Computing Infeasibility Certificates for Combinatorial Problems through Hilbert's Nullstellensatz

Jesús A. De Loera

Department of Mathematics, University of California, Davis, Davis, CA

Jon Lee

IBM T.J. Watson Research Center, Yorktown Heights, NY

Peter N. Malkin

Department of Mathematics, University of California, Davis, Davis, CA

Susan Margulies

Computational and Applied Math Department, Rice University, Houston, TX

Abstract

Systems of polynomial equations with coefficients over a field $\mathbb{K}$ can be used to concisely model combinatorial problems. In this way, a combinatorial problem is feasible (e.g., a graph is 3-colorable, hamiltonian, etc.) if and only if a related system of polynomial equations has a solution over the algebraic closure of the field $\mathbb{K}$. In this paper, we investigate an algorithm aimed at proving combinatorial infeasibility based on the observed low degree of Hilbert's Nullstellensatz certificates for polynomial systems arising in combinatorics, and based on fast large-scale linear-algebra computations over $\mathbb{K}$. We also describe several mathematical ideas for optimizing our algorithm, such as using alternative forms of the Nullstellensatz for computation, adding carefully constructed polynomials to our system, branching and exploiting symmetry. We report on experiments based on the problem of proving the non-3-colorability of graphs. We successfully solved graph instances with almost two thousand nodes and tens of thousands of edges.

Key words: combinatorics, systems of polynomials, feasibility, Non-linear Optimization, Graph 3-coloring

1. Introduction

It is well known that systems of polynomial equations over a field can yield compact models of difficult combinatorial problems. For example, it was first noted by D. Bayer that the 3-

Email addresses: deloera@math.ucdavis.edu (Jesús A. De Loera), jonlee@us.ibm.com (Jon Lee), malkin@math.ucdavis.edu (Peter N. Malkin), susan.margulies@rice.edu (Susan Margulies)

colorability of graphs can be modeled via a system of polynomial equations [2]. More generally, one can easily prove the following lemma:

Lemma 1.1. *A graph G is k-colorable if and only if the system of $n + m$ equations in n variables $x_i^k - 1 = 0, \forall i \in V(G)$, and $\sum_{l=0}^{k-1} x_i^{k-1-l} x_j^l = 0, \forall \{i, j\} \in E(G)$ has a complex solution. Moreover, the number of solutions equals the number of distinct k-colorings multiplied by $k!$.*

Although such polynomial system encodings have been used to prove combinatorial results (see [1, 10] and the references therein), they have not been widely used for computation. The key issue that we investigate here is the use of such polynomial systems to effectively decide whether a graph, or other combinatorial structure, has a property captured by the polynomial system and its associated ideal. We call this the *combinatorial feasibility problem*. We are particularly interested in whether this can be accomplished in practice for large combinatorial structures such as graphs with many nodes.

Certainly, using standard tools in computational algebra such as Gröbner bases, one can answer the combinatorial feasibility problem by simply solving the system of polynomials. Nevertheless, it has been shown by experiments that current Gröbner bases implementations often cannot directly solve polynomial systems with hundreds of polynomials. This paper proposes another approach that relies instead on the nice low degree of the Hilbert's Nullstellensatz for combinatorial polynomial systems and on large-scale linear-algebra computation.

For a hard combinatorial problem (e.g., 3-colorability of graphs), we associate a system of polynomial equations $J = \{f_1(x) = 0, \ldots, f_s(x) = 0\}$ such that the system J has a solution if and only if the combinatorial problem has a feasible solution. The Hilbert Nullstellensatz (see e.g.,[7]) states that the system of polynomial equations with coefficients over a field $\mathbb{K}$ has *no* solution over its algebraically-closure $\bar{\mathbb{K}}$ if and only if there exist polynomials $\beta_1, \ldots, \beta_s \in \mathbb{K}[x_1, \ldots, x_n]$ such that $1 = \sum \beta_i f_i$. Thus, if the polynomial system J has no solution, then there exists a *certificate* that J has no solution, and thus a proof that the combinatorial problem is infeasible.

The key idea that we explore in this article is to use the Nullstellensatz to generate a finite sequence of linear algebra systems, of increasing size, which will eventually become *feasible* if and only if the combinatorial problem is *infeasible*. Roughly speaking, given a system of polynomial equations, we fix a tentative degree d for the certificate meaning $\deg(\beta_i f_i) = d$ for every $i = 1, ..., s$. Then, we can decide whether there is a Nullstellensatz certificate of degree d by solving a system of *linear* equations over the field $\mathbb{K}$ whose variables are in bijection with the coefficients of the monomials of the polynomials $\beta_1, \ldots, \beta_s$. If this linear system has a solution, we have found a certificate; otherwise, we try a higher degree for the certificate. This process is guaranteed to terminate because, for a Nullstellensatz certificate to exist, the degrees of the certificate cannot be more than known bounds (see e.g., [16] and references therein). We explain the details of the algorithm, which we call **N**ul**LA**, in Section 2.

Our method can be seen as a general-field variation of recent exciting work by Lasserre [17], Laurent [18], Parrilo [26] and many others, who studied the problem of minimizing a general polynomial function $f(x)$ over a real algebraic variety with finitely many points. Laurent proved that when the variety consists of the solutions of a zero-dimensional ideal I, one can set up the optimization problem $\min\{f(x) : x \in \text{variety}(I)\}$ as a finite sequence of semidefinite programs terminating with the optimal solution (see [18]). In our case, we only desire to decide combinatorial feasibility (e.g., is this graph 3-colorable?), thus there are two key observations that speed up practical calculations considerably: (1) when dealing with feasibility, instead of optimization, linear algebra replaces semidefinite programming and (2) there are many ways of

controlling the size of the sequence of linear-algebra systems. We discuss details of a variety of mathematical ideas for controlling the size of the sequence in Section 3. These ideas include the following: computing over finite fields instead of over the reals, designing carefully-constructed polynomials that can actually decrease the length of the sequence in some cases, exploring alternative forms of Hilbert's Nullstellensatz more suitable for computation in a particular instance, branching to create polynomial subsystems with smaller sequences of linear-algebra systems, and exploiting symmetries in the linear system. These ideas are new developments or extensions of the ideas presented in our previous paper [11].

Our algorithm has very good practical performance and numerical stability. Although known theoretical bounds for degrees of the Nullstellensatz coefficients are doubly exponential in the size of the polynomial system (and indeed there exist pathological examples that attain such a large bound and make **N**ul**LA** useless in general), our experiments demonstrate that very low degrees suffice for systems of polynomials coming from graph theory, even for very large graphs. We have implemented an exact-arithmetic linear system solver optimized for these Nullstellensatz-based systems. We performed many experiments using **N**ul**LA**, focusing on the problem of deciding graph 3-colorability (note nevertheless that the method presented here is applicable to any combinatorial problem for which a polynomial system encoding is known). We conclude with a report on these experiments in Section 4.

2. Nullstellensatz Linear Algebra (NulLA) Algorithm

We start by recalling Hilbert's Nullstellensatz in the traditional statement found in most textbooks (for a proof see e.g., [7]): A system of polynomial equations $f_1(x) = 0, \ldots, f_s(x) = 0$, where $f_i \in \mathbb{K}[x_1, \ldots, x_n]$ and $\mathbb{K}$ is an algebraically closed field, has no solution in $\mathbb{K}^n$ if and only if there exist polynomials $\beta_1, \ldots, \beta_s \in \mathbb{K}[x_1, \ldots, x_n]$ such that $1 = \sum \beta_i f_i$.

In this paper, we will use a slightly stronger form that is much more useful for our purposes and can be easily derived from the classical statement above. This stronger form allows us to perform calculations over any field $\mathbb{K}$ even if $\mathbb{K}$ is not algebraically closed.

Lemma 2.1. *Let $\mathbb{K}$ be a field and $\overline{\mathbb{K}}$ its algebraic closure. Given $f_1, f_2, \ldots, f_s \in \mathbb{K}[x_1, \ldots, x_n]$, the system of polynomial equations $f_1(x) = 0, \ldots, f_s(x) = 0$, with $f_i \in \mathbb{K}[x_1, \ldots, x_n]$ and has no solution in $\overline{\mathbb{K}}^n$ if and only if there exist polynomials $\beta_1, \ldots, \beta_s \in \mathbb{K}[x_1, \ldots, x_n]$ such that $1 = \sum \beta_i f_i$.*

In other words, there exists a Nullstellensatz certificate $1 = \sum \beta_i f_i$ where $\beta_i \in \overline{\mathbb{K}}[x_1, \ldots, x_n]$ if and only if there exists a Nullstellensatz certificate $1 = \sum \beta_i' f_i$ where $\beta_i' \in \mathbb{K}[x_1, \ldots, x_n]$.

Definition 2.2. *The polynomial identity $1 = \sum \beta_i f_i$ is called a* Nullstellensatz certificate*, which has degree d if $\max_i\{\deg(\beta_i f_i)\} = d$.*

Now we describe the simple Nullstellensatz Linear Algebra (**N**ul**LA**) algorithm. It accepts as input a system of polynomial equations and outputs either a *yes answer*, if the system of polynomial equations has a solution, or a *no answer*, along with a Nullstellensatz infeasibility certificate, if the system has no solution. Before stating the algorithm in pseudocode, we clarify the connection to linear algebra. Suppose the input polynomial system is infeasible over $\mathbb{K}$, and suppose further that an oracle has told us the certificate has degree d but that we do not know the actual coefficients of the polynomials β_i. Thus, we have the polynomial identity $1 = \sum \beta_i f_i$. If we expand the identity into monomials, the coefficients of a monomial are linear expressions

in the coefficients of the β_i. Since two polynomials over a field are identical precisely when the coefficients of corresponding monomials are identical, from the $1 = \sum \beta_i f_i$, we get a system of linear equations whose variables are the coefficients of the β_i. Here is an example:

Example 2.3. Consider the polynomial system $x_1^2 - 1 = 0, x_1 + x_2 = 0, x_1 + x_3 = 0, x_2 + x_3 = 0$. This system has no solution, and a Nullstellensatz certificate of degree two.

$$\begin{aligned} 1 = &\underbrace{(c_0)}_{\beta_1}\underbrace{(x_1^2 - 1)}_{f_1} + \underbrace{(c_1x_1 + c_2x_2 + c_3x_3 + c_4)}_{\beta_2}\underbrace{(x_1 + x_2)}_{f_2} \\ &+ \underbrace{(c_5x_1 + c_6x_2 + c_7x_3 + c_8)}_{\beta_3}\underbrace{(x_1 + x_3)}_{f_3} + \underbrace{(c_9x_1 + c_{10}x_2 + c_{11}x_3 + c_{12})}_{\beta_4}\underbrace{(x_2 + x_3)}_{f_4}. \end{aligned}$$

Expanding the tentative Nullstellensatz certificate into monomials and grouping like terms, we arrive at the following polynomial equation:

$$\begin{aligned} 1 = &- c_0 + (c_4 + c_8)x_1 + (c_4 + c_{12})x_2 + (c_8 + c_{12})x_3 \\ &+ (c_0 + c_1 + c_5)x_1^2 + (c_1 + c_2 + c_6 + c_9)x_1x_2 + (c_3 + c_5 + c_7 + c_9)x_1x_3 \\ &+ (c_2 + c_{10})x_2^2 + (c_3 + c_6 + c_{10} + c_{11})x_2x_3 + (c_7 + c_{11})x_3^2. \end{aligned}$$

From this, we extract a system of *linear* equations. Since a Nullstellensatz certificate is identically one, all monomials except the constant term must be equal to zero; namely:

$$\begin{aligned} -c_0 = 1, \quad & c_4 + c_8 = 0, & c_4 + c_{12} = 0, & & c_8 + c_{12} = 0, \\ & c_0 + c_1 + c_5 = 0, & c_1 + c_2 + c_6 + c_9 = 0, & & c_3 + c_5 + c_7 + c_9 = 0, \\ & c_2 + c_{10} = 0, & c_3 + c_6 + c_{10} + c_{11} = 0, & & c_7 + c_{11} = 0. \end{aligned}$$

By solving the system of linear equations, we reconstruct the Nullstellensatz certificate from the solution:

$$1 = -(x_1^2 - 1) + \frac{1}{2}x_1(x_1 + x_2) + \frac{1}{2}x_1(x_1 + x_3) - \frac{1}{2}x_1(x_2 + x_3).$$

In general, one does not know the degree of the Nullstellensatz certificate in advance. What one can do is to start with a tentative degree, say start at degree $\max_i\{\deg(f_i)\}$, produce the corresponding linear system, and solve it. If the system has a solution, then we have found a Nullstellensatz certificate demonstrating that the original input polynomials do not have a common root. Otherwise, we increment the degree until we can be sure that there will not be a Nullstellensatz certificate at all, and thus we can conclude the system of polynomials has a solution. The number of iterations of the above steps determines the running time of **NulLA**. For this, there are well-known upper bounds on the degree of the β_i in the Nullstellensatz certificate (see Kollár [16] and references therein), and thus on the degree of the certificate. These upper bounds for the degrees of the β_i in the Hilbert Nullstellensatz certificates for *general* systems of polynomials are doubly-exponential in the number of input polynomials and their degree.

Unfortunately, Kollár's bounds [16] are known to be sharp for some specially-constructed systems. Although this immediately says that **NulLA** is not practical for arbitrary polynomial systems, this is far from the end for computing with *combinatorial* polynomial systems. First of all, a fundamental result by D. Lazard [19] provides ideals like ours (ideals that can be homogenized with the addition of one or more variables such that there no common zeros at infinity) with a *linear* bound.

Lemma 2.4 (Lazard [19]). *Let $f_1, \ldots, f_k$ be homogeneous polynomials of $\mathbb{K}[x_0, \ldots, x_n]$ that generate an ideal I, let d_i be the degree of f_i and assume that $d_1 \geq d_2 \geq \cdots \geq d_k \geq 1$ and $k \geq n+1$. Then the following conditions are equivalent:*

1) *The k projective hypersurfaces defined by $f_1, \ldots, f_k$ have no point in common over the algebraic closure of $\mathbb{K}$ (in particular, they have no point in common at infinity).*

2) *The ideal I contains a power of the maximal ideal $M = \langle x_0, x_1, \ldots, x_n \rangle$; namely, for some power p, $x_i^p \in I$ for all x_i.*

3) *$M^p \subset I$ with $p = d_1 + d_2 + \cdots + d_{n+1} - n \leq (n+1)(\max_{1 \leq i \leq n+1}\{d_i\} - 1) + 1$.*

4) *The map $\phi : (\beta_1, \ldots, \beta_k) \rightarrow \sum \beta_i f_i$ is surjective among all polynomials of degree p, when, for all i, β_i is a homogeneous polynomial of degree $p - d_i$.*

The proof of Lemma 2.4 relies on advanced techniques in commutative and homological algebra, and is presented in [19], pg. 169. As a consequence of Lemma 2.4, when given polynomials $f_i \in \mathbb{K}[x_1, \ldots, x_n]$, we can consider their homogenization $\bar{f}_i$, using an extra variable x_0 (e.g., $x^2 - x$ can be homogenized to $x^2 - xx_0$). If we are able to find a "projective" Nullstellensatz of the form

$$x_0^p = \sum \beta_i \bar{f}_i \, ,$$

then we can substitute $x_0 = 1$ in the above equation and obtain the form of the Nullstellensatz that is more desirable for computation (e.g., $1 = \sum \beta_i' f_i$). Furthermore, the degree of β_i' is less than or equal to the degree of β_i.

We can summarize the Lazard lemma as follows (see also Brownawell [4]):

Corollary 2.5. *Given polynomials $f_1, \ldots, f_s \in \mathbb{K}[x_1, \ldots, x_n]$ where $\mathbb{K}$ is an algebraically-closed field and $d = \max\{\deg(f_i)\}$, if $f_1, \ldots, f_s$ have no common zeros and $f_1, \ldots, f_s$ have no common zeros at infinity, then $1 = \sum_{i=1}^{s} \beta_i f_i$ where*

$$\deg(\beta_i) \leq n(d-1) \, .$$

Therefore, the bound on Nullstellensatz described by combinatorial ideals (for example, see Lemma 3.1) gives linear growth on the degree of the Nullstellensatz certificates. This a considerable improvement on the exponential bound predicted by Kollár, but our second point is that, in practice, polynomial systems for combinatorial questions are extremely specialized, and the degree growth is often *very* slow, and is much better than even Lazard's bound — enough to deal with very large graphs or other combinatorial structures. Now we describe **Nul**l**LA** in pseudocode:

********** **********

ALGORITHM: **Null**stellensatz **L**inear **A**lgebra (**NullLA**) Algorithm
INPUT: A system of polynomial equations $F = \{f_1(x) = 0, \ldots, f_s(x) = 0\}$
OUTPUT: YES, if F has solution, else NO along with a Nullstellensatz certificate of infeasibility.
  $d \leftarrow \max_i\{\deg(f_i)\}$.
  $K \leftarrow$ known upper bounds on degree of Nullstellensatz for F (see e.g., [16])
  while $d \leq K$ **do**
    CERT $\leftarrow \sum_{i=1}^{s} \beta_i f_i$ (where β_i are degree $(d - \deg(f_i))$ polynomials with unknowns for coefficients).

Extract a system of linear equations from CERT with columns corresponding to unknowns, and rows corresponding to monomials.
Solve the linear system.
if the linear system is consistent **then**
CERT $\leftarrow \sum_{i=1}^{s} \beta_i f_i$ (with unknowns in β_i replaced with linear system solution values.)
print "The system of equations F is infeasible."
return NO with CERT.
end if
$d \leftarrow d + 1$.
end while
print "The system of equations F is feasible."
return YES.

This opens several theoretical questions. It is natural to ask about lower bounds on the degree of the Nullstellensatz certificates. Little is known, but recently it was shown in [10], that for the problem of deciding whether a given graph G has an independent set of a given size, a minimum-degree Nullstellensatz certificate for the non-existence of an independent set of size greater than $\alpha(G)$ (the size of the largest independent set in G) has β_i with degree less than or equal to $\alpha(G)$, and it is very dense; specifically, it contains at least one term per independent set in G. For polynomial systems coming from logic there has also been an effort to show degree growth in related polynomial systems (see [5, 13] and the references therein). Another question is to provide tighter, more realistic upper bounds for concrete systems of polynomials. It is a challenge to settle it for any concrete family of polynomial systems.

3. Some mathematical ideas to optimize NulLA

Since we are interested in practical computational problems, it makes sense to explore refinements and variations that make **NulLA** robust and much faster for concrete challenges. The main computational component of **NulLA** is to construct and solve linear systems for finding Nullstellensatz certificates of increasing degree. These linear systems are typically very large for reasonably-sized problems, even for certificate degrees as low as six, which can produce linear systems with millions of variables (see Section 4). Furthermore, the size of the linear system increases dramatically with the degree of the certificate. In particular, the number of variables in the linear system to find a Nullstellensatz certificate of degree d is precisely $\sum_i \binom{n+d_i}{d_i}$ where n is the number of variables in the polynomial system and $d_i = d - \deg(f_i)$ is the degree of β_i. Note that $\binom{n+d}{d}$ is the number of possible monomials of degree d or less. Also, the number of non-zero entries in the constraint matrix is precisely $\sum_i M_i \binom{n+d_i}{d_i}$ where M_i is number of monomials in f_i.

For this reason, in this section, we explore mathematical approaches for solving the linear system more efficiently and robustly, for decreasing the size of the linear system for a given degree, and for decreasing the degree of the Nullstellensatz certificate for infeasible polynomial systems thus significantly reducing the size of the largest linear system that we need to solve to prove infeasibility. Note that these approaches to reduce the degree of the Nullstellensatz certificates do not decrease the available upper bound on the degree of the Nullstellensatz certificate required for proving feasibility, but they work in particular instances.

The mathematical ideas we explain in this section can be applied to arbitrary polynomial systems, but to implement them, one has to look for the right structures in the polynomials.

In what follows we illustrate this with the problem of deciding whether the vertices of a graph permit a proper 3-coloring.

3.1. NulLA over Finite Fields

The first idea is that, for combinatorial problems, one can often carry out calculations over finite fields instead of relying on unstable floating-point calculations. The following encoding (a variation of [2] over the complex numbers) allows us to compute over $\mathbb{F}_2$, which is robust and much faster in practice (also see [12]):

Lemma 3.1. *The graph G is* 3*-colorable if and only if the zero-dimensional system of equations $x_i^3 + 1 = 0, \forall i \in V(G)$, and $x_i^2 + x_i x_j + x_j^2 = 0, \forall \{i, j\} \in E(G)$, has a solution over $\overline{\mathbb{F}}_2$, the algebraic closure of $\mathbb{F}_2$.*

Before we prove Lemma 3.1, we introduce a convenient notation: Let α be an algebraic element over $\overline{\mathbb{F}}_2$ such that $\alpha^2 + \alpha + 1 = 0$. Thus, although $x_i^3 + 1$ has only one root over $\mathbb{F}_2$, since $x_i^3 + 1 = (x_i + 1)(x_i^2 + x_i + 1)$, the polynomial $x_i^3 + 1$ has three roots over $\overline{\mathbb{F}}_2$, which are $1, \alpha$ and $\alpha + 1$.

Proof. If the graph G is 3-colorable, simply map the three colors to $1, \alpha$ and $\alpha + 1$. Clearly, the vertex polynomial equations $x_i^3 + 1 = 0$ are satisfied. Furthermore, given an edge $\{i, j\}$, $x_i + x_j \neq 0$ since variable assignments correspond to a proper 3-coloring and adjacent vertices are assigned different roots. This implies that $x_i^3 + x_j^3 = (x_i + x_j)(x_i^2 + x_i x_j + x_j^2) = 1 + 1 = 0$. Therefore, $x_i^2 + x_i x_j + x_j^2 = 0$ and the edge polynomial equations are satisfied.

Conversely, suppose that there exists a solution to the system of polynomial equations. Clearly, every vertex is assigned either $1, \alpha$ or $\alpha + 1$. We will show that adjacent vertices are assigned different values. Our proof is by contradiction: Assume that two adjacent vertices i, j are assigned the same value β. Then, $0 = x_i^2 + x_i x_j + x_j^2 = \beta^2 + \beta^2 + \beta^2 = 3\beta^2 \neq 0$. Therefore, adjacent vertices are assigned different roots, and a solution to the system corresponds directly to a proper 3-coloring. □

We remark that this result can be extended to k-colorability and $\overline{\mathbb{F}}_q$, when q is relatively prime to k. Lemma 3.1 allows us to certify graph non-3-colorability very rapidly over $\mathbb{F}_2$ instead of working over its algebraic closure. Namely,

Corollary 3.2. *A graph G is* non*-3-colorable if and only if there exists a Nullstellensatz certificate $1 = \sum \beta_i f_i$ where $\beta_i \in \mathbb{F}_2[x_1, \ldots, x_n]$ where the polynomials $f_i \in \mathbb{F}_2[x_1, \ldots, x_n]$ are as defined in Lemma 3.1.*

This corollary enables us to compute over $\mathbb{F}_2$, which is extremely fast in practice (see Section 4).

Finally, the degree of Nullstellensatz certificates necessary to prove infeasibility can indeed be lower over $\mathbb{F}_2$ than over the rationals. For example, over the rationals, every odd-wheel has a minimum non-3-colorability certificate of degree six [10]. However, over $\mathbb{F}_2$, every odd-wheel has a Nullstellensatz certificate of degree three. Therefore, not only are the mathematical computations more efficient over $\mathbb{F}_2$ as compared to the rationals, but the algebraic properties of the certificates themselves are sometimes more favorable for computation as well.

3.2. Reducing the Nullstellensatz degree by appending polynomial equations

We have discovered that by *appending* certain valid but redundant polynomial equations to the system of polynomial equations described in Lemma 3.1, we have been able to *decrease* the degree of the Nullstellensatz certificate necessary to prove infeasibility. A valid but redundant polynomial equation is any polynomial equation $g(x) = 0$ that is true for all the zeros of the polynomial system $f_1(x) = 0, ..., f_s(x) = 0$, i.e., $g \in \sqrt{I}$, the radical ideal of I, where I is the ideal generated by $f_1, ..., f_s$. We refer to a redundant polynomial equation appended to a system of polynomial equations, with the goal of reducing the degree of a Nullstellensatz certificate, as a *degree-cutter*. Note that appending an equation could never increase the necessary degree of a Nullstellensatz certificate.

For example, for 3-coloring, consider a triangle described by the vertices $\{x, y, z\}$. Whenever a triangle appears as a subgraph in a graph, the vertices of the triangle must be colored differently. We capture that additional requirement with the equation

$$x^2 + y^2 + z^2 = 0, \tag{1}$$

which is satisfied if and only if $x \neq y \neq z \neq x$ since x, y and z are third roots of unity. It is worth remarking that the equation $x + y + z = 0$ also implies $x \neq y \neq z \neq x$. We use the equation $x^2 + y^2 + z^2 = 0$ instead, which is homogeneous of degree two, because the edge equations from Lemma 3.1 are also homogeneous of degree two, and this helps preserve the balance of monomials in the final certificate.

Consider the Koester graph [15] from Figure 1, a graph with 40 vertices and 80 edges. This graph has chromatic number four, and a corresponding non-3-colorability certificate of degree six. The size (after preprocessing) of the associated linear system required by **N**ul**LA** to produce this certificate was $8,724,468 \times 10,995,831$ and required 5 hours and 17 minutes of computation time.

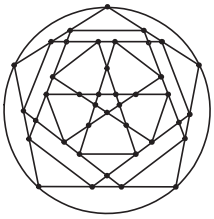

Figure 1: Koester graph

When we inspect the Koester graph in Figure 1, we can see that this graph contains 25 triangles. When we append these additional 25 equations to the system of polynomial equations describing this graph, the degree of the Nullstellensatz certificate drops from six to three, and now, with the addition of the 25 triangle equations, **N**ul**LA** only needs to solve a $4,626 \times 4,346$ linear system to produce a degree one certificate, which takes 0.2 seconds of computation time. Note that even though we have *appended* equations to the system of polynomial equations, because the degree of the overall certificate is drastically *reduced*, the size of the resulting linear system is still much, much smaller.

These degree-cutter equations for 3-colorability (1) can be extended to k-colorability. A $(k-1)$-clique implies that all nodes in the clique have a different color. Then, given the $(k-1)$-clique

with the vertices x_1 through x_{k-1}, the equation $x_1^{k-1} + x_2^{k-1} + ... + x_{k-1}^{k-1} = 0$ is valid. We conjecture that these equations may also decrease the minimal degree of the Nullstellensatz certificate if one exists.

The degree-cutter equations for 3-colorability (1) are not always sufficient to reduce the degree of the Nullstellensatz. Consider the graph from Figure 2. Using only the polynomials from Lemma 3.1, the graph in Figure 2 has a degree six certificate. The graph contains three triangles: $\{1, 2, 6\}, \{2, 5, 6\}$ and $\{2, 6, 7\}$. In this case, after appending the degree-cutter equations for 3-colorability (1) the degree of the minimal Nullstellensatz certificate for this graph is still six. However, for this graph, there are other types of equations that we can append to lower the degree, which we discuss below.

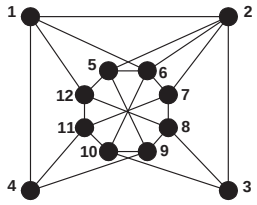

Figure 2: A graph with a degree four certificate.

The polynomial equation $g(x) = 0$ that we append to the system of equations need not belong to the radical $\sqrt{I}$ as above, but instead, we only require the weaker condition that $f_1(x) = 0, ..., f_s(x) = 0$ is feasible if and only if $f_1(x) = 0, ..., f_s(x) = 0, g(x) = 0$ is feasible.

For example, if a graph has a k-coloring, then it still has a k-coloring after fixing the color of one of the vertices, which means that the polynomial system encoding graph coloring has a solution if and only if it has a solution after appending the equation $g(x) = x_i - \alpha = 0$ for some vertex i where α is a kth root of unity, say 1. Note that appending the polynomial $g(x) = x_i - \alpha = 0$ is the same as fixing the value of x_i to α in the polynomial system thereby eliminating x_i from the system, which is a more efficient approach in practice. We found that even fixing just one variable can lead to a lower certificate degree for non-trivial graphs.

For example, consider the graph in Figure 2, which has a degree six certificate of non-3-colorability. This graph has a degree four certificate after fixing x_1 to 1, that is, after appending the equation $x_1 - 1 = 0$ to the system of polynomial equations encoding graph 3-colorability.

Moreover, for graph coloring, we can fix two variables corresponding to two adjacent nodes to two different roots of unity without affecting the feasibility of the polynomial system, and furthermore, we can fix k variables corresponding to a k-clique to k different roots of unity. Fixing variables to roots of unity other than 1 complicates the polynomial system since then the coefficient field must be extended to include the roots of unity. Specifically, if we wish to fix variables to kth roots of unity other than one, then we need to extend the field $\mathbb{K}$ to the splitting field of $x_0^k - 1$ over $\mathbb{K}$, which is the smallest field containing $\mathbb{K}$ and all the kth roots of unity. Note that, for 3-coloring, the splitting field of $x_0^3 - 1$ over $\mathbb{F}_2$ is isomorphic to $\mathbb{F}_{2^2}$. Performing the linear algebra operations over the field extension is slower, but if fixing more variables leads to a lower Nullstellensatz degree, then it may be computationally worthwhile doing so.

For instance, consider again the graph in Figure 2, which has a degree four certificate after fixing x_1. There is a degree three certificate if instead we fix the variables x_2, x_5 and x_6 to three different roots of unity since they correspond to a triangle in the graph.

The difficulty with the degree-cutter approach is in finding candidate degree-cutters and in

determining how many of the candidate degree-cutters to append to the system. There is an obvious trade-off here between the time spent finding degree-cutters together with the time penalty incurred related to the increased size of the linear system that must be solved versus the benefit of reducing the degree of the Nullstellensatz certificate.

3.3. Branching

Branching is another way of appending polynomial equations to reduce the degree of the Nullstellensatz certificate required to prove infeasibility. The well-known main fact behind branching is the following: given $g_1(x), g_2(x) \in \mathbb{K}[x_1, ..., x_n]$ such that $g_1(x)g_2(x) \in I$ where I is the ideal generated by $f_1, ..., f_k$, the polynomial system $f_1(x) = 0, ..., f_k(x) = 0$ is infeasible if and only if both the subsystem $f_1(x) = 0, ..., f_k(x) = 0, g_1(x) = 0$ is infeasible and the subsystem $f_1(x) = 0, ..., f_k(x) = 0, g_2(x) = 0$ is infeasible. The obvious choice for $g_1(x)$ and $g_2(x)$ is where one of the polynomials f_i factors as $f_i(x) = g_1(x)g_2(x)$. Thus, to check for infeasibility of a polynomial system, we can check for infeasibility of two more constrained polynomial subsystems in the hope that the more constrained subsystems have lower minimal degrees than the original system such that it is faster to prove infeasibility of the two subsystems than the original system.

This approach of creating two more constrained polynomials system from one can be applied recursively leading to the following general branching scheme. First, we try to find a Nullstellensatz certificate of infeasibility of a particular degree of the original system, and then, if this fails, instead of increasing the degree and trying again, we branch and attempt to find a certificate of the same degree for the two subsystems. If we fail to find a certificate for one or both of the subsystems, then again, we branch on the failed subsystem and try again to find a certificate of the same degree, and so on. If all generated subsystems are infeasible, the original system is infeasible. If, however, we reach a subsystem for which we can no longer branch on and we cannot prove infeasibility, then we must start the branching process again with a higher degree. We must keep increasing the degree until infeasibility is shown or until the degree is high enough to prove feasibility.

We applied this branching approach to the case of 3-coloring of a graph $G = (V, E)$ where we tried to find a degree three certificate of infeasibility for the polynomial system encoding 3-coloring over $\mathbb{F}_2$. Here, to branch on a subsystem, we choose a variable x_i and branch on the two separate cases for $g_1 = x_i + 1$ and $g_2 = x_i^2 + x_i + 1$ where in the first case x_i is fixed to 1 and in the second case x_i is constrained to be a root of unity other than 1. The graph below has a degree six certificate of non-3-colorability, which takes 6.33 seconds to compute on a machine with dual Opteron nodes, 2GHz clock speed, and 12 GB of RAM. If we run the branching algorithm above, then we can prove infeasibility of subsystems in 0.01 seconds by proving infeasibility of 9 subsystems via degree three certificates. See Section 4 for more results for the branching algorithm.

Interestingly, the above branching algorithm for 3-colorability has the important property that if we reach a subsystem where we have branched on every variable but we cannot find a degree three certificate, then the graph is 3-colorable – we have proven feasibility and we do not need to increase the degree and try again. If we have branched on every variable, then every variable is either is fixed to be 1 or not 1, and this subsystem is infeasible if and only if two adjacent vertices have been fixed to be 1 or the subgraph induced by the vertices that are fixed to be not 1 is not 2-colorable, and in either of these two cases, there exists a degree three certificate attesting infeasibility as shown below: Firstly, if two adjacent vertices $i, j \in V$ are fixed to 1, then the

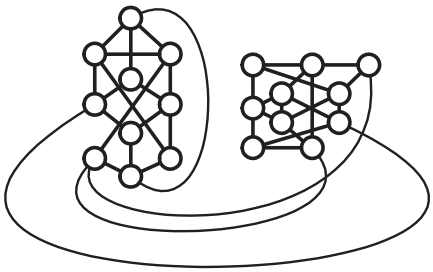

Figure 3: An example of a Liu-Zhang 4-CGU.

following is a degree two certificate of infeasibility:

$$(1 + x_i + x_j)(x_i + 1) + (x_i^2 + x_i x_j + x_j^2) + (x_j)(x_j + 1) = 1.$$

Secondly, a graph is not 2-colorable if and only if there exists an odd length cycle in the graph. Now, if $C = (v_1, v_2, ..., v_s) \subseteq V$ is an odd length cycle among the vertices fixed to not 1, then the following is a degree three certificate of infeasibility:

$$\sum_{r=1}^{s-1} \left[(x_{v_r} + x_{v_{r+1}} + 1)(x_{v_r}^2 + x_{v_r} + 1) + (x_{v_r})(x_{v_{r+1}}^2 + x_{v_{r+1}} + 1) + (x_{v_r})(x_{v_r}^2 + x_{v_r} x_{v_{r+1}} + x_{v_{r+1}}^2) \right] = 1$$

Branching can also be applied for arbitrarily many subsystems: given $g_1, ..., g_s \in \mathbb{K}[x_1, ..., x_n]$ such that $g_1 \cdots g_s \in I$ where I is the ideal generated by $f_1, ..., f_k$, the polynomial system $f_1(x) = 0, ..., f_k(x) = 0$ is infeasible if and only if each subsystem $f_1(x) = 0, ..., f_k(x) = 0, g_i(x) = 0$ is infeasible for all $i = 1, ..., s$.

3.4. Alternative Nullstellensätze

There is another approach we have found to decrease the minimal degree of the Nullstellensatz certificate. We now introduce the idea of an *alternative Nullstellensatz*, which follows from the Hilbert Nullstellensatz.

Corollary 3.3 (Alternative Nullstellensatz)**.** *A system of polynomial equations $f_1(x) = 0, \ldots, f_s(x) = 0$ where $f_i \in \mathbb{K}[x_1, \ldots, x_n]$ and $\mathbb{K}$ is an algebraically closed field has no solution in $\mathbb{K}^n$ if and only if there exist polynomials $\beta_1, \ldots, \beta_s \in \mathbb{K}[x_1, \ldots, x_n]$ and $g \in \mathbb{K}[x_1, ..., x_n]$ such that $g = \sum \beta_i f_i$ and the system $f_1(x) = 0, \ldots, f_s(x) = 0$ and $g(x) = 0$ has no solution.*

The Hilbert Nullstellensatz is a special case of this alternative Nullstellensatz where $g(x) = 1$. We can easily adapt the **NulLA** algorithm to use this alternative Nullstellensatz given the polynomial g. Here, the polynomial g determines the constant terms of the linear system that we need to solve to find a certificate of infeasibility. The idea here is that the minimal degree of the alternative Nullstellensatz certificate is sometimes smaller than the minimal degree of the ordinary Nullstellensatz certificate.

In the case of 3-colorability (and also more generally k-colorability), we may choose g as any non-trivial monomial since $g(x) = 0$ implies that $x_i = 0$ for some $i = 1, ..., n$, which contradicts that $x_i^3 - 1 = 0$. For the graph in Figure 2, if we choose $g(x) = x_1 x_8 x_9$, then the minimal degree of

the Nullstellensatz certificate drops to three (after appending degree-cutter polynomial equations to the system).

$$
\begin{aligned}
x_1x_8x_9 = {} & (x_1+x_2)(x_1^2+x_1x_2+x_2^2)+(x_4+x_9+x_{12})(x_1^2+x_1x_4+x_4^2)\\
& +(x_1+x_4+x_8)(x_1^2+x_1x_{12}+x_{12}^2)+(x_2+x_7+x_8)(x_2^2+x_2x_3+x_3^2)\\
& +(x_3+x_8)(x_2^2+x_2x_7+x_7^2)+(x_{10}+x_{12})(x_4^2+x_4x_{11}+x_{11}^2)\\
& +(x_1+x_4+x_{10})(x_4^2+x_4x_9+x_9^2)+(x_2+x_7+x_8)(x_3^2+x_3x_8+x_8^2)\\
& +(x_2+x_{10})(x_5^2+x_5x_6+x_6^2)+(x_5+x_{10})(x_5^2+x_5x_9+x_9^2)\\
& +(x_2+x_3+x_{12})(x_7^2+x_7x_8+x_8^2)+(x_1+x_7+x_8)(x_8^2+x_8x_{12}+x_{12}^2)\\
& +(x_2+x_{10})(x_6^2+x_6x_7+x_7^2)+(x_{10}+x_{12})(x_7^2+x_7x_{11}+x_{11}^2)\\
& +(x_5)(x_2^2+x_2x_5+x_5^2)+(x_5+x_7)(x_6^2+x_6x_{10}+x_{10}^2)\\
& +(x_4+x_7)(x_{10}^2+x_{10}x_{11}+x_{11}^2)+(x_4+x_5)(x_9^2+x_9x_{10}+x_{10}^2)\\
& +(x_1)(x_8^2+x_8x_9+x_9^2)+(x_4+x_7)(x_{11}^2+x_{11}x_{12}+x_{12}^2)+(x_5+x_7)(x_2^2+x_2x_6+x_6^2)\\
& +(x_8+x_9)\underbrace{(x_1^2+x_2^2+x_6^2)}_{\text{degree-cutter}}+(x_9)\underbrace{(x_2^2+x_5^2+x_6^2)}_{\text{degree-cutter}}+(x_8)\underbrace{(x_2^2+x_6^2+x_7^2)}_{\text{degree-cutter}}.
\end{aligned}
$$

We note $g(x) = x_1x_8x_9$ was not the only alternative Nullstellensatz certificate that we were able to find: $g(x) = x_7x_4x_9$ also produced a certificate. □

The apparent difficulty in using the alternative Nullstellensatz approach is in choosing $g(x)$. One solution to this problem is to try and find a Nullstellensatz certificate for a set of $g(x)$ including $g(x) = 1$. For example, for the graph in Figure 2, we tried to find a certificate of degree three for the set of all possible monomials of degree three. Since choosing different $g(x)$ only means changing the constant terms of the linear system in **NulLA** (the other coefficients remain the same), solving for a set of $g(x)$ can be accomplished very efficiently.

3.5. Deleting equations and exploiting linear dependencies

Here are two more ideas on how to reduce the size of the linear system to find a Nullstellensatz certificate of infeasibility.

First, one way to reduce the size of the linear system is to remove all polynomial equations $f_i(x) = 0$ for which there exists $h_1, ..., h_{i-1}, h_{i+1}, ..., h_k \in \mathbb{K}[x_1, ..., x_n]$ such that $f_i = \sum_{j\neq i} h_j f_j$ and $\deg(h_j f_j) \leq \deg(f_i)$ for all $j \neq i$. If the above condition holds for f_i, then the polynomial is redundant since f_i is in the ideal generated by $f_1, ..., f_{i-1}, f_{i+1}, ..., f_k$. Moreover, removing f_i can never increase the degree of a certificate since replacing f_i with $\sum_{j\neq i} h_j f_j$ in a given certificate gives another certificate of the same degree but without f_i. Note that the degree-cutting polynomials that we add in Section 3.2 are chosen specifically so that they do not satisfy the above condition, and thus, those polynomials, although redundant, may still reduce the degree.

For the case of k-coloring for a connected graph $G = (V, E)$, this means we can remove all but one of the vertex polynomials $x_i^k - 1$ using the above condition as follows: Let $P = (v_1, v_2, ..., v_s) \subseteq V$ be a path from vertex i to j in G. Then,

$$(x_j^k - 1) = (x_i^k - 1) + \sum_{r=1}^{s-1}(x_{v_r} - x_{v_{r+1}})(x_{v_r}^{k-1} + x_{v_r}^{k-2}x_{v_{r+1}} + ... + x_{v_r}x_{v_{r+1}}^{k-2} + x_{v_{r+1}}^{k-1}).$$

So, we can remove all vertex equations $x_j^k - 1$ where $j \neq i$.

To present the second idea it is best to consider the matrix associated to the linear system of the Nullstellensatz. Consider the input polynomial system $F = \{f_1, \ldots, f_s\}$. As we observed in Section 2, for a given fixed positive integer d serving as a tentative degree for the Nullstellensatz certificate, the Nullstellensatz coefficients come from the solution of a system of linear equations. We now take a closer look at the matrix equation $M_{F,d}\, y = b_{F,d}$ defining the system of linear equations. First of all, the matrix $M_{F,d}$ has one row per monomial x^α of degree less than or equal to d on the n variables and one column per polynomial of the form $x^\delta f_i$, i.e., the product of a monomial x^δ of degree less than or equal to $d - \deg(f_i)$ and a polynomial $f_i \in F$. Thus, $M_{F,d} = (M_{x^\alpha, x^\delta f_i})$ where $M_{x^\alpha, x^\delta f_i}$ equals the coefficient of the monomial x^α in the polynomial $x^\delta f_i$. The variable y has one entry for every polynomial of the form $x^\delta f_i$ denoted $y_{x^\delta f_i}$, and the vector $b_{F,d}$ has one entry for every monomial x^α of degree less than or equal to d where $(b_{F,d})_{x^\alpha} = 0$ if $\alpha \neq 0$ and $(b_{F,d})_1 = 1$.

Example 3.4. Consider the complete graph K_4. The shape of a degree-three Hilbert Nullstellensatz certificate over $\overline{\mathbb{F}}_2$ for non-3-colorability is as follows:

$$
\begin{aligned}
1 = {} & (c_0)(x_1^3 + 1) \\
& + (c_{12}^1 x_1 + c_{12}^2 x_2 + c_{12}^3 x_3 + c_{12}^4 x_4)(x_1^2 + x_1 x_2 + x_2^2) \\
& + (c_{13}^1 x_1 + c_{13}^2 x_2 + c_{13}^3 x_3 + c_{13}^4 x_4)(x_1^2 + x_1 x_3 + x_3^2) \\
& + (c_{14}^1 x_1 + c_{14}^2 x_2 + c_{14}^3 x_3 + c_{14}^4 x_4)(x_1^2 + x_1 x_4 + x_4^2) \\
& + (c_{23}^1 x_1 + c_{23}^2 x_2 + c_{23}^3 x_3 + c_{23}^4 x_4)(x_2^2 + x_2 x_3 + x_3^2) \\
& + (c_{24}^1 x_1 + c_{24}^2 x_2 + c_{24}^3 x_3 + c_{24}^4 x_4)(x_2^2 + x_2 x_4 + x_4^2) \\
& + (c_{34}^1 x_1 + c_{34}^2 x_2 + c_{34}^3 x_3 + c_{34}^4 x_4)(x_3^2 + x_3 x_4 + x_4^2)
\end{aligned}
$$

Note that we have preprocessed the certificate by removing the redundant polynomials $x_i^3 + 1$ where $i \neq 1$ and removing some variables that we know a priori can be set to zero, which results in a matrix with less columns. As we explained in Section 2, this certificate gives a linear system of equations in the variables c_0 and c_{ij}^k (note that k is a superscript and not an exponent). This linear system can be captured as the matrix equation $M_{F,1} c = b_{F,1}$ where the matrix $M_{F,1}$ is as follows.

	c_0	c_{12}^1	c_{12}^2	c_{12}^3	c_{12}^4	c_{13}^1	c_{13}^2	c_{13}^3	c_{13}^4	c_{14}^1	c_{14}^2	c_{14}^3	c_{14}^4	c_{23}^1	c_{23}^2	c_{23}^3	c_{23}^4	c_{24}^1	c_{24}^2	c_{24}^3	c_{24}^4	c_{34}^1	c_{34}^2	c_{34}^3	c_{34}^4
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x_1^3	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$x_1^2x_2$	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$x_1^2x_3$	0	0	0	1	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
$x_1^2x_4$	0	0	0	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
$x_1x_2^2$	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
$x_1x_2x_3$	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
$x_1x_2x_4$	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
$x_1x_3^2$	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0
$x_1x_3x_4$	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
$x_1x_4^2$	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0
x_2^3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
$x_2^2x_3$	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0
$x_2^2x_4$	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0
$x_2x_3^2$	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0
$x_2x_3x_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0
$x_2x_4^2$	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	1	0	0
x_3^3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
$x_3^2x_4$	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1
$x_3x_4^2$	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	1
x_4^3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1

There are often many columns in the constraint matrix of the linear system that are linear combinations of other columns, and if we could avoid creating these columns in the first place, then solving the linear system would be more efficient. Recall that each column of the matrix corresponds to the polynomial $x^\alpha f_i$ for some monomial x^α and some polynomial f_i where $\deg(x^\alpha f_i) \leq d$. The column $x^\alpha f_i$ is thus a linear combination of the other columns of the matrix if there exists $h_1, ..., h_k \in \mathbb{K}[x_1, ..., x_n]$ such that $x^\alpha f_i = \sum_j h_j f_j$ where $\deg(h_j f_j) \leq d$ and the monomial x^α does not appear in the polynomial h_i.

There is a simple way of finding columns that are linear combinations of other columns in many cases: Let cx^α be a non-zero term in f_1 where $\deg(x^\alpha) = \deg(f_1)$. Then, for every $x^\gamma f_i$ $(i > 1)$ where $x^\alpha | x^\gamma$ and $\deg(x^\gamma f_i) \leq d$, we have $x^\gamma f_i = h_1 f_1 + h_i f_i$ where $h_1 = x^\gamma - x^{\gamma-\alpha} f_i / c$ and $h_i = x^{\gamma-\alpha} f_1 / c$. Note that x^γ does not appear in h_i, $\deg(h_1 f_1) \leq \deg(x^\gamma f_i) \leq d$ and $\deg(h_i f_i) \leq \deg(x^\gamma f_i) \leq d$. Thus, $x^\gamma f_i$ corresponds to a column in the matrix that is a linear combination of other columns and can therefore be eliminated.

Hence, a general approach to avoid generating many columns of the matrix is thus as follows. Select a monomial x^α in f_1 where $\deg(x^\alpha) = \deg(f_1)$. Then, from above, we can remove all monomials from β_i $(i > 1)$ that are divisible by x^α. Repeating this, for every $i = 1, ..., k$, we can we can choose a monomial x^α in f_i where $\deg(x^\alpha) = \deg(f_i)$, and we can remove all monomials from β_j $(j > i)$ that are divisible by x^α; thus eliminating potentially many rows from the constraint matrix. Note that we must be careful to avoid circular dependencies, which is why we only eliminate monomials from β_j where $j > i$.

3.6. NulLA with symmetries

Certainly the matrix $M_{F,d}$ we presented above is rather large already for small systems of polynomials. The main point of this section is to demonstrate how to reduce the size of the matrix by using a group action on the variables, e.g., using symmetries or automorphisms in a graph. Suppose we have a finite permutation group G acting on the variables $x_1, \ldots, x_n$. Clearly G induces an action on the set of monomials with variables $x_1, x_2, \ldots, x_n$ of degree t. We will assume that the set F of polynomials is invariant under the action of G, i.e., $g(f_i) \in F$ for each

$f_i \in F$. Denote by x^δ, the monomial $x_1^{\delta_1} x_2^{\delta_2} \dots x_n^{\delta_n}$, a monomial of degree $\delta_1 + \delta_2 + \cdots + \delta_n$. Denote by $Orb(x^\alpha), Orb(x^\delta f_i)$ the orbit under G of monomial x^α and, respectively, the orbit of the polynomial obtained as the product of the monomial x^δ and the polynomial $f_i \in F$.

We now introduce a new matrix equation $\bar{M}_{F,d,G}\, \bar{y} = \bar{b}_{F,d,G}$. The rows of the matrix $\bar{M}_{F,d,G}$ are indexed by the orbits of monomials $Orb(x^\alpha)$ where x^α is a monomial of degree less than or equal to d, and the columns of $\bar{M}_{F,d,G}$ are indexed by the orbits of polynomials $Orb(x^\delta f_i)$ where $f_i \in F$ and the degree of the monomial x^δ less than or equal to $d - \deg(f_i)$. Then, let $\bar{M}_{F,d,G} = (\bar{M}_{Orb(x^\alpha),Orb(x^\delta f_i)})$ where

$$\bar{M}_{Orb(x^\alpha),Orb(x^\delta f_i)} = \sum_{x^\gamma f_j \in Orb(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j}.$$

Note that $M_{x^\alpha, x^\delta f_i} = M_{g(x^\alpha), g(x^\delta f_i)}$ for all $g \in G$ meaning that the coefficient of the monomial x^α in the polynomial $x^\delta f_i$ is the same as the coefficient of the monomial $g(x^\alpha)$ in the polynomial $g(x^\delta f_i)$. So, $\forall x^d \in Orb(x^\alpha)$,

$$\sum_{x^\gamma f_j \in Orb(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j} = \sum_{x^\gamma f_j \in Orb(x^\delta f_i)} M_{x^d, x^\gamma f_j},$$

and thus, $\bar{M}_{Orb(x^\alpha),Orb(x^\delta f_i)}$ is well-defined. We call the matrix $\bar{M}_{F,d,G}$ the *orbit matrix*. The variable $\bar{y}$ has one entry for every polynomial orbit $Orb(x^\delta f_i)$ denoted $\bar{y}_{Orb(x^\delta f_i)}$. The vector $\bar{b}_{F,d}$ has one entry for every monomial orbit $Orb(x^\alpha)$, and let $(\bar{b}_{F,d})_{Orb(x^\alpha)} = (b_{F,d})_{x^\alpha} = 0$ if $\alpha \neq 0$ and $(\bar{b}_{F,d})_{Orb(1)} = (b_{F,d})_1 = 1$. The main result in this section is that, under some assumptions, the system of linear equations $\bar{M}_{F,d,G}\, \bar{y} = \bar{b}_{F,d,G}$ has a solution if and only if the larger system of linear equations $M_{F,d}\, y = b_{F,d}$ has a solution.

Theorem 3.5. *Let $F = \{f_1, \dots, f_s\} \subset \mathbb{K}[x_1, \dots, x_n]$, be a polynomial system, and let $\mathbb{K}$ be an algebraically-closed field, and a finite group of permutations $G \subset S_n$. Let $M_{F,d}, \bar{M}_{F,d,G}$ denote the matrices defined above. Suppose that the polynomial system F is closed under the action of the group G permuting the indices of variables $x_1, \dots, x_n$. Suppose further that the order of the group $|G|$ and the characteristic of the field $\mathbb{K}$ are relatively prime. The degree d Nullstellensatz linear system of equations $M_{F,d}\, y = b_{F,d}$ has a solution over $\mathbb{K}$ if and only if the system of linear equations $\bar{M}_{F,d,G}\, \bar{y} = \bar{b}_{F,d,G}$ has a solution over $\mathbb{K}$.*

Proof. To simplify notation, let $M = M_{F,d}$, $b = b_{F,d}$, $\bar{M} = \bar{M}_{F,d,G}$ and $\bar{b} = \bar{b}_{F,d,G}$. First, we show that if the linear system $My = b$ has a solution, then there exists a *symmetric* solution y of the linear system $My = b$ meaning that $y_{x^\delta f_i}$ is the same for all $x^\delta f_i$ in the same orbit, i.e., $y_{x^\gamma f_j} = y_{x^\delta f_i}$ for all $x^\gamma f_j \in Orb(x^\delta f_i)$. The converse is also trivially true.

Since the rows and columns of the matrix M are labeled by monomials x^α and polynomials $x^\delta f_i$ respectively, we can think of the group G as acting on the matrix M, permuting the entries M, i.e., applying $g \in G$ to M gives the permuted matrix $g(M)$ where

$$g(M)_{g(x^\alpha), g(x^\delta f_i)} = M_{x^\alpha, x^\delta f_i}.$$

Moreover, since $M_{x^\alpha, x^\delta f_i} = M_{g(x^\alpha), g(x^\delta f_i)}$ for all $g \in G$, we must have $g(M) = M$, so the matrix M is invariant under the action of the group G. Also, since the entries of the variable y are labeled by polynomials of the form $x^\alpha f_i$, we can also think of the group G as acting on the vector y, permuting the entries of the vector y, i.e., applying $g \in G$ to y gives the permuted vector $g(y)$

where $g(y)_{g(x^\delta f_i)} = y_{x^\delta f_i}$. Similarly, G acts on the vector b, and in particular, $g(b) = b$. Next, we show that if $My = b$, then $Mg(y) = b$ for all $g \in G$ accordingly:

$$My = b \;\Rightarrow\; g(My) = g(b) \;\Rightarrow\; g(M)g(y) = b \;\Rightarrow\; Mg(y) = b,$$

for all $g \in G$. Now, let

$$y' = \frac{1}{|G|} \sum_{g \in G} g(y).$$

Note we need that $|G|$ is relatively prime to the characteristic of the field $\mathbb{K}$ so that $|G|$ is invertible. Then,

$$My' = \frac{1}{|G|} \sum_{g \in G} Mg(y) = \frac{1}{|G|} \sum_{g \in G} b = b,$$

so y' is a solution. Also, $y'_{x^\delta f_i} = \frac{1}{|G|} \sum_{g \in G} y_{g(x^\delta f_i)}$, so $y'_{x^\delta f_i} = y'_{x^\gamma f_j}$ for all $x^\gamma f_j \in Orb(x^\delta f_i)$. Therefore, y' is a symmetric solution as required.

Now, assume that there exists a solution of $My = b$. By the above argument, we can assume that the solution is symmetric, i.e., $y_{x^\delta f_i} = y_{x^\gamma f_j}$ where $g(x^\delta f_i) = x^\gamma f_j$ for some $g \in G$. From this symmetric solution of $My = b$, we can find a solution of $\bar{M}\bar{y} = \bar{b}$ by setting

$$\bar{y}_{Orb(x^\delta f_i)} = y_{x^\delta f_i}.$$

To show this, we check that $(\bar{M}\bar{y})_{Orb(x^\alpha)} = \bar{b}_{Orb(x^\alpha)}$ for every monomial x^α.

$$\begin{aligned}
(\bar{M}\bar{y})_{Orb(x^\alpha)} &= \sum_{\text{all } Orb(x^\delta f_i)} \bar{M}_{Orb(x^\alpha),Orb(x^\delta f_i)}\, \bar{y}_{Orb(x^\delta f_i)} \\
&= \sum_{\text{all } Orb(x^\delta f_i)} \left(\sum_{x^\gamma f_j \in Orb(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j} \right) \bar{y}_{Orb(x^\delta f_i)} \\
&= \sum_{\text{all } Orb(x^\delta f_i)} \left(\sum_{x^\gamma f_j \in Orb(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j}\, y_{x^\gamma f_j} \right) \\
&= \sum_{\text{all } x^\delta f_i} M_{x^\alpha, x^\delta f_i}\, y_{x^\delta f_i} \;=\; (My)_{x^\alpha}.
\end{aligned}$$

Thus, $(\bar{M}\bar{y})_{Orb(x^\alpha)} = \bar{b}_{Orb(x^\alpha)}$ since $(My)_{x^\alpha} = b_{x^\alpha} = \bar{b}_{Orb(x^\alpha)}$.

Next, we establish the converse more easily. Recall that the columns of $\bar{M}$ are labeled by orbits. If there is a solution for $\bar{M}\bar{y} = \bar{b}$, then to recover a solution of $My = b$, we set

$$y_{x^\delta f_i} = \bar{y}_{Orb(x^\delta f_i)}.$$

Note that y is a symmetric solution. Using the same calculation as above, we have that $(My)_{x^\alpha} = (\bar{M}\bar{y})_{Orb(x^\alpha)}$, and thus, $My = b$. □

Example 3.6 (Continuation of Example 3.4)**.** Now consider the action of the symmetry group G generated by the cycle (2,3,4) (a cyclic group of order three). The permutation of variables permutes the monomials and yields a matrix $M_{F,1,G}$. We have now grouped together monomials and terms within orbit blocks in the matrix below. The blocks will be later replaced by a single entry, shrinking the size of the matrix.

	c_0	c^1_{12}	c^1_{13}	c^1_{14}	c^2_{12}	c^3_{13}	c^4_{14}	c^3_{12}	c^4_{13}	c^2_{14}	c^4_{12}	c^2_{13}	c^3_{14}	c^1_{23}	c^1_{34}	c^1_{24}	c^2_{23}	c^3_{34}	c^4_{24}	c^2_{24}	c^3_{23}	c^4_{34}	c^2_{34}	c^3_{24}	c^4_{23}
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x_1^3	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$x_1^2x_2$	0	1	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
$x_1^2x_3$	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
$x_1^2x_4$	0	0	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$x_1x_2^2$	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
$x_1x_3^2$	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
$x_1x_4^2$	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
$x_1x_2x_3$	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
$x_1x_2x_4$	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
$x_1x_3x_4$	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
x_2^3	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
x_3^3	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
x_4^3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
$x_2^2x_3$	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0
$x_3^2x_4$	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1
$x_2x_4^2$	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0
$x_2^2x_4$	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	1
$x_2x_3^2$	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	1	0	0
$x_3x_4^2$	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	1	0
$x_2x_3x_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

The action of the symmetry group generated by the cycle (2,3,4) yields an orbit matrix $\bar{M}_{F,q,G}$ of about a third the size of the original one:

	$\bar{c}_0$	$\bar{c}^1_{12}$	$\bar{c}^2_{12}$	$\bar{c}^3_{12}$	$\bar{c}^4_{12}$	$\bar{c}^1_{23}$	$\bar{c}^2_{23}$	$\bar{c}^2_{24}$	$\bar{c}^2_{34}$
$Orb(1)$	1	0	0	0	0	0	0	0	0
$Orb(x_1^3)$	1	3	0	0	0	0	0	0	0
$Orb(x_1^2x_2)$	0	1	1	1	1	0	0	0	0
$Orb(x_1x_2^2)$	0	1	1	0	0	2	0	0	0
$Orb(x_1x_2x_3)$	0	0	0	1	1	1	0	0	0
$Orb(x_2^3)$	0	0	1	0	0	0	1	1	0
$Orb(x_2^2x_3)$	0	0	0	1	0	0	1	1	1
$Orb(x_2^2x_4)$	0	0	0	0	1	0	1	1	1
$Orb(x_2x_3x_4)$	0	0	0	0	0	0	0	0	3

$$\overset{(\mathrm{mod}\ 2)}{\equiv}$$

	$\bar{c}_0$	$\bar{c}^1_{12}$	$\bar{c}^2_{12}$	$\bar{c}^3_{12}$	$\bar{c}^4_{12}$	$\bar{c}^1_{23}$	$\bar{c}^2_{23}$	$\bar{c}^2_{24}$	$\bar{c}^2_{34}$
$Orb(1)$	1	0	0	0	0	0	0	0	0
$Orb(x_1^3)$	1	1	0	0	0	0	0	0	0
$Orb(x_1^2x_2)$	0	1	1	1	1	0	0	0	0
$Orb(x_1x_2^2)$	0	1	1	0	0	0	0	0	0
$Orb(x_1x_2x_3)$	0	0	0	1	1	1	0	0	0
$Orb(x_2^3)$	0	0	1	0	0	0	1	1	0
$Orb(x_2^2x_3)$	0	0	0	1	0	0	1	1	1
$Orb(x_2^2x_4)$	0	0	0	0	1	0	1	1	1
$Orb(x_2x_3x_4)$	0	0	0	0	0	0	0	0	1

If $|G|$ is not relatively prime to the characteristic of the field $\mathbb{K}$, then it is still true that, if $\bar{M}y = \bar{b}$ has a solution, then $My = b$ has a solution. Thus, even if $|G|$ is not relatively prime to

the characteristic of the field $\mathbb{K}$, we can still prove that the polynomial system F is infeasible by finding a solution of the linear system $\bar{M}y = \bar{b}$.

4. Experimental results

In this section, we present our experimental results, including a comparison between **NulLA** and other graph coloring algorithms such as DSATUR, *Branch-and-Cut* [24], and the Alon-Tarsi [1] and Gröbner basis methods. Given a certificate $1 = \sum \beta_i f_i$ for graph non-3-colorability, the degree of the f_i input polynomials is constant over all input graphs. Thus, the degree affecting **NulLA** computation time is the *coefficient degree*, defined to be $\max\{\deg(\beta_i)\}$. In this way, almost all of the graphs tested by **NulLA** had *degree one or less* coefficients in their certificates. This algebraic property, coupled with our ability to compute over $\mathbb{F}_2$, allowed us to prove the non-3-colorability of graphs with almost two thousand nodes.

4.1. Methods

Our computations were performed on machines with dual Opteron nodes, 2 GHz clock speed, and 12 GB of RAM. No branching, degree-cutter equations or alternative Nullstellensatz certificates were used unless explicitly specified. We also eliminated redundant equations, and monomials whose coefficients could be set to zero.

4.2. Test cases

We tested the following graphs:

1. **DIMACS:** The graphs from the DIMACS Computational Challenge (1993, 2002) are described in detail at `http://mat.gsia.cmu.edu/COLORING02/`. This set of graphs is the standard benchmark for graph coloring algorithms. We tested every DIMACS graph whose associated **NulLA** matrix could be instantiated within 12 GB of RAM. For example, we did *not* test `C4000.5.clq`, which has 4,000 vertices and 4,000,268 edges, yielding a degree one **NulLA** matrix of 758 million non-zero entries and 1 trillion columns.
2. **Mycielski:** The Mycielski graphs are known for the gap between their clique and chromatic number. The Mycielski graph of order k is a triangle-free graph with chromatic number k. The first few instances and the algorithm for their construction can be seen at `http://mathworld.wolfram.com/MycielskiGraph.html`.
3. **Kneser:** The nodes of the Kneser-(t, r) graph are represented by the r-subsets of $\{1, \ldots, t\}$. Two nodes are adjacent if and only if their subsets are disjoint.
4. **Random:** We tested random graphs in 16 nodes with an edge probability of .27. This probability was experimentally selected based on the boundary between 3-colorable and non-3-colorable graphs and is explained in detail in Section 4.3.
5. **Hard Instances:** We also tested purported *hard instances of 3-colorability*. The algorithms behind the generation of these graphs, and the associated experimental results are described in detail in Section 4.5.

4.3. Results

In this section, we present our experimental results on graphs with and without 4-cliques. We also point out certain properties of **N**ul**LA**-constructed certificates, and conclude with tests on random graphs. Surprisingly, all but four of the DIMACS, Mycielski and Kneser graphs tested with **N**ul**LA** have degree three certificates, which implies that the β coefficients present in the certificates have *degree one or less*.

The DIMACS graphs are primarily benchmarks for graph k-colorability, and thus contain many graphs with large chromatic number. Such graphs often contain 4-cliques. Although testing for graph 3-colorability is well-known to be NP-Complete, there exist many efficient (and even trivial), polynomial-time algorithms for finding 4-cliques in a graph. Thus, we break our computational investigations into two tables: Table 1 contains graphs *without* 4-cliques, and Table 3 contains graphs *with* 4-cliques (considered "easy" instances of 3-colorability). For space considerations, we only display representative results for graphs of varying size for each family. The

Graph	*vertices*	*edges*	*rows*	*cols*	$\frac{coeff}{deg}$	*sec*
m7 (Mycielski 7)	95	755	64,281	71,726	1	.46
m9 (Mycielski 9)	383	7,271	2,477,931	2,784,794	1	268.78
m10 (Mycielski 10)	767	22,196	15,270,943	17,024,333	1	14835
$(8,3)$-Kneser	56	280	15,737	15,681	1	.07
$(10,4)$-Kneser	210	1,575	349,651	330,751	1	3.92
$(12,5)$-Kneser	792	8,316	7,030,585	6,586,273	1	466.47
$(13,5)$-Kneser	1,287	36,036	45,980,650	46,378,333	1	216105
ash331GPIA.col	662	4,185	3,147,007	2,770,471	1	13.71
ash608GPIA.col	1,216	7,844	10,904,642	9,538,305	1	34.65
ash958GPIA.col	1,916	12,506	27,450,965	23,961,497	1	90.41
1-Insertions_5.col	202	1,227	268,049	247,855	1	1.69
2-Insertions_5.col	597	3,936	2,628,805	2,349,793	1	18.23
3-Insertions_5.col	1,406	9,695	15,392,209	13,631,171	1	83.45

Table 1: Graphs without 4-cliques.

size of the linear systems involved ranged from $15,737 \times 15,681$ up to $45,980,650 \times 46,378,333$ (for the $(8,3)$-Kneser and $(13,5)$-Kneser graphs, respectively).

However, not all of the DIMACS challenge graphs had degree one coefficient certificates. We were unable to produce certificates for `mug88_1`, `mug88_25`, `mug100_1` or `mug100_25`, even when using degree-cutters and searching for alternative Nullstellensatz certificates. When testing for a degree six certificate, the smallest of these graphs (`mug88_1` with 88 vertices and 146 edges) yielded a linear system with 1,170,902,966 non-zero entries and 390,340,149 columns. A matrix of this size is not computationally tractable at this time because it cannot be instantiated within available memory. Branching was also not successful on these graphs. The runs were terminated after solving over 5 million subproblems. Section 4.5 investigates graphs from this family in greater detail.

Recall that the certificates returned by **N**ul**LA** consist of a single vertex polynomial (via preprocessing), and edge polynomials describing either the original graph in its entirety, or a non-3-colorable subgraph from the original graph. For example, if the graph contains a 4-clique as a subgraph, often the Nullstellensatz certificate will only display the edges contained in the

4-clique. In this case, we say that **NulLA** *isolates* a non-3-colorable subgraph from the original graph. The size difference between these subgraphs and the input graphs is often dramatic, as shown in Table 2.

Graph	*vertices*	*edges*	*subgraph vertices*	*subgraph edges*
miles1500.col	128	10,396	6	10
hamming8-4.clq	256	20,864	19	33
m10 (Mycielski 10)	767	22,196	11	20
$(12, 5)$-Kneser	792	8,316	53	102
dsjc1000.1.col	1,000	49,629	15	24
ash608GPIA.col	1,216	7,844	23	44
3-Insertions_5.col	1,406	9,695	56	110
ash958GPIA.col	1,916	12,506	24	45

Table 2: Original graph vs. non-3-colorable subgraph.

An overall analysis of these computational experiments shows that **NulLA** performs best on sparse graphs. For example, the `3-Insertions_5` graph (with 1,406 nodes and 9,695 edges) runs in 83 seconds, while the `3-FullIns_5` graph (with 2,030 nodes and 33,751 edges) runs in 15027 seconds. Another example is `p_hat700-2` (with 700 nodes and 121,728 edges) and `will199GPIA` (with 701 nodes and 7,065 edges). **NulLA** proved the non-3-colorability of `will199GPIA` in 35 seconds, while `p_hat700-2` took 30115 seconds.

Finally, as an informal measure of the distribution of degree three certificates (certificates with β coefficients of degree one or less), we generated random graphs of 16 nodes with edge probability .27. We selected this probability because it lies on the boundary between feasible and infeasible instances. In other words, graphs with edge probability less than .27 were almost always 3-colorable, and graphs with edge probability greater than .27 were almost always non-3-colorable. However, we experimentally found that an edge probability of .27 created a distribution that was almost exactly half and half. Of 100 trials, 48 were infeasible. Of those 48 graphs, 40 had degree three certificates and 8 had degree six certificates. Of these remaining 8 instances, we were able to find degree three certificates for all 8 by appending degree-cutters or by finding alternative Nullstellensatz certificates. This tentative measure indicates that non-3-colorability certificates of degrees greater than three may be rare.

4.4. NulLA vs. other algorithms

In this section, we compare **NulLA** to two other algebraic methods for detecting 3-colorability: the Alon-Tarsi (AT) method, and the Gröbner basis (GB) method. We also briefly comment on **NulLA**'s relation to well-known graph coloring heuristics such as DSATUR and Branch-and-Cut [24]. We implemented the Alon-Tarsi method in C++, and used CoCoA Lib [6] to test the Gröbner basis method. For brevity, we do not record any "internal data" about the various algorithmic runs, such as the size of the underlying linear systems solved by **NulLA** or the maximum number of monomials in the normal forms produced by the Alon-Tarsi method. In the tables below, all certificates have degree three (β coefficients of degree one or less) and a "–" signifies that the method was terminated after 4 hours of computation.

The Gröbner basis method refers to simply taking the Gröbner basis of the ideal defined in Lemma 3.1. By Hilbert's Nullstellensatz, the Gröbner basis is a constant if and only if the graph is non-3-colorable.

Graph	*vertices*	*edges*	*rows*	*cols*	$\overset{coeff}{deg}$	*sec*
miles500.col	128	2,340	143,640	299,521	1	1.35
miles1000.col	128	6,432	284,042	823,297	1	7.52
miles1500.col	128	10,396	349,806	1,330,689	1	24.23
mulsol.i.5.col	197	3,925	606,959	773,226	1	6
zeroin.i.1.col	211	4,100	643,114	865,101	1	6
queen16_16.col	256	12,640	1,397,473	3,235,841	1	106
hamming8-4.clq	256	20,864	2,657,025	5,341,185	1	621.1
school1_nsh.col	352	14,612	4,051,202	5,143,425	1	210.74
MANN_a27.clq	378	70,551	9,073,144	26,668,279	1	9809.22
brock400_4.clq	400	59,765	10,579,085	23,906,001	1	4548.59
gen400_p0.9_65.clq	400	71,820	10,735,248	28,728,001	1	9608.85
le450_5d.col	450	9,757	4,168,276	4,390,651	1	304.84
fpsol2.i.1.col	496	11,654	4,640,279	57,803,85	1	93.8
C500.9.clq	500	112,332	20,938,304	56,166,001	1	72752
homer.col	561	3,258	1,189,065	1,827,739	1	8
p_hat700-2.clq	700	121,728	48,301,632	85,209,601	1	30115
will199GPIA.col	701	7,065	5,093,201	4,952,566	1	35
inithx.i.1.col	864	18,707	13,834,511	16,162,849	1	1021.76
qg.order30.col	900	26,100	23,003,701	23,490,001	1	13043
wap06a.col	947	43,571	37,703,503	41,261,738	1	1428
dsjc1000.1.col	1,000	49,629	45,771,027	49,629,001	1	2981.91
5-FullIns_4.col	1,085	11,395	13,149,910	12,363,576	1	200.09
3-FullIns_5.col	2,030	33,751	70,680,086	68,514,531	1	15027.9

Table 3: Graphs with 4-cliques.

The Alon-Tarsi method is based on the following (see Section 7 of [1] and references therein):

Theorem 4.1. *Given a graph G with n vertices, let* $I_G = \langle x_1^3 - 1, \ldots, x_n^3 - 1 \rangle$. *Additionally, let*

$$P_G = \prod_{(i,j) \in E(G)} (x_i - x_j)$$

Then $P_G \in I_G$ *if and only if G is non-3-colorable*

In order to compute with the Alon-Tarsi method, we note that the set $B = \{x_1^3 - 1, \ldots, x_n^3 - 1\}$ is a Gröbner basis for I_G. Thus, we simply take the normal form of P_G with respect to B. If the normal form is zero, $P_G \in I_G$, and the graph is non-3-colorable. The efficiency of the Alon-Tarsi method can be increased by incrementally constructing P_G [12]: we order the edges, and then find the normal form of $(x_{i_1} - x_{j_1})$ with respect to B, and then the normal form of $(x_{i_1} - x_{j_1})(x_{i_2} - x_{j_2})$ with respect to B, etc.

We compared **NulLA** to the Gröbner basis and Alon-Tarsi methods on graphs with and without 4-cliques; results are displayed in Tables 6 and 7, respectively. **NulLA** consistently outperformed the Gröbner basis method. For example, on `zeroin.i.1`, **NulLA** ran in 6 seconds, while CoCoA Lib took almost one hour. These experimental results indicate that **NulLA** scales better than the Gröbner basis method.

NulLA also compared extremely favorably with the Alon-Tarsi method, which usually did not terminate within the requisite time bounds. However, in the special case when the first few

vertices and edges of the graph happen to describe a non-3-colorable subgraph (such as a 4-clique, or the Grötzch graph), the Alon-Tarsi method ran very quickly, because of the iterative approach incorporated during implementation. Consider the example of the ninth Mycielski graph (383 vertices and 7,271 edges): the Alon-Tarsi method terminated in .24 seconds, but after we permuted the vertices and edges, the method consumed 9 GB of RAM over 4 hours of computation and only processed 30 edges. This example shows that the Alon-Tarsi method is extremely sensitive to the vertex and edge ordering. If a similar iterative approach was incorporated either into **N**ul**LA** or the Gröbner basis method, these algorithms would likewise terminate early in this special case.

As another example of the draw-backs of the Alon-Tarsi method, we considered edge-critical graphs, where the entire input must be read. For example, the odd wheels form a trivial family of edge-critical non-3-color-able graphs. The Alon-Tarsi method was unable to determine the non-3-colorability of the 17-odd-wheel (18 vertices and 34 edges): after two hours of computation, the normal form contained over 19 million monomials, and had consumed over 8 GB of RAM. The experimental results are displayed in Table 4.

odd-wheels	*vertices*	*edges*	**N**ul**LA**	*GB*	*AT*
9	10	18	0	0	.05
11	12	22	0	0	.74
13	14	26	0	0	8.47
15	16	30	0	0	369.45
17	18	34	0	0	–
151	152	302	.21	2.21	–
501	502	1,002	15.58	126.83	–
1001	1,002	2,002	622.73	1706.69	–
2001	2,002	4,002	12905.6	–	–

Table 4: **N**ul**LA**, GB and AT on odd-wheel graphs.

We conclude with a short comment about **N**ul**LA**'s relation to DSATUR and Branch-and-Cut [24]. These heuristics return bounds on the chromatic number. In Table 5 (data taken from [24]), we display the bounds returned by Branch-and-Cut (B&C) and DSATUR, respectively. In the case of these graphs, **N**ul**LA** determined non-3-colorability very rapidly (establishing a lower bound of four), while the two heuristics returned lower bounds of three and two, respectively. Thus, **N**ul**LA** returned a tighter lower bound on the chromatic number than B&C or DSATUR. We note that this example does not constitute a rigorous comparison between **N**ul**LA** and B&C or DSATUR.

			B&C		DSATUR		**N**ul**LA**
Graph	*vertices*	*edges*	*lb*	*up*	*lb*	*up*	*sec*
4-Insertions_3	79	156	3	4	2	4	0
3-Insertions_4	281	1,046	3	5	2	5	1
4-Insertions_4	475	1,795	3	5	2	5	3
2-Insertions_5	597	3,936	3	6	2	6	12
3-Insertions_5	1,406	9,695	3	6	2	6	83

Table 5: **N**ul**LA** vs. Branch-and-Cut and DSATUR.

Graph	*vertices*	*edges*	**NulLA**	*GB*	*AT*
miles500	128	2,340	1.35	133.91	.07
miles1000	128	6,432	7.52	802.23	0
miles1500	128	10,396	24.23	2598.84	.01
mulsol.i.5	197	3,925	6	18804.5	0
zeroin.i.1	211	4,100	6	2753.37	0
queen16_16	256	12,640	106	59466.9	0
hamming8-4	256	20,864	621.1	–	–
le450_5d	450	9,757	304.84	–	–
homer	561	3,258	8	–	–
dsjc1000.1	1,000	49,629	2981.91	–	–
5-FullIns_4	1,085	11,395	200.09	–	557.12
3-FullIns_5	2,030	33,751	15027.9	–	3.97

Table 6: **NulLA**, GB, AT on graphs with 4-cliques.

Graph	*vertices*	*edges*	**NulLA**	*GB*	*AT*
Mycielski 4	11	20	0	.01	.22
Mycielski 5	23	71	0	.08	.23
Mycielski 6	47	236	.04	3.99	.22
Mycielski 7	95	755	.46	179.94	.23
Mycielski 8	191	2,360	7.72	9015.06	.23
Mycielski 9	383	7,271	268.78	–	.22
Mycielski 9 permuted	383	7,271	497.47	–	–
$(6,2)$-Kneser	15	45	0	.03	1.87
$(8,3)$-Kneser	56	280	.07	18.39	–
$(10,4)$-Kneser	210	1,575	3.92	9771.76	–
$(12,5)$-Kneser	792	8,316	466.47	–	–
ash331GPIA	662	4,185	13.71	–	–
1-Insertions_4	67	232	.04	3.71	–
2-Insertions_4	149	541	.26	32.42	–
1-Insertions_5	202	1,227	1.69	940.7	–
3-Insertions_4	281	1,046	.97	237.69	–
4-Insertions_4	475	1,795	3.02	1596.35	–
2-Insertions_5	597	3,936	18.23	–	–

Table 7: **NulLA**, GB, AT on graphs without 4-cliques.

4.5. Hard Instances of 3-colorability

The question of whether "hard" instances of graph 3-colorability have specific, identifiable, and systematically reproducible properties is an area of active research. Examples of graph-theoretic properties proposed as *order parameters* separating "easy" instances from "hard" include 3-paths [27], minimal unsolvable subproblems [22] and frozen developments [8]. Some of these proposed order parameters have resulted in algorithms [27] [25] [20] for generating infinite families of non-3-colorable graphs conjectured (and computationally verified) to be "hard". In this section, we investigate a link between Nullstellensatz certificate coefficient degree and "hard" non-3-colorable graphs.

We begin by describing the algorithms for generating "hard" instances that we tested, which were the minimum unsolvable graphs (MUGs) from [25], and the 4-critical graph units (4-CGUs) from [20]. We conclude by displaying our experimental results, and comparing **N**ul**LA** with the Gröbner basis method on these instances.

4.5.1. Minimal Unsolvable (non-3-colorable) Subgraphs (MUGs)

In [25], a randomized algorithm for generating infinitely large instances of quasi-regular, 4-critical graphs is described. These quasi-regular, 4-critical graphs are referred to by the authors as *minimal unsolvable subgraphs*, where the term "unsolvable" refers to the non-3-colorability of the graph. In this case, *quasi-regular* refers to graphs containing only vertices of degree three or four, and *4-critical* refers to graphs with chromatic number four such that the removal of any edge decreases the chromatic number from four to three. The MUG generation algorithm relies on five core 4-critical, quasi-regular minimal unsolvable graphs (displayed in Figure 4), which are randomly chosen and then iteratively constructed using the Hajós calculus, creating larger and larger 4-critical graphs. The Hajós calculus is a particular construction used to generate the entire class of non-3-colorable graphs (see [14] and references therein).

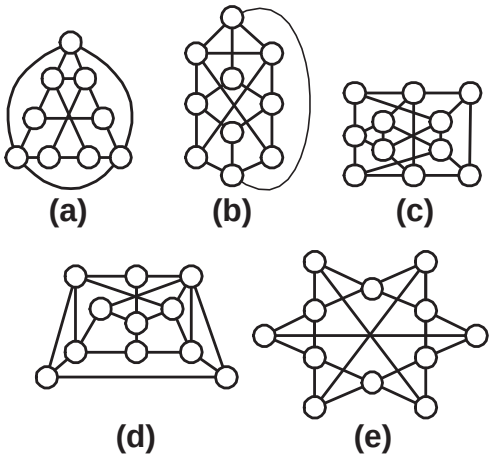

Figure 4: 4-critical, near-4-clique-free minimum unsolvable graphs (MUGs).

*4.5.2. 4-critical graph units (*4*-CGUs)*

In [20], a randomized algorithm for generating infinitely large instances of triangle-free, 4-critical graphs is described. The 4-CGU algorithm constructs a particular 4-critical core, which is than joined to the previous graph in the sequence using the Hajós calculus. An example of a 4-CGU is displayed in Figure 5, and the algorithm for generating a sequence of 4-CGUs follows below.

4.5.3. Experimental Results on Hard Instances of 3-colorability

We implemented both the MUG hard instance generation algorithm, and the 4-CGU hard instance generation algorithm. We tested both families with **N**ul**LA**, and also with the Gröbner basis method using CoCoA Lib. In [25], the MUG instances were tested with the Smallk [9] and

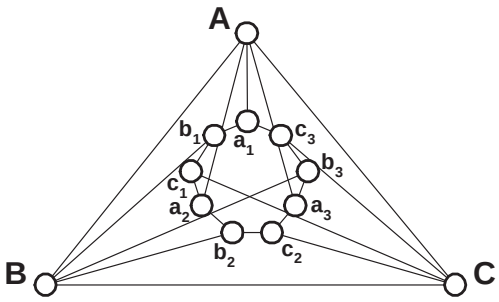

Figure 5: An example of a Liu-Zhang 4-CGU.

Brélaz heuristics [3], as well as with six major constraint satisfaction problem (CSP) solvers. In each case, exponential runtime growth was reported by the authors.

When we tested the MUG random instances using **N**ul**LA**, we immediately saw corresponding growth in the degree of the Nullstellensatz. We were only able to compute the degrees of the first few certificates in the sequence; thus, it is impossible to infer a precise rate of growth for the MUG family. Furthermore, the use of triangle equations as degree-cutters did not reduce the degree, and we were also unable to find alternative Nullstellensatz certificates of lower degree for these graphs. However, **N**ul**LA** with branching proved extremely successful. For example, on MUG G_4, **N**ul**LA** without branching took 773.16 seconds, while **N**ul**LA** with branching only took 53.48 seconds to solve 6,131 subproblems. Furthermore, **N**ul**LA** with branching compared favorably to the Gröbner bases method using CoCoA Lib: for example, MUG G_7 took 7058.14 seconds using **N**ul**LA** with branching, but took 19837.4 with CoCoA Lib. We report on these results in Table 8.

			Nul**LA** without branching				**N**ul**LA** with branching		GB
Graph	*n*	*m*	*rows*	*cols*	*coeff deg*	*sec*	*# of subprobs*	*sec*	*sec*
MUG G_0	10	18	198	181	1	0	1	0	0
MUG G_1	20	37	178,012	329,916	4	6.33	9	.01	.05
MUG G_2	30	55	1,571,328	2,257,211	4	52.83	83	.31	.46
MUG G_3	39	72	6,481,224	8,072,429	4	201.96	479	2.86	5.5
MUG G_4	49	90	22,054,196	24,390,486	≥ 7	773.16	6,131	53.48	150.47
MUG G_5	60	110	–	–	–	–	67,163	946.66	1718.62
MUG G_6	69	127	–	–	–	–	103,787	2031.98	3806.17
MUG G_7	78	144	–	–	–	–	297,371	7058.14	19837.4

Table 8: Hard instances of graph 3-colorability: MUGs.

In Table 9, we report the results of the **N**ul**LA** experiments on the 4-CGU hard instances of graph 3-colorability. The 4-CGU instance generation algorithm has not been tested as thoroughly with multiple graph coloring algorithms as compared to the MUGs in [25]. However, the 4-CGUs were tested with Smallk, and exponential running times were reported in [20]. When we tested the 4-CGU algorithm with **N**ul**LA**, we immediately found corresponding growth in the degree of the Nullstellensatz certificates, at a rate of growth very similar to the rate of growth in the MUG family. We also note that the 4-CGUs are triangle-free. Thus, no reductions in degree via triangle

degree-cutter equations are possible. Furthermore, as in the case of the MUGs, we could not find alternative Nullstellensatz certificates for the 4-CGUs. However, branching again proved very successful on these graphs. Finally, we note that the running times returned by CoCoA Lib in the Gröbner basis experiments were very different between the MUG and 4-CGU families: for example, CoCoA Lib found a Gröbner basis for the 4-CGU G_7 (74 vertices and 139 edges) in 75.02 seconds, as compared with 19837.4 seconds for the MUG G_7 (78 vertices and 144 edges).

			NulLA without branching				**NulLA** with branching		GB
Graph	*n*	*m*	*rows*	*cols*	$\overset{coeff}{deg}$	*sec*	*# of subprobs*	*sec*	*sec*
4-CGU G_0	11	20	247	221	1	0	1	0	0
4-CGU G_1	20	37	177,760	329,916	4	7.35	9	.02	.1
4-CGU G_2	29	54	1,306,695	1,947,902	4	82.77	329	1.18	.75
4-CGU G_3	38	71	5,621,140	7,202,749	4	364.23	3,161	18.6	1.65
4-CGU G_4	47	88	17,629,974	20,288,961	≥ 7	688.35	21,161	183.01	10.46
4-CGU G_5	56	105	–	–	–	–	92,633	1167.01	13.41
4-CGU G_6	65	122	–	–	–	–	92,641	1679.15	20.82
4-CGU G_7	74	139	–	–	–	–	3,938,023	84326.9	75.02
4-CGU G_8	83	156	–	–	–	–	$> 5{,}148{,}710$	–	570.96

Table 9: Hard instances of graph 3-colorability: 4-CGUs.

The underlying cause in the degree growth of graph 3-colorability certificates remains an open question. It is interesting to note that of the hundreds of graphs present in the DIMACS computational challenge, the only graphs with degrees greater than three were the MUG graphs, specifically proposed as "hard" instances of graph 3-colorability.

5. Conclusion

We presented a general algebraic method to prove combinatorial infeasibility. We showed that even though the worst-case known Nullstellensatz degree upper bounds are doubly exponential, in practice for useful combinatorial systems, they are often much smaller and can be used to solve even large problem instances. Our experimental results illustrated that many benchmark non-3-colorable graphs have degree three certificates (β coefficients of degree one or less); indeed, non-3-colorable graphs with coefficient certificate degrees larger than three appear to be rare. We also showed that **NulLA** compares well with other algebraic methods and popular heuristics for colorability.

References

[1] N. Alon, Combinatorial Nullstellensatz, *Combinatorics, Probability and Computing*, 8, (1999), 7–29.

[2] D.A. Bayer, "The Division Algorithm and the Hilbert Scheme," Ph.D. Thesis, Harvard University, (1982).

[3] D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22:251–256, 1979.

[4] W.D. Brownawell. Bounds for the degrees in the Nullstellensatz. *Annals of Mathematics*, 126(3):577–591, 1987.

[5] S. Buss and T. Pitassi, Good degree bounds on nullstellensatz refutations of the induction principle, *IEEE Conference on Computational Complexity*, (1996), 233–242.

[6] CoCoATeam, CoCoA: a system for doing Computations in Commutative Algebra, available at `http://cocoa.dima.unige.it`.

[7] D. Cox, J. Little and D. O'Shea, "Ideals,Varieties and Algorithms," Springer Undergraduate Texts in Mathematics, Springer-Verlag, New York, (1992).
[8] J. Culberson and I. Gent. Frozen development in graph coloring. *Theoretical Computer Science*, 265:227–265, 2001.
[9] J. Culberson. Overview of the smallk graph coloring program. `http://www.cs.ualberta.ca/\~joe/Coloring/Colorscr/smallk.html788`.
[10] J.A. De Loera, J. Lee, S. Margulies, S. Onn, Expressing Combinatorial Optimization Problems by Systems of Polynomial Equations and the Nullstellensatz, to appear in the Journal of Combinatorics, Probability and Computing, `http://arxiv.org/abs/0706.0578`
[11] J.A. De Loera, J. Lee, P.N. Malkin, and S. Margulies. Hilbert's Nullstellensatz and an algorithm for proving combinatorial infeasibility, Proceedings of the Twenty-first International Symposium on Symbolic and Algebraic Computation (ISSAC 2008), `http://arxiv.org/abs/0801.3788`.
[12] C.J. Hillar and T. Windfeldt, "An algebraic characterization of uniquely vertex colorable graphs," Journal of Combinatorial Theory Series B, 98 (2008), 400–414.
[13] P. Impagliazzo, P. Pudlák and J. Sgall, Lower bounds for polynomial calculus and the Groebner basis algorithm, *Computational Complexity*, 8, (1999), 127–144.
[14] K. Iwama and T. Pitassi. Exponential lower bounds for the tree-like Hajós calculus. *Information Processing Letters*, 54:289–294, 1995.
[15] E. Koester, On 4-critical planar graphs with high edge density, *Discrete Mathematics*, 98, (1991), 147–151.
[16] J. Kollár, Sharp effective Nullstellensatz, *Journal of the AMS*, 1(4), (1988), 963–975.
[17] J.B. Lasserre, Polynomials nonnegative on a grid and discrete optimization, *Transactions of the AMS*, 354(2), (2001), 631–649.
[18] M. Laurent, Semidefinite representations for finite varieties, *Mathematical Programming*, 109, (2007), 1–26.
[19] D. Lazard. Algèbre linéaire sur $\mathbb{K}[x_1, \ldots, x_n]$ et élimination. *Bulletin de las S.M.F*, 105:165–190, 1977.
[20] S. Liu and J. Zhang. Using hajós construction to generate hard graph 3-colorability instances. *Lecture Notes in Computer Science*, 4120:211–225, 2006.
[21] L. Lovász, Stable sets and polynomials, *Discrete Mathematics*, 124, (1994), 137–153.
[22] D. Mammen and T. Hogg. A new look at easy-hard-easy pattern of cominatorial search difficulty. *Journal of Artificial Intelligence Research*, 7:47–66, 1997.
[23] S. Margulies, "Computer Algebra, Combinatorics, and Complexity: Hilbert's Nullstellensatz and NP-Complete Problems," UC Davis Ph.D. dissertation, in preparation, (2008).
[24] I. Méndez-Díaz and P. Zabala, "A branch-and-cut algorithm for graph coloring," *Discrete Applied Mathematics*, Vol. 154(5), (2006), 826–847.
[25] K. Mizuno and S. Nishihara. Constructive generation of very hard 3-colorability instances. *Discrete Applied Mathematics*, 156(2):218–229, 2008.
[26] P. Parrilo, Semidefinite programming relaxations for semialgebraic problems," *Mathematical Programming, Series B*, 96(2), (2003), 293–320.
[27] R. Vlasie. Systematic generation of very hard cases for graph 3-colorability. In *Tools with Artificial Intelligence*, pages 114–119. Seventh International Conference on Artificial Intelligence, 1995.