Approximate Graph Colouring and the Hollow Shadow

Lorenzo Ciardo
lorenzo.ciardo@cs.ox.ac.uk
University of Oxford
Oxford, U.K.

ABSTRACT
We show that approximate graph colouring is not solved by constantly many levels of the lift-and-project hierarchy for the combined basic linear programming and affine integer programming relaxation. The proof involves a construction of tensors whose fixed-dimensional projections are equal up to reflection and satisfy a sparsity condition, which may be of independent interest.

CCS CONCEPTS
• Theory of computation → Approximation algorithms analysis; Problems, reductions and completeness.

KEYWORDS
approximate graph colouring, Sherali-Adams linear programming relaxations, affine integer programming relaxations, Diophantine equations, promise constraint satisfaction

INTRODUCTION
The approximate graph colouring problem (AGC) consists in finding a \(d\)-colouring of a given \(c\)-colourable graph, where \(3 \leq c \leq d\). There is a huge gap in our understanding of this problem. For an \(n\)-vertex graph and \(c = 3\), the currently best known polynomial-time algorithm of Kawarabayashi and Thorup [51] finds a \(d\)-colouring of the graph with \(d = O(n^{0.1999})\), building on a long line of works started by Wigderson [70]. It was conjectured by Garey and Johnson [42] that the problem is NP-hard for any fixed constants \(3 \leq c \leq d\) even in the decision variant: Given a graph, output Yes if it is \(c\)-colourable and output No if it is not \(d\)-colourable.

For \(c = d\), the problem becomes the classic \(c\)-colouring problem, which appeared on Karp’s original list of 21 NP-complete problems [50]. The case \(c = 3, d = 4\) was only proved to be NP-hard in 2000 by Khanna, Linial, and Safra [52] (and a simpler proof was given by Guruswami and Khanna in [44]); more generally, [52] showed hardness of the case \(d = c + 2\lfloor c/3 \rfloor - 1\). This was improved to \(d = 2c - 2\) in 2016 by Brakensiek and Guruswami [14], and recently to \(d = 2c - 1\) by Barto, Bulín, Krokhin, and Oprášl [8]. In particular, this last result implies hardness of the case \(c = 3, d = 5\); the complexity of the case \(c = 3, d = 6\) is still open. Building on the work of Khot [53] and Huang [48], Krokhin, Oprášl, Wrochna, and Živný established NP-hardness for \(d = (\lceil c/2 \rceil)^2 - 1\) for \(c \geq 4\) in [59]. NP-hardness of AGC was established for all constants \(3 \leq c \leq d\) by Dinur, Mossel, and Regev in [39] under a non-standard variant of the Unique Games Conjecture, by Guruswami and Sanjeev in [45] under the \(d\)-to-1 conjecture [54] for any fixed \(d\), and (an even stronger statement of distinguishing 3-colourability from not having an independent set of significant size) by Braverman, Khot, Lifshitz, and Minzer in [21] under the rich 2-to-1 conjecture of Braverman, Khot, and Minzer [22].

AGC is a prominent example of so called promise constraint satisfaction problems (PCSPs), which we define next. A directed graph (digraph) \(A\) consists of a set \(V(A)\) of elements called vertices and a set \(E(A) \subseteq V(A)^2\) of pairs of vertices called edges. Given two digraphs \(A\) and \(B\), a map \(f : V(A) \to V(B)\) is a homomorphism from \(A\) to \(B\) if \((f(u), f(v)) \in E(B)\) for any \((u, v) \in E(A)\). We shall indicate the existence of a homomorphism from \(A\) to \(B\) by writing \(A \to B\). Let \(A\) and \(B\) be two fixed finite digraphs with \(A \to B\); we call the pair \((A, B)\) a template. The PCSP parameterised by the template \((A, B)\), denoted by PCSP\((A, B)\), is the following decision problem: Given a finite digraph \(X\) as input, answer Yes if \(X \to A\) and No if \(X \not\to B\). 1 A \(p\)-colouring of a digraph \(X\) is precisely a homomorphism from \(X\) to the clique \(K_p\) – i.e., the digraph on vertex set \(\{1, \ldots, p\}\) such that any pair of distinct vertices is a (directed) edge. Hence, AGC is PCSP\((K_c, K_d)\).

By letting \(A = B\) in the definition of a PCSP, one obtains the standard (non-promise) constraint satisfaction problem (CSP) [40]. PCSPs were introduced by Austrin, Guruswami, and Håstad [5] and Brakensiek and Guruswami [16] as a general framework for studying approximability of perfectly satisfiable CSPs and have emerged as a new exciting direction in constraint satisfaction that requires different techniques than CSPs.2 Recent works on PCSPs include those using analytical methods [12, 13, 17, 22] and those building on algebraic methods [3, 7, 10, 15, 18, 19, 26, 33, 45, 63] developed in [8]. However, most basic questions are still left open, including existence of algorithmic reductions and applicability of different types of algorithms.

Two main algorithmic techniques have been utilised for solving CSPs and their variants: enforcing (some type of) local consistency, and solving (generalisations of) linear equations. The first type of algorithms divides a given CSP into multiple small CSPs, each of which requires meeting local constraints on a portion of the instance

---

1The requirement \(A \to B\) implies that the two cases cannot happen simultaneously, as homomorphisms compose; the promise is that one of the two cases always happens.

2It is customary to study \((P')\)CSPs on more general objects than digraphs, known as relational structures [5], which consist of a collection of relations of arbitrary arities on a vertex set, cf. [5].
of bounded size, and then enforces consistency between all solutions (called partial homomorphisms); i.e., it requires that they agree on the intersection of their domains. Instead, the second type of algorithms seeks a global solution that satisfies a linearised version of the constraints. More precisely, it is always possible to formulate a CSP (and, in fact, any homomorphism problem) as a system of linear equations over \( \{0, 1\} \); then, the algorithms of the second type work by suitably modifying the system (in particular, extending the domain of its variables) in a way that it can be efficiently solved through variants of Gaussian elimination.

Remarkably, all algorithms hitherto proposed in the literature on (variants of) CSPs can be broadly classified as instances of one of the two aforementioned techniques, or a combination of both. A primary example of the first type is the bounded width algorithm, which outputs Yes if a consistent collection of partial homomorphisms exists [40]. More powerful versions of the local consistency technique require that the partial homomorphisms be sampled according to a probability distribution (which results in the Sherali-Adams LP hierarchy [67]), and that the probabilities be treated as vectors satisfying certain orthogonality requirements (which gives the sum-of-squares or Lasserre SDP hierarchy [60, 64, 68]). As for the second type, the linear-system formulation of a CSP can be efficiently solved in \( \mathbb{Z} \) by computing the Hermite or the Smith canonical forms of the corresponding coefficient matrix [66]; this results in the affine integer programming (AIP) relaxation, studied in the context of PCSPs in [8, 15, 16, 18]. A hierarchy of linear Diophantine equations was also studied by Berkholz and Grohe [11] in the context of the graph isomorphism problem.

Since polynomial-time algorithms are not expected to solve NP-hard problems, a well-established line of work has sought lower bounds on the efficacy of these algorithms; see [2, 20, 28, 43, 57] for lower bounds on LPs arising from lift-and-project hierarchies such as that of Sherali-Adams, [27, 62, 69] for lower bounds on SDPs, and [11] for lower bounds on linear Diophantine equations. If, as conjectured by Garey and Johnson [42], AGC is NP-hard and P \( \neq \) NP, neither of the two algorithmic techniques should be able to solve it. This is indeed the case! In a striking sequence of works by Dinur, Khot, Kindler, Minzer, and Safra [37, 38, 55, 56], the 2-dimensional existence of matrices (i.e., 2-dimensional tensors) over a certain domain having prescribed row- and column-sums (i.e., 1-dimensional projections) and a fixed pattern, i.e., a fixed set of entries allowed (or required) to be nonzero. Examples include 0-1 matrices with zero trace (i.e., adjacency matrices of digraphs) [41], with at most one fixed zero in each column [1], or with a fixed zero block [23], real matrices with a fixed pattern [49], and integral matrices with fixed lower and upper bounds on each entry [30]; see also related work in [24, 29, 36]. We believe that our work will stimulate further progress within that trend since, for the first time to our best knowledge, it extends the investigation from matrices to tensors of arbitrary dimension.

2 OVERVIEW OF RESULTS AND TECHNIQUES

Let \( X \) and \( A \) be two digraphs. We can cast the question “Is \( X \) homomorphic to \( A \)” as the question of checking whether a system of linear equations has a solution in the set \( \{0, 1\} \). Indeed, introduce the variables \( \lambda_{x,a} \) for all vertices \( x \in V(X), a \in V(A) \), and the variables \( \mu_{y,b} \) for all edges \( y \in E(X), b \in E(A) \), and consider the equations

\[
(\text{IP}_1) \quad \sum_{a \in V(A)} \lambda_{x,a} = 1 \quad \forall x \in V(X)
\]

\[
(\text{IP}_2) \quad \sum_{b \in E(A) \, | \, b_i = a} \mu_{y,b} = \lambda_{y_i,a} \quad \forall y \in E(X), \forall i \in \{1, 2\}, \forall a \in V(A).
\]

One readily checks that \( X \rightarrow A \) if and only if (IP) has a solution in \( \{0, 1\} \). Unless P \( \neq \) NP, this system is not solvable in polynomial time over \( \{0, 1\} \). Relaxing it by allowing that the variables can be assigned rational nonnegative values (resp. integer values) results in the so-called basic linear programming (BLP) relaxation (resp. affine integer programming (AIP) relaxation). Note that both these
relaxations (as well as all relaxations we shall use in this work) result in algorithms that are complete but not necessarily sound, in the sense that they always output Yes if \( X \to A \), but may fail to output No if \( X \not\to A \). The BA relaxation described in [18] combines the two relaxations mentioned above as follows: It outputs Yes if and only if there exist a solution to BLP and a solution to AIP such that the following so-called refinement condition holds: Whenever a variable is zero in the first solution, it is zero in the second solution. It follows that BA is at least as strong as both BLP and AIP; in fact, as shown in [18], it is strictly stronger, in the sense that there exist templates that are solved by BA but not by BLP or AIP.

The system (IP) can be refined by replacing the variables \( \lambda_{x,a} \) with variables \( \lambda_{x,f} \), where \( S \) is a set of vertices of \( X \) of size at most \( k \) and \( f \) is a function from \( S \) to \( V(A) \). Solving such refined system over the set of nonnegative rational numbers (resp. integer numbers) would then mean finding rational nonnegative (resp. integer) distributions over the set of partial assignments from portions of the instance of size at most \( k \) to \( A \). The former choice results in the Sherali-Adams LP hierarchy [67], which we call the BLP hierarchy; the latter results in the affine integer programming hierarchy [32], which we call the AIP hierarchy. Crucially, the former but not the latter choice ensures local consistency. Each assignment receiving nonzero weight in the BLP hierarchy corresponds to a partial homomorphism, while the same is not true for the AIP hierarchy. Equivalently, the BLP hierarchy is at least as strong as the bounded-width algorithm [6, 9, 40] (and, in fact, strictly stronger, see [4]). It is also worth noting that these hierarchies are still complete but not necessarily sound, and they become progressively stronger as the level \( k \) increases. In particular, the BLP hierarchy is “sound in the limit”, in the sense that its \( k \)-th level correctly classifies all instances of size \( k \) or less – which is clear from the fact that a partial homomorphism over the whole domain is a homomorphism.

The BA hierarchy we consider in this work consists in applying the BA relaxation of [18] to progressively larger portions of the instances, in the same spirit as the BLP and AIP hierarchies. Equivalently, the BA hierarchy can be described as follows: Its \( k \)-th level, applied to two digraphs \( X \) and \( A \), outputs Yes if and only if (i) the \( k \)-th level of both BLP and AIP outputs Yes when applied to \( X \) and \( A \), and (ii) the two solutions they provide satisfy the refinement condition. In this case, we write \( BA^k(X, A) = Yes \). Given two digraphs \( A, B \) such that \( A \to B \), we say that the \( k \)-th level of BA solves \( P_{\text{CS}}(A, B) \) if, for any instance \( X \), \( BA^k(X, A) = Yes \) implies \( X \to B \).

The main result of our work is that no constant level of the BA hierarchy solves the approximate graph colouring problem.

**Theorem 1.** For any fixed \( 3 \leq c \leq d \), there is no \( k \in \mathbb{N} \) such that \( BA^k \) solves \( P_{\text{CS}}(K_c, K_d) \).

A way to prove that approximate graph colouring is not solved by the BA hierarchy is to present fooling instances – digraphs with a large chromatic number but yet whose structure meets all constraints of the hierarchy. More precisely, it would suffice to build, for every \( c, d \), and \( k \), a digraph \( G \) whose chromatic number is higher than \( d \) and such that \( BA^k(G, K_c) = Yes \). Thus our goal is the following:

“Find a fooling instance for the BA hierarchy applied to AGC.”

Instead of directly looking for instances that fool the hierarchy, our approach shall be to consider the following questions: How does a certificate of acceptance for the BA hierarchy look like? Can we, tell, from the shape of such a certificate, what the limits of the hierarchy applied to AGC are? The first step of our analysis is to translate the problem of whether the BA hierarchy accepts an input into a problem having a different, multilinear nature. Building on the framework developed in [34], we find that BA acceptance is implied by the existence of a family of tensors having certain special characteristics. First of all, they need to satisfy (i) a system of symmetries. This is essentially the result of the marginality constraints that are enforced by all “lift-and-project” hierarchies such as the BLP, AIP, and Lasserre SDP hierarchies [61], and is common to all algorithmic hierarchies studied in [34] through the tensor approach. There is, however, a feature that is typical of the BA hierarchy. Not only does BA require that both a linear program and a system of Diophantine equations have a solution; it also requires that any variable that is assigned zero weight by the former should be also assigned zero weight by the latter. This refinement condition of the relaxation introduced in [18] blends together the consistency-enforcing and linear-equation-solving techniques, to produce an algorithm that, as discussed above, is provably strictly stronger than both. The translation of the refinement condition into the multilinear framework is (ii) a hollowness requirement: Each tensor certifying BA acceptance needs to be hollow; i.e., it needs to contain zeros in certain prescribed locations. Summarising, the original problem has now become the following:

“Produce a family of hollow tensors satisfying a system of symmetries.”

There is a natural way to produce a family \( \{T_i\} \) of tensors satisfying such symmetries: One starts with a high-dimensional tensor \( C \) whose low-dimensional oriented projections (i.e., projections onto oriented hyperplanes) are equal. Then, the family of all (not necessarily oriented) low-dimensional projections of \( C \) satisfies the required symmetries. We call such a tensor \( C \) a crystal (in accordance with [32]), while the shadow of \( C \) is any of its oriented projections. We then reformulate the problem to its final form; the solution of this problem is the main technical result of the paper.

“Find a crystal whose shadow is hollow.”

The rest of the paper is conceptually organised in three parts, each corresponding to a different phase of the proof of Theorem 1: (1) a pre-processing phase, where \( BA^k \) acceptance is turned into a multilinear problem; (2) a multilinear phase, where the multilinear problem is solved (i.e., hollow-shadowed crystals are built); (3) a post-processing phase, where the solution of the previous problem is translated back to the algorithmic framework, and it is used to recover a fooling instance. Full details of the three phases are discussed in the full version of this paper [31]. Sections [2.1, 2.2, and 2.3 below] give a more intuitive overview of the contents of each of the phases.

### 2.1 The BA Hierarchy through Tensors

All hitherto studied relaxation algorithms for (promise) CSPs, including the BLP, AIP, and BA algorithms, are captured algebraically through the notion of linear mimin – an algebraic structure consisting of matrices having a fixed number of columns and a variable
number of rows, that is closed under the application of elementary row operations (summing up or swapping two rows, inserting an extra zero row). Given a linear minion $\mathcal{M}$ and a digraph $A$ with $n$ vertices and $m$ edges, there exists a natural way of simulating the structure of $A$ in $\mathcal{M}$, by defining a new (potentially infinite) digraph $\mathbb{F}_\mathcal{M}(A)$ (the free structure of $\mathcal{M}$ generated by $A$) whose vertices are the matrices in $\mathcal{M}$ having $n$ rows, and whose edges are pairs of matrices $(M, N)$ such that both $M$ and $N$ can be obtained from some matrix $Q$ having $m$ rows through certain elementary row operations. Then, the relaxation corresponding to $\mathcal{M}$ works as follows: Given an instance $X$, rather than directly checking whether $X \rightarrow A$, one checks whether $X \rightarrow \mathbb{F}_\mathcal{M}(A)$. The reason for doing so is that, for certain linear minions, the latter homomorphism problem is always tractable. As an example, stochastic rational vectors form a linear minion (since they are preserved under elementary row operations) named $\mathcal{Z}$, whose corresponding relaxation is BLP. Similarly, integer vectors whose entries sum up to 1 form the linear minion $\mathcal{Z}_{\text{aff}}$ corresponding to AIL. The framework developed in [34] allows to systematically strengthen the relaxation corresponding to any linear minion, by making use of the notion of tensor power of a digraph: For $k \in \mathbb{N}$, the $k$-th tensor power of $A$ is the hypergraph $A^\otimes k$ whose vertices are $k$-tuples of vertices of $A$, and whose edges are $k$-dimensional tensors obtained by “scattering” the edges of $A$ in $k$ dimensions. The $k$-th level of the hierarchy of the relaxation corresponding to some linear minion $\mathcal{M}$ essentially consists in applying the relaxation to the tensorised digraphs rather than the original digraphs; in other words, one checks if there exists a homomorphism $X^\otimes k \rightarrow \mathbb{F}_{\mathcal{M}}(A^\otimes k)$. In addition, the homomorphism needs to preserve the tensor structure of the two hypergraphs (intuitively, it must “behave well with respect to projections”) – in which case, we say that it is a $k$-tensorial homomorphism. The algorithm obtained in this way is progressively stronger as $k$ increases, and it still runs in polynomial time (for a fixed $k$) since the size of the tensorised digraph is polynomial in the size of the original digraph. In particular, if the matrices of $\mathcal{M}$ satisfy a certain positivity requirement – in which case we say that the linear minion is conic – the hierarchy is sound in the limit, in the sense that its $k$-th level correctly classifies all instances $X$ on at most $k$ vertices. In fact, the hierarchies based on conic minions enforce local consistency [34].

How does the BA hierarchy fit within this framework? Given two linear minions $\mathcal{M}$ and $\mathcal{N}$ such that $\mathcal{M}$ is conic, we define their semi-direct product $\mathcal{M} \ltimes \mathcal{N}$ as a new linear minion that is essentially designed in a way to make the corresponding relaxation stronger than the relaxations associated with $\mathcal{M}$ and $\mathcal{N}$. In particular, one easily checks that $\mathcal{M} \ltimes \mathcal{N}$ is in fact a conic minion. Hence, this operation can be viewed as a standard way of making a given linear minion $\mathcal{N}$ conic; or, in other words, a way to make a given relaxation locally consistent. The minion introduced in [18] corresponding to the first level of BA is the semi-direct product of the conic minion $\mathcal{Z}_{\text{conv}}$ and the linear minion $\mathcal{Z}_{\text{aff}}$. We can then capture algebraically the BA hierarchy as follows.

We note that now $\mathbb{F}_{\mathcal{M}}(A^\otimes k)$ is a hypergraph rather than a digraph; the definitions are analogous.

Proposition 2. Let $X, A$ be two digraphs and let $k \in \mathbb{N}$. Then $BA^k(X, A) = \text{Yes}$ if and only if there exists a $k$-tensorial homomorphism $\theta : X^\otimes k \rightarrow \mathbb{F}_{\mathcal{M}}(A^\otimes k).

Recall that the goal of this pre-processing phase is to come up with a multilinear criterion to check whether $BA^k$ accepts an instance of AGC. From the way the semi-direct product is defined, it follows that a homomorphism $\theta$ from $X^\otimes k$ to $\mathbb{F}_{\mathcal{M}}(A^\otimes k)$ can be decoupled into a homomorphism $\xi$ to $\mathbb{F}_{\mathcal{N}}(A^\otimes k)$ and a homomorphism $\eta$ to $\mathbb{F}_{\mathcal{Z}_{\text{aff}}}(A^\otimes k)$. If $\eta$ is a clique – as it happens when the BA hierarchy is applied to AGC – one can design a simpler sufficient criterion, based on the fact that one can always assume $\xi$ to be the homomorphism mapping a tuple of vertices of $X$ to a tensor in $\mathbb{F}_{\mathcal{Z}_{\text{aff}}}(A^\otimes k)$ that is uniform on its support. After dealing with some combinatorial technicalities, this fact produces the following criterion of acceptance.

Theorem 3. Let $2 \leq k \leq n \in \mathbb{N}$, let $X$ be a loopless digraph, and let $\xi : X^\otimes k \rightarrow \mathbb{F}_{\mathcal{Z}_{\text{aff}}}(K_n^\otimes k)$ be a $k$-tensorial homomorphism such that $E_a \star \xi(x) = 0$ for any $x \in V(X)^k$ and $a \in \{1, \ldots, n\}^k$ for which $a \not= x$. Then $BA^k(X, K_n) = \text{Yes}$.

2.2 Crystals with Hollow Shadows

The criterion of acceptance for $BA^k$ stated in Theorem 3 is multilinear. Indeed, $\mathbb{F}_{\mathcal{Z}_{\text{aff}}}(K_n^\otimes k)$ is a space of integer affine tensors (i.e., whose entries sum up to 1), and the existence of a $k$-tensorial homomorphism from $X^\otimes k$ to $\mathbb{F}_{\mathcal{Z}_{\text{aff}}}(K_n^\otimes k)$ corresponds to the existence of a family of tensors satisfying a specific system of symmetries. Letting $g$ be the number of vertices in $X$, such a family can be realised as the family of $k$-dimensional projections of a single affine $q$-dimensional crystal tensor, which we next informally define. We let $T^n_\text{cr}^q(Z)$ denote the set of all integer cubical tensors of dimension $q$ and width $n$ – i.e., $n \times n \times \cdots \times n$ arrays of integer numbers, where $n$ appears $q$ times. The notion of projecting should intuitively be thought of as “summing up all entries of a tensor along a certain set of directions”; the formal definition shall make use of the operation of tensor contraction, which is defined in the full version [31]. By oriented projection we mean that the directions are considered to be ordered.

Definition 4 (Informal). Let $q, n \in \mathbb{N}$ and $k \in \{0, \ldots, q\}$. A cubical tensor $C \in T^n_\text{cr}^q(Z)$ is a $k$-crystal if all its $k$-dimensional oriented projections are equal. In this case, the $k$-shadow of $C$ is this common oriented projection.

Equivalently, a $k$-crystal is required to have equal $k$-dimensional projections up to reflection – a reflection is a higher-dimensional analogue of the transpose operation. Let $\zeta_C$ be the map associated with an affine $k$-crystal $C$, which takes a $k$-tuple of vertices of $X$ and maps it to the projection of $C$ onto the hyperplane generated by $x$. By construction, $\zeta_C$ behaves well with respect to projections, so it is automatically $k$-tensorial. In order to yield a certificate of acceptance for $BA^k(X, K_n)$, according to Theorem 3, $\zeta_C$ also needs to be a homomorphism and satisfy the extra condition $a \not= x \Rightarrow E_a \star \zeta_C(x) = 0$. It turns out that both these requirements translate

---

3We note that now $\mathbb{F}_{\mathcal{M}}(A^\otimes k)$ is a hypergraph rather than a digraph; the definitions are analogous.

4Intuitively, a linear minion is conic if any matrix is nonzero and has the property that, whenever some of its rows sum up to the zero vector, each of those rows is the zero vector.

5Here, $E_a \star \zeta_C(x)$ denotes the $a$-th entry of the tensor $\zeta_C(x)$, while $a \not= x$ means that there exist two indices $i, j$ for which $a_i = a_j$ but $x_i \not= x_j$. 
as a condition on the k-shadow S of C: The only entries of S allowed to be nonzero are those whose coordinates are all distinct. We say that a tensor having this property is hollow. As an example, if k = 2, the condition means that the n × n matrix S needs to have zero diagonal; if k = 3, three diagonal planes of the n × n × n tensor S of the form (a, a, b), (a, b, a), (b, a, a) should be set to zero, and so on.

In summary, the discussion above indicates that an affine k-crystal of dimension q and width n whose k-shadow is hollow yields a certificate that BAk(X, Kn) = YES for any loopless digraph X with q vertices. The problem is now to verify whether hollow-shadowed crystals exist. It is not hard to check that such crystals cannot exist for all choices of k, q, and n; this parallels the fact that the BA hierarchy is sound in the limit, so it cannot be the case that any X is accepted by any level of BA applied to any clique Kn. This is in sharp contrast with the weaker AIP hierarchy, for which a similar acceptance result holds, cf. [32]. Hence, unlike for AIP, one cannot simply take large cliques as fooling instances for BA.

The key to establishing Theorem 5 is proving the following.

**Theorem 5.** For any k ≤ q ∈ N there is an affine k-crystal C ∈ T^{q × 1}_{k+1} such that C has hollow k-shadow.

The key to establishing Theorem 5 is proving the following.

**Theorem 6.** For any k ∈ N there exists a hollow affine (k−1)-crystal C ∈ T^{k−1, q}_{k−1}.

We now discuss the main ideas of the proof of Theorem 6 for the case k = 3.

**Figure 1: The crystal W.**

Our goal is to find a hollow affine 2-crystal C ∈ T^{6, 1}_{5} with k − 1 = 2. In other words, C must be a three-dimensional cubical tensor of width 6, such that (i) C is hollow, i.e., the only entries allowed to be nonzero are the ones whose three coordinates are all distinct; (ii) C is affine, i.e., its entries sum up to 1; and (iii) C is a 2-crystal, i.e., projecting it onto the xy-, yz-, and xz-planes results in the same 6 × 6 "shadow" matrix. By induction, we can assume that Theorem 6 holds for k = 2. In fact, it is not hard to find by inspection that the matrix U =

\[
\begin{bmatrix}
0 & 0 & 1 \\
1 & 0 & -1 \\
0 & 0 & 0
\end{bmatrix}
\]

is a hollow affine 1-crystal in T^{3, 1}_{3}.

**The next step is to build a (not necessarily hollow) 3-dimensional 2-crystal having shadow U.** This can be done using the method developed in [32] as a black box, and it results in the tensor V =

\[
\begin{bmatrix}
-1 & 0 & 1 & 0 & 0 & 0 \\
2 & 0 & -1 & 0 & 0 & 0 \\
-1 & 1 & 0 & 0 & 0 & 1 \\
-1 & 0 & 0 & 1 & -1 & 0
\end{bmatrix}
\]

Clearly, V is not hollow - for example, its (1, 1, 1)-th coordinate is ≠ 0. In fact, it is not hard to check that a hollow affine 2-crystal of dimension 3 and width 3 cannot exist.

We need to increase the width to "make more space"; we do so by padding V with three layers of zeros along each of the three dimensions. The tensor W we obtain in this way (see Figure 1) is clearly still a 2-crystal. We can view W as a block tensor with eight 3 × 3 × 3 blocks; note that all non-zero entries of W are in one block. The idea is now to "spread" these entries in the other blocks, in a way that they occupy positions whose indices have no repetitions. To this end, we make use of a particular class of "transparent" crystals that we call quartzes. Such crystals are designed in a way that the shadow they project is identically zero, meaning that we can freely add them (or their integer multiples) to a given crystal without changing its shadow and maintaining it a crystal.

**A quartz can be built by choosing two cells a and b having disjoint coordinates, considering the parallelepiped generated by a and b, assigning value 1 or −1 to its vertices in a way that two adjacent vertices get values of opposite sign, and assigning value 0 to all other cells; we refer to such quartz as to Q_{a,b}.**

See Figure 2. (This construction is easily generalised to an arbitrary dimension.) Quartzes yield a method to relocate some nonzero entry of W, while leaving the rest of W almost untouched. More precisely, if the a-th entry of W has value w_{a} ≠ 0, the a-th entry of W − w_{a} · Q_{a,b} is zero, and this operation modifies the value of only 8 cells of W. The idea is then to perturb W with suitable quartzes, so as to transfer all nonzero entries to positions where they do not violate the hollowness requirement.

To this end, we take as b a fixed cell that generates the smallest number of ties and that lies in the block of W opposite to the one

**Figure 2: The quartz Q_{a,b}.**

**Figure 3: W − w_{(1,1,1)} · Q_{(1,1,1),b}.**
containing the nonzero entries – for example, the cell \( b = (4, 5, 6) \). Even with such a choice, it can happen that adding a multiple of a quartz introduces new nonzero entries in positions that violate hollowness. For example, Figure 3 shows the tensor \( W = w_{1,1,1} \cdot Q_{(1,1,1),b} \). The cell \((1, 1, 1)\) has become zero, as wanted, but three new forbidden cells \((1, 1, 1), (1, 5, 1), (4, 1, 1)\) now have nonzero values. However, the nonzero values in these forbidden cells cancel out once this procedure is applied to all entries in the nonzero block of \( W \). In other words, the affine 2-crystal

\[
C = W - \sum_{a \in \{1,2,3\}} w_a \cdot Q_{a,b}
\]

is hollow (see Figure 4).

### 2.3 Fooling the BA Hierarchy

Let \( C \) be an affine \( k \)-crystal of dimension \( q \) and width \( \frac{k^{2+k}}{2} \) whose \( k \)-shadow is hollow, as in Theorem 5. Let \( X \) be a loopless digraph on vertex set \( V(X) = \{1, \ldots, q\} \). Consider the map \( \zeta_C \) taking as input a tuple \( x \) of \( k \) vertices of \( X \) (i.e., a tuple of \( k \) numbers in \( \{1, \ldots, q\} \)) and returning the \( k \)-dimensional projection of \( C \) onto the hyperplane corresponding to \( x \). As discussed earlier, \( \zeta_C \) yields a \( k \)-tensorial homomorphism from \( X^k \) to \( \mathbb{F}_2^{2^d} \), and the fact that the shadow of \( C \) is hollow translates as \( \zeta_C \) satisfying the extra requirement of Theorem 3. Hence, we obtain the following.

**Proposition 7.** Let \( 2 \leq k \in \mathbb{N} \) and let \( X \) be a loopless digraph. Then \( \text{BA}^k(X, K_{(k^{2+k})/2}) = \text{Yes} \).

To prove Theorem 1, we need to show that \( \text{BA}^k \) does not solve \( \text{PCSP}(K_c, K_d) \) for all choices of \( k \in \mathbb{N} \) and \( 3 \leq c \leq d \in \mathbb{N} \). If \( c = \frac{k^{2+k}}{2} \), any graph with chromatic number bigger than \( d \) (for example, the clique \( K_{d+1} \)) would then yield a fooling instance. Since increasing \( c \) can only make AGC harder, this argument shows that \( \text{BA}^k \) does not solve \( \text{PCSP}(K_c, K_d) \) whenever \( c \geq \frac{k^{2+k}}{2} \), and the fooling instances are simply cliques.

In order to establish Theorem 1 in full generality, however, we shall pick the fooling instances from a more refined class of digraphs: the so-called shift digraphs (see Figure 5).

**Definition 8.** The line digraph of a digraph \( X \) is the digraph \( \delta X \) defined by \( V(\delta X) = E(X) \) and \( E(\delta X) = \{(x, y), (y, z) : (x, y, z) \in E(X)\} \).

**Definition 9.** Let \( q \in \mathbb{N} \) and \( i \in \mathbb{N}_0 \). The shift digraph \( S_{q,i} \) is recursively defined by setting \( S_{q,0} = K_q \), \( S_{q,i} = \delta S_{q,i-1} \) for each \( i \geq 1 \).

It is not hard to verify that the following non-recursive description of shift digraphs is equivalent to Definition 9 for \( i \geq 1 \): \( S_{q,i} \) is the digraph whose vertex set consists of all strings of length \( i+1 \) over the alphabet \( \{1, \ldots, q\} \) such that consecutive letters are distinct, and whose edge set consists of all pairs of strings of the form \((a_1 \ldots a_k, a_2 \ldots a_{k+1})\). In particular, it is clear from this description that the edge set of \( S_{q,i} \) is nonempty for \( q \geq 2 \).

The line digraph construction has been utilised in [45, 59] as a polynomial-time (and in fact log-space) reduction between PCSPs. In particular, the construction changes the chromatic number in a controlled way, as we now describe. Consider the integer functions \( a \) and \( b \) defined by \( a(p) = 2^p \) and \( b(p) = \left\lfloor \frac{p}{\log_2 p} \right\rfloor \) for \( p \in \mathbb{N} \), and notice that \( a(p) \geq b(p) \) for each \( p \). Let also \( a^{(i)} \) (resp. \( b^{(i)} \)) be the function obtained by iterating \( a \) (resp. \( b \)) \( i \)-many times, for \( i \in \mathbb{N} \). The following result bounds the chromatic number of the line digraph in terms of that of the original digraph.

**Theorem 10 ([46]).** Let \( X \) be a digraph and let \( p \in \mathbb{N} \). If \( \delta X \rightarrow K_p \), then \( X \rightarrow K_{a(p)} \); if \( \delta X \rightarrow K_{b(p)} \), then \( \delta X \rightarrow K_{b} \).\(^{1}\)

---

\(^{1}\text{In [47, \S 2.5], a slightly different definition of shift digraphs is given, where the case } i = 0 \text{ is a transitive tournament rather than a clique; equivalently, the vertex set of } S_{q,i} \text{ only includes monotonically increasing strings.} \)
An interesting feature of the line digraph operator is that it preserves acceptance by hierarchies of relaxations corresponding to conic minions, at the only cost of halving the level. As stated next, this in particular holds for the BA hierarchy, whose corresponding minion \( \mathcal{L}_{	ext{conic}} \equiv \mathcal{L}_{\text{aff}} \) is conic.

**Proposition 11.** Let \( 2 \leq k \in \mathbb{N} \), let \( X, A \) be digraphs, and suppose that \( \text{BA}^k(\delta X, \delta A) \equiv \text{Yes} \). Then \( \text{BA}^k(\delta X, \delta A) \equiv \text{Yes} \).

The key point is that, under the application of the line digraph operator, a digraph decreases exponentially fast in terms of chromatic number, but only polynomially fast in terms of BA acceptance level. Intuitively, our strategy to fool \( \text{BA}^k \) as an algorithm to solve PCSP\((K_3, K_4)\) will be to take as the fooling instance a shift digraph \( S_{q,i} \) where \( q \sim \exp(i)(d+1) \), rather than the clique \( K_{d+1} \). Categorically, this digraph is similar to \( K_{d+1} \) by Theorem 10, so it is not \( d \)-colourable. On the other hand, for large enough \( i \), the difference in speed decrease guarantees that \( \text{BA}^{\text{pol}^{i}(e)}(K_q, K_{\exp(i)(e)}) \equiv \text{Yes} \) by Proposition 7 – which, through Proposition 11, eventually implies \( \text{BA}^k(S_{q,i}, K_q) \equiv \text{Yes} \). We note that this argument crucially depends on the fact that the size \( \frac{2^d}{i} \) of the clique in Proposition 7 – i.e., the width of the hollow-shadowed crystals mined in Section 2.2 – is sub-exponential in \( k \). Proving Theorem 1 in full detail, we present a result, true for all linear minions, stating that acceptance of some instance \( X \) by some level of the hierarchy is preserved under homomorphisms of the template.

**Proposition 12.** Let \( k \in \mathbb{N} \), let \( X, A, B \) be digraphs such that \( A \rightarrow B \), and suppose that \( \text{BA}^k(X, A) \equiv \text{Yes} \). Then \( \text{BA}^k(X, B) \equiv \text{Yes} \).

**Proof of Theorem 1.** Since \( \text{BA}^2 \) is at least as powerful as \( \text{BA}^1 \), we can assume that \( k \geq 2 \). Suppose first that \( k \geq 4 \). In this case, we can find \( i \in \mathbb{N} \) such that \( b(i)(c) \geq k^24^i \). Take \( q \sim a(i)(d) \). We claim that the shift digraph \( S_{q,i} \) is a fooling instance for the \( k \)-th level of BA applied to PCSP\((K_3, K_4)\); in other words, we claim that \( (i) \text{ BA}^k(S_{q,i}, K_q) \equiv \text{Yes} \) and \( (ii) S_{q,i} \not\rightarrow K_4 \).

For \( (i) \), we start by applying Proposition 7 to find that
\[
\text{BA}^{k^{2i}}(K_q, K_{k^{2i}4^i+k^{2i}4^i}) \equiv \text{Yes}.
\]
Observe that \( \frac{k^{2i}4^i+k^{2i}4^i}{2} \leq k^{2i}4^i \leq b(i)(c) \), so
\[
K_{k^{2i}4^i+k^{2i}4^i} \rightarrow K_{k^{2i}4^i} \rightarrow K_{b(i)(c)}.
\]
By Proposition 12, we deduce that \( \text{BA}^{k^{2i}}(K_q, K_{b(i)(c)}) = \text{Yes} \). Applying Proposition 11 repeatedly, we obtain \( \text{BA}^k(S_{q,i}, K_{b(i)(c)}) = \text{Yes} \). Noticing that \( K_{b(i)(c)} \rightarrow K_{q,i} \) and applying the second part of Theorem 10 repeatedly, we find \( S_{q,i} \rightarrow K_4 \). Again by Proposition 12, we conclude that \( \text{BA}^k(S_{q,i}, K_4) = \text{Yes} \), as required. For \( (ii) \), we first note that \( K_4 \not\rightarrow K_{b(i)(d)} \) as \( q \sim a(i)(d) \). Applying the (contrapositive of the) first part of Theorem 10 repeatedly, we deduce that \( S_{q,i} \not\rightarrow K_4 \), as required.

Suppose now that \( c = 3 \). Assume, for the sake of contradiction, that the \( k \)-th level of BA solves PCSP\((K_3, K_4)\). Let \( X \) be a digraph such that \( \text{BA}^{4k}(X, K_4) = \text{Yes} \). Applying Proposition 11 twice, we find that \( \text{BA}^k(\delta X, S_{8,2}) = \text{Yes} \). We now use the fact, observed in [59, Lemma 4.19], that \( S_{8,2} \rightarrow K_3 \); combining this with Proposition 12 yields \( \text{BA}^k(\delta (\delta X), K_3) = \text{Yes} \). Since we are assuming that \( \text{BA}^k \) solves PCSP\((K_3, K_4)\), we must have \( \delta (\delta X) \rightarrow K_4 \), whence it follows, through a double application of the first part of Theorem 10, that \( X \rightarrow K_{b(i)(d)} \). Thus, we have shown that the \( (4k) \)-th level of BA solves PCSP\((K_4, K_{b(i)(d)}\)), which is a PCSP template as \( d \geq c = 3 \) implies \( a(2)(d) = 2^d \geq 2^3 \geq 4 \), so \( K_4 \rightarrow K_{b(i)(d)} \). This contradicts the argument above establishing the case of \( c \geq 4 \).

**ACKNOWLEDGMENTS**

This research was funded in whole by UKRI EP/X024431/1. For the purpose of Open Access, the authors have applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission. All data is provided in full in the results section of this paper. The authors are grateful to Jakub Opršal, who suggested to us a connection between the line digraph reduction and the Sherali-Adams LP hierarchy for approximate graph colouring.

**REFERENCES**

[1] Richard P Anan’ev. 1982. Properties of a class of (0, 1)-matrices covering a given matrix. Canadian Journal of Mathematics 34, 2 (1982), 488–495. https://doi.org/10.4153/CJM-1982-029-3

[2] Sanjeev Arora, Béla Bollobás, László Lovász, and Iannis Tourlakis. 2006. Proving Integrality Gaps without Knowing the Linear Program. Theory Comput. 2, 2 (2006), 19–51. https://doi.org/10.4086/toc.2006.v002a002

[3] Kristina Asami and Libor Barto. 2021. Finutely Tractable Promise Constraint Satisfaction Problems. In Proc. 46th International Symposium on Mathematical Foundations of Computer Science (MFCS’21) (LIPIcs, Vol. 202). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 11:1–11:16. https://doi.org/10.4230/LIPIcs.MFCS.2021.11

---

\[\text{Figure 5: Shift digraphs.}\]
Approximate Graph Colouring and the Hollow Shadow

Workshop on Randomization and Computation (APPROX-RANDOM ’23). Springer. 233–243. https://doi.org/10.1007/978-3-642-40328-6_17 arXiv:1801.02106

[40] Charles R. Johnson and David P. Stanford. 2000. Patterns that allow given row and column sums. Linear Algebra Appl. 311, 1-3 (2000), 97–105. https://doi.org/10.1016/S0024-3795(00)00071-9

[41] Richard M. Karp. 1972. Reducibility Among Combinatorial Problems. In Proc. Complexity of Computer Computations. 85–103. https://doi.org/10.1007/978-1-4684-2001-2_9

[42] Ken-ichi Kawarabayashi and Mikkel Thorup. 2017. Coloring 3-Colorable Graphs with Less than \( n^{1/5} \) Colors. J. ACM 64, 1 (2017), 4:1–4:23. https://doi.org/10.1145/3001582

[43] Sanjeev Khanna, Nathaniel Linial, and Shmuel Safra. 2000. On the Hardness of Approximating the Chromatic Number. J. ACM 47, 3 (2000), 424–446. https://doi.org/10.1145/339329.339339

[44] Subhash Khot. 2002. On the power of unique 2-prover 1-round games. In Proc. 44th Annual ACM Symposium on Theory of Computing (STOC’02). ACM, 767–775. https://doi.org/10.1145/509907.510017

[45] Subhash Khot, Dor Minzer, and Muli Safra. 2017. On independent sets, 2-to-2 games, and Grassmann graphs. In Proc. 49th Annual ACM Symposium on Theory of Computing (STOC’17). ACM, 576–589. https://doi.org/10.1145/3055399.3055432

[46] Subhash Khot, Dor Minzer, and Muli Safra. 2018. Pseudorandom Sets in Grassmann Graph Have Near-Perfect Expansion. In Proc. 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS’18). IEEE Computer Society, 576–589. https://doi.org/10.1109/FOCS.2018.00062

[47] Pravesh K. Kothari, Raghu Meka, and Prasad Raghavendra. 2017. Approximating Subhash Khot, Dor Minzer, and Muli Safra. 2017. On independent sets, 2-to-2 games, and Grassmann graphs. In Proc. 49th Annual ACM Symposium on Theory of Computing (STOC’17). ACM, 576–589. https://doi.org/10.1145/3055399.3055432

[48] Andrei Krokhin and Jakub Opršal. 2022. An invitation to the promise constraint satisfaction problem. ACM SIGLOG News 9, 3 (2022), 30–59. arXiv:2208.13538

[49] Andrei A. Krokhin, Jakub Opršal, Marcin Wrochna, and Stanislav Živný. 2022. Topology and adjunction in promise constraint satisfaction. SIAM J. Comput. (2022). arXiv:2003.11351

[50] James R. Lee, Prasad Raghavendra, and David Steurer. 2015. Lower Bounds on the Size of Semidefinite Programming Relaxations. In Proc. 47th Annual ACM on Symposium on Theory of Computing (STOC’15). ACM, 567–576. https://doi.org/10.1145/2746539.2746599

[51] Tamio-Vesa Nakajima and Stanislav Živný. 2022. Linearly Ordered Colourings of Hypergraphs. In Proc. 49th International Colloquium on Automata, Languages, and Programming (ICALP’22) (LIPIcs, Vol. 229). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 128:1–128:18. https://doi.org/10.4230/LIPIcs.ICALP.2022.128 arXiv:2204.05628

[52] Pablo A Parrilo. 2000. Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. California Institute of Technology. http://www.cds.caltech.edu/~parrilo/thesis.pdf

[53] Monique Laurent. 2003. A Comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre Relaxations for 0-1 Programming. Math. Oper. Res. 28, 3 (2003), 470–496. https://doi.org/10.1287/moor.28.3.470.16391

[54] Jean B. Lasserre. 2002. A Proof of the CSP Dichotomy Conjecture. J. ACM 49, 3 (2002), 30:1–30:78. https://doi.org/10.1145/509907.510017

[55] Jean B. Lasserre. 2002. An Explicit Equivalent Positive Semidefinite Program for Nonlinear 0-1 Programs. SIAM J. Optim. 12, 3 (2002), 756–769. https://doi.org/10.1137/S1052623400380079

[56] H. D. Sherali and W. P. Adams. 1990. A Hierarchy of Relaxations between the Continuous and Convex Hull Representations for Zero-One Programming. SIAM J. Optim. 1, 1 (1990), 101–112. https://doi.org/10.1137/0801008

[57] Thanh-Vien Nguyen and Stanislav Živný. 2020. A Proof of the CSP Dichotomy Conjecture. J. ACM 67, 5 (2020), 30:1–30:78. https://doi.org/10.1145/3402029 arXiv:1704.01914

Received 2022-11-07; accepted 2023-02-06

631