Maximum Bipartite vs. Triangle-Free Subgraph

Tamio-Vesa Nakajima
Department of Computer Science, University of Oxford, UK

Stanislav Živný
Department of Computer Science, University of Oxford, UK

Abstract

Given a (multi)graph G which contains a bipartite subgraph with ρ edges, what is the largest triangle-free subgraph of G that can be found efficiently? We present an SDP-based algorithm that finds one with at least 0.8823ρ edges, thus improving on the subgraph with 0.878ρ edges obtained by the classic Max-Cut algorithm of Goemans and Williamson. On the other hand, by a reduction from Håstad's 3-bit PCP we show that it is NP-hard to find a triangle-free subgraph with $(25/26 + \varepsilon)\rho \approx (0.961 + \varepsilon)\rho$ edges.

As an application, we classify the *Maximum Promise Constraint Satisfaction Problem*, denoted byMaxPCSP(G, H), for all bipartite G: Given an input (multi)graph X which admits a G-colouring satisfying ρ edges, find an H-colouring of X that satisfies ρ edges. This problem is solvable in polynomial time, apart from trivial cases, if H contains a triangle, and is NP-hard otherwise.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Problems, reductions and completeness

Keywords and phrases approximation, promise constraint satisfaction, triangle-free subgraphs

Digital Object Identifier 10.4230/LIPIcs.ICALP.2025.17

Category Track A: Algorithms, Complexity and Games

Related Version *Full version*: `https://arxiv.org/abs/2406.20069`

Funding *Tamio-Vesa Nakajima*: UKRI EP/X024431/1, Clarendon Fund Scholarship
Stanislav Živný: UKRI EP/X024431/1

Acknowledgements We thank the anonymous reivewers for their feedback. We also thank Jakub Opršal, who asked us about the complexity of maximisation PCSPs for graphs beyond cliques.

1 Introduction

Given an undirected (multi)graph G, what is the bipartite subgraph of G with the most edges? This problem, known as the maximum cut problem — or Max-Cut for short — is one of the most fundamental problems in computer science. Max-Cut was among the 21 problems shown to be NP-hard by Karp [22]. Papadimitriou and Yannakakis showed that Max-Cut is APX-hard [28] and thus does not admit a polynomial-time approximation scheme, unless P = NP. However, there are several simple 0.5-approximation algorithms. Goemans and Williamson used semidefinite programming and randomised rounding to design a 0.878-approximation algorithm [19]. Khot, Kindler, Mossel, O'Donnell, and Oleszkiewicz established the optimality of this algorithm [24, 26] under Khot's Unique Games Conjecture [23].

What if the task is merely finding a large triangle-free subgraph (rather than a bipartite one)?

While the Goemans-Williamson algorithm can still be used, as our main result we design an algorithm with a better approximation guarantee: If G contains a bipartite subgraph with

52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025).
Editors: Keren Censor-Hillel, Fabrizio Grandoni, Joel Ouaknine, and Gabriele Puppis; Article No. 17; pp. 17:1–17:16
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

ρ edges, our algorithm efficiently finds a triangle-free subgraph of G with 0.8823ρ edges.[1] Our algorithm is a randomised combination of the Goemans-Williamson original "random hyperplane algorithm", and an algorithm that first selects "long edges" (meaning edges for which the angle between the corresponding vectors from the SDP solution is above a certain threshold) and then applies a random hyperplane rounding, selecting "shorter edges" (still longer than some other threshold). The probability of the biased coin that selects one of the two algorithms depends on certain geometric quantities which guarantee that the resulting subgraph is indeed triangle-free. We complement our tractability result by showing that it is NP-hard to find a triangle-free subgraph with $(25/26 + \varepsilon)\rho \approx (0.961 + \varepsilon)\rho$ edges. This result is obtained by a reduction from Håstad's 3-bit PCP [20].

Our work falls within the broad framework of *constraint satisfaction problems.* A constraint satisfaction problem (CSP) is a problem like Max-Cut, i.e., a problem about finding an assignment of values to the given variables subject to the given constraints. In Max-Cut, the variables correspond to the vertices of the input graph, the values are just 0 and 1 (corresponding to the two sides of a cut), and the constraints are binary disequalities associated with the edges of the graph. Given a CSP, the computational task could be to find a solution maximising the number of satisfied constraints as in Max-Cut, or finding a (perfect) solution satisfying all constraints as in (hyper)graph colouring problems. A *promise* CSP (PCSP) is a CSP in which each constraint comes in two forms, a strong one and a weak one. The promise is that a solution exists using the strong versions of the constraints, while the (possibly easier) task is to find a solution using the weak constraints. A recent line of work by Austrin, Guruswami, and Håstad [2], Brakensiek and Guruswami [12], and Barto, Bulín, Krokhin, and Opršal [3] initiated a systematic study of PCSPs with perfect completeness, i.e., finding a solution satisfying all weak constraints given the promise that a solution satisfying all strong constraints exists. Canonical examples include approximate graph [18] and hypergraph [15, 1] colouring problems, e.g., finding a 6-colouring of a given 3-colourable graph.

The *approximation* of promise CSPs includes interesting computational problems, as discussed in the recent work of Barto, Butti, Kazda, Viola, and Živný [4]. One example is the following. Given a graph G that admits a 2-colouring of the vertices with a ρ-fraction of the edges coloured properly, find a 3-colouring of G with an $\alpha\rho$-fraction of the edges coloured properly, where $0 < \alpha \leq 1$ is the approximation factor. As a side result, we show that there is a 1-approximation algorithm; i.e., given a graph with a 2-colouring with ρ-fraction of non-monochromatic edges, one can efficiently find a 3-colouring of the graph with the same fraction of non-monochromatic edges. Our algorithm solves the Goemans-Williamson SDP for Max-Cut [19] and then rounds like Frieze and Jerrum for Max-3-Cut [17], cf. the full version [27] for details.

As can be easily shown (cf. Section 3), this implies that the following problem is also solvable efficiently. Fix a bipartite graph G with at least one edge and another graph H that *contains a triangle* and has a homomorphism from G.[2] Then, given an input graph X, finding a partial homomorphism from X to H[3] is solvable efficiently given the promise that such a partial homomorphism exists from X to G. Our hardness result immediately implies that this problem is NP-hard if H is *triangle-free.*

---

[1] We note that our algorithm is easy to extend to the case where edges have positive weights.
[2] I.e. there is an edge preserving map from $V(G)$ to $V(H)$.
[3] I.e. a map from $V(X)$ to $V(H)$ preserving ρ of the edges of X. When $\rho = |E(X)|$, such a map is known as an H-colouring [21].

In other words, we obtain a computational complexity dichotomy for the problem MaxPCSP(G, H) with a bipartite G (cf. Section 3 for all formal definitions and the precise statement of our result). This should be compared with the conjecture of Brakensiek and Guruswami [12] on the decision version of the approximate graph homomorphism problem; in their conjecture the "bipartite G" case is (trivially) tractable but the maximisation version, which we settle in the present paper, was open. (The precise statement of their conjecture and the relationship with our work can be found in Section 3.)

Related work

Bhangale and Khot gave an optimal inapproximability result for Max-3-LIN over non-Abelian groups [6]. The series of works by Bhangale, Khot, and Minzer [7, 8, 9, 10] established, among other things, hardness of approximation of certain perfectly satisfiable CSPs. We already mentioned in passing the very recent work of Barto et al. [4], which extends the algebraic approach from [3] to approximability of PCSPs.

The notion of Max-PCSPs is a natural generalisation of the well-studied notion of Max-CSPs. For finite-domain Max-CSPs, it is know that a certain rounding of the basic SDP relaxation gives, up to some ε, the UGC-optimal approximation ratio (in time doubly exponential in $1/\varepsilon$) [29, 30]. Recent work of Brakensiek et al. [13] states that parts of this hardness result are verbatim applicable to approximating certain Max-PCSPs. However, the Raghavendra-Steurer algorithm does not immediately give a 1-approximation algorithm due to the above-mentioned ε, even for Max-CSPs. Moreover, that result is established only for finite-domain Max-CSPs. Furthermore, the hardness results are conditional on the UGC. Our results (i) give approximation ratio 1, for MaxPCSP(K_2, K_3), (ii) give algorithms for infinite-domain structures, for MaxPCSP$(K_2, \mathfrak{G}_3)$, and (iii) involve unconditional NP-hardness, for the hardness of MaxPCSP$(K_2, \mathfrak{G}_3)$.

If one removes the promise of a large cut in G, then the problem of simply finding the largest triangle-free subgraph of G is NP-hard [33].

As a final remark, we note that the ultimate goal of our work is to understand the precise approximation factor for all promise CSPs, and thus identify where the transition from tractability to intractability occurs. This is an ambitious, long-term goal that would encompass many existing fundamental results. Theorem 2 gives an improved algorithmic and hardness result for the very concrete CSP from the title, i.e., finding a large triangle-free subgraph of a given graph (as measured with respect to the size of a largest bipartite subgraph). Theorem 3, together with Theorem 2, makes a substantial progress on the special case of the general goal, namely for which promise graph problems 1-approximation is possible (stated below as Corollary 4, cf. also Conjecture 6).

2 Preliminaries

For two nonzero vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$, we denote by $\angle(\mathbf{x}, \mathbf{y})$ the angle between $\mathbf{x}$ and $\mathbf{y}$ in radians; i.e., $\angle(\mathbf{x}, \mathbf{y}) = \arccos\left(\frac{\mathbf{x}\cdot\mathbf{y}}{||\mathbf{x}||||\mathbf{y}||}\right)$. The following useful fact is well-known, cf. [16, Book XI, Proposition 21].

▶ **Lemma 1.** *For any three nonzero vectors $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3 \in \mathbb{R}^N$, we have $\angle(\mathbf{x}_1, \mathbf{x}_2) + \angle(\mathbf{x}_2, \mathbf{x}_3) + \angle(\mathbf{x}_3, \mathbf{x}_1) \leq 2\pi$.*

Graphs and (partial) homomorphisms

All graphs will be undirected but with possibly multiple edges. Fix two graphs $G = (V, E)$, $H = (U, F)$, and $\rho \in \mathbb{N}$. We say that there exists a *partial homomorphism* of weight ρ from G to H, and write $G \xrightarrow{\rho} H$, if there exists a mapping $h : V \to U$ such that for ρ edges $(x, y) \in E$ we have $(h(x), h(y)) \in F$. If $G \xrightarrow{|E|} H$, we say that there exists a homomorphism from G to H and write $G \to H$. (Note that for any G, H, I, if $G \xrightarrow{\rho} H \to I$ then $G \xrightarrow{\rho} I$.)

We denote by K_2 a clique on two vertices. A partial homomorphism $h : G \xrightarrow{\rho} K_2$ represents a cut of weight ρ, namely the edges (x, y) with $h(x) \neq h(y)$. Equivalently, it represents a bipartite subgraph of G with weight ρ. We now introduce a graph that similarly captures triangle-free subgraphs. Let $\mathfrak{G}_3$ be the direct sum of all finite triangle-free graphs. In other words, for every finite triangle-free graph $G = (V, E)$, the graph $\mathfrak{G}_3$ contains vertices x_G for $x \in V$, and edges (x_G, y_G) for $(x, y) \in E$. Then, for finite G, a partial homomorphism $h : G \xrightarrow{\rho} \mathfrak{G}_3$ represents a triangle-free subgraph of G with weight ρ: all the edges that connect vertices that are mapped by h to neighbouring vertices in $\mathfrak{G}_3$ form a triangle-free subgraph of G.

Maximum PCSPs

Fix two (possibly infinite) graphs $G \to H$. Then the *maximum promise constraint satisfaction problem* (MaxPCSP) for undirected graphs, denoted by $\mathrm{MaxPCSP}(G, H)$, is defined as follows. In the search version of the problem, we are given a (multi)graph X such that $X \xrightarrow{\rho} G$, and must find $h : X \xrightarrow{\rho} H$; this problem can be approximated with the approximation ratio α if we can find $h : X \xrightarrow{\lceil \alpha\rho \rceil} H$. In the decision version, we are given a (multi)graph X and a number $\rho \in \mathbb{N}$ and must output YES if $X \xrightarrow{\rho} G$, and NO if not even $X \xrightarrow{\rho} H$. This problem can be approximated with approximation ratio α if we can decide between $X \xrightarrow{\rho} G$ and not even $X \xrightarrow{\lceil \alpha\rho \rceil} H$. (In all cases, ρ is *not* part of the input.)

In particular, approximating the problem $\mathrm{MaxPCSP}(K_2, \mathfrak{G}_3)$ with approximation ratio α means the following. In the search version: given a graph G that contains a cut of weight ρ, find a triangle-free subgraph of weight $\alpha\rho$. In the decision version: given a graph G and a number $\rho \in \mathbb{N}$, output YES if it has a cut of weight ρ, and NO if it has no triangle-free subgraph of weight $\alpha\rho$.

We define the problem $\mathrm{PCSP}(G, H)$ identically to $\mathrm{MaxPCSP}(G, H)$, except that it is guaranteed that ρ is the number of edges of G. Thus observe that $\mathrm{PCSP}(G, H)$ reduces to $\mathrm{MaxPCSP}(G, H)$ trivially, in the sense that there is a polynomial-time reduction from $\mathrm{PCSP}(G, H)$ to $\mathrm{MaxPCSP}(G, H)$ that does not change the input.

Suppose $G \to G' \to H' \to H$. Then, it follows that $\mathrm{PCSP}(G, H)$ polynomial-time reduces to $\mathrm{PCSP}(G', H')$ and $\mathrm{MaxPCSP}(G, H)$ polynomial-time reduces to $\mathrm{MaxPCSP}(G', H')$ (and the same holds for α-approximation). Furthermore, the decision version of $\mathrm{PCSP}(G, H)$ and $\mathrm{MaxPCSP}(G, H)$ polynomial-time reduces to the search version of $\mathrm{PCSP}(G, H)$ and $\mathrm{MaxPCSP}(G, H)$, respectively. In other words, the decision version is no harder than the search version. Hence by proving our tractability results for the search version, and our hardness results for the decision version, we prove them for both versions of the problems.

SDP

For the Max-Cut problem, which is just $\mathrm{MaxPCSP}(K_2, K_2)$, the *basic SDP relaxation* for a graph $G = (V, E)$ with n vertices, which can be solved within additive error ε in polynomial time with respect to the size of G and $\log(1/\varepsilon)$,[4] is as follows:

$$\begin{aligned} \max \quad & \sum_{(u,v)\in E} \frac{1-\mathbf{x}_u\cdot\mathbf{x}_v}{2} \\ \text{s.t.} \quad & \mathbf{x}_u\cdot\mathbf{x}_u = 1, \\ & \mathbf{x}_u \in \mathbb{R}^n. \end{aligned} \tag{1}$$

Goemans and Williamson [19] gave a rounding algorithm for the SDP (1) with approximation ratio

$$\alpha_{GW} = \left(\max_{0\le\tau\le\pi} \frac{\pi}{2}\frac{1-\cos\tau}{\tau}\right)^{-1} = 0.878\cdots,$$

thus beating the trivial approximation ratio of 1/2 obtained by, e.g., a random cut. Their algorithm solves the SDP (1), selects a uniformly random hyperplane in $\mathbb{R}^N$, and returns the cut induced by the hyperplane.

3 Results

Our main result is the following.

▶ **Theorem 2.** MaxPCSP$(K_2, \mathfrak{G}_3)$ *is* 0.8823*-approximable (in the search version) in polynomial time, and it is* NP*-hard to* $(25/26+\varepsilon)$*-approximate (even in the decision version) for any fixed* $\varepsilon > 0$.

Note that crucially $0.8823 > 0.878\ldots$, thus our algorithm beats the Goemans-Williamson algorithm.

We prove this result in two parts: the tractability side in Section 4 and the hardness side in Section 5. We quickly give an intuitive explanation of why such an algorithm is possible. Define

$$\tau_{GW} = \arg\max_{0\le\tau\le\pi} \frac{\pi}{2}\frac{1-\cos\tau}{\tau} \approx 0.742\pi.$$

The function $\tau \mapsto (\pi/2)(1-\cos\tau)/\tau$, depicted in Figure 1, is increasing up to τ_{GW}, then decreasing. Why would MaxPCSP$(K_2, \mathfrak{G}_3)$ be easier to approximate than MaxPCSP(K_2, K_2)? Consider the Goemans-Williamson algorithm for MaxPCSP(K_2, K_2); the worst-case performance of this algorithm appears in a graph where the embedding into $\mathbb{R}^N$ given by solving the SDP (1) gives all edges an angle of approximately τ_{GW}. Observe that $\tau_{GW} > 2\pi/3$ — so immediately (cf. Lemma 1) this instance is triangle-free. So, for this instance an algorithm for MaxPCSP$(K_2, \mathfrak{G}_3)$ could just return the entire graph! Indeed, in order to create an instance that contains triangles one needs to introduce shorter edges. This suggests that a hybrid algorithm, that either selects "long edges" or some appropriate selection of "shorter edges" (still longer than some threshold), should have better performance.

Our next result is a combination of the Goemans-Williamson SDP for Max-Cut [19] and a rounding scheme due to Frieze and Jerrum for Max-3-Cut [17], cf. [27] for details.

▶ **Theorem 3.** MaxPCSP(K_2, K_3) *is solvable in polynomial time (in the search version).*

[4] Throughout we will ignore issues of real precision.

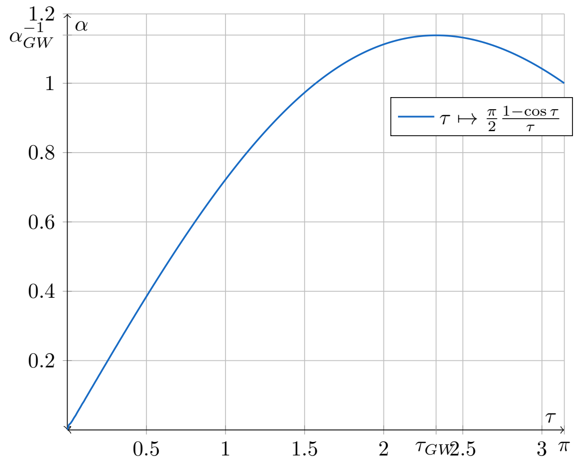


Figure 1 Function giving rise to α_{GW}, τ_{GW}.

This algorithm is somewhat similar to the Goemans-Williams algorithm, except that rather than selecting a uniformly random hyperplane, it selects three normally distributed vectors, and partitions the vertices according to which vector they are closest to (where closeness is measured in terms of inner products). Thus, while this algorithm solves the same SDP as the algorithm from Theorem 2, it rounds the solution differently.

Combining Theorem 3 with (the hardness part of) Theorem 2, we obtain a complexity classification:

▶ **Corollary 4.** *Let G be bipartite graph with at least one edge. Then,* $\mathrm{MaxPCSP}(G, H)$ *is solvable in polynomial time if H has a loop or contains a triangle (in the search version), and is* NP*-hard to solve if H is triangle-free (in the decision version).*[5]

Proof. If H contains a triangle then $G \to K_2 \to K_3 \to H$, so $\mathrm{MaxPCSP}(G, H)$ reduces to $\mathrm{MaxPCSP}(K_2, K_3)$, which can be solved in polynomial time by Theorem 3. Conversely, if H is triangle-free then $K_2 \to G \to H \to \mathfrak{G}_3$, so $\mathrm{MaxPCSP}(K_2, \mathfrak{G}_3)$ reduces to $\mathrm{MaxPCSP}(G, H)$. Hence, by Theorem 2, it is NP-hard even to approximate $\mathrm{MaxPCSP}(G, H)$ with approximation ratio $25/26 + \varepsilon$ (in the decision version), and thus in particular $\mathrm{MaxPCSP}(G, H)$ is NP-hard (in the decision version). ◀

Brakensiek and Guruswami conjectured the tractability boundary of promise CSPs on undirected graphs.

▶ **Conjecture 5** ([12])**.** *If G, H are undirected graphs, then* $\mathrm{PCSP}(G, H)$ *is tractable if G is bipartite or H has a loop; and is* NP*-hard otherwise.*

As $\mathrm{PCSP}(G, H)$ reduces to $\mathrm{MaxPCSP}(G, H)$, we can see that it follows from Conjecture 5 that $\mathrm{MaxPCSP}(G, H)$ is NP-hard whenever G is non-bipartite and H is loopless. Since we already classified the case when G is bipartite but not edgeless, we conjecture that the tractability boundary for maximum PCSPs on undirected graphs is the following:

▶ **Conjecture 6.** *If G, H are undirected graphs, then* $\mathrm{MaxPCSP}(G, H)$ *is tractable if (i) G has no edges, or (ii) H has a loop, or (iii) G is bipartite and H contains a triangle; and is* NP*-hard otherwise.*

[5] Note that if G has no edges or H has a loop then $\mathrm{MaxPCSP}(G, H)$ is trivially tractable.

Resolving Conjecture 6 should be easier than resolving Conjecture 5, similarly to how the complexity of finite-valued CSPs [31] was resolved before the complexity of decision CSPs [14, 34]

We note that our hardness result depends in an essential way on the fact that the input graph can have multiple edges; we equivalently could have allowed non-negative integer weights on the edges. This variant of the problem is most natural when looking at it as a constraint satisfaction problem. It is interesting to ask what the complexity of $\mathrm{MaxPCSP}(K_2, \mathfrak{G}_3)$ is if the input graph is both weightless and without multiple edges.

4 Tractability

Our tractability result is based on the following lemma.

▶ **Lemma 7.** *There exist $\alpha, P, Q, \tau \in \mathbb{R}$ with $P + Q = 1, P \geq 0, Q \geq 0,$ $\tau \in [2\pi/3, \tau_{GW}]$, such that the following hold*

$$P\frac{\theta}{\pi} + Q \geq \alpha\frac{1-\cos\theta}{2} \qquad \theta \in [\tau, \pi] \tag{2}$$

$$P\frac{\varphi}{\pi} + Q\frac{\varphi}{\pi} \geq \alpha\frac{1-\cos\varphi}{2} \qquad \varphi \in [\pi - \tau/2, \tau] \tag{3}$$

$$P\frac{\psi}{\pi} \geq \alpha\frac{1-\cos\psi}{2} \qquad \psi \in [0, \pi - \tau/2]. \tag{4}$$

In particular, we can take $\alpha \geq 0.88232, \tau = 2.18746$, $Q = 1 - P$ and

$$P = \frac{\alpha\pi}{2}\left(\frac{1-\cos(\pi-\tau/2)}{\pi-\tau/2}\right) \approx 0.987535.$$

Proof. Firstly (4) is most tight when $\psi = \pi - \tau/2$, so it reduces to

$$P \geq \frac{\alpha\pi}{2}\underbrace{\left(\frac{1-\cos(\pi-\tau/2)}{\pi-\tau/2}\right)}_{X(\tau)}.$$

Next, (3) is actually independent of P, Q, since $P + Q = 1$ and thus the left-hand side is just φ/π. Since we will take $\tau \leq \tau_{GW}$, it is also most tight when $\varphi = \tau$ (cf. Figure 1), so it becomes

$$\alpha \leq \frac{2}{\pi}\frac{\tau}{1-\cos\tau}.$$

For (2) we see that it becomes easier to satisfy if P is smaller and Q is larger, so we will choose P to be the minimum value it could have vis-à-vis (3) and (4), namely

$$P = \frac{\alpha\pi}{2}X(\tau).$$

Hence, as $P + Q = 1$, the first constraint becomes

$$\frac{\alpha\theta}{2}X(\tau) + 1 - \frac{\alpha\pi}{2}X(\tau) \geq \alpha\frac{1-\cos\theta}{2}.$$

Simplifying, we get

$$\alpha\frac{\theta-\pi}{2}X(\tau) + 1 \geq \alpha\frac{1-\cos\theta}{2}.$$

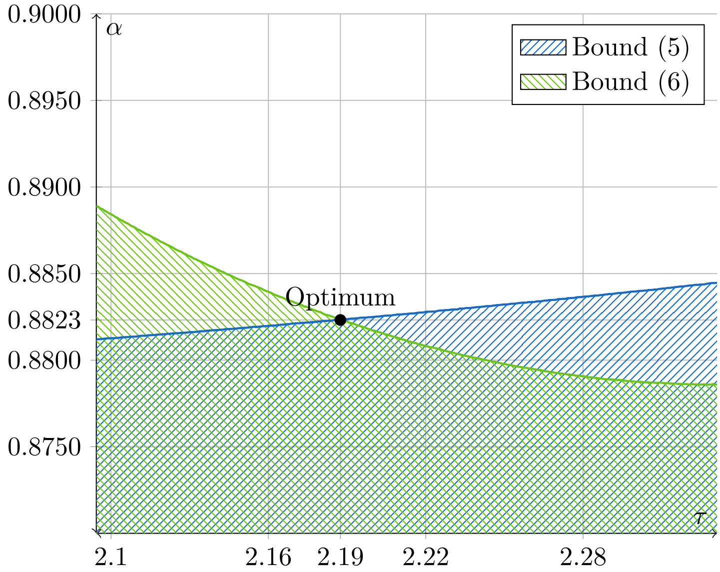


Figure 2 Bounds from Lemma 7

This is equivalent to

$$\alpha \leq \frac{2}{(\pi - \theta)X(\tau) - \cos\theta + 1}$$

Separating out α, we have

$$\alpha \leq \min_{\theta} \frac{2}{(\pi - \theta)X(\tau) - \cos\theta + 1} = 2\left(\max_{\theta}(\pi - \theta)X(\tau) - \cos\theta + 1\right)^{-1}$$

Now we can observe that $0.75 > X(\tau) > 0.6$ for $2\pi/3 \leq \tau \leq \pi$ numerically. The derivative of the function we are maximising is $-X(\tau) + \sin\theta$ — this is necessarily positive at $\theta = 2\pi/3$ and negative for $\theta = \pi$, because of the approximation of $X(\tau)$ above. So the maximum is hit at $\pi - \arcsin X(\tau)$, as this is the solution to the equation within the bound for θ. Thus the bound is

$$\alpha \leq \frac{2}{X(\tau)\arcsin X(\tau) + \cos\arcsin X(\tau) + 1}$$

So we now want to find τ that maximises α such that

$$\alpha \leq \frac{2}{X(\tau)\arcsin X(\tau) + \cos\arcsin X(\tau) + 1} \tag{5}$$

$$\alpha \leq \frac{2}{\pi}\frac{\tau}{1 - \cos\tau} \tag{6}$$

We can see this situation in Figure 2. Numerically we compute that if we choose $\tau = 2.18746$, then we get $\alpha \geq 0.88232$. ◀

We now prove our tractability result.[6]

[6] We remark in passing that *no* SDP-based algorithm can have performance greater than $8/9 = 0.888\ldots$, since for the triangle K_3 the SDP value is $9/4$, yet the largest triangle-free subgraph has weight 2 (and $2/(9/4) = 8/9$).

▶ **Theorem 8.** $\mathrm{MaxPCSP}(K_2, \mathfrak{G}_3)$ *can be* 0.8823*-approximated in polynomial time.*

Proof. Take α, P, Q, τ as in Lemma 7. Consider the following randomised algorithm.

1. Input: a graph $G = (V, E)$, which admits a cut of weight $\rho \geq |E|/2$.
2. Solving SDP (1) to within error ε, we get a set of vectors $\mathbf{x}_v$ for $v \in V$, with $||\mathbf{x}_v||^2 = 1$ and
$$\sum_{(u,v)\in E} \frac{1 - \mathbf{x}_u \cdot \mathbf{x}_v}{2} \geq \rho - \varepsilon.$$
3. Flip a biased coin, randomly choosing from the following two cases.
   (i) With probability P, sample a uniformly random hyperplane H, and compute the set of edges (u, v) with $\mathbf{x}_u$ on the opposite side of H as $\mathbf{x}_v$. Return this set of edges.
   (ii) With probability Q, return all the edges (u, v) with $\angle(\mathbf{x}_u, \mathbf{x}_v) > \tau$, then sample a uniformly random hyperplane H and additionally return all the edges (u, v) with $\angle(\mathbf{x}_u, \mathbf{x}_v) \geq \pi - \tau/2$ and with $\mathbf{x}_u, \mathbf{x}_v$ on opposite sides of H.

First, let us verify that our algorithm returns a triangle-free subgraph. First, in Case (i), we return a bipartite subgraph, so we certainly return a triangle-free subgraph. The reasoning for Case (ii) is more geometric. Consider any three edges returned in this case. If the three edges are of angle between $\pi - \tau/2$ and τ, then they cannot form a triangle, since the edges of this kind that we return form a bipartite graph (determined by H). Conversely, suppose that at least one edge has angle greater than τ, and the other two have angle at least $\pi - \tau/2$. Then the sum of the angles of the edges are $> 2\pi$, and hence by Lemma 1, they cannot form a triangle.

Now, let us compute the expected performance of our algorithm. Recall that for a uniformly random hyperplane H, two unit vectors $\mathbf{x}, \mathbf{y}$ are on opposite sides of H with probability $\angle(\mathbf{x}, \mathbf{y})/\pi$ [19]. Consider any edge (u, v) in our graph. We will show that the probability that the edge is included in the cut is at least $\alpha(1 - \mathbf{x}_u \cdot \mathbf{x}_v)/2 = \alpha(1 - \cos\angle(\mathbf{x}_u, \mathbf{x}_v))/2$; there are three cases depending on $\angle(\mathbf{x}_u, \mathbf{x}_v)$.

$\angle(\mathrm{x}_u, \mathrm{x}_v) \in (\tau, \pi]$ In this case, the edge is included with probability $\angle(\mathbf{x}_u, \mathbf{x}_v)/\pi$ in Case (i), and with probability 1 in Case (ii). Thus, and by applying (2) from Lemma 7, we find that the edge is included with probability
$$P\frac{\angle(\mathbf{x}_u, \mathbf{x}_v)}{\pi} + Q \geq \alpha\frac{1 - \cos\angle(\mathbf{x}_u, \mathbf{x}_v)}{2}.$$

$\angle(\mathrm{x}_u, \mathrm{x}_v) \in [\pi - \tau/2, \tau]$ In this case, the edge is included with probability $\angle(\mathbf{x}_u, \mathbf{x}_v)/\pi$ both in Case (i) and in Case (ii). Thus, by (3) from Lemma 7, the edge is included with probability
$$P\frac{\angle(\mathbf{x}_u, \mathbf{x}_v)}{\pi} + Q\frac{\angle(\mathbf{x}_u, \mathbf{x}_v)}{\pi} \geq \alpha\frac{1 - \cos\angle(\mathbf{x}_u, \mathbf{x}_v)}{2}.$$

$\angle(\mathrm{x}_u, \mathrm{x}_v) \in [0, \pi - \tau/2)$ In this case, the edge is included with probability $\angle(\mathbf{x}_u, \mathbf{x}_v)/\pi$ in Case (i) and never included in Case (ii). Thus, by (4) from Lemma 7, the edge is included with probability
$$P\frac{\angle(\mathbf{x}_u, \mathbf{x}_v)}{\pi} \geq \alpha\frac{1 - \cos\angle(\mathbf{x}_u, \mathbf{x}_v)}{2}.$$

Hence overall we include an edge (u, v) with probability at least $\alpha(1 - \mathbf{x}_u \cdot \mathbf{x}_v)/2$. Since

$$\sum_{(u,v)\in E} \frac{1 - \mathbf{x}_u \cdot \mathbf{x}_v}{2} \geq \rho - \varepsilon,$$

it follows that we return a triangle-free subgraph with expected weight $\alpha(\rho - \varepsilon)$.

Now, we derandomise our algorithm. First, note that rather than randomly choosing between Case (i) and Case (ii), we can just run both cases and return the better of the two solutions. Secondly, using the techniques of [25] (or, more efficiently [11]), we can derandomise the choice of random hyperplane in polynomial time, at the cost of losing ε' potential value, in polynomial time in $\log(1/\varepsilon')$. Hence, in polynomial time in the size of the graph and $\varepsilon, \varepsilon'$ we return a triangle-free subgraph with weight $\alpha(\rho - \varepsilon) - \varepsilon' \geq \alpha\rho - (\varepsilon + \varepsilon')$.

To complete the proof, note that we take $\alpha \geq 0.88232$, but we only advertise an approximation ratio of 0.8823. Hence, the subgraph we return has weight $0.8823\rho + 2 \cdot 10^{-5}\rho - (\varepsilon + \varepsilon')$, and since $\rho \geq |E|/2 \geq 1/2$ (since if $|E| = 0$ the problem is trivial), this is at least $0.8823\rho + (10^{-5} - \varepsilon - \varepsilon')$. Hence it is sufficient to choose $\varepsilon + \varepsilon' = 10^{-5}$. The algorithm then runs in polynomial time in the size of G, and finds a triangle-free subgraph with weight 0.8823ρ, as required. ◀

5 Hardness results

Our general strategy will be to gadget reduce from the 3-bit PCP of Håstad [20], similarly to [32] or [5]. The main difficulty comes in from the fact that it is not possible to "negate" variables in an obvious way, since "negation" is not globally preserved by the property of being triangle-free, as opposed to that of being bipartite. Some mild complications will be forced by this. Recall first the definition of exactly-3 linear equations.

▶ **Definition 9.** *In the problem* $\mathsf{E3Lin}_\delta$*, one is given a system of mod-2 linear equations with exactly 3 variables per equation; i.e.* $x + y + z \equiv 0 \bmod 2$ *or* $x + y + z \equiv 1 \bmod 2$*. If it is possible to simultaneously solve a* $1 - \delta$ *fraction of all the equations, one must answer* Yes*; otherwise, if it is not even possible to simultaneously solve a* $\frac{1}{2} + \delta$ *fraction of the equations, one must answer* No*.*

▶ **Theorem 10** ([20])**.** *For every small enough* δ*, the problem* $\mathsf{E3Lin}_\delta$ *is* NP*-hard.*

To deal with our negation problems, we will need a "balanced" version of this problem.

▶ **Definition 11.** *In the problem* $\mathsf{BalancedE3Lin}_\delta$*, one is given a system of mod-2 linear equations with exactly 3 variables per equation; i.e.* $x+y+z \equiv 0 \bmod 2$ *or* $x+y+z \equiv 1 \bmod 2$*. Furthermore, the number of equations of the two types is equal. A* balanced solution *to such a system of equations is one that satisfies exactly as many equations of form* $x+y+z \equiv 0 \bmod 2$ *as those of form* $x + y + z \equiv 1 \bmod 2$*.*

If it is possible to find a balanced solution that satisfies a $1 - \delta$ *fraction of all the equations, one must answer* Yes*; otherwise, it if is not even possible to find any (*possibly even unbalanced*) solution that satisfies a* $\frac{1}{2} + \delta$ *fraction of the equations, one must answer* No*.*

We believe that [20] proves, without being explicit, NP-hardness of $\mathsf{BalancedE3Lin}_\delta$, although it is not straightforward to see it from the proof in [20]. For completeness, we provide a simple, self-contained reduction.

▶ **Lemma 12.** *For every small enough* δ*, the problem* $\mathsf{BalancedE3Lin}_\delta$ *is* NP*-hard.*

Proof. We reduce from $\mathsf{E3Lin}_\delta$ to $\mathsf{BalancedE3Lin}_\delta$.

Reduction

Given a system of m equations $\mathcal{E}$ on n variables, which contains the equations $x_i + y_i + z_i \equiv p_i \bmod 2$ for $i \in [m]$,[7] we define the system of equations $\mathcal{E}'$ on n variables, which contains the equations $x_i' + y_i' + z_i' \equiv 1 - p_i \bmod 2$ for $i \in [m]$. We then return the system of equations $\mathcal{E} \sqcup \mathcal{E}'$ i.e. the disjoint union of the two systems.

Completeness

Suppose that $\mathcal{E}$ has a solution $x_i \mapsto c(x_i)$ that satisfies a $1-\delta$ fraction of the equations. Then the system $\mathcal{E} \sqcup \mathcal{E}'$ has a balanced solution that also satisfies a $1 - \delta$ fraction of its equations, namely the one that sends x_i to $c(x_i)$ and x_i' to $1 - c(x_i)$. This solution is balanced since every equation $x_i + y_i + z_i \equiv p_i \bmod 2$ that it solves within $\mathcal{E}$ can be paired up with an equation $x_i' + y_i' + z_i' \equiv 1 - p_i \bmod 2$ which is solved in $\mathcal{E}'$.

Soundness

Suppose $\mathcal{E} \sqcup \mathcal{E}'$ has a solution $x_i \mapsto c(x_i)$, $x_i' \mapsto d(x_i')$, which satisfies a $\frac{1}{2} + \delta$ fraction of the equations. Such a solution must either satisfy a $\frac{1}{2} + \delta$ fraction of the equations within $\mathcal{E}$, or a $\frac{1}{2} + \delta$ fraction of the equations within $\mathcal{E}'$. In the first case, c is the required solution for the original problem; in the second, $x_i \mapsto 1 - d(x_i')$ is the required solution. ◀

We now define the notion of "gadget" that we will need for this particular reduction. This is along the same lines as [5, 32], but (i) generalised to deal with promise problems and (ii) specialised to our particular promise problem.

For the following, if $G = (V, E)$ is any bipartite graph, and $V' \subseteq V$, then we say that a function $c : V' \to \{0, 1\}$ is compatible with G if it is possible to extend c into a 2-colouring of G.

▶ **Definition 13.** *A gadget with performance $\alpha \in \mathbb{N}$ and parity $p \in \{0, 1\}$ is a graph $G = (V, E)$, with $0, x, y, z \in V$, where the following hold.*

1. *For any function $c : \{0, x, y, z\} \to \{0, 1\}$ such that $c(0) + c(x) + c(y) + c(z) \equiv p \bmod 2$, there exists a bipartite subgraph H of G with α edges, such that c is compatible with H.*
2. *Any triangle-free subgraph of G has at most α edges.*
3. *Every triangle-free subgraph H of G with strictly more than $\alpha - 1$ edges puts $0, x, y, z$ in the same connected component C, and the distance from 0 to x, y, z respectively is at most 2. Furthermore C is bipartite, and for any $c : \{0, x, y, z\} \to \{0, 1\}$ that is compatible with C, we have that $c(0) + c(x) + c(y) + c(z) \equiv p \bmod 2$.*

▶ **Lemma 14.** *Suppose we have n containers with capacities $c_1, \ldots, c_n \geq 0$. Suppose we distribute a volume of $c_1 + \cdots + c_n - n + a$ among the containers, distributing $v_i \leq c_i$ volume to container i. Then the number of containers i for which $v_i > c_i - 1$ is at least a.*

Proof. If we want to minimise the number of containers with $v_i > c_i - 1$, it is optimal first to distribute $c_i - 1$ volume to each container i, and then for those containers where we are forced to distribute more to distribute up to c_i. The first step uses up $c_1 + \cdots + c_n - n$ volume, leaving us with a volume to distribute. This volume of a cannot push fewer than $\lceil a \rceil$ of the containers above $c_i - 1$, since once pushed above this one can then add only 1 further unit to that container. ◀

[7] As usual, we denote by $[m]$ the set $\{1, \ldots, m\}$.

The next theorem encodes our reduction from the 3-bit PCP of [20] to $\mathrm{MaxPCSP}(K_2, \mathfrak{G}_3)$. This reduction is standard, needing only some care to deal with the fact that the triangle-free graph selected in the soundness case must be "connected enough".

▶ **Theorem 15.** *Suppose that for $i \in \{0, 1\}$ there exist gadgets G_i with performance α_i and parity i. Then it is* NP*-hard to approximate* $\mathrm{MaxPCSP}(K_2, \mathfrak{G}_3)$ *with approximation ratio* $1 - 1/(\alpha_0 + \alpha_1) + \varepsilon$.

Proof. We reduce $\mathsf{BalancedE3Lin}_\delta$ to approximate $\mathrm{MaxPCSP}(K_2, \mathfrak{G}_3)$ with approximation ratio $1 - 1/(\alpha_0 + \alpha_1) + \varepsilon$. Our choice of δ will be bounded above by a value that depends on ε. We now describe our gadget reduction.

Reduction

Suppose we are given an instance of $\mathsf{BalancedE3Lin}_\delta$, with $2m$ constraints and n variables V. Suppose that the constraints are $x_i^0 + y_i^0 + z_i^0 \equiv 0 \bmod 2$ and $x_i^1 + y_i^1 + z_i^1 \equiv 1 \bmod 2$ for $i \in [m]$. Our reduction first creates $n + 1$ vertices, one for every variable, plus a special vertex denoted by $0'$. For every constraint $x_i^p + y_i^p + z_i^p = p \bmod 2$ we create a copy of G_p, identifying vertices $0, x, y, z$ with $0', x_i^p, y_i^p, z_i^p$.

Completeness

Suppose that our $\mathsf{BalancedE3Lin}$ instance has a balanced solution that satisfies at least a $1 - \delta$ fraction of the constraints. We claim that the graph outputted by our algorithm has a cut with at least $m(1 - \delta)(\alpha_0 + \alpha_1)$ edges. Indeed, this cut is guaranteed by Item 1 in Definition 13: place the vertices corresponding to variables in the original problem in the cut according to their value; then the vertices in the gadgets are placed according to the cut guaranteed by this assumption. The cut then has at least $m(1 - \delta)(\alpha_0 + \alpha_1)$ edges because the solution is guaranteed to be balanced.

Soundness

Suppose that the graph outputted by our reduction has a triangle-free subgraph S with $m(\alpha_0 + \alpha_1 - 1 + \delta) = m(\alpha_0 + \alpha_1) - 2m + m + 2m\delta$ edges. We call a constraint with parity i "good" if the intersection of S with the gadget corresponding to that constraint has weight greater than $\alpha_i - 1$. Observe that at least $m + 2m\delta$ constraints must be good by Lemma 14: the gadgets for the $2m$ constraints are the containers; by Item 2 from Definition 13, m of the capacities are α_0, and m are α_1; finally, a constraint is "good" if it's container is allocated strictly more volume than it's capacity minus one. We now show how to create a solution to the original $\mathsf{BalancedE3Lin}$ instance that satisfies all the good constraints: this solution then satisfies a $(m + 2m\delta)/2m = 1/2 + \delta$ fraction of the constraints.

To create our solution, consider the subgraph S. Find the shortest path from $0'$ to every variable, then set the variable to the parity of the length of that path. (If there is no path, then we can set that variable arbitrarily.) Consider now any good constraint: suppose it is $x + y + z \equiv p \bmod 2$. Suppose S' is the intersection of S with the gadget for this constraint. Since S is triangle-free, S' must be triangle-free; since the constraint is good, the weight of S' is strictly greater than $\alpha_i - 1$. By Item 3 of Definition 13, S' is bipartite; furthermore S' connects $0'$ to x, y, z by paths of length at most 2. Thus the shortest path from $0'$ to x, y, z *in S (as well as S')* is also of length at most 2 as well. Since S is triangle-free, the parity of the length of the shortest path from $0'$ to x, y, z in S matches the parity of any path from $0'$ to x, y, z in S'. (To see why: if this were not the case, then for one of x, y, z, there are two paths from $0'$ of different parities, both of length at most 2. The first path is the one that exists in S by assumption, the second one is the one that exists in S' by Item 3 of Definition 13. This implies a triangle in S, a contradiction.) Colouring according to the sides of the bipartite graph S' satisfies the constraint according to Item 3 of Definition 13, and our colouring is the same as this. So all good constraints are satisfied.

Hardness factor

Thus we have shown that given a BalancedE3Lin instance with a balanced solution that satisfies a $1-\delta$ fraction of the equations, our reduction yields a graph with a cut with $m(1-\delta)(\alpha_0+\alpha_1)$ edges; and if the graph we output has a triangle-free subgraph with at least $m(\alpha_0+\alpha_1-1+\delta)$ edges the original instance must have had a solution that satisfies at least a $1/2+\delta$ fraction of equations. It follows that $\mathrm{MaxPCSP}(K_2,\mathfrak{G}_3)$ is NP-hard to approximate with approximation ratio

$$\frac{m(\alpha_0+\alpha_1-1+\delta)}{m(1-\delta)(\alpha_0+\alpha_1)}=\left(1-\frac{1}{\alpha_0+\alpha_1}+O(\delta)\right)(1+O(\delta))=1-\frac{1}{\alpha_0+\alpha_1}+O(\delta).$$

This can be made to be less than $1-\frac{1}{\alpha_0+\alpha_1}+\varepsilon$ by setting δ small enough. ◀

We now exhibit the gadgets. The first gadget is identical to a gadget of Bellare, Goldreich and Sudan [5] (although our analysis is slightly more complicated). In [5], this gadget is called “PC-CUT”, defined immediately before [5, Claim 4.17]. The second gadget is a generalisation of the first. Recall that the gadgets of [5] were improved in [32], and indeed the results of [32] indicate a generic method to find optimal gadgets for finite-domain CSPs. We do not believe this approach directly applies to our case because the property of being triangle-free is not captured by any finite CSP template (indeed, $\mathfrak{G}_3$ is infinite, and any homomorphism-equivalent structure must also be).

We will write our gadgets as graphs with non-negative integer weights for simplicity of presentation. These gadgets can then be implemented by adding edges multiple times.

▶ **Lemma 16.** *There exists a gadget with performance 9 and parity 1.*

Proof. Consider the complete graph on $\{0,a,x,y,z\}$. Suppose $w(a)=2, w(0)=w(x)=w(y)=w(z)=1$, and give an edge (i,j) weight $w(i,j)=w(i)w(j)$. We now show that this satisfies the conditions from Definition 13.

1. Without loss of generality suppose $c(0)=0$. We have two cases. First, suppose that $c(x)=c(y)=c(z)=1$. Then our cut is $(\{x,y,z\},\{a,0\})$ which has weight 9. Conversely, suppose without loss of generality that $c(x)=c(y)=0, c(z)=1$. Then our cut is $(\{0,x,y\},\{a,z\})$, which also has weight 9.
2. Consider any triangle-free subgraph of the gadget. This graph is either bipartite or it is C_5. If it is bipartite, suppose it has parts (A,B). Then

$$\sum_{i\in A,j\in B} w(i,j)=\left(\sum_{i\in A} w(i)\right)\left(\sum_{j\in B} w(j)\right),$$

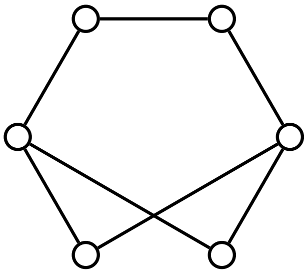
   and the optimal cut puts a (with weight 2) and one other vertex on one side, and the other three vertices on the other side — this cut has weight 9, as required (all other cuts have weight at most 8). On the other hand, if the triangle-free subgraph is C_5 it has weight at most $2+2+1+1+1=7\leq 9$.

3. Consider any triangle-free subgraph of the gadget. By the analysis in the previous item, the only triangle-free subgraphs with weight greater than 8 are isomorphic to $K_{2,3}$, which put a and one of $0, x, y, z$ on one side, and the other three vertices on the other side. It is not difficult to check that all of these graphs connect 0 to x, y, z with paths of length at most 2, and that the sides of the cut exhibit the correct parity requirements. ◀

▶ **Lemma 17.** *There exists a gadget with performance 17 and parity 0.*

Proof. Consider the complete graph on $\{0, 1, a, x, y, z\}$. Let $w(a) = w(1) = 2$, $w(0) = w(x) = w(y) = w(z) = 1$, $\delta(0,1) = \delta(1,0) = 1$, whereas otherwise $\delta(i,j) = 0$. Then define $w(i,j) = w(i)w(j) + \delta(i,j)$.

1. Without loss of generality suppose $c(0) = 0$. There are two cases. First suppose $c(x) = c(y) = c(z) = 0$. Then our cut is $(\{x, y, z, 0\}, \{a, 1\})$ with weight 17. Otherwise, suppose without loss of generality that $c(x) = 0, c(y) = c(z) = 1$. Then our cut is $(\{1, y, z\}, \{0, a, x\})$, with weight 17.
2. Consider all the triangle-free subgraphs of the gadget. This subgraph is either bipartite, or it is a subgraph of the following graph



First we deal with the bipartite case. The weight of a complete bipartite graph with parts (A, B) is

$$\left(\sum_{i\in A, j\in B} \delta(i,j)\right) + \left(\sum_{i\in A} w(i)\right)\left(\sum_{j\in B} w(j)\right).$$

The first part of the sum is 1 if and only if 0 and 1 are in different sides of the cut, and the second is maximised exactly when the two sides of the cut are as even as possible i.e. have weight 4 and 4 respectively — cuts that satisfy both of these conditions weight at most $17 = 4 \times 4 + 1$ edges, while all other cuts have weight at most 16.
Now consider a non-bipartite triangle-free subgraph. The largest this could be is the graph pictured above. First consider the contribution of $w(i)w(j)$ to the total weight. Suppose for contradiction that the contribution is > 14; since there are 7 edges, there must exist one edge whose contribution is > 2. Since $w(i)w(j) \in \{1, 2, 4\}$, and $w(i)w(j) = 4$ only for $\{i, j\} = \{a, 1\}$, a and 1 are adjacent within the subgraph. Since any way of placing $a, 1$ adjacently within the subgraph leads to them having at most 3 other neighbours, the contribution of $w(i)w(j)$ is at most $4 + 3 \times 2 + 3 \times 1 = 13 \not> 14$, a contradiction. So the contribution of $w(i)w(j)$ is at most 14, and the total weight is at most 15.
3. Consider now any triangle-free subgraph of the gadget with weight greater than 16. By the analysis in the previous item, this subgraph must be a complete bipartite subgraph where the two parts both have weight 4, and 0 and 1 are put on opposite sides of the cut. Consider the side of a: if it is on the same side as 1, then x, y, z are all on the opposite side i.e. together with 0. Conversely, if a is on the same side as 0, then two of x, y, z must

be on the same side as 1, and the remaining vertex among x, y, z must be on the same side as 0. It is not difficult to check that these bipartite subgraphs satisfy the desired connectivity and parity conditions. ◀

References

1 Per Austrin, Amey Bhangale, and Aditya Potukuchi. Improved inapproximability of rainbow coloring. In *Proc. 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'20)*, pages 1479–1495, 2020. arXiv:1810.02784, doi:10.1137/1.9781611975994.90.

2 Per Austrin, Venkatesan Guruswami, and Johan Håstad. (2+ε)-Sat is NP-hard. *SIAM J. Comput.*, 46(5):1554–1573, 2017. doi:10.1137/15M1006507.

3 Libor Barto, Jakub Bulín, Andrei A. Krokhin, and Jakub Opršal. Algebraic approach to promise constraint satisfaction. *J. ACM*, 68(4):28:1–28:66, 2021. arXiv:1811.00970, doi:10.1145/3457606.

4 Libor Barto, Silvia Butti, Alexandr Kazda, Caterina Viola, and Stanislav Živný. Algebraic approach to approximation. In *39th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'24)*, pages 10:1–10:14. ACM, 2024. arXiv:2401.15186, doi:10.1145/3661814.3662076.

5 Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free Bits, PCPs, and Nonapproximability — Towards Tight Results. *SIAM J. Comput.*, 27(3):804–915, 1998. doi:10.1137/S0097539796302531.

6 Amey Bhangale and Subhash Khot. Optimal Inapproximability of Satisfiable k-LIN over Non-Abelian Groups. In *Proc. 53rd Annual ACM Symposium on Theory of Computing (STOC'21)*, pages 1615–1628. ACM, 2021. arXiv:2009.02815, doi:10.1145/3406325.3451003.

7 Amey Bhangale, Subhash Khot, and Dor Minzer. On Approximability of Satisfiable k-CSPs: I. In *Proc. 54th Annual ACM Symposium on Theory of Computing (STOC'22)*, pages 976–988. ACM, 2022. doi:10.1145/3519935.3520028.

8 Amey Bhangale, Subhash Khot, and Dor Minzer. On Approximability of Satisfiable k-CSPs: II. In *Proc. 55th Annual ACM Symposium on Theory of Computing (STOC'23)*, pages 632–642. ACM, 2023. doi:10.1145/3564246.3585120.

9 Amey Bhangale, Subhash Khot, and Dor Minzer. On Approximability of Satisfiable k-CSPs: III. In *Proc. 55th Annual ACM Symposium on Theory of Computing (STOC'23)*, pages 643–655. ACM, 2023. doi:10.1145/3564246.3585121.

10 Amey Bhangale, Subhash Khot, and Dor Minzer. On Approximability of Satisfiable k-CSPs: IV. In *Proc. 56th Annual ACM Symposium on Theory of Computing (STOC'24)*, pages 1423–1434. ACM, 2024. doi:10.1145/3618260.3649610.

11 Ankur Bhargava and S. Rao Kosaraju. Derandomization of dimensionality reduction and SDP based algorithms. In *Proc. 8th International Workshop on Algorithms and Data Structures (WADS'05)*, pages 396–408. Springer Berlin Heidelberg, 2005. doi:10.1007/11534273_35.

12 Joshua Brakensiek and Venkatesan Guruswami. Promise Constraint Satisfaction: Algebraic Structure and a Symmetric Boolean Dichotomy. *SIAM J. Comput.*, 50(6):1663–1700, 2021. arXiv:1704.01937, doi:10.1137/19M128212X.

13 Joshua Brakensiek, Venkatesan Guruswami, and Sai Sandeep. SDPs and Robust Satisfiability of Promise CSP. In *Proc. Annual 55th ACM Symposium on Theory of Computing (STOC'23)*, pages 609–622. ACM, 2023. arXiv:2211.08373, doi:10.1145/3564246.3585180.

14 Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *Proc. 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS'17)*, pages 319–330, USA, 2017. IEEE. arXiv:1703.03021, doi:10.1109/FOCS.2017.37.

15 Irit Dinur, Oded Regev, and Clifford Smyth. The hardness of 3-uniform hypergraph coloring. *Comb.*, 25(5):519–535, September 2005. doi:10.1007/s00493-005-0032-4.

16 Euclid. *The Thirteen Books of Euclid's Elements.* Cambridge University Press, Cambridge, second edition, 1926. Trans. Sir Thomas Little Heath.

17 Alan M. Frieze and Mark Jerrum. Improved Approximation Algorithms for MAX k-CUT and MAX BISECTION. *Algorithmica*, 18(1):67–81, 1997. doi:10.1007/BF02523688.

18 M. R. Garey and D. S. Johnson. The complexity of near-optimal graph coloring. *J. ACM*, 23(1):43–49, 1976. doi:10.1145/321921.321926.

19 Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. doi:10.1145/227683.227684.

20 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 7 2001. doi:10.1145/502090.502098.

21 Pavol Hell and Jaroslav Nešetřil. On the Complexity of H-coloring. *J. Comb. Theory, Ser. B*, 48(1):92–110, 1990. doi:10.1016/0095-8956(90)90132-J.

22 Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations*, pages 85–103. Springer US, 1972. doi:10.1007/978-1-4684-2001-2_9.

23 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proc. 34th Annual ACM Symposium on Theory of Computing (STOC'02)*, pages 767–775. ACM, 2002. doi:10.1145/509907.510017.

24 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal Inapproximability Results for MAX-CUT and Other 2-Variable CSPs? *SIAM J. Comput.*, 37(1):319–357, 2007. doi:10.1137/S0097539705447372.

25 Sanjeev Mahajan and H. Ramesh. Derandomizing approximation algorithms based on semidefinite programming. *SIAM J. Comput.*, 28(5):1641–1663, 1999. doi:10.1137/S0097539796309326.

26 Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. *Ann. of Math. (2)*, 171(1):295–341, 2010. doi:10.4007/annals.2010.171.295.

27 Tamio-Vesa Nakajima and Stanislav Živný. Maximum bipartite vs. triangle-free subgraph. Technical report, 2024. arXiv:2406.20069.

28 Christos H. Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.*, 43(3):425–440, 1991. doi:10.1016/0022-0000(91)90023-X.

29 Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proc. 40th Annual ACM Symposium on Theory of Computing (STOC'08)*, pages 245–254, 2008. doi:10.1145/1374376.1374414.

30 Prasad Raghavendra and David Steurer. How to round any CSP. In *Proc. 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS'09)*, pages 586–594. IEEE Computer Society, 2009. doi:10.1109/FOCS.2009.74.

31 Johan Thapper and Stanislav Živný. The Complexity of Finite-Valued CSPs. *J. ACM*, 63(4):37:1–37:33, 2016. doi:10.1145/2974019.

32 Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson. Gadgets, approximation, and linear programming. *SIAM J. Comput.*, 29(6):2074–2097, 2000. doi:10.1137/S0097539797328847.

33 Mihalis Yannakakis. Edge-deletion problems. *SIAM J. Computing*, 10(2):297–309, 1981. doi:10.1137/0210021.

34 Dmitriy Zhuk. A proof of the CSP dichotomy conjecture. *J. ACM*, 67(5):30:1–30:78, 2020. arXiv:1704.01914, doi:10.1145/3402029.