

An Efficient Sparse Regularity Concept

Amin Coja-Oghlan*

Colin Cooper†

Alan Frieze‡

Abstract

Let \mathbf{A} be a 0/1 matrix of size $m \times n$, and let p be the density of \mathbf{A} (i.e., the number of ones divided by $m \cdot n$). We show that \mathbf{A} can be approximated in the cut norm within $\varepsilon \cdot mnp$ by a sum of cut matrices (of rank 1), where the number of summands is independent of the size $m \cdot n$ of \mathbf{A} , provided that \mathbf{A} satisfies a certain boundedness condition. The decomposition can be computed in polynomial time. This result extends the work of Frieze and Kannan (Combinatorica 1999) to *sparse* matrices. As an application, we obtain efficient $1 - \varepsilon$ approximation algorithms for “bounded” instances of MAX CSP problems.

1 Introduction and results.

For many fundamental optimization problems there are *NP-hardness of approximation* results known, showing that not only is it NP-hard to compute the optimum exactly, but even to approximate the optimum within a factor bounded away from 1. For instance, in the MAX k -SAT problem it is NP-hard to achieve an approximation ratio better than $1 - 2^{-k}$ [18]. Furthermore, it is NP-hard to approximate MAX CUT within better than $16/17 \approx 0.94118$ [18, 23] (which can be tightened to ≈ 0.87856 under a stronger hypothesis [19]).

Frieze and Kannan [14] showed that the situation is much better for *dense* problem instances. For example, if $G = (V, E)$ is a graph on n vertices of density $p = 2n^{-2}|E|$, then its MAX CUT can be approximated within a factor of $1 - \varepsilon$ in time $\text{poly}(\exp((\varepsilon p)^{-2}) \cdot n)$. Hence, if $p > \delta$ for some fixed number $\delta > 0$, then this algorithm has a polynomial running time. Similarly, if F is a k -SAT formula with at least $\binom{\delta 2^k n}{k}$ clauses (i.e., at least a constant fraction of all possible clauses is present), then the maximum number of simultaneously satisfiable clauses can be approximated within $1 - \varepsilon$ in polynomial time for any fixed $\varepsilon > 0$.

The key ingredient in [14] is an algorithm for approximating a dense matrix \mathbf{A} by a sum of a bounded number of “cut matrices”. Applied to the adjacency matrix of a graph, this yields the aforementioned algorithm for MAX CUT. Moreover, an extension of this matrix algorithm to k -dimensional tensors yields the approximation algorithms for dense instances of MAX CSP problems.

To explain the matrix decomposition, let us consider a 0/1-matrix \mathbf{A} of size $m \times n$, and let $0 \leq p \leq 1$ be the *density* of \mathbf{A} , i.e., the number of ones in \mathbf{A} divided by $m \cdot n$. A *cut matrix* is a matrix \mathbf{D} such that there are sets $S \subset [m]$, $T \subset [n]$ and a number d such that the entry \mathbf{D}_{ij} is equal to d if $(i, j) \in S \times T$ and 0 otherwise. We denote such a matrix by $\mathbf{D} = \text{CUT}(d, S, T)$, and observe that cut matrices have rank one. The *cut norm* of a $m \times n$ matrix $\mathbf{M} = (\mathbf{M}_{ij})_{i \in [m], j \in [n]}$ is

$$\|\mathbf{M}\|_{\square} = \max_{S \subset [m], T \subset [n]} |\mathbf{M}(S, T)|,$$

where $\mathbf{M}(S, T) = \sum_{(s,t) \in S \times T} \mathbf{M}_{st}.$

Frieze and Kannan proved that for any \mathbf{A} and any $\varepsilon > 0$ there exist cut matrices $\mathbf{D}_1, \dots, \mathbf{D}_s$ such that

$$\|\mathbf{A} - (\mathbf{D}_1 + \dots + \mathbf{D}_s)\|_{\square} < \varepsilon \cdot mn,$$

where $s \leq c\varepsilon^{-2}$ for a constant $c > 0$. Indeed, such a decomposition can be computed in time $\varepsilon^{-2} \cdot \text{poly}(mn)$ (or even in “constant” expected time $O(\varepsilon^{-2} \cdot \text{polylog}(1/\varepsilon))$ by sampling). Hence, if $p \geq \delta$ for some fixed $\delta > 0$, i.e., if \mathbf{A} is a *dense* matrix, then setting $\varepsilon' = \varepsilon p$ we can use this algorithm to find a decomposition of \mathbf{A} within $\varepsilon \|\mathbf{A}\|_{\square} = \varepsilon \cdot mnp$ efficiently by a sum of at most $c\varepsilon'^{-2} = c(\varepsilon p)^{-2} \leq c(\varepsilon \delta)^{-2}$ cut matrices. The crucial point here is that the number of cut matrices is bounded *independently* of the size $m \cdot n$ of \mathbf{A} .

The goal of the present paper is to extend this result to *sparse* matrices, where the density p of \mathbf{A} is no longer bounded below by a fixed number. Thus, in asymptotic terms, we are interested in $p = o(1)$ as $m, n \rightarrow \infty$. Clearly, in this case the bound $c(\varepsilon p)^{-2}$ on the number of cut matrices in the decomposition guaranteed by [14] is no longer “constant”, but grows with the size $m \cdot n$ of \mathbf{A} . Of course, we cannot expect to obtain the same

*School of Informatics, University of Edinburgh, UK. email: acoghlan@inf.ed.ac.uk. Supported by DFG CO 646. Research done while visiting Carnegie Mellon University.

†Department of Computer Science, King’s College, University of London, UK, email: ccooper@dcs.kcl.ac.uk. Supported by Royal Society Grant 2006/R2-IJP.

‡Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA, USA, email: alan@random.math.cmu.edu. Supported in part by NSF grant CCF0502793.

results in the sparse as in the dense case for *arbitrary* sparse matrices; for in the light of the aforementioned hardness results this would imply $P=NP$. Hence, our main result is that even in the sparse case a 0/1 matrix \mathbf{A} (or, more generally, a k -dimensional tensor) can be approximated in the cut norm by a sum of cut matrices with a number of summands independent of m , n , and p , provided that \mathbf{A} satisfies a certain boundedness condition. This condition basically requires that \mathbf{A} does not feature relatively large, extraordinarily dense spots. In addition, we shall use these decomposition results to obtain $(1-\varepsilon)$ -approximation algorithms for instances of MAX CSP problems that have a suitable boundedness property. In a sense these results mediate between the “average” and the worst case analysis of algorithms.

In the rest of this section we state the main results. In Section 2 we discuss related work, and Section 3 contains some preliminary remarks. Section 4 deals with the algorithm for approximating sparse bounded matrices, and in Section 5 we present the algorithm for approximating bounded CSP instances. Finally, Section 6 contains some examples of how to apply the results from the previous section.

1.1 Approximating 0/1 matrices. Let \mathbf{A} be a 0/1 matrix of size $m \times n$ and density p . Given $C, \gamma > 0$, we say that \mathbf{A} is (C, γ) -bounded if for any two sets $S \subset [m]$ and $T \subset [n]$ of sizes $|S| \geq \gamma m$, $|T| \geq \gamma n$ we have

$$(1.1) \quad \mathbf{A}(S, T) = \sum_{(s,t) \in S \times T} \mathbf{A}_{st} \leq C \cdot |S| \cdot |T| \cdot p.$$

In words, for any two sufficiently large sets S, T the number $\mathbf{A}(S, T)$ of ones in the square $S \times T$ must not exceed the number $|S| \cdot |T| \cdot p$ that we would expect if S, T were *random* sets by more than a factor of C .

THEOREM 1.1. *There is an algorithm `ApXMatrix`, absolute constants $\zeta, \zeta' > 0$, and a polynomial Π such that the following holds. Suppose that $0 < \varepsilon < \frac{1}{2}$, $C > 1$. Let*

$$(1.2) \quad \kappa = \frac{\zeta C^2}{\varepsilon^2} \quad \text{and} \quad \gamma = \gamma(\varepsilon, C) = \frac{\zeta' \varepsilon}{2^{10\kappa} C}.$$

If \mathbf{A} is a (C, γ) -bounded 0/1 matrix, then in time $\kappa \cdot \Pi(m \cdot n)$, `ApXMatrix`($\mathbf{A}, C, \varepsilon$) outputs cut matrices $\mathbf{D}_1, \dots, \mathbf{D}_s$ such that $s \leq \kappa$ and

$$\|\mathbf{A} - (\mathbf{D}_1 + \dots + \mathbf{D}_s)\|_{\square} \leq \varepsilon \|\mathbf{A}\|_{\square}.$$

We emphasize that the upper bound κ on the number of cut matrices depends *only* on C and ε , but not on the size of \mathbf{A} or the density p . Moreover, while for the sake of simplicity we assume that `ApXMatrix` is given the boundedness parameter C as an input, this can easily be avoided by performing a binary search (details omitted).

Given the 0/1 matrix \mathbf{A} and partitions \mathcal{S} of $[m]$ and \mathcal{T} of $[n]$, we define a matrix $\mathbf{A}_{\mathcal{S} \times \mathcal{T}}$ as follows. If $s \in S \in \mathcal{S}$ and $t \in T \in \mathcal{T}$, then the corresponding entry $(\mathbf{A}_{\mathcal{S} \times \mathcal{T}})_{s,t}$ equals $|S|^{-1}|T|^{-1}\mathbf{A}(S, T)$. Hence, on each square $S \times T$ the matrix $\mathbf{A}_{\mathcal{S} \times \mathcal{T}}$ is constant, and the value it takes is just the average of \mathbf{A} over that square.

COROLLARY 1.1. *There is an algorithm `PartMatrix` and a polynomial Π that satisfy the following. Suppose that $\varepsilon, C > 0$, let κ, γ be as in (1.2), and assume that \mathbf{A} is a (C, γ) -bounded 0/1 matrix of size $m \times n$. Then in time $2^{\kappa} \cdot \Pi(m \cdot n)$ `PartMatrix`($\mathbf{A}, C, \varepsilon$) computes partitions \mathcal{S} of $[m]$ and \mathcal{T} of $[n]$ such that $\|\mathbf{A} - \mathbf{A}_{\mathcal{S} \times \mathcal{T}}\|_{\square} \leq 2\varepsilon \|\mathbf{A}\|_{\square}$. The number of classes in each partition is at most 2^{κ} .*

1.2 Weak regular partitions of graphs. Let $G = (V, E)$ be a graph on n vertices, and let $0 \leq p \leq 1$ be such that $|E| = n^2 p / 2$; we refer to p as the *density* of G . Moreover, we assume that $V = [n]$. In addition, let $\mathbf{A} = \mathbf{A}(G)$ be the adjacency matrix of G . Then we say that G is (C, γ) -bounded if \mathbf{A} has this property. Thus, if G is (C, γ) -bounded, then for any two sets $S, T \subset V$ of size at least γn we have $e_G(S, T) \leq C\gamma|S||T|p$, where $e_G(S, T)$ is the number of S - T -edges in G .

We call a partition \mathcal{V} of V a *weak ε -regular partition* of G if $\|\mathbf{A} - \mathbf{A}_{\mathcal{V} \times \mathcal{V}}\|_{\square} \leq \varepsilon \|\mathbf{A}\|_{\square} = 2\varepsilon|E|$. Hence, if, for instance, $S, T \subset V$ are disjoint sets of vertices, then the number $\mathbf{A}(S, T)$ of S - T -edges is within $2\varepsilon|E|$ of $\mathbf{A}_{\mathcal{V} \times \mathcal{V}}(S, T)$. As we shall see below, this definition is related to the notion of regular partitions introduced by Szemerédi.

COROLLARY 1.2. *There is an algorithm `WeakPart` and a polynomial Π that satisfy the following. Suppose that $C > 1$, $0 < \varepsilon < \frac{1}{2}$, let $\kappa, \gamma > 0$ be as in (1.2), and let $G = (V, E)$ be a (C, γ) -bounded graph on n vertices. Then `WeakPart`(G, C, ε) computes a weak 4ε -regular partition of G in time $2^{2\kappa} \cdot \Pi(n)$. This partition has at most $2^{2\kappa}$ classes.*

The algorithm `WeakPart` basically just applies `ApXMatrix` to the adjacency matrix of the input graph.

1.3 Approximating k -dimensional 0/1 tensors. A k -dimensional tensor is a map $\mathbf{M} : R_1 \times R_2 \times \dots \times R_k \rightarrow \mathbf{R}$, where R_1, \dots, R_k are finite index sets. Moreover, extending the matrix case in the obvious way to k dimensions, we say that a tensor $\mathbf{C} : R_1 \times R_2 \times \dots \times R_k \rightarrow \mathbf{R}$ is a *cut tensor* if there exist sets $S_i \subseteq R_i$ for $i = 1, 2, \dots, k$ and a real number d such that

$$\mathbf{C}(i_1, i_2, \dots, i_k) = \begin{cases} d & \text{if } (i_1, i_2, \dots, i_k) \in \prod_{j=1}^k S_j \\ 0 & \text{otherwise.} \end{cases}$$

In this case we write $\mathbf{C} = \text{CUT}(d, S_1, \dots, S_k)$. Further, we define the cut norm of a tensor as

$$\|\mathbf{M}\|_{\square} = \max_{S_i \subseteq R_i} |\mathbf{M}(S_1, S_2, \dots, S_k)|,$$

where

$$\mathbf{M}(S_1, \dots, S_k) = \sum_{(s_1, \dots, s_k) \in S_1 \times \dots \times S_k} \mathbf{M}(s_1, \dots, s_k).$$

Let $\mathbf{A} : R_1 \times R_2 \times \dots \times R_k \rightarrow \{0, 1\}$ be a 0/1 tensor. Set $k_1 = \lfloor k/2 \rfloor$. Then letting $\mathcal{R} = R_1 \times R_2 \times \dots \times R_{k_1}$ and $\mathcal{C} = R_{k_1+1} \times R_{k_1+2} \times \dots \times R_k$, we define a $(2^k - 1)$ -dimensional matrix $\mathbf{B} = \mathbf{B}(\mathbf{A}) : \mathcal{R} \times \mathcal{C} \rightarrow \{0, 1\}$ by

$$(1.3) \quad \mathbf{B}((i_1, \dots, i_{k_1}), (i_{k_1+1}, \dots, i_k)) = \mathbf{A}(i_1, \dots, i_k).$$

We say that \mathbf{A} is (C, γ) -bounded if $\mathbf{B}(\mathbf{A})$ has this property.

THEOREM 1.2. *There is an algorithm ApxTensor , a polynomial Π and a constant $\Gamma > 0$ such that the following is true. Suppose that $C > 1$ and $0 < \varepsilon < \frac{1}{2}$. Let*

$$\gamma = \exp(-\Gamma(C/\varepsilon)^2).$$

If $\mathbf{A} : R_1 \times R_2 \times \dots \times R_k \rightarrow \{0, 1\}$ is a (C, γ) -bounded 0/1 tensor, then $\text{ApxTensor}(\mathbf{A}, C, \varepsilon)$ outputs cut tensors

$$\mathbf{D}_i = \text{CUT}(d_i, S_i^1, \dots, S_i^k) \quad (S_i^1 \subset R_1, \dots, S_i^k \subset R_k)$$

for $i = 1, \dots, s$ with $s \leq (\Gamma C/\varepsilon)^{2(k-1)}$ such that

$$\|\mathbf{A} - (\mathbf{D}_1 + \dots + \mathbf{D}_s)\|_{\square} \leq \varepsilon \|\mathbf{A}\|_{\square}.$$

Moreover, $\sum_{i=1}^s d_i^2 \leq (Cp)^2 \Gamma^{2k}$. The running time is $(2^{(C/\varepsilon)^2} + (C/\varepsilon)^{3k}) \cdot \Pi(|R_1 \times \dots \times R_k|)$.

ApxTensor applies the algorithm ApxMatrix to the matrix $\mathbf{B}(\mathbf{A})$. The resulting cut norm approximation of $\mathbf{B}(\mathbf{A})$ is then used to obtain a dense tensor $\hat{\mathbf{A}}$ such that approximating $\hat{\mathbf{A}}$ by cut tensors is equivalent to approximating \mathbf{A} (up to scaling). Finally, ApxTensor applies an algorithms for dense tensors from [14] to $\hat{\mathbf{A}}$.

If $\mathcal{R}_1, \dots, \mathcal{R}_k$ are partitions of R_1, \dots, R_k , then we define a tensor $\mathbf{A}_{\mathcal{R}_1 \times \dots \times \mathcal{R}_k} : R_1 \times \dots \times R_k \rightarrow [0, 1]$ as follows: if $t_i \in \rho_i \in \mathcal{R}_i$ for $i = 1, \dots, k$, then we set

$$\mathbf{A}_{\mathcal{R}_1 \times \dots \times \mathcal{R}_k}(t_1, \dots, t_k) = \frac{\mathbf{A}(\rho_1, \dots, \rho_k)}{\prod_{i=1}^k |\rho_i|}.$$

COROLLARY 1.3. *There is an algorithm PartTensor , a polynomial Π and a constant $\Gamma > 0$ such that the following is true. Suppose that $C > 0$ and $0 < \varepsilon < \frac{1}{2}$. Let $\gamma = \exp(-\Gamma(C/\varepsilon)^2)$. If $\mathbf{A} : R_1 \times \dots \times R_k \rightarrow \{0, 1\}$ is*

a (C, γ) -bounded 0/1 tensor, then $\text{PartTensor}(\mathbf{A}, C, \varepsilon)$ computes partitions $\mathcal{R}_1, \dots, \mathcal{R}_k$ of R_1, \dots, R_k such that

$$\|\mathbf{A} - \mathbf{A}_{\mathcal{R}_1 \times \dots \times \mathcal{R}_k}\|_{\square} < \varepsilon \|\mathbf{A}\|_{\square}.$$

Each \mathcal{R}_i consists of at most $\exp((\Gamma C/\varepsilon)^{2(k-1)})$ classes. The running time is bounded by

$$\left[\exp((\Gamma C/\varepsilon)^{2(k-1)}) + (C/\varepsilon)^{3k} \right] \Pi(n^k).$$

1.4 An approximation algorithm for bounded Max CSPs. Let $V = \{x_1, \dots, x_n\}$ be a set of n Boolean variables. A (binary) k -constraint over V is a map $\phi : \{0, 1\}^{V_\phi} \rightarrow \{0, 1\}$ that is not identically zero, where $V_\phi \subset V$ is a set of size k . For an assignment $\sigma \in \{0, 1\}^V$ we let $\phi(\sigma) = \phi(\sigma(x))_{x \in V_\phi}$. Further, a k -CSP instance over V is a set \mathcal{F} of k -constraints over V , and we define

$$\text{OPT}(\mathcal{F}) = \max_{\sigma \in \{0, 1\}^V} \sum_{\phi \in \mathcal{F}} \phi(\sigma).$$

We let $\Psi = \Psi_k$ be the set of all $2^{2^k} - 1$ non-zero maps $\{0, 1\}^k \rightarrow \{0, 1\}$. Let $\psi \in \Psi$ and let $\phi : \{0, 1\}^{V_\phi} \rightarrow \{0, 1\}$ be a k -constraint, where $V_\phi = \{x_{i_1}, \dots, x_{i_k}\}$ with $1 \leq i_1 < \dots < i_k \leq n$. Then we say that ϕ is of type ψ if for any $\sigma : V_\phi \rightarrow \{0, 1\}$ we have $\phi(\sigma(x_{i_1}), \dots, \sigma(x_{i_k})) = \psi(\sigma)$. With this notion we can represent a k -CSP instance \mathcal{F} by a family $(\mathbf{A}_{\mathcal{F}}^\psi)_{\psi \in \Psi}$ of $2^{2^k} - 1$ k -tensors as follows. We let $\mathbf{A}_{\mathcal{F}}^\psi(i_1, \dots, i_k) = 1$ if there is a $\phi \in \mathcal{F}$ of type ψ with $V_\phi = \{x_{i_1}, \dots, x_{i_k}\}$ and set $\mathbf{A}_{\mathcal{F}}^\psi(i_1, \dots, i_k) = 0$ otherwise. Further, we say that \mathcal{F} is (C, γ) -bounded if the tensors $\mathbf{A}_{\mathcal{F}}^\psi$ are (C, γ) -bounded for all $\psi \in \Psi$.

THEOREM 1.3. *There are an algorithm ApxCSP , a constant $\Gamma > 0$, and a polynomial Π such that for any $k, C > 1$, $0 < \varepsilon < \frac{1}{2}$ there is a number $n_0 = n_0(C, \varepsilon, k)$ such that the following is true. Let*

$$\gamma = \exp(-\Gamma 2^{-2^k - 2k - 2} (C/\varepsilon)^2).$$

If \mathcal{F} is a (C, γ) -bounded k -CSP instance over $V = \{x_1, \dots, x_n\}$ for some $n \geq n_0$, then $\text{ApxCSP}(\mathcal{F}, C, \varepsilon)$ outputs an assignment $\sigma : V \rightarrow \{0, 1\}$ such that

$$\sum_{\phi \in \mathcal{F}} \phi(\sigma) \geq (1 - \varepsilon) \text{OPT}(\mathcal{F}).$$

The running time is at most

$$\Pi \left[\exp(k 2^k 2^{2^k} (C/\varepsilon)^{2k} \ln(C/\varepsilon)) n^k \right].$$

2 Related work.

2.1 Approximating dense matrices and tensors.

As mentioned earlier, Frieze and Kannan [14] dealt with *dense* matrices and tensors. More precisely, they showed that for a tensor $\mathbf{A} : R_1 \times \cdots \times R_k \rightarrow [0, 1]$ and an $\varepsilon > 0$ one can compute cut tensors $\mathbf{D}_1, \dots, \mathbf{D}_s$ such that $\|\mathbf{A} - \sum_{i=1}^s \mathbf{D}_i\|_{\square} < \varepsilon |R_1 \times \cdots \times R_k|$ in time $O(\varepsilon^{2(1-k)} \text{polylog}(1/\varepsilon))$ with $s \leq O(\varepsilon)^{2(1-k)}$ as $\varepsilon \rightarrow 0$. Let us point out two things.

- The running time of their algorithm depends *only* on ε , and not on the size of \mathbf{A} . To achieve this, the algorithm just works with a bounded (by a function of ε only) size sample of the input data and produces an implicit representation of the desired decomposition. Further results of this type can be found in [2, 5, 11, 12, 16]. If $\mathbf{A} : R_1 \times \cdots \times R_k \rightarrow \{0, 1\}$ is a sparse 0/1 tensor with density $p = \|\mathbf{A}\|_{\square} / |R_1 \times \cdots \times R_k| = o(1)$ as the problem size $N = |R_1 \times \cdots \times R_k| \rightarrow \infty$, then this sampling approach cannot yield an approximation within $\varepsilon N p$, because a constant sized sample of \mathbf{A} is likely to be just identically 0.
- The error term $\varepsilon |R_1 \times \cdots \times R_k|$ does not account for the density of \mathbf{A} . For example, suppose that \mathbf{A} is the adjacency matrix of a graph $G = (V, E)$ on n vertices with density $p = 2n^{-2}|E|$. Then the algorithm from [14] can be used to compute a cut norm approximation of \mathbf{A} to within εn^2 for any $\varepsilon > 0$. Hence, we can use this approximation to solve graph partitioning problems such as MAX CUT within an additive error of εn^2 (edges). This is why [14] is limited to *dense* problem instances (i.e., $p = \Omega(1)$ as $n \rightarrow \infty$).

In spite of these differences, some of the algorithms that we consider here are very similar to those from [14]. In a sense our main contribution is to *analyze* these algorithms on sparse matrices/graphs/tensors. For instance, the matrix approximation algorithm for Theorem 1.1 is almost identical to the procedure described in [14, Section 4.1]. The only difference is that [14] employs as a subroutine a combinatorial procedure for approximating the cut norm of a given $m \times n$ matrix within an *additive* error of εmn , whereas here we need to approximate the cut norm within a constant *multiplicative* factor. To this end, we rely on an algorithm of Alon and Naor [4] (which is based on semidefinite programming). Nonetheless, as we shall see in Section 4 new ideas are necessary to analyze, e.g., the number of cut matrices that are necessary to approximate the input matrix \mathbf{A} within the desired $\varepsilon \|\mathbf{A}\|_{\square}$ in the cut norm (rather than within εmn).

2.2 Szemerédi’s regularity lemma. Theorem 1.2 and the concept of weak regular partitions is related to Szemerédi’s well-known regularity lemma [22]. While [22] only deals with “dense” graphs, Kohayakawa [20] and Rödl [21] independently extended the regularity lemma to the sparse case; for a comprehensive account see Gerke and Steger [15]. The papers [20, 21] show that for any $\varepsilon > 0$ and any $C > 0$ there is a number $\gamma > 0$ such any (C, γ) -bounded graph has a regular partition (V_1, \dots, V_s) in the following sense.

- We have $|V_i - n/s| \leq 1$ for all i .
- All but εs^2 pairs (V_i, V_j) satisfy the following. For any two sets $S \subset V_i, T \subset V_j$ of size $|S| \geq \varepsilon |V_i|, |T| \geq \varepsilon |V_j|$ we have

$$(2.4) \quad \left| e_G(S, T) - \frac{|S \times T|}{|V_i \times V_j|} \cdot e_G(V_i, V_j) \right| \leq \varepsilon e_G(V_i, V_j).$$

The number s of classes is bounded by $\mathcal{T}((C/\varepsilon)^3)$, i.e., it is *independent* of n . This is the key fact that makes Szemerédi’s lemma so useful in extremal combinatorics. Here \mathcal{T} is the rather fast growing *tower function* defined by $\mathcal{T}(0) = 1$ and $\mathcal{T}(j) = 2^{\mathcal{T}(j-1)}$ for $j \geq 1$. Hence, from an algorithmic perspective the bound $\mathcal{T}((C/\varepsilon)^3)$ is somewhat disappointing. However, it is essentially best possible, as there is an infinite family of graphs for which the number of classes in the smallest ε -regular Szemerédi partition is $\mathcal{T}(C/\varepsilon)$ [17]. Moreover, the number γ required in the boundedness condition is as tiny as $\mathcal{T}((C/\varepsilon)^3)^{-1}$.

While [20, 21, 22] focus on proving that a regular partition exists, [1, 3] deal with algorithmic versions of the regularity lemma. In the dense case (i.e., $|E| = \Omega(n^2)$) there is a purely combinatorial algorithm [3] with running time $\mathcal{T}(\varepsilon^{-3}) \cdot \text{poly}(n)$. Moreover, an algorithm for the sparse case was presented in [1]; the running time is $\mathcal{T}((C/\varepsilon)^3) \cdot \text{poly}(n)$ for (C, γ) -bounded graphs, and the algorithm is based on the semidefinite programming algorithm for approximating the cut norm from [4].

Corollary 1.2 relates to [1] as follows. While the “strong” regularity condition (2.4) takes into account the “microscopic” edge distribution within (almost) each pair (V_i, V_j) , the “weak” regularity concept from Corollary 1.2 just provides a “macroscopic” approximation w.r.t. the cut norm. This approximation is sufficiently strong for algorithmic applications such as MAX CUT. In effect, the algorithm is more efficient. Indeed, instead of scaling as a tower function $\mathcal{T}((C/\varepsilon)^3)$, the running time of the algorithm **WeakPart** from Corollary 1.2 grows like $\exp(O(C/\varepsilon)^2)$ in terms of C, ε . Furthermore, as Theorem 1.1 shows, one can approximate a (C, γ) -bounded adjacency matrix by a sum of $O(C/\varepsilon)^2$

cut matrices (if the actual partition of the vertex set is not needed), thus even avoiding the exponential dependence on C/ε . Similarly, the parameter γ required in the boundedness condition is just $\gamma = \exp(-O(C/\varepsilon)^2)$, rather than $\gamma = 1/T((C/\varepsilon)^3)$ as in [1]. Consequently, Corollary 1.2 applies to a larger class of graphs. A further novel aspect here is that we extend our results to k -dimensional tensors (or k -uniform hypergraphs). This point is not addressed in [1].

As far as the techniques (both for the algorithms and for the proofs) are concerned, our algorithm `ApxMatrix` and the algorithm from [1] follow a somewhat similar general strategy: starting from a trivial partition, the algorithm uses the cut norm approximation from [4] to find a “witness of irregularity”, refines the partition, and iterates. To analyze this procedure, some function of the present approximation is used to measure the algorithm’s “progress”. However, the details of how this strategy is implemented differ considerably. The algorithms aim to compute rather different objects. Hence, the present algorithm maintains a real “error matrix” \mathbf{A}_j and the “witness of irregularity” is a set of rows/columns of \mathbf{A}_j where the maximum for the cut norm is attained. The function to measure the progress is (essentially) the Frobenius norm of \mathbf{A}_j . By contrast, in [1] the algorithm maintains a (huge) partition of the vertices, the witness of irregularity is a (huge) family of subsets of the partition classes, and the progress function is the measured by the “index” of the partition as introduced by Szemerédi [22].

3 Preliminaries and Notation.

An important ingredient to the algorithm `ApxMatrix` for Theorem 1.1 is the the following algorithmic version of Grothendieck’s inequality from Alon and Naor [4].

THEOREM 3.1. *There are a polynomial time algorithm and a number $\alpha_0 > 0$ that have the following property. Given a $m \times n$ matrix \mathbf{M} , the algorithm outputs sets $S \subset [m]$ and $T \subset [n]$ such that $|\mathbf{M}(S, T)| \geq \alpha_0 \|\mathbf{M}\|_{\square}$.*

Alon and Naor present a randomized algorithm with $\alpha_0 > 0.56$, and a deterministic one with $\alpha_0 \geq 0.03$.

If \mathbf{M} is a real $m \times n$ matrix, then we let $\|\mathbf{M}\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n \mathbf{M}_{ij}^2}$ signify the Frobenius norm of \mathbf{M} . Moreover, if G is a graph, then we denote the vertex set of G by $V(G)$ and the edge set by $E(G)$. For sets $S, T \subset V(G)$ we let $e_G(S, T)$ signify the number of S - T -edges of G , and $e_G(S)$ signifies the number of edges spanned by S .

Suppose that X is a set and that $\mathcal{P}_1, \mathcal{P}_2$ are partitions of X . We say that \mathcal{P}_1 is *coarser* than \mathcal{P}_2 if each class of \mathcal{P}_2 is contained in a class of \mathcal{P}_1 . If \mathcal{S} is an arbitrary

set of subsets of X , then there is a unique partition \mathcal{P} of X such that each set in X is a union of classes of \mathcal{P} , and \mathcal{P} is coarser than any other partition that has this property. This partition \mathcal{P} has at most $2^{|\mathcal{S}|}$ classes.

4 Approximating 0/1 matrices.

Let $C > 1$ and $0 < \varepsilon < \frac{1}{2}$. Moreover, let α_0 be the constant from Theorem 3.1 and set

$$\kappa = \frac{513C^2}{\varepsilon^2\alpha_0}, \quad \gamma = \frac{\varepsilon\alpha_0}{2^{10\kappa}C}, \quad \gamma' = 2^\kappa\gamma.$$

Throughout this section we assume that \mathbf{A} is a (C, γ) bounded 0/1 matrix of size $m \times n$.

ALGORITHM 4.1. `ApxMatrix`($\mathbf{A}, C, \varepsilon$)

Input: A 0/1 matrix \mathbf{A} of size $m \times n$, numbers $C, \varepsilon > 0$.

Output: A sequence of cut matrices.

1. Set $\mathbf{A}_0 = \mathbf{A}$.
2. For $j = 0, 1, 2, \dots, \kappa$ do
3. Compute two sets S_{j+1}, T_{j+1} of sizes $|S_{j+1}| \geq m/2, |T_{j+1}| \geq n/2$ such that $|\mathbf{A}_j(S_{j+1}, T_{j+1})| \geq \alpha_0 \|\mathbf{A}_j\|_{\square} / 4$.
4. If $|\mathbf{A}_j(S_{j+1}, T_{j+1})| < \alpha_0 \varepsilon m n p / 4$ and $j \geq 1$, then output the cut matrices $\mathbf{D}_1, \dots, \mathbf{D}_j$ and halt.
5. Else, compute

$$d_{j+1} = \frac{\mathbf{A}_j(S_{j+1}, T_{j+1})}{|S_{j+1}| |T_{j+1}|},$$

set $\mathbf{D}_{j+1} = \text{CUT}(d_{j+1}, S_{j+1}, T_{j+1})$, and let $\mathbf{A}_{j+1} = \mathbf{A}_j - \mathbf{D}_{j+1}$.

6. Output “failure”.

In order to approximate the given 0/1 matrix \mathbf{A} by a sum $\mathbf{D}_1 + \dots + \mathbf{D}_j$ of cut matrices, `ApxMatrix` proceeds as follows. After j iterations, $\mathbf{A}_j = \mathbf{A} - \sum_{i=1}^j \mathbf{D}_i$ is the “error term” that results from approximating \mathbf{A} by $\sum_{i=1}^j \mathbf{D}_i$. Thus, the goal is to eventually achieve an error term \mathbf{A}_j that has a small cut norm. Therefore, Step 3 computes sets S_{j+1}, T_{j+1} of rows and columns such that $|\mathbf{A}_j(S_{j+1}, T_{j+1})|$ is a good approximation of the cut norm of \mathbf{A}_j . If the term $|\mathbf{A}_j(S_{j+1}, T_{j+1})|$ (and hence the cut norm of \mathbf{A}_j) is small, then Step 4 terminates and outputs the cut matrices $\mathbf{D}_1, \dots, \mathbf{D}_j$. Otherwise, S_{j+1}, T_{j+1} witness a set of rows/columns on which $\sum_{i=1}^j \mathbf{D}_i$ does not provide a good enough approximation. Therefore, Step 5 adds a further “patch” \mathbf{D}_{j+1} , which is a cut matrix whose value on $S_{j+1} \times T_{j+1}$ is just the average d_{j+1} of \mathbf{A}_j over that square (note that d_{j+1} may be negative). This ensures that $\mathbf{A}_{j+1}(S_{j+1}, T_{j+1}) = 0$, and thus takes care of the discrepancy witnessed by S_{j+1}, T_{j+1} .

If the algorithm outputs cut matrices $\mathbf{D}_1, \dots, \mathbf{D}_j$, then clearly

$$\|\mathbf{A} - (\mathbf{D}_1 + \dots + \mathbf{D}_j)\|_{\square} = \|\mathbf{A}_j\|_{\square} \leq \varepsilon m n p = \varepsilon \|\mathbf{A}\|_{\square},$$

because of the halting condition in Step 4. Hence, in order to establish Theorem 1.1, we need to prove that

- (a) Step 3 of `ApxMatrix` can be implemented by a polynomial time algorithm,
- (b) the halting condition in Step 4 is satisfied for some $1 \leq j \leq \kappa$.

PROPOSITION 4.1. *In Step 3 the sets S_{j+1}, T_{j+1} can be computed in time $\text{poly}(mn)$.*

Proof To obtain S_{j+1}, T_{j+1} , we use the polynomial time algorithm from Theorem 3.1, which yields sets S'_{j+1}, T'_{j+1} such that $|\mathbf{A}_j(S'_{j+1}, T'_{j+1})'| \geq \alpha_0 \|\mathbf{A}_j\|_{\square}$. If $|S'_{j+1}| \geq n/2$ then we take $S_{j+1} = S'_{j+1}$. If $|S'_{j+1}| < n/2$ then since

$$\mathbf{A}(R, T'_{j+1}) = \mathbf{A}(S'_{j+1}, T'_{j+1}) + \mathbf{A}(R \setminus S'_{j+1}, T'_{j+1})$$

we get

$$\max\{|\mathbf{A}(R, T'_{j+1})|, |\mathbf{A}(R \setminus S'_{j+1}, T'_{j+1})|\} \geq \alpha_0 \|\mathbf{A}_0\|_{\square} / 2.$$

We can therefore take either R or $R \setminus S'_{j+1}$ as our set S_{j+1} and note it is at least $n/2$ in size. We perform the same operation to get T_{j+1} , losing (at most) another factor 2 in the approximation. \square

With respect to (b), we will study the Frobenius norm of \mathbf{A}_j . Namely, it is not difficult to show that $\|\mathbf{A}_j\|_F^2 \leq \|\mathbf{A}\|_F^2(1 - j \cdot \alpha_0^2 \varepsilon^2 p/4)$. Since trivially $\|\mathbf{A}_j\|_F \geq 0$, this implies that the total number of iterations is at most $4/(\alpha_0^2 \varepsilon^2 p)$. Hence, if p is bounded from below by a constant, then this argument shows that the total number of iterations is bounded by a number that does not depend on n, m . In fact, this is the basic argument used to establish the matrix decomposition theorem in [14, Section 4.1].

But in the present work we do *not* assume that p remains bounded away from 0. In effect, the aforementioned argument does not apply. As it turns out, the problem is that the above argument just uses the trivial lower bound $\|\mathbf{A}_j\|_F^2 \geq 0$. By contrast, the basic idea here is to use the boundedness condition to establish $\|\mathbf{A}_F\|^2(1 - C^2 p)$ as a lower bound. Indeed, if we could show that $\|\mathbf{A}_j\|_F^2 \geq \|\mathbf{A}_F\|^2(1 - C^2 p)$ for all j , then the bound $\|\mathbf{A}_j\|_F^2 \leq \|\mathbf{A}\|_F^2(1 - j \cdot \alpha_0^2 \varepsilon^2 p/4)$ would imply that the number of iterations is at most $4C^2/(\alpha_0^2 \varepsilon^2)$, and thus independent of m, n, p .

However, we can't quite use the boundedness condition to prove that $\|\mathbf{A}_j\|_F^2 \geq \|\mathbf{A}_F\|^2(1 - C^2 p)$. The reason is that the boundedness condition only applies to "sufficiently large" sets, i.e., sets of size at least γn . Therefore, to show that `ApxMatrix` stops after at most κ iterations, we will consider slightly different sequences

of matrices $\mathbf{D}'_j, \mathbf{A}'_j$, to which the boundedness condition applies. The matrices $\mathbf{D}'_j, \mathbf{A}'_j$ will be "close" to $\mathbf{D}_j, \mathbf{A}_j$ in cut norm, and to bound the number of iterations we are going to investigate the Frobenius norm of \mathbf{A}'_j .

We construct the matrices $\mathbf{D}'_j, \mathbf{A}'_j$ as follows. Let us assume (for contradiction) that `ApxMatrix` outputs "failure", i.e., the number of iterations performed by Steps 2–5 is κ . Then during these κ iterations the algorithm constructed sets S_1, \dots, S_{κ} of rows and T_1, \dots, T_{κ} of columns. Let \mathcal{S} be the coarsest partition of the set $[m]$ of row indices such that each S_i is a union of classes of \mathcal{S} . Similarly, let \mathcal{T} be the coarsest partition of the columns set $[n]$ such that every T_i is a union of classes of \mathcal{T} . Clearly, both \mathcal{S} and \mathcal{T} have at most 2^{κ} classes. The reason why the boundedness condition does not imply directly that $\|\mathbf{A}_j\|_F^2 \geq \|\mathbf{A}_F\|^2(1 - C^2 p)$ is that some classes of \mathcal{S} and \mathcal{T} may have size less than γm or γn . Therefore, we let

$$R_0 = \bigcup_{S \in \mathcal{S}: |S| < \gamma m} S, \quad C_0 = \bigcup_{T \in \mathcal{T}: |T| < \gamma n} T$$

comprise the "small" classes of the partitions \mathcal{S}, \mathcal{T} . Setting $\gamma' = 2^{\kappa} \gamma$, we have

$$(4.5) \quad |R_0| \leq \gamma' m, \quad |C_0| \leq \gamma' n.$$

Further, let $\mathbf{A}'_0 = \mathbf{A}'$ be the matrix obtained from \mathbf{A} by replacing all rows in R_0 and all columns in C_0 by 0. In addition, define inductively sets $S'_j = S_j \setminus R_0$ and $T'_j = T_j \setminus T_0$ and

$$\begin{aligned} d'_{j+1} &= \frac{\mathbf{A}'_j(S'_{j+1}, T'_{j+1})}{|S'_{j+1}| |T'_{j+1}|}, \\ \mathbf{D}'_{j+1} &= \text{CUT}(S'_{j+1}, T'_{j+1}, d'_{j+1}), \\ \mathbf{A}'_{j+1} &= \mathbf{A}'_j - \mathbf{D}'_{j+1}. \end{aligned}$$

Let \mathcal{S}' be the coarsest partition of $[m] \setminus R_0$ such that each S'_j is a union of classes of \mathcal{S}' , and define a partition \mathcal{T}' of $[n] \setminus C_0$ analogously w.r.t. the sets T'_j . Then the construction of the sets S'_j, T'_j readily implies:

FACT 4.1. *All classes of \mathcal{S}' (resp. \mathcal{T}') have size at least γm (resp. γn).*

Consequently, we can use the boundedness condition to infer the following

LEMMA 4.1. *For all $1 \leq j \leq \kappa$ we have*

$$\|\mathbf{A}'_j\|_F^2 \geq \|\mathbf{A}'\|_F^2(1 - 2C^2 p).$$

Proof Let $\mathbf{M} = \sum_{i=1}^j \mathbf{D}'_i$. Then for any two sets $S \in \mathcal{S}', T \in \mathcal{T}'$ the matrix \mathbf{M} is constant on the square $S \times T$, because every \mathbf{D}'_j is a cut matrix on the

square $S'_j \times T'_j$, and S'_j, T'_j are unions of classes of \mathcal{S}' , \mathcal{T}' . Thus, letting $m_{S \times T}$ signify the value that \mathbf{M} takes on $S \times T$, we obtain

$$\begin{aligned} \|\mathbf{A}'_j\|_F^2 &= \|\mathbf{A}' - \mathbf{M}\|_F^2 \\ &= \sum_{S \in \mathcal{S}', T \in \mathcal{T}'} \sum_{(v,w) \in S \times T} (\mathbf{A}'(v,w) - m_{S \times T})^2. \end{aligned}$$

For any $S \in \mathcal{S}'$, $T \in \mathcal{T}'$ the sum $\sum_{(v,w) \in S \times T} (\mathbf{A}'(v,w) - m_{S \times T})^2$ is minimized iff

$$m_{S \times T} = m_{S \times T}^* = \mathbf{A}'(S, T) / (|S| \cdot |T|).$$

Therefore,

$$\begin{aligned} &\sum_{(v,w) \in S \times T} (\mathbf{A}'(v,w) - m_{S \times T})^2 \\ &\geq \sum_{(v,w) \in S \times T} (\mathbf{A}'(v,w) - m_{S \times T}^*)^2 \\ &= \sum_{(v,w) \in S \times T} \mathbf{A}'(v,w)^2 - 2\mathbf{A}'(S, T)m_{S \times T}^* \\ &\quad + m_{S \times T}^{*2} |S| \cdot |T| \\ &= \sum_{(v,w) \in S \times T} \mathbf{A}'(v,w)^2 - m_{S \times T}^{*2} |S| \cdot |T|. \end{aligned}$$

Since \mathbf{A}' is (C, γ) bounded and because $|S| \geq \gamma m$, $|T| \geq \gamma n$ by Lemma 4.1, we get $m_{S \times T}^* \leq Cp$. Hence,

$$(4.6) \quad \|\mathbf{A}'_j\|_F^2 \geq \|\mathbf{A}'\|_F^2 - (Cp)^2 mn.$$

Finally, using the fact that \mathbf{A} and \mathbf{A}' are 0, 1 matrices, we have

$$\begin{aligned} \|\mathbf{A}\|_F^2 - \|\mathbf{A}'\|_F^2 &= \sum_{i=1}^m \sum_{j=1}^n \mathbf{A}_{ij}^2 - \mathbf{A}'_{ij}{}^2 \\ &= \mathbf{A}(R_0, [n]) + \mathbf{A}([m], C_0) - \mathbf{A}(R_0, C_0) \\ &\leq \mathbf{A}(R_0, [n]) + \mathbf{A}([m], C_0) \\ (4.5) \quad &\leq 2C\gamma' mnp < mnp/2, \end{aligned}$$

whence $\|\mathbf{A}'\|_F^2 \geq \|\mathbf{A}\|_F^2/2 = \frac{mnp}{2}$. Thus, the assertion follows from (4.6). \square

To show that our assumption that **ApxMatrix** performs at least κ iterations yields a contradiction, we need the following upper bound on $\|\mathbf{A}'_j\|_F^2$. Its proof employs similar arguments as presented in [14, Section 4.1].

LEMMA 4.2. *For all $1 \leq j \leq \kappa$ we have*

$$\|\mathbf{A}'_j\|_F^2 \leq \|\mathbf{A}'\|_F^2 (1 - j \cdot \alpha_0^2 \varepsilon^2 p / 256).$$

Combining Lemmas 4.1 and 4.2 and setting $j = \kappa$, we conclude $2C^2 \geq \kappa \cdot \alpha_0^2 \varepsilon^2 / 256$, which contradicts our choice of κ (cf. (1.2)). This completes the proof of Theorem 1.1.

5 Approximating Max CSP problems.

Throughout this section we keep the notation from Section 1.4. Given $0 < \varepsilon < \frac{1}{2}$, $C > 1$, we set $\gamma = \exp(-\Gamma(C/\varepsilon)^2)$, where Γ is the constant from Theorem 1.2. Moreover, we assume that \mathcal{F} is a (C, γ) -bounded k -CSP instance on n variables $V = \{x_1, \dots, x_n\}$, where $n > n_0$ for some sufficiently large number $n_0 = n_0(C, \varepsilon, k)$. Let $m = |\mathcal{F}|$ be the number of constraints.

ALGORITHM 5.1. **ApxCSP**($\mathcal{F}, C, \varepsilon$)

Input: A k -CSP instance \mathcal{F} over $V = \{x_1, \dots, x_n\}$, numbers $C, \varepsilon > 0$.

Output: An assignment $\hat{\sigma} : V \rightarrow \{0, 1\}$.

1. Set up the tensors $\mathbf{A}'_{\mathcal{F}}^{\psi}$ for all $\psi \in \Psi$.

Let $\alpha = \varepsilon 2^{-2k-2k-2}$.

Call **ApxTensor**($\mathbf{A}'_{\mathcal{F}}^{\psi}, C, \alpha$) for each $\psi \in \Psi$ to obtain tensors

$$\mathbf{B}^{\psi} = \sum_{i=1}^s \mathbf{D}_i^{\psi}, \text{ where } \mathbf{D}_i^{\psi} = \text{CUT}(d_i^{\psi}, S_{i1}^{\psi}, \dots, S_{ik}^{\psi}).$$

Let \mathcal{P} be the coarsest partition of V such that each set S_{ih}^{ψ} is a union of classes of \mathcal{P} ($1 \leq i \leq s$, $1 \leq h \leq k$, $\psi \in \Psi$).

2. Let $\delta = C^{-1} \Gamma^{-k} s^{-1} 2^{-2k-4k-4}$ and $\nu = \lceil \delta n \rceil$.

Compute an optimal solution $(\hat{\tau}_{ih}^{\psi}(1), \hat{\tau}_{ih}^{\psi}(0), \hat{z}_P)$ to the following optimization problem.

$$\begin{aligned} \text{OPT}'' &= \max \sum_{\psi \in \Psi} \sum_{i=1}^s \sum_{y \in \{0,1\}^k} d_i^{\psi} \psi(y) \nu^k \\ &\quad \times \prod_{h=1}^k \tau_{ih}^{\psi}(y_h) \\ \text{s.t.} \quad &0 \leq \tau_{ih}^{\psi}(1) \leq \|S_{ih}^{\psi}\|/\nu \\ &\tau_{ih}^{\psi}(0) = \nu^{-1} S_{ih}^{\psi} - \tau_{ih}^{\psi}(1) \\ &\tau_{ih}^{\psi}(1)\nu \leq \sum_{P \in \mathcal{P}: P \subset S_{ih}^{\psi}} z_P \leq (\tau_{ih}^{\psi}(1) + 1)\nu \\ &0 \leq z_P \leq |P| \quad \text{for all } P \in \mathcal{P}, \end{aligned}$$

where $1 \leq i \leq s$, $1 \leq h \leq k$, $\psi \in \Psi$ and the numbers $\tau_{ih}^{\psi}(1)$ are constrained to be integers.

Output an assignment $\hat{\sigma} : V \rightarrow \{0, 1\}$ such that $\|\hat{\sigma}^{-1}(1) \cap P| - \hat{z}_P| \leq 1$ for all $P \in \mathcal{P}$.

The first step of **ApxCSP** relies on the procedure **ApxTensor** from Theorem 1.2. Since we assume that all the tensors $\mathbf{A}'_{\mathcal{F}}^{\psi}$ are (C, γ) -bounded, we can apply **ApxTensor** to each of them to obtain an approximation \mathbf{B}^{ψ} consisting of a bounded number of cut tensors \mathbf{D}_i^{ψ} . The basic idea is to approximate the MAX CSP

problem, i.e., the optimization problem

$$\begin{aligned} \text{OPT} &= \max_{\sigma \in \{0,1\}^V} \sum_{\phi \in \mathcal{F}} \phi(\sigma) \\ &= \max_{\sigma \in \{0,1\}^V} \sum_{\psi \in \Psi} \sum_{(z_1, \dots, z_k) \in V^k} \mathbf{A}^\psi(z_1, \dots, z_k) \\ &\quad \times \psi(\sigma(z_1), \dots, \sigma(z_k)) \end{aligned}$$

by the optimization problem

$$\text{OPT}' = \max_{\sigma \in \{0,1\}^V} \sum_{\psi \in \Psi} \sum_{(z_1, \dots, z_k) \in V^k} \mathbf{B}^\psi(z_1, \dots, z_k) \times \psi(\sigma(z_1), \dots, \sigma(z_k)).$$

The following lemma shows that any assignment σ that approximates OPT' well also provides a good approximation for OPT .

LEMMA 5.1. *Let $\sigma \in \{0,1\}^V$ be such that*

$$(5.7) \quad \sum_{\psi \in \Psi} \sum_{z \in V^k} \mathbf{B}^\psi(z) \psi(\sigma(z)) \geq \text{OPT}' - 2^{-k-1} \varepsilon m.$$

Then $\sum_{\phi \in \mathcal{F}} \phi(\sigma) \geq (1 - \varepsilon) \text{OPT}(\mathcal{F})$.

It is worth pointing out that OPT' and OPT'' can be solved in “polynomial” time. This is because the tensors \mathbf{B}^ψ consist of only a bounded (w.r.t. n) number of cut tensors. More precisely, the partition \mathcal{P} constructed in Step 1 of the algorithm has the following property: if $S_1, \dots, S_k \in \mathcal{P}$, then all the tensors \mathbf{B}^ψ , $\psi \in \Psi$, are constant on the rectangle $S_1 \times \dots \times S_k$. Therefore, as far as OPT' is concerned, the individual variables in each set $S \in \mathcal{P}$ are completely indistinguishable. More precisely, consider an assignment $\sigma : V \rightarrow \{0,1\}$ and let

$$(5.8) \quad \mathcal{Z}_P = |\{v \in P : \sigma(v) = 1\}|,$$

$$(5.9) \quad \mathcal{T}_{ih}^\psi(1) = \sum_{P \in \mathcal{P}: P \subset S_{ih}^\psi} \mathcal{Z}_P,$$

$$(5.10) \quad \mathcal{T}_{ih}^\psi(0) = |S_{ih}^\psi| - \mathcal{T}_{ih}^\psi(1)$$

for each $P \in \mathcal{P}$, $1 \leq i \leq s$, $1 \leq h \leq k$, and $\psi \in \Psi$. In words, $\mathcal{T}_{ih}^\psi(y)$ is the number of variables in S_{ih}^ψ that attain the value y under σ ($y = 0, 1$). Let us further define

$$\sigma(y) = (\sigma(y_1), \dots, \sigma(y_k)) \quad \text{for } y \in V^k.$$

Then

$$\begin{aligned} &\sum_{\psi \in \Psi} \sum_{z \in V^k} \mathbf{B}^\psi(z) \psi(\sigma(z)) \\ &= \sum_{\psi \in \Psi} \sum_{i=1}^s \sum_{z \in \prod_{h=1}^k S_{ih}^\psi} d_i^\psi \psi(\sigma(z)) \\ &= \sum_{\psi \in \Psi} \sum_{i=1}^s \sum_{y \in \{0,1\}^k} d_i^\psi \psi(y) \prod_{h=1}^k \mathcal{T}_{ih}^\psi(y_h). \end{aligned}$$

Hence, to solve OPT' optimally, we could just try all possible tuples $(\mathcal{Z}_P)_{P \in \mathcal{P}}$ such that $0 \leq \mathcal{Z}_P \leq |P|$ is an integer. Since the number of such tuples is at most $n^{|\mathcal{P}|}$ and the number $|\mathcal{P}|$ of classes is independent of n , this yields a polynomial time algorithm for any fixed ε, k, C .

To speed things up, we use an idea developed in [14] for dense MAX CSP problems; this will eventually lead to the problem OPT'' detailed in Step 2 of **ApXCSP**. Instead of optimizing over all possible $(\mathcal{Z}_P)_{P \in \mathcal{P}}$, we could just enumerate all tuples $(\mathcal{T}_{ih}^\psi(1))_{i,h,\psi}$ with $0 \leq \mathcal{T}_{ih}^\psi(1) \leq |S_{ih}^\psi|$. The issue is that not all such tuples correspond to an assignment $\sigma : V \rightarrow \{0,1\}$ as in (5.8) and (5.9). Hence, for each tuple $(\mathcal{T}_{ih}^\psi(1))_{i,h,\psi}$ we will have to check feasibility, i.e., if there is a tuple $(\mathcal{Z}_P)_P$ such that (5.9) holds. Since we are just aiming to solve OPT' approximately, we can drop the requirement that all \mathcal{Z}_P must be integral. Thus, checking (5.9) turns into a linear programming problem. In effect, we can reduce the running time from $\exp(|\mathcal{P}| \cdot \ln n)$ to $\exp(sk2^{2^k} \cdot \ln n)$. (Remember that in general $|\mathcal{P}|$ is exponential in $sk2^{2^k}$.)

Finally, to remove the $\ln n$ factor, we chop each set S_{ih}^ψ into chunks of size $\nu = \lceil \delta n \rceil$, where $\delta > 0$ is bounded by a function of C, ε, k only. Hence, instead of optimizing over the number $0 \leq \mathcal{T}_{ih}^\psi(1) \leq |S_{ih}^\psi|$ of variables to be set to 1 in each S_{ih}^ψ , we optimize over the number $0 \leq \tau_{ih}^\psi(1) = \lfloor \mathcal{T}_{ih}^\psi(1) / \nu \rfloor \leq \lfloor |S_{ih}^\psi| / \nu \rfloor$ of chunks set to 1. This is sufficient because we just need to solve OPT' within an additive $\varepsilon 2^{-k-1} m$ (cf. (5.7)). Of course, for each $\tau_{ih}^\psi(1)$ the number of possible values is at most $1 + \delta^{-1}$, i.e., independent of n . To check feasibility, we then have to verify that there are $0 \leq z_P \leq 1$ ($P \in \mathcal{P}$) such that $\tau_{ih}^\psi(1) \nu \leq \sum_{P \in \mathcal{P}: P \subset S_{ih}^\psi} z_P \leq (\tau_{ih}^\psi(1) + 1) \nu$ for all i, h, ψ , which is again an LP problem. This leaves us with the optimization problem OPT'' quoted in Step 2 of **ApXCSP**. After finding an optimal solution to OPT'' , the algorithm sets up the assignment $\hat{\sigma}$ that mirrors the resulting z_P values. Bounding the errors incurred in the course of passing from OPT to OPT'' carefully, we obtain the following estimate.

PROPOSITION 5.1. *The assignment $\hat{\sigma}$ satisfies (5.7).*

Thus, to complete the proof of Theorem 1.3, we just need to bound the running time of **ApXCSP**. The argument is based on the bound on the number of classes in the partition \mathcal{P} from Theorem 1.2 (details omitted).

6 Examples.

6.1 Max Cut. To illustrate the use of **ApXCSP**, we present an example of bounded problem instances of **MAX CUT**. This shows how the present techniques provide a unified approach to problems that were previ-

ously studied by individually tailored methods, in particular in the average case analyses of algorithms.

Let $0 \leq p = p(n) \leq 1$ be a sequence of edge probabilities, and let $G_{n,p}$ be a random graph on n vertices $V = \{1, \dots, n\}$ obtained by including each of the $\binom{n}{2}$ possible edges with probability p independently. We say that $G_{n,p}$ has some property \mathcal{E} *with high probability* (“**whp**”) if the probability that \mathcal{E} holds tends to 1 as $n \rightarrow \infty$. For any graph G we let $\mathcal{I}(G)$ denote the set of all subgraphs H of G such that $|E(H)| \geq 0.01|E(G)|$. Furthermore, for a fixed $\varepsilon > 0$ we say that an algorithm \mathcal{A} *approximates MAX CUT within $1 - \varepsilon$ on $G_{n,p}$ -bounded graphs* if the following two conditions are satisfied:

1. For *any* input graph G the algorithm \mathcal{A} either outputs a cut that is within a $1 - \varepsilon$ factor of the maximum cut, or just outputs “fail”. In the first case we say that the algorithm *succeeds*, in the second case it *fails*.
2. If $G = G_{n,p}$ is a random graph, then with high probability \mathcal{A} succeeds for all graphs in $\mathcal{I}(G)$.

Thus, the algorithm *never* outputs a solution that is off by more than $1 - \varepsilon$, and for almost all outcomes $G = G_{n,p}$ it succeeds on all subgraphs $G_* \subset G$ that contain at least 1% of the edges of G . One can think of G_* being constructed by a malicious adversary, starting from the random graph G .

THEOREM 6.1. *Suppose that $np \geq c_0(\varepsilon)$ for a number $c_0(\varepsilon)$ that only depends on $\varepsilon > 0$. The polynomial time algorithm **ApxCSP** from Theorem 1.3 approximates MAX CUT within $1 - \varepsilon$ on $G_{n,p}$ -bounded graphs.*

Proof MAX CUT fits into the general CSP framework discussed in Section 1.4 as follows. The set of variables is the vertex set of the input graph $G_* = (V, E_*)$. Moreover, each edge $e = \{v, w\} \in E_*$ yields the (binary) constraint

$$\sigma \in \{0, 1\}^V \mapsto \begin{cases} 1 & \text{if } \sigma(v) \neq \sigma(w), \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the objective function value of the resulting CSP \mathcal{F} is just the number of crossing edges of a maximum cut of G_* .

Let $\varepsilon > 0$, and let γ be as in Theorem 1.3 with $C = 360$. We claim that if $np \geq c_0(\varepsilon)$ for a sufficiently large $c_0(\varepsilon) > 0$, then **whp** $G = G_{n,p}$ has the property that for any $G_* \in \mathcal{I}(G)$ the CSP instance \mathcal{F} is $(360, \gamma)$ -bounded. By the construction of \mathcal{F} , it is sufficient to show that the adjacency matrix $A = A(G_*)$ is $(180, \gamma)$ -bounded. To see this, consider any two sets $S, T \subset V$

of sizes $|S|, |T| \geq \gamma n$. Then

$$A(S, T) \leq 2e_{G_*}(S, T) \leq 2e_G(S, T).$$

Since $G = G(n, p)$ is a random graph, we have $E(2e_G(S, T)) \leq 2|S \times T|p$. Moreover, as $e_G(S, T)$ is binomially distributed, Chernoff bounds entail that

$$\begin{aligned} \mathbb{P}[2e_G(S, T) > 3|S \times T|p] &\leq \exp(-0.01|S \times T|p) \\ &\leq \exp(-0.01 \cdot \gamma^2 n^2 p) \\ &\leq \exp(-0.01\gamma^2 c_0(\varepsilon) \cdot n). \end{aligned}$$

Hence, if $c_0(\varepsilon)$ is sufficiently large, then $A(S, T) \leq 3|S \times T|p$ with probability at least $1 - \exp(-2n)$. Since there are at most 2^n ways to choose S, T , the union bound entails that **whp** for all pairs of sets S, T of size at least γn we have

$$(6.11) \quad A(S, T) \leq 3|S \times T|p.$$

Finally, let q be the density of A . Since the number of edges of $G(n, p)$ is $\binom{(1+o(1))n}{2p}$ **whp** (by Chernoff bounds), and since $G_* \in \mathcal{I}(G)$, we have $0.009p \leq q$ **whp**. Hence, (6.11) entails that $A(S, T) \leq 180|S \times T|q$ for all S, T of size at least γn **whp**, i.e., A is $(180, \gamma)$ -bounded. \square

Theorem 6.1 readily yields a result on the “planted model” for MAX CUT. In this model a random graph $G = G_{n,p,q}$ is generated by partitioning the vertex set $V = \{1, \dots, n\}$ randomly into two parts V_1, V_2 , inserting each possible V_1 - V_2 -edge with probability p , and each possible edge inside V_1, V_2 with probability $q < p$ independently. Improving upon prior work by Boppana [6], Coja-Oghlan [7] showed that a MAX CUT of $G_{n,p,q}$ can be computed in polynomial time **whp**, provided that $n(p - q) \geq \zeta \sqrt{np \ln(np)}$ for a certain constant $\zeta > 0$ (actually [6, 7] are stated in terms of MIN BISECTION, but things carry over to MAX CUT easily). Since the random graph $G_{n,p,q}$ can be obtained by first choosing $G_{n,p}$, then choosing a random partition (V_1, V_2) , and finally removing random edges inside of V_1, V_2 , Theorem 6.1 encompasses this model. In fact, Theorem 6.1 comprises various generalizations of the “planted cut” model (e.g., instead of planting a single cut, we could plant an arbitrary number of cuts, etc.).

6.2 MAX k -SAT. Let $V = \{x_1, \dots, x_n\}$ be a set of n propositional variables, and let $F_k(n, p)$ signify a k -SAT formula obtained by including each of the $(2n)^k$ possible k -clauses with probability $0 \leq p \leq 1$ independently (hence, we think of each clause as an order k -tuple of literals). Let $m = (2n)^k p$ denote the expected number of clauses. We say that $F_k(n, p)$ has some property \mathcal{E} *with high probability* if the probability

that \mathcal{E} holds tends to one as $n \rightarrow \infty$. Moreover, for any k -SAT formula F we let $\mathcal{I}(F)$ denote the set of all sub-formulas F_* of F that contain at least $0.01m$ clauses. Furthermore, for a fixed $\varepsilon > 0$ we say that an algorithm \mathcal{A} *approximates MAX k -SAT within $1 - \varepsilon$ on $F_k(n, p)$ -bounded formulas* if the following two conditions are satisfied:

1. For *any* input F the algorithm \mathcal{A} either outputs an assignment such that the number of satisfied clauses is within a $1 - \varepsilon$ factor of the optimum for MAX k -SAT or just outputs “fail”.
2. If $F = F_k(n, p)$, then **whp** \mathcal{A} succeeds on all formulas in $\mathcal{I}(F)$.

Similar arguments as in Section 6.1 show the following.

THEOREM 6.2. *Suppose that $k \geq 2$ is fixed and that $c_0(\varepsilon)n^{\lceil k/2 \rceil} \leq m = o(n^k)$ for a number $c_0(\varepsilon)$ that only depends on ε . The polynomial time algorithm **ApxCSP** from Theorem 1.3 approximates MAX k -SAT within $1 - \varepsilon$ on $F_k(n, p)$ -bounded formulas.*

In particular, Theorem 6.2 applies to the plainly random formula $F_k(n, p)$, in which case the algorithm yields a lower *and* an upper bound on the number of simultaneously satisfiable clauses. If $k \geq 3$, then for $m \geq c_0(\varepsilon)n^{\lceil k/2 \rceil}$ the optimal assignment of $F_k(n, p)$ satisfies a $1 - 2^{-k} + o(1)$ fraction of the clauses **whp**. Hence, **whp** the polynomial time algorithm **ApXSAT** yields a *proof* that there is no assignment satisfying more than a $1 - 2^{-k} + \varepsilon$ fraction of all clauses. The problem of deriving such a proof in polynomial time is known as the “strong refutation problem” for random k -SAT [9], and a number of authors have tailored algorithms specifically for this problem [8, 10, 13]. For even values of k , Theorem 6.1 matches the best known result [8]. However, for $k = 3$ the strong refutation problem can be solved under the weaker assumption $m \gg n^{3/2} \ln^6 n$ by other techniques [8].

References

- [1] N. Alon, A. Coja-Oghlan, H. Han, M. Kang, V. Rödl, M. Schacht: Quasi-randomness and algorithmic regularity for graphs with general degree distributions. Proc. 34th ICALP (2007) 789–800
- [2] N. Alon, W. Fernandez de la Vega, R. Kannan, M. Karpinski: Random Sampling and approximation of MAX-CSPs, J. Computer and System Sciences **67** (2003) 212–243
- [3] N. Alon, R.A. Duke, H. Lefmann, V. Rödl, R. Yuster: The algorithmic aspects of the regularity lemma. J. of Algorithms **16** (1994) 80–109
- [4] N. Alon, A. Naor: Approximating the cut-norm via Grothendieck’s inequality. Proc. 36th STOC (2004) 72–80
- [5] S. Arora, D. Karger, M. Karpinski: Polynomial time approximation schemes for dense instances of NP-hard problems, Journal of Computer and System Sciences **58** (1999) 193–210
- [6] R. Boppana: Eigenvalues and graph bisection: an average-case analysis. Proc 28th FOCS (1987) 280–285
- [7] A. Coja-Oghlan: A spectral heuristic for bisecting random graphs. Random Structures and Algorithms **29** (2006) 351–398
- [8] A. Coja-Oghlan, A. Goerdt, A. Lanka: Strong refutation heuristics for random k -SAT. Combinatorics, Probability, and Computing **16** (2007) 5–28
- [9] U. Feige: Relations between average case complexity and approximation complexity. Proc. 34th STOC (2002) 534–543
- [10] U. Feige, J.H. Kim, E. Ofek: Witnesses for non-satisfiability of dense random 3CNF formulas. Proc. 47th FOCS (2006) 497–508
- [11] W. Fernandez de la Vega: MAX-CUT has a randomized approximation scheme in dense graphs. Random Structures and Algorithms **8** (1996) 187–199
- [12] W. Fernandez de la Vega, R. Kannan, M. Karpinski, S. Vempala: Tensor decomposition and approximation algorithms for constraint satisfaction problems. Proc. 37th STOC (2005) 747–754
- [13] J. Friedman, A. Goerdt, M. Krivelevich: Recognizing more unsatisfiable random k -SAT instances efficiently. SIAM Journal on Computing **35** (2006) 408–430
- [14] A. Frieze, R. Kannan: Quick approximation to matrices and applications. Combinatorica **19** (1999) 175–220
- [15] S. Gerke, A. Steger: The sparse regularity lemma and its applications In: B. Webb (ed.): Surveys in combinatorics. Cambridge University Press (2005) 227–258
- [16] O. Goldreich, S. Goldwasser, D. Ron: Property testing and its connection to learning and approximation, Journal of the ACM **45** (1998) 653–750
- [17] T. Gowers: Lower bounds of tower type for Szemerédi’s uniformity lemma. Geometric and Functional Analysis **7** (1997) 322–337
- [18] J. Hastad: Some optimal inapproximability results. Journal of the ACM **48** (2001) 798–859
- [19] S. Khot, G. Kindler, E. Mossel, R. O’Donnell: Optimal inapproximability results for MAX-CUT and other 2-variable CSPs? Proc. 45th FOCS (2004) 146–154
- [20] Y. Kohayakawa: Szemerédi’s regularity lemma for sparse graphs. In F. Cucker, M. Shub (eds.): Foundations of computational mathematics (1997) 216–230
- [21] V. Rödl: unpublished.
- [22] E. Szemerédi: Regular partitions of graphs, Problèmes Combinatoires et Théorie des Graphes Colloques Internationaux CNRS **260** (1978) 399–401
- [23] L. Trevisan, G. Sorkin, M. Sudan, D. Williamson: Gadgets, approximation, and linear programming. SIAM Journal on Computing **29** (2000) 2074–2097