our base case is the *root*, which vacuously satisfies the inductive hypothesis since $V \setminus T = \{\}$ and $depth = 0$. We show that if a node $(T, y, depth)$ (in the recursion tree) satisfies the invariant then its *child* $(T', y', depth + 1)$ does as well. We hereafter analyze a particular $(T, y, depth)$ and assume the inductive hypothesis holds for them. There is only something to prove if a child exists, so we hereafter assume the additive error answer is *not* returned from this node. We now prove a number of Lemmas in this context, from which the fact that $T', y', depth + 1$ satisfies the inductive hypothesis will trivially follow.

Lemma 27. *The number of $\delta^2|T|/216D^2$-clear variables that are corrupted in $x^{(1)}$ is at most $\delta|T|/72|D|$ with probability at least $1 - 1/10|D|$.*

Proof. Essentially the same proof as for fragile MIN-kCSP, and the recursion invariant, shows $\hat{b}(v, i)$ is an unbiased estimator of $b(x^*, v, i)$.

This time Azuma-Hoeffding yields

$$\Pr\left[|\hat{b}(v,i) - b(x^*, v, i)| \geq \lambda|T|\right] \quad \leq \quad 2e^{-2\lambda^2 s}$$

Choose $\lambda = \frac{\delta^2}{432|D|^2}$ and recall $s = \frac{432^2|D|^4 \log(1440|D|^3/\delta)}{2\delta^4}$, yielding.

$$\Pr\left[|\hat{b}(v,i) - b(x^*, v, i)| \geq \delta^2|T|/432|D|^2\right] \quad \leq \quad \frac{\delta}{720|D|^3}$$

By clearness we have $b(x^*, v, j) > b(x^*, v, x^*_v) + \delta^2|T|/216|D|^2$ for all $j \neq x^*_v$. Therefore, the probability that $\hat{b}(v, x_v)$ is not the smallest $\hat{b}(v, j)$ is bounded by $|D|$ times the probability that a particular $\hat{b}(v, j)$ differs from its mean by at least $\delta^2|T|/432|D|^2$. Therefore $\Pr\left[x^{(1)}_v \neq x^*_v\right] \leq |D|\frac{\delta}{720|D|^3} = \frac{\delta}{720|D|^2}$. Therefore, by Markov bound, with probability $1 - 1/10|D|$ the number of corrupted $\delta^2|T|/216D^2$-clear variables is at most $\delta|T|/72|D|$. □

There are two types of bad events: the additive error algorithm failing and our own random samples failing. We choose constants so that each of these events has probability at most $1/10|D|$. This path has length at most $|D|$, so the overall probability of a bad event is at most $2/10$. We hereafter assume no bad events occur.

Lemma 28. *The number of $\delta c/3$-unclear variables in clusters of size at least c is at most $\frac{6OPT}{\delta c}$.*

Let *confusing* variable refer to a $\delta c/3$-unclear variable in a cluster of size at least c. Let v be such a variable, in cluster $\mathcal{C}$ in OPT. By unclearness,

Proof.

$$b(x^*, v, x_v^*) \geq b(x^*, v, j) - \delta c/3$$

for appropriate $j \neq x_v^*$ and by rigidity

$$b(x^*, v, x_v^*) + b(x^*, v, j) \geq \delta|\mathcal{C}|.$$

Adding these inequalities we see $b(x^*, v, x_v^*) \geq \delta c/3$.

$OPT = 1/2 \sum_v b(x^*, v, x_v^*) \geq 1/2 \sum_{v \text{ confusing}} \delta c/3 = |\{v \in T \ : \ v \text{ confusing}\}|\delta c/6$ so $|\{v \in T \ : \ v \text{ confusing}\}| \leq \frac{6OPT}{\delta c}$. □

Lemma 29. *For all v, i, $|b(x^{(1)}, v, i) - b(x^*, v, i)| \leq \frac{\delta|T|}{24|D|}$*

Proof. First we show bounds on three classes of corrupted variables:

1. The number of $\delta^2|T|/216D^2$-clear corrupted vertices is bounded by $\delta|T|/72|D|$ using Lemma 27
2. The number of vertices in clusters of size at most $\delta|T|/72|D|^2$ is bounded by $\delta|T|/72|D|$.
3. The number of $\delta^2|T|/216D^2$-unclear corrupted vertices in clusters of size at least $\delta|T|/72|D|^2$ is bounded by, using Lemma 28, $\frac{6OPT}{\delta}\frac{72|D|^2}{\delta|T|} \leq \frac{\delta^3|T|^2}{6\cdot 72^2|D|^3} \cdot \frac{6\cdot 72|D|^2}{\delta^2|T|} = \frac{\delta|T|}{72|D|}$.

Therefore the total number of corrupted variables in $x^{(1)}$ is at most $\frac{\delta|T|}{72|D|} + \frac{\delta|T|}{72|D|} + \frac{\delta|T|}{72|D|} = \frac{\delta|T|}{24|D|}$. The easy observation that $|b(x^{(1)}, v, i) - b(x^*, v, i)|$ is bounded by the number of corrupted variables in $x^{(1)}$ proves the Lemma. □

Lemma 30. *There exists an obvious vertex in T that is in a cluster of size at least $|T|/2|D|$.*

Proof. Simple counting shows there are at most $|T|/2$ vertices of T in clusters of size less than $|T|/2|D|$.

We say a vertex v is *confusing'* if it is non-obvious and its cluster in OPT has size at least $|T|/2|D|$. By Lemma 28

$$|\{v \in T \ : \ v \text{ confusing'}\}| \leq \frac{12|D|}{\delta|T|}OPT \leq \frac{12|D|}{\delta|T|}\frac{\delta^3|T|^2}{6\cdot 72^2|D|^3} < |T|/2$$

Therefore by counting there must be an obvious vertex in a big cluster of OPT. □

Lemma 31. *The number of finished clusters w.r.t. T' strictly exceeds the number of finished clusters w.r.t. T.*

Proof. Let v be the vertex promised by Lemma 30 and $\mathcal{C}_i$ its cluster in OPT. For any obvious vertex u in $\mathcal{C}_i$ note that u is $\delta|\mathcal{C}_i|/3 \geq \delta|T|/6|D|$-clear, so Lemma 29 implies

$$\begin{aligned} b(x^{(1)}, u, i) &\leq b(x^*, u, i) + \frac{\delta|T|}{24|D|} < b(x^*, u, j) + \frac{\delta|T|}{24|D|} - \frac{\delta|T|}{6|D|} \\ &\leq b(x^{(1)}, u, j) + 2\frac{\delta|T|}{24|D|} - \frac{\delta|T|}{6|D|} = b(x^{(1)}, u, j) - \frac{\delta|T|}{12|D|} \end{aligned}$$

hence $u \in C$. Therefore, no obvious vertices in $\mathcal{C}_i$ are in T' so $\mathcal{C}_i$ is finished w.r.t. T'. The existence of v implies $\mathcal{C}_i$ is not finished w.r.t. T, so $\mathcal{C}_i$ is newly finished. To complete the proof note that $T' \subseteq T$ so finished is a monotonic property. □

Lemma 32. *(T', y') satisfy the invariant $v \in V \setminus T' \to y'_v = x^*_v$.*

Proof. Fix $v \in V \setminus T'$. If $v \in T$ the conclusion follows from the invariant for (T, y). If $v \in T \setminus T' = C$ we need to show $y'_v = x^*_v$.

Let $i = y'_v$. For any $j \neq i$, use Lemma 29 to obtain

$$b(x^*, v, i) \leq b(x^{(1)}, v, i) + \frac{\delta|T|}{24|D|} < b(x^{(1)}, v, j) + \frac{\delta|T|}{24|D|} - \frac{\delta|T|}{12|D|} \leq b(x^*, v, j) + 2\frac{\delta|T|}{24|D|} - \frac{\delta|T|}{12|D|} = b(x^*, v, j)$$

so by optimality of x^* we have the Lemma. □

Lemmas 31 and 32 complete the inductive proof of Lemma 26.

Acknowledgements

We would like to thank Claire Mathieu and Joel Spencer for raising a question on approximability status of the Gale-Berlekamp game and Alex Samorodintsky for interesting discussions.

References

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

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

[3] N. Alon, R. Panigrahy, and S. Yekhanin. Deterministic Approximation Algorithms for the Nearest Codeword Problem. Technical report, Elec. Coll. on Comp. Compl., ECCC TR08-065, 2008.

[4] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The Hardness of Approximate Optima in Lattices, Codes, and Systems of Linear Equations. In *Foundations of Computer Science*, pages 724–733, Nov 1993.

[5] S. Arora, A. Frieze, and H. Kaplan. A New Rounding Procedure for the Assignment Problem with Applications to Dense Graph Arrangement Problems. In *Foundations of Computer Science*, pages 21–30, Oct 1996.

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

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

[8] P. Berman and M. Karpinski. Approximating Minimum Unsatisfiability of Linear Equations. In *Procs. 13th ACM-SIAM SODA*, pages 514–516, 2002.

[9] P. Berman and M. Karpinski. Approximation Hardness of Bounded Degree MIN-CSP and MIN-BISECTION. In *Procs. 29th ICALP, LNCS 2380*, pages 623–632. Springer, 2002.

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

[11] I. Dinur, G. Kindler, R. Raz, and S. Safra. Approximating CVP to Within Almost-Polynomial Factors is NP-Hard. *Combinatorica*, 23(2):205–243, 2003.

[12] U. Feige and L. Lovasz. Two prover one round proof systems: Their power and their problems. In *Procs. 24th STOC*, pages 733–741, 1992.

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

[14] 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.

[15] P. C. Fishburn and N. J. Sloane. The Solution to Berlekamp's Switching Game. *Discrete Math.*, 74(3):263–290, 1989.

[16] A. M. Frieze and R. Kannan. Quick Approximation to Matrices and Applications. *Combinatorica*, 19(2):175–220, 1999.

[17] I. Giotis and V. Guruswami. Correlation clustering with a fixed number of clusters. *Theory of Computing*, 2(1):249–266, 2006.

[18] A. Gupta and K. Talvar. Approximating unique games. In *Procs. 17th ACM-SIAM SODA*, pages 99–106, 2006.

[19] S. V. Lokam. Spectral Methods for Matrix Rigidity with Applications to Size-Depth Tradeoffs and Communication Complexity. In *36th IEEE FOCS*, pages 6–15, 1995.

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

[21] 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.

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

[23] J. Spencer. *Ten Lectures on the Probabilistic Method*. SIAM, second edition, 1994.