2020

Domination problems in directed graphs and inducibility of nets

Adam Blumenthal
Iowa State University

Domination problems in directed graphs and inducibility of nets

by

Adam Blumenthal

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Mathematics

Program of Study Committee:
Bernard Lidický, Co-major Professor
Michael Young, Co-major Professor
Steve Butler
Jonas Hartwig
Jack Lutz

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2020

DEDICATION

I would like to dedicate this thesis to my fiancée Heather, my parents Rik and Cathy, and my sister Erin. Without all of their support, this thesis would not have been possible.

I dedicate this thesis to all of the academic role models in my life, both listed and unlisted. Thank you all for instilling in me not only a passion for learning, but also countless examples of how to positively impact the lives of my students.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to take this opportunity to express my thanks to those who helped me develop as a researcher and an educator. Dr. Bernard Lidický and Dr. Michael Young for their guidance, patience and support throughout this research and the writing of this thesis. Thanks also to Michael Phillips, who kept me honest and was an incredible collaborator throughout our work together. I would like to thank each of my committee members, Dr. Jonas Hartwig, Dr. Steve Butler, and Dr. Jack Lutz for their support through my graduate studies.

Thanks to Dr. Timothy Hayes. Through his teaching in my English Composition course, I learned how critical and fulfilling it is to support students with interests that differ from my own expertise. Without his support, I may not have continued my academic pursuits. Thanks to Dr. Ulrich Albrecht, who taught me what positive leadership with undergraduates in academia looked like and encouraged me to expand my teaching horizons from an early age. Thanks to Dr. Peter D. Johnson, who demonstrated to me how fun mathematical research can be. Thanks to Dr. Kat Perry, who taught me that in academia, a large and diverse support network is not only academically necessary, but also enriches our lives.

ABSTRACT

In this thesis we discuss two topics: domination parameters and inducibility. In the first chapter, we introduce basic concepts, definitions, and a brief history for both types of problems. We will first inspect domination parameters in graphs, particularly independent domination in regular graphs and we answer a question of Goddard and Henning [23]. Additionally, we provide some constructions for regular graphs of small degree to provide lower bounds on the independent domination ratio of these classes of graphs. In Chapter 3 we expand our exploration of independent domination into the realm of directed graphs. We will prove several results including providing a fastest known algorithm for determining existence of an independent dominating set in directed graphs with minimum in-degree at least one and period not eqeual to one. We also construct a set of counterexamples to the analogue of Vizing's Conjecture for this setting. In the fourth chapter, we pivot from independent domination to split domination in directed graphs, where we introduce the split domination sequence. We will determine that almost all possible split domination sequences are realizable by some graphs, and state several open questions that would be of interest to continue in this field. In the fifth chapter we will provide a brief introduction to flag algebras, then determine the unique maximizer of induced net graphs in graphs of order 6^k for each k.

CHAPTER 1. DEFINITIONS AND INTRODUCTION

This thesis will focus largely on two subjects. First, we will explore several domination parameters of graphs, then we will provide an extremal graph theory result through the use of flag algebras. This chapter will provide basic definitions and some samples of techniques used to show results used later in the thesis. We will begin with fundamental graph theory definitions. A *graph* is an ordered pair (V, E), where V is a set of objects called vertices, and E is a set of pairs of V called edges.

Graphs are well suited to help study the relationships between possible states in a discrete system. Often, graphs are introduced by imagining that vertices may be people and a pair of people is included in the edge set if they are friends. In this way, we could use model a social network as a large graph and ask questions about its structure. For example, one might be interested in the expected number of pages you might have to click through on your favorite social network until you end up at Adam Blumenthal's page. Another problem that can be modeled with graph theory is known as the 8 Queen's Problem, posed by de Jaenisch in 1862 [11]:

Question 1. *Can you place* 8 *queens on a chessboard such that no two queens can attack each other?*

More generally we can ask instead how many queens can we place on a chessboard such that no two queens can attack each other? The answer is that one can place 8 such queens. It is also easy to see that one can certainly place one queen and satisfy the non-attacking condition. This condition is a well studied graph parameter (if we model the chessboard in a particular way) called independence. We say a set of vertices S in a graph $G = (V, E)$ is *independent* if for all $\{u, v\} \in S \times S$ $\{u, v\} \notin E$. That is, an independent set is a set of vertices which contains no edges. Finding the maximum size of an independent set in a graph, called the *independence number* of the graph, is one of the most heavily studied graph parameters due to its applications and relationships

to many other parameters. Much of this thesis will be based on this idea of optimizing such a graph parameter. We now turn our attention to domination of graphs.

In a graph $G = (V, E)$, a dominating set of G is a set of vertices $S \subseteq V$ such that for each $v \in V \setminus S$ there exists some $u \in S$ such that $\{u, v\} \in E$. Intuitively, this means that a set of vertices is dominating if it has an edge to every vertex outside of the set. We note now that like independence, there is a trivial way to guarantee the existence of such a set. Namely, we may choose $S = V$, so there do not exist vertices outside of the set chosen and the condition is vacuously satisfied. Therefore, the interesting question to ask about domination in a graph is to determine the size of a smallest dominating set, called the *domination number* of G denoted $\gamma(G)$.

It is easy to see that for de Jaenisch's question, finding a set of 8 non-attacking queens would be a maximum independent set, since each queen attacks every other square in its column. Slightly weaker, we notice that this property makes such a set a *maximal independent set*. That is, an independent set S such that for any vertex $v \notin S$, $v \cup S$ is not independent. From this definition, we can see that every maximal independent set is a dominating set. We note now that not all dominating sets are independent. We define an *independent dominating set* as a set of vertices that is both independent and dominating, and observe that an independent dominating set is a maximal independent set.

Observation 1. *Let $S \subseteq V(G)$ for some graph G. Then S is an independent dominating set if and only if S is a maximal independent set.*

Proof. Let $S \subseteq V(G)$. S is independent dominating if and only if S is independent and for every $v \in V(G) \setminus S$, there exists some $u \in S$ such that $\{u, v\} \in E(G)$ if and only if S is independent and for every $v \in V(G) \setminus S$, $v \cup S$ is not independent if and only if S is a maximal independent set. □

We now slightly modify de Jaenisch's question for an interesting graph theoretic optimization problem which allows us to ask for either a maximum or a minimum set size. For a graph G, we now provide several definitions to make the optimization version of these questions easier to discuss. The *independent domination number* of a graph, denoted $i(G)$ is the size of a smallest independent

dominating set. The *independence number*, $\alpha(G)$ is the size of a largest independent set, and the *domination number*, $\gamma(G)$ is the size of a smallest dominating set.

Question 2. *Can we determine the independent domination number of a graph?*

This question is the minimization version of de Jaenisch's question, and is what we will focus on in Chapters 2 and 3. Chapter 2 will focus on the case of regular graphs with large degree and Chapter 3 focuses on independent domination number in directed graphs. In Chapter 4 we will move to discuss a different variation of domination, called split domination in directed graphs. A set of vertices $S \subseteq V(G)$ is a split dominating set if it is both dominating and its removal leaves the graph disconnected. We will take this notion into the setting of directed graphs and build the notion of a split domination sequence of a directed graph.

Finally, in Chapter 5 we will employ the method of flag algebras to answer a question on the inducibility of a graph. To introduce this question, we need several more definitions, but technical definitions and description of the method will be discussed in that chapter. We say that a graph G is a *subgraph* of a graph H if there exists an injective function $f : V(G) \to V(H)$ such that for every edge $\{u, v\} \in E(G)$, $\{f(u), f(v)\} \in E(H)$. We say that a subgraph is *induced* if there exists an injective function $f : V(G) \to V(H)$ such that for every edge $\{u, v\} \in E(G)$, $\{f(u), f(v)\} \in E(H)$ and for each non-edge $\{x, y\} \notin E(G)$, $\{f(x), f(y)\} \notin E(H)$. For a host graph H and potential subgraph G, we may be interested in finding the number of mappings which testify that it is a subgraph or induced subgraph. As the host graphs grow in size, the simple number of mappings becomes less clearly informative so we focus instead on the density with which the mapping satisfies the conditions. In particular, way say that the density of G in H is the number of distinct mappings which testify that G is a subgraph (or an induced subgraph) divided by the total number of injective mappings. We will denote the density of G in a graph H as $d_H(G)$, and when context is clear we will drop the subscript corresponding to the host graph. Often we think about this as a probabilistic approach: "if I were to pick $|V(G)|$ vertices at random, how likely is it that those vertices are isomorphic to G?" We may now state the one of the driving questions behind extremal graph theory in a general form.

Question 3. *For a given graph G, which graphs with n vertices maximize the density of induced copies of G?*

One of the most famous results is due to Turán, which is one of the first theorems which began the field of extremal graph theory. Turán determined the graphs that maximized the number of edges while also forbidding induced complete graphs (or clique). We say a graph G is a *complete graph (or clique)* if for all $u, v \in G$, $\{u, v\} \in E(G)$. A complete graph on n vertices will be denoted K_n.

Theorem 1 (Turán's Theorem)**.** *Let G be a graph on n vertices that contains no K_{r+1} as a subgraph. Then G contains at most $\frac{r-1}{r}\frac{n^2}{2}$ edges.*

This theorem can be viewed as an answer to the following optimization question: What is the maximum number of edges in a graph on n vertices which has $d_H(K_{r+1}) = 0$? Several generalizations of this theorem exist, the most famous being a result of Erdős and Stone which uses the chromatic number to determine the maximum number of edges a graph can contain before any graph G must appear as a subgraph. The theorem of Erdős and Stone still has its limitations, namely that for bipartite graphs the bound is not terribly meaningful. Advances toward understanding the extremal nature of bipartite graphs continue to be of significant interest in the field. One property that one might be interested to find is which graphs are maximized simply by bigger versions of themselves? To formalize this question, we define the iterated blow-up of a graph G as follows: For each $v \in V(G)$, replace V with several vertices $\{v_1, v_2, \ldots, v_{|V(G)|}\}$ all adjacent to the same vertices as v, and such that $\{v_1, v_2, \ldots, v_{|V(G)|}\}$ is isomorphic to G. Call this new graph G_1 and we can repeat this process, replacing each vertex with a copy of G as many times as we like to create graphs $G_2, G_3, \ldots, G_n$. Each of these graphs is what is called an *iterated blow up of* G. We will say that a graph G that has its density maximized only the iterated blow ups of G are called fractalizers. This turns out to be the natural formalization of our question due to the surprising theorem of Fox, Huang, and Lee [19] which states that almost all graphs are fractalizers. The peculiarity of this result is that it uses random graphs, which means that the only known fractalizers are the empty graphs (graph with no edges) and complete graphs. In Chapter 5 in joint

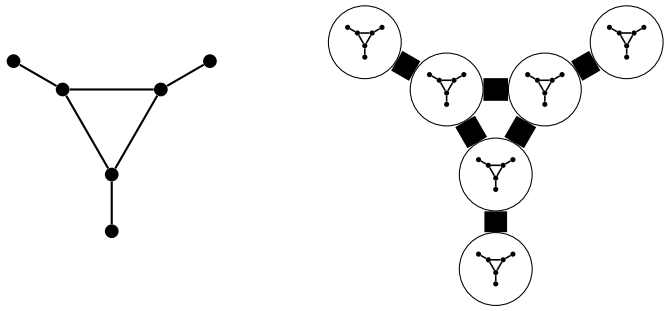

Figure 1.1 The Net Graph and a Blow Up of the Net

work with Michael Philips, we sought to find another fractalizer (see Figure 1.1). We will instead show that the net is not a fractalizer, but for certain graph sizes, the iterated blow up is the only maximizer for the density of the net.

CHAPTER 2. INDEPENDENT DOMINATION IN REGULAR GRAPHS

2.1 Introduction

In this chapter, we explore independent domination in regular graphs, proving that there exist finite graphs G with independent domination number arbitrarily close to $|V(G)|/2$ which are not complete bipartite graphs. The study of independent domination in regular graphs began with a result of Rosenfeld [37] which showed that in any regular graph, the largest independent dominating set is of size at most $|V(G)|/2$. Indeed this bound is achieved by the complete bipartite graph $K_{\frac{n}{2},\frac{n}{2}}$, but if we exclude complete bipartite graphs, the question becomes more interesting. Let c_r denote the supremum of $i(G)/n$ taken over all connected r-regular graphs G of order n except $K_{r,r}$. Note that $c_r \leq 1/2$ for all r by the theorem of Rosenfeld. It has been proven that $c_3 = 2/5$ [30, 12] and it can be shown the $c_2 = \frac{3}{7}$. Further problems have been considered with different classes of forbidden graphs beyond just the complete bipartite graph. This problem is motivated by a question of Goddard and Henning from 2013 [22].

Question 4 ([22]). *Is it true that c_r tends to $\frac{1}{2}$ as r goes to ∞?*

2.2 c_r as r Approaches Infinity

In this section we expand on the results of P.C.B. Lam et al. (On independent domination number of regular graphs) [30] in which they prove that for all $\varepsilon > 0$, there exists some $r > 0$ such that $c_r \geq \frac{1}{2+\varepsilon}$.

We notice first that there are some simple results that suggest that Question 4 should be answered affirmatively. It is known that c_r is, in some very weak way, non-decreasing.

Theorem 2 ([22]). *For all positive integers r and s, $c_{rs} \geq c_r$.*

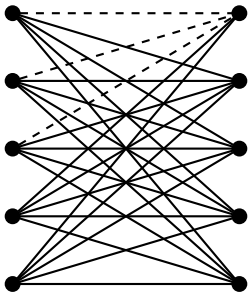

Figure 2.1 The Graph $K^4_{5,5}$

Theorem 3.

$$\lim_{r\to\infty}(c_r) = 1/2.$$

We will prove the theorem by producing a family of graphs $\mathcal{F}$ such that for every $\varepsilon \geq 0$, there exists $G \in \mathcal{F}$ such that $i(G)/n \geq \frac{1}{2} - \varepsilon$. The construction of $\mathcal{F}$ hinges on the following seemingly odd lemma. For all $r > k$, let $K^k_{r,r}$ be the graph $K_{r,r}$ with the edges of a star with $k-1$ leaves removed.

Lemma 1. *Any independent set of $K^k_{r,r}$ with S being the star removed which dominates all vertices of degree r is a subset S containing the center or has order $\geq r-k-1$.*

Proof. Notice also that any subset of S containing the center is such a set. Let $D \subset S$ not containing the center. We notice that D is not a dominating set of all vertices of degree R since $r > k$.

Let R be the vertices in $K^k_{r,r}$ having degree r.

Let the bipartition of $K_{r,r}$ be $\{X, Y\}$ and without loss of generality, let the center of the star removed be in X. Suppose $x \in X \setminus v$ is in an independent dominating set. Since x is adjacent to all vertices in Y, we must select all vertices of X to dominate X, so any such independent set dominating R has size $r-1$. Suppose instead that $y \in Y \setminus (S \cap Y)$ is in the independent dominating set. Similarly y is adjacent to all vertices in X so all of $Y \setminus (S \cap Y)$ must be chosen, so any such independent set dominating R has size $r-k-1$.

To see the second part of the lemma, we note that by this proof if any vertex not from the star is chosen, the independent set dominating R must be of size at least $r-k-1$. □

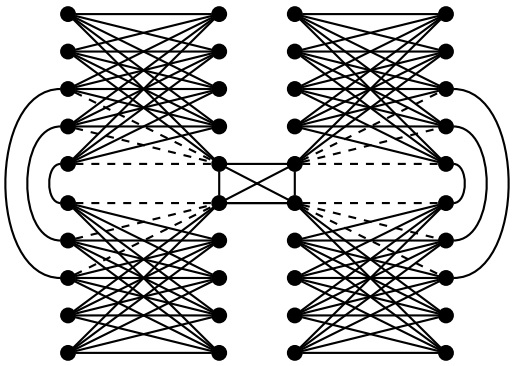

Figure 2.2 The Graph $G_{5,4}$.

We now construct the family $\mathcal{F}$ as follows. Let $r > k \geq 2$ with k even. We begin with K_k and let M be a perfect matching of K_k. For each vertex $v \in K_k$ we add $2r-1$ additional vertices, inducing a copy of $K^k_{r,r}$ where v is the center of the star removed. In each $K^k_{r,r}$ we see that all vertices have degree r except those which are not adjacent to the center of the star removed. Using the matching M, we now add a matching between the vertices missing one degree to those in the corresponding $K^k_{r,r}$ from the matching. The constructed graph $G_{r,k}$ is now r-regular.

Proposition 1. *$G_{r,k}$ is an r-regular connected graph of order $2rk$.*

Proof. Easy to see order. Looking at $K^k_{r,r}$ subgraphs, we see most vertices are r regular, those involved in stars as the center gain $k-1$ degree from the complete graph. Vertices involved as pendants of the star are paired off using the matching to regain the one deficient degree.

To see that the graph is connected, we note that since $r > k$ there exists a vertex in each $K^k_{r,r}$ adjacent to the complete graph, and $K^k_{r,r}$ is connected, hence the entire graph is connected. □

Lemma 2. *For $r >> k$, $i(G_{r,k}) \geq k + (r-k-1)(k-1)$.*

Proof. We observe first that by construction, at most one induced copy of $K^k_{r,r}$ can achieve an independent dominating set of size k. By Lemma 1, any set of size $\leq r-k-1$ that is independent and dominates the vertices not involved in the removed star contains the center of the star. Hence, to maintain independence, at most one induced copy of $K^k_{r,r}$ has a dominating set of size k. Notice

also that for each of the remaining induced $K^k_{r,r}$, we must dominate the vertices of degree R which are adjacent only to vertices in the induced $K^k_{r,r}$. By Lemma 1, such an independent dominating set must have size at least $r-k-1$.

Therefore $i(G_{r,k}) \geq k + (r-k-1)(k-1)$, as desired. □

Theorem 4. *c_r tends to $\frac{1}{2}$ as $r \to \infty$.*

Proof. Let $\varepsilon > 0$. We seek to find a regular graph G such that $i(G) > \frac{1}{2} - \varepsilon$. Let $k \geq \frac{1}{\varepsilon}$, $r \geq k^2$. Then

$$\begin{aligned} c_r &\geq \frac{i(G_{r,k})}{2rk} \\ &\geq \frac{k + (r-k-1)(k-1)}{2rk} \\ &\geq \frac{rk + 2k - r - k^2 - k + 1}{2rk} \\ &= \frac{1}{2} + \frac{1}{r} - \frac{1}{2k} - \frac{k}{2r} - \frac{1}{2r} + \frac{1}{2kr} \\ &\geq \frac{1}{2} + \varepsilon^2 - \frac{\varepsilon}{2} - \frac{\varepsilon}{2} - \frac{\varepsilon^2}{2} + \frac{\varepsilon^3}{2} \\ &> \frac{1}{2} - \varepsilon. \end{aligned}$$

□

We note that with this, the question of Goddard and Henning [22] is resolved. We see though, that as a limit argument, this answers the question of independent domination ratio in large graphs. It remains to get a better understanding of independent domination in the context of graphs with fixed regularity r for specific small values of r.

2.3 Graphs with Fixed Regularity

We continue this chapter with a brief discussion of the construction technique which was used above roughly based on the ideas of the construction above. That is, given a regular graph G, we can construct a graph of higher regularity by replacing each vertex with a copy of a graph close to a complete bipartite graph.

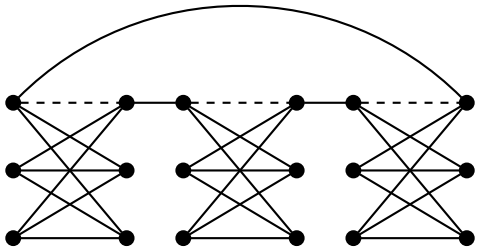

Figure 2.3 The Graph G_3.

We will construct a connected r-regular graph G_r for each $r > 2$. First let G be some k regular graph for $k < r$. For each $v \in V(G)$, replace v with $2(k-r)+2$ vertices labeled $\{v_0, v_1, \ldots, v_{2(k-r)+2}\}$. Add edges between $v_i, v_j \in \{v_1, v_2, \ldots, v_{2(k-r)+2}\}$ if and only if $i \not\equiv j \mod 2$. and make v_0 adjacent to all vertices v was adjacent to as well as all odd labeled vertices in $\{v_{2(k-r)+2}, \ldots, v_{2k+2}, v_{2k+1}\}$. We notice that all vertices in $\{v_1, v_3, v_5, \ldots, v_{2k-1}\}$ are one degree short. Repeat this process for all vertices in G. To fix this, we note that there must be an even number of total vertices missing degree 1, therefore we can find a matching between them (such that no two vertices in the same vertex image are paired). Note that this construction is how the previous section was proven, using the base graph as K_{k+1}. Below we provide an argument with a more simple construction which gives a lower bound general r for small r.

Begin with three disjoint copies of $K_{r,r} - e$, call them H_i for $i \in \{1, 2, 3\}$. Pick one vertex in H_1 with degree 2, and add an edge to a vertex of degree 2 in H_2, then do the same for H_2 and H_3 and the same for H_3 and H_1. Notice that there will always be a vertex of degree 2 to pick in this process since we pick exactly 2 vertices of degree 2 from each H_i.

Lemma 3. *For each subgraph $H = K_{r,r} - e$ in G, any independent set dominating vertices of degree r contains either 2 or $\geq r-1$ vertices of H. Furthermore, the only such set of size 2 is the vertices incident to the removed edge.*

Proof. Let x_e and y_e be the vertices incident to e in part X and Y respectively. If $x \in X - x_e$ is chosen, it is adjacent to all of Y, hence no vertices of Y can be chosen and be independent of x,

so at least $r-1$ vertices must have been chosen to cover $X - x_e$, so at least $r-1$ vertices must be chosen. Similarly if a vertex of $y \in Y - y_e$ is chosen, at least $r-1$ vertices must be chosen. Otherwise x_e, y_e are chosen as the dominating set. □

Lemma 4. *If in a subgraph $H = K_{r,r} - e$, an independent dominating set of order 2 is chosen, each remaining subgraph must have at least $2(r-1)$ vertices chosen.*

Proof. Suppose 2 vertices are chosen in H. We look at the two adjacent copies of $K_{r,r}-e$ subgraphs. By construction, each is forbidden from taking an endpoint of missing edge, so by above lemma, each requires at least $r-1$ vertices to be chosen. □

Corollary 1. *For all r, $c_r \geq \frac{1}{3}$.*

Proof. By the lemmas above $i(G_r) \geq n/3$ and is r-regular. □

2.4 Conclusion

Theorem 4 provides a lower bound for c_r for large r, but it remains to be seen what the best possible bound is for a fixed r. In fact, a significant portion of the study of c_r has been dedicated entirely to fixing a single r value and trying to find the exact value. As such, it would be interesting to determine an upper bound for c_r which matches some construction.

Question 5. *Does there exist an integer n_0 such that for any connected graph G on $n > n_0$ vertices other than complete bipartite graphs, $i(G) < 2n$?*

I suspect that this is not the optimal construction for any r, though it does have the same limit as the known best upper bound. There is only one known three regular graph witnessing the bound of 2/5 and upon forbidding this graph (like the complete bipartite graphs) what better bounds can be found. The current state of the art is that $c_3 = 2/5$, and there is a standing question for a modified version of c_3 by Goddard and Henning where instead of forbidding only the complete bipartite graph, also $C_5 \square K_2$ is forbidden.

Conjecture 1 ([22]). *If G is a connected cubic graph on n vertices other than $K_{3,3}$ and $C_5\square K_2$ then $i(G) \leq 3n/8$.*

Goddard and Henning made this conjecture upon finding an infinite family of 3-regular graphs with $i(G) = 3n/8$. In particular, it would be interesting to find an infinite family of graphs which all have the same ratio, and a matching upper bound for graphs forbidding some finite number of anamolous graphs (like $C_4\square K_2$). The first step which would need to be answered is if such a goal can even be achieved. That is, does there exist an infinite family of 3-regular graphs $\{G_i\}_{i=1}^{\infty}$ such that $\frac{i(G_k)}{|V(G_k)|} > \frac{i(G_{k+1})}{|V(G_{k+1})|}$ and $\frac{i(G_k)}{|V(G_k)|} > c > \frac{3}{8}$ for all k.

A remaining open question for further research in this field is to determine c_r for small r. Namely, the value of c_4 is still unknown, and c_5 has largely been untouched. The best known conjecture for c_4 is $3/7$ which was found by an exhaustive search of all graphs up to around twenty five vertices [22]. It would also be interesting to find an analogous construction to that of Goddard and Henning's 3-regular construction of an infinite family to provide a lower bound on c_4 which cannot be lowered by forbidding a finite number of graphs.

CHAPTER 3. INDEPENDENT DOMINATION IN DIRECTED GRAPHS

After this research was conducted, it was determined that independent domination had been studied under a different name (kernels of directed graphs) and has a long history. We will keep our discussion as some of the methods of proof are short and unique, and a discussion of what has been contributed to the field is included in the conclusion, as well as what results in this chapter are known to have been proven before.

Both the dominating set problem and independent set problem have been studied extensively in graphs. Independence has been widely studied for its relation to chromatic number, while domination has a deep relationship with communication in networks. The study of sets that are both independent and dominating (or independent dominating sets) has history dating back to 1862, when de Jaenisch [11] asked for the minimum number of non-attacking queens which can be placed on a chessboard such that every other square is threatened. We note also that both independence and domination are classic examples of NP-complete problems, as is finding the smallest independent dominating set [21]. It has been proven that determining the minimum size of an independent dominating set is NP-complete even in restricted families including bipartite graphs or line graphs [31, 42, 10]. The minimum size of a dominating set is used as a measure of efficiency of backbones for communications networks, and independent domination can be used for communication networks in which interference or fading can occur. Further results include Nordhaus-Gaddum type results [22, 24], and results for claw-free graphs [1], as well as random graphs [12]. For a thorough survey of the history and results in independent domination theory, we direct the reader to the paper [23].

In directed graphs independence is no different from the question in undirected graphs. On the other hand dominating sets are drastically affected by direction. There is a long history of dominating set problems in directed graphs, but frequently they are restricted to certain families

of graphs. In particular, domination in tournaments has been studied for decades, including questions of Erdős [14] and Gyárfás [32]. More recently, Caro and Henning [6] continued the study of dominating set theory in directed graphs, providing some general bounds as well as relating the directed domination number to the independence number in bipartite graphs.

In 2019, Cary, Cary, and Prabhu [7] introduced independent domination in directed graphs. This problem has relations to finding communication points for information transmission, particularly when information can only be sent in one direction at a time in a network. As such, they explore the parameter with respect to oriented graphs since they correlate to ad-hoc networks [13].

3.1 Introduction

We define a *directed graph* $D = (V, A)$ to be an ordered pair, where V is a set called vertices ($V(G)$) and A is a set of pairs of vertices called the edge set or arc set ($A(G)$). A set of vertices S to be *independent* in a directed graph D if there does not exist $u, v \in S$ such that (u, v) is an arc in D. A set of vertices S to be *dominating* in a directed graph D if for every $v \in V(G) \setminus S$ there exists some $v \in S$ such that $(u, v) \in A(D)$. A set of vertices S to be *independent dominating* in a directed graph D if S is both independent and dominating.

Cary, Cary, and Prabhu [7] provide results on certain families of graphs including orientations of bipartite graphs and cycles as well as directed acyclic graphs. In this paper, we extend the study of independent domination into directed graphs which allow antiparallel edges, noting that parallel edges do not affect independent domination in directed graphs. All directed graphs will be assumed to be finite. We will provide a result which generalizes several of the results of the previous paper, namely determining the number of pairwise disjoint independent dominating sets, called *idomatic number*, for directed graphs with certain periods. We additionally provide some alternative, algorithmically focused, proofs of similar results to Cary, Cary, and Prabhu. We also begin the study of time complexity of independent dominating sets, showing that determining the smallest size of an independent dominating set in a directed graph in is NP-complete and providing an algorithm which answers this question in $O(1.26^n)$ time when the period of the graph is not

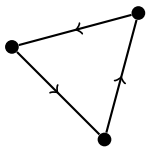

Figure 3.1 An Example of a IDS-Free Digraph.

one. Cary, Cary, and Prabhu also introduce the concept of idomatic number of a graph G, and explore the parameter in some families of graphs. In the conclusion, we suggest possible avenues for furthering the theory of independent domination in directed graphs.

3.2 A Greedy Heuristic

In this section we will provide a simple heuristic for finding an independent dominating set, which gives some short alternative proofs to those given in [7]. Our goal throughout this section is to provide a tool for determining the existence of an independent dominating set in a directed graph, with the goal of classifying graphs which contain no independent dominating set which we call *independent dominating set-free (IDS-free).*

Note that in undirected graphs, there always exists an independent dominating set which can be made by greedily adding vertices until we reach a maximal independent set. In directed graphs, this is not the case. Notice that, for example, a directed 3-cycle has no independent dominating set. We seek to provide conditions for when a digraph D has an independent dominating set.

In a directed graph D we call a vertex v a *source* if it has $d^-(v) = 0$. We define the *source-greedy algorithm (SGA)* as follows: for D, while there exists a source in the graph, choose one to be placed in the IDS, then remove it and all of its out neighbors. This returns a graph with no sources.

Claim 1. *The vertices chosen by the source-greedy algorithm are independent.*

Proof. We consider the step at which the vertex v is chosen by the SGA. Since v is chosen, it remains in the graph, so there are no edges from previously chosen sources to v. Also v cannot have had an edge to any previously chosen vertex, else it would not have been a source. □

Note that with the source-greedy algorithm guaranteeing an independent set, we can now refine our search to source-free graphs.

Claim 2. *All oriented bipartite graphs have an IDS.*

Proof. First we run SGA. What remains after SGA is a source-free graph. Now we may simply take one side of the bipartition in the independent dominating set. Note that since there are no sources and the graph is now isolate free, each vertex has at least one in-neighbor on the other side, so by taking an entire side, either the vertex is chosen or its in-neighbor is chosen. □

Observation 2. *A graph is IDS-free if and only if every execution of the SGA leaves a source-free graph with no IDS. In particular, every vertex minimal IDS-free graph is source-free.*

Proof. By contraposition, if there exists an execution which leaves a source-free graph with an IDS, we run that execution and add the remaining IDS.

Suppose the graph has an IDS. Notice that each source must be taken in the independent dominating set, reducing the problem to a subgraph. Repeat. □

A digraph is said to be *acyclic* if it does not contain any subgraphs isomorphic to a directed cycle.

Theorem 5. *Every Directed acyclic graph contains an independent dominating set.*

Proof. Consider a topological ordering of the vertices. The source greedy algorithm will provide an independent dominating set, since at each stage that a vertex set is removed, no cycles are created and we have reduced the problem to another directed acyclic graph. Since every directed acyclic graph contains a source, this process will only terminate when no vertices remain, namely every vertex was either chosen, or was deleted by a chosen vertex which dominates it. □

Corollary 2. *Every oriented tree contains an independent dominating set.*

We now build to the main theorem, expanding the source-greedy algorithm to strongly connected components of a graph. Call a graph G *vertex minimal IDS-free* if D has no IDS and for every

subset $S \subseteq V(D)$, $D \setminus N^+[S]$ has an IDS I such that $I \cap (N^-(S)) = \emptyset$. We define vertex minimal in this way as a generalization of the source-greedy algorithm, where S is acting as a source which can be removed.

Theorem 6. *Any vertex minimal IDS-free digraph is strongly connected.*

Proof. Let D be a vertex minimal IDS-free digraph. Consider the strongly connected components of G. The reduced graph generated by contracting the strongly connected components is acyclic, hence there exists a source vertex. The strongly connected component corresponding to this source vertex, C, can be dominated only by other vertices in C. If C has an IDS, then $G - C \cup (N^+[C])$ has no IDS, else G has an independent dominating set. Otherwise, C has no IDS, a contradiction with minimality of D unless $D = C$. □

Claim 3. *Every vertex in a strongly connected digraph has at least one in edge and at least one out edge.*

Proof. Clear, since if a vertex has $d^+(v) = 0$ there does not exist a path from v to any other vertex, and if $d^-(v)$ there does not exist a path from any other vertex to v. □

Since odd cycles are a problem for independent domination, we explore the digraphs with specific periods. We define the *period* of a digraph D to be the greatest common divisor among all lengths of directed cycles which appear as subgraphs in D. As convention, we will say that the period of a directed acyclic graph is 0.

We now introduce some tools of linear algebra, which will come in handy for the next proof. For a directed graph D on n vertices, we define the *adjacency matrix* of D, A_D (or just A if context is clear) to be the $n \times n$ matrix with (i, j) entry 1 if $(i, j) \in A(D)$ and 0 otherwise. We say that a square matrix is *irreducible* if it is not similar via a permutation matrix to a block upper triangluar matrix. The following well known theorem is a fundamental result of spectral graph theory, relating linear algebra and directed graphs:

Theorem 7 ([25])**.** *A directed graph G is strongly connected if and only if A_G is irreducible.*

Perron-Frobenius theory provides a deeper relationship between graph and digraph properties and their respective adjacency matrices. For more information about this relationship, we direct the reader to the textbook [25]. In particular, the period of a strongly connected digraph creates rich structure in the adjacency matrix, as evidenced by the following theorem.

Theorem 8 ([20])**.** *If G is a digraph with period $h > 1$, there exists some permutation matrix P such that PAP^{-1} is a block matrix*

$$PAP^{-1} = \begin{pmatrix} 0 & A_1 & 0 & & 0 \\ & & A_2 & \ddots & \\ \vdots & & \ddots & \ddots & 0 \\ 0 & & & & A_{h-1} \\ A_h & 0 & & \cdots & 0 \end{pmatrix}$$

where each diagonal block is square zero matrix.

We notice that this provides a way to partition our digraph D into h independent sets, which we will call $S_0, \ldots S_{h-1}$ corresponding to the vertices of the diagonal zero blocks. With this structure theorem, we may now prove the main theorem of the paper.

Theorem 9. *Every strongly connected directed graph with even period has an independent dominating set.*

Proof. If D has period h, as above the graph can be partitioned into h independent sets $S_0, \ldots, S_{h-1}$ such that there exists an edge from u to v only if $u \in S_i$ and $v \in S_{i+1}$ for some $i \in [h]$ with addition modulo h. Therefore, we can create an independent dominating set by taking all S_i such that i is even (or odd).

To see that this set is indeed an IDS, since we take only independent sets of the same parity, and h is even there are no parts which are taken that share any adjacencies. Furthermore, since every vertex has at least one in degree, we observe one vertex v. Either v is included in the set, or it is in S_i which is not included and has at least one in degree from S_{i-1}, say from u. But if S_i is not included in our set, S_{i-1} is included in our set, hence u is in the set and dominates v. □

Corollary 3. *If G is vertex minimal IDS-free, G has odd period.*

Corollary 4. *Every oriented bipartite graph has an independent dominating set.*

Cary, Cary, and Prabhu [7] define the maximum number of vertex disjoint independent dominating sets in a digraph G as the *Idomatic Number*, written $id(G)$. We note that Corollary 4 was proven in this paper as they worked towards determining graphs with $id(G) = 1$. Our proof provides a bound for the idomatic number of digraphs with even period.

Corollary 5. *Every strongly connected digraph D with even period has $id(D) \geq 2$.*

3.3 Vizing's Conjecture

In this section, we show that the analogous statement to the famous Vizing's conjecture does not hold with independent dominating sets. Vizing's conjecture is about the relationship between domination number (the smallest size of a dominating set of a graph G, $\gamma(G)$) of graphs with their Cartesian product.

We define the Cartesian product of directed graphs with vertex set $V(G) \times V(H)$ with edges defined by :

$$A(G\Box H) = \{(x,u)(y,v) | xy \in A(G) \text{ and } u = v \text{ or } uv \in A(H) \text{ and } x = y\}$$

Conjecture 2 (Vizing [40])**.** *For any undirected graphs G and H, $\gamma(G\Box H) \geq \gamma(G)\gamma(H)$.*

This also has an analogous conjecture in independent domination, asked by Goddard and Henning, which would imply Vizing's conjecture. For the *independent domination number*, the smallest size of a dominating set of a graph G, denoted $i(G)$. Vizing's Conjecture is altered in the case of independent domination since it has been proven that there exist graphs G, H such that $i(G\Box H) < i(G)i(H)$ [4].

Conjecture 3 ([4])**.** *For any undirected graphs G and H,*

$$i(G\Box H) \geq \min\{i(G)\gamma(H), \gamma(G)i(H)\}.$$

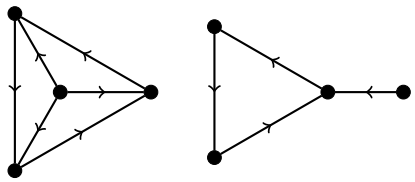

Figure 3.2 The Graphs W_3' and P'.

We will show that the possibility of a directed graph containing no independent dominating set will provide examples that ensure that no such inequality holds in directed graphs. One may wonder how to define the independent domination number for directed graphs without independent dominating sets. Some natural candidates for G IDS-free would be $i(G) = 0$, $i(G) = n + 1$, or $i(G) = \infty$. The following corollary shows that the direct translation of the conjecture of Goddard and Henning into directed graphs cannot hold regardless of which convention is chosen. In the case that $i(G) = 0$ is chosen, Claim 4 provides a family of counterexamples, and in the other two cases Claim 5 provides a family of counterexamples.

To provide a family of directed graphs which contain independent dominating sets whose Cartesian product does not contain an independent dominating set we define the following graphs. Define W_n' to be a directed wheel on $n+1$ vertices in which the center vertex is dominating and the outside cycle is directed. We define P', as an oriented paw with directed edges as in Figure 3.3.

Claim 4. *$W_n'\Box P'$ is IDS-free for all n odd.*

Proof. Let $n \in \mathbb{Z}$ be odd. We notice that both W_n' and P' have unique independent dominating sets by following the source greedy algorithm. Let the dominating vertex of W_n' be v_d Also, in $W_n'\Box P'$, the copy of P' that appears in place of the dominating vertex of W_n' must be dominated only by vertices of the form (v_d, u) for some $u \in P'$. Hence the unique dominating set of P' must be chosen for this copy of P'. Therefore, all copies of the dominating set of P' around the cycle cannot be included in an independent dominating set.

We now look at the strongly connected components of the graph induced by the vertices which have yet to be dominated. There are two strongly connected components, both of which are cycles

on n vertices. One of these vertices acts as a source in the directed acyclic graph created by contracting strongly connected components, hence it must be dominated only by vertices in its own strongly connected component. This is impossible, since it is an odd cycle which is known to have no independent dominating set. □

Claim 5. *$C_n \square C_n$ where n is odd contains an independent dominating set.*

Proof. Let n be odd. It has been observed that each directed odd cycle does not have an independent dominating set. It remains to provide an independent dominating set for $C_n \square C_n$. Label the vertices of one $V(C_n) = \{v_0, \ldots, v_{n-1}\}$ such that $(v_i, v_{i+1}) \in A(G)$ with addition modulo n, and for the other copy of C_n, $V(C_n) = \{u_0, \ldots, u_{n-1}\}$ similarly. We construct an independent dominating set of $C_n \square C_n$ as $D = \{(v_i, u_{i+2j}) | 0 \leq i \leq n-1 \mod n, 0 \leq j \leq \lfloor \frac{n}{2} \rfloor\}$. To see that the set is dominating, we notice that for any $0 \leq i \leq 4$, (v_i, u_i) and (v_i, u_{i+2}) dominate all (v_i, u_k) for $i \leq k \leq i + n - 2 \rfloor \mod n$, leaving only (v_i, u_{i-1}) not dominated. But we have that $(v_{i-1}, u_{i-1}) \in D$ which dominates (v_i, u_{i-1}). Therefore D is dominating. For i fixed, we have $\{(v_i, u_{i+2j}) | 0 \leq j \leq \lfloor \frac{n}{2} \rfloor\}$ is independent since edges occur if and only if $(u_k, u_\ell) \in E(C_n)$, but we have only taken vertices of the same parity, without taking a full trip around the vertex set. That is, in each v_i we take only vertices (v_i, u_j) where $i = j \mod 2$. Hence, vertices (v_i, u_j) and (v_{i+1}, u_k) we have no edges, since $j \neq k$. Therefore D is independent, thus an independent dominating set. □

Theorem 10. *There exist infinitely many pairs of graphs (G, H) such that*

$$i(G \square H) > \min\{i(G)\gamma(H), \gamma(G)i(H)\}$$

and infinitely many pairs of graphs (G', H') such that

$$i(G' \square H') < \min\{i(G')\gamma(H'), \gamma(G')i(H')\}.$$

Proof. This is a direct consequence of Claims 4 and 5. □

3.4 Time Complexity

We note that finding the size of an independent dominating set in undirected graphs is a well known NP-complete problem, for example it is proven in the textbook of Garey and Johnson [21].

Theorem 11 (Garey and Johnson [21])**.** *Given a graph G and constant k, determining existence of an independent dominating set S such that $|S| \leq k$ is NP-complete.*

Corollary 6. *Given a directed graph D and constant k, determining existence of an independent dominating set S such that $|S| \leq k$ is NP-complete.*

Proof. Suppose that we have some oracle f for directed independent dominating sets. For G, an undirected graph, we may create a corresponding directed graph by replacing every edge with a pair of antiparallel edges, creating a graph G'. We run f on G'. By returning whichever result comes from running f on G', we have answered the problem for the undirected graph G. We see this, since S is an independent set in G if and only if S is an independent set in G' by construction. Also S is an dominating set in G if and only if S is an dominating set in G' by construction. □

We notice that the existence of an independent dominating set in a graph G of order n is equivalent to determining if there exists an independent dominating set of order at most n. In particular, this problem is trivial for undirected graphs since all graphs contain an independent dominating set. We seek to determine if for directed graphs determining the existence of an independent dominating set S such that $|S| \leq n$ is NP-complete.

Claim 6. *Given a directed acyclic graph D, determining the existence of an independent is in P.*

Proof. By the proof of Theorem 5, we provide an algorithm that is polynomial in time. □

Theorem 12. *Given a directed graph D with even period h, determining the existence of an independent dominating set is in P.*

Proof. Since we know the period of D is h, we can construct h independent sets using breadth first search by creating layers modulo h (that is, the h^{th} layer is the same as the first vertex chosen in $O(n^2)$ time. Then we select all vertices in even layers as our independent set. □

We note that the period of a digraph can be determined in polynomial time, as proven by Jarvis and Sheir [27].

Theorem 13. *There exists an $O(2^{\frac{n}{h}})$ algorithm for determining the existence of an independent dominating set in a graph D of period h.*

Proof. Similar to the above proof, we may partition the vertices of D into h independent sets $S_0, \ldots, S_{h-1}$ such that edges follow cyclically. We note now that if h is even, we have an independent dominating set, so we may assume that h is odd.

Let S_k be the smallest independent set in our partition of the vertices, then $|S_k| \leq \frac{n}{h}$. We notice now that the selection of vertices in one part forces the structure of the rest of the independent dominating set. That is, let D be an independent dominating set of G, then D is the union of $S_i \cap D$, $S_{i+1} - N^+(S_i \cap D)$, $S_{i+2} - N^+(S_{i+1} - N^+(S_i \cap D))$, *dots*. Note that this observation gives us that $S_{i-1} - (S_{i-1} \cap D) \subseteq N^-(S_i \cap D)$. In particular, we may search among only the smallest independent set for the independent set giving the desired bound. Since there are h parts, there exists at least one part of size at most n/h, and a brute force search among each of the subsets of these vertices will be $O(2^{\frac{n}{h}})$ time. □

Corollary 7. *For any digraph D with period $h \neq 1$, there exists an $O(1.26^n)$ algorithm to determine existence of an independent dominating set.*

Proof. The algorithm provided in the proof above for odd degree is $h = 3$, yielding an $O(2^{\frac{n}{3}}) \leq O(1.26^n)$ algorithm, since directed acyclic graphs and graphs with even period are in P. □

3.5 Additional Constructions

One may wonder if all graphs with odd period have no independent dominating set. We now provide examples for each odd period of infinite families of graphs which have independent dominating sets and which do not have independent dominating sets. We start with a few lemmas to work toward constructions of infinite families of graphs with specific period that contain independent dominating sets, and that do not contain independent dominating sets.

Lemma 5. *Let D be a digraph with odd period h and vertex partition $S_1, \ldots, S_h$ such that for every edge $(u, v) \in A(D)$, $u \in S_i$ and $v \in S_{i+1}$ for some $i \in [h]$ with addition modulo h. Every independent dominating set I has $S_i \cap I \neq \emptyset$ and $S_i \cap I \neq S_i$ for all $i \in [h]$.*

Proof. Let D a digraph with odd period h and S_i be as in the statement of the theorem for $1 \leq i \leq h$. If $h = 1$, the statement is clear.

Suppose $h > 1$ and assume for contradiction that there exists some S_i such that $S_i \cap I = \emptyset$. We notice that the only vertices which can dominate the vertices of S_{i+1} are in S_i or the vertices themselves. Therefore, $S_{i+1} \cap I = S_{i+1}$. Since the digraph is strongly connected, every vertex in S_{i+2} has a neighbor in S_{i+1}, hence $S_{i+2} \cap I = \emptyset$. By a similar argument, we see that $S_{i+2l} \cap I = \emptyset$ for all l, with addition modulo h. Since h is odd, for each $1 \leq j \leq h$ there exists some k such that $S_j = S_{i+2k}$. Therefore $S_i \cap I = \emptyset$ for all $1 \leq i \leq h$, a contradiction with I being an independent dominating set. The argument that $S_i \cap I \neq S_i$ is similar. □

This lemma gives us a simple way to create infinite families of graphs which do not contain independent dominating sets for each period. Namely, any strongly connected digraph with odd period h in which the decomposition into h independent sets has at least one set of size 1 cannot have an independent dominating set. We seek to find a family more rich in structure which has no independent dominating set, which will lead to a very similar family that does contain independent dominating sets.

Lemma 6. *For each odd integer $h > 1$, there exists an infinite family of graphs $\mathcal{F}$ with period h such that for all $D \in \mathcal{F}$, D is independent dominating set-free.*

Proof. We will construct a graph $D_{h,k}$ with period h for any $2 \leq k$ which has no independent dominating set. We will use the fact that since D has period h, it can be partitioned into h independent sets $S_0, \ldots, S_{h-1}$ such that for all edges (u, v), $u \in S_i$ and $v \in S_{i+1}$ for some $0 \leq i \leq h-1$ with addition modulo h. We will create $S_0, \ldots, S_{h-1}$ as such a partition. Let $k \geq 2$.

We create a special graph for the case $h = 3$. Let the vertices of S_0 be k vertices labelled 1 to k, the vertices of S_1 be all subsets of $[k]$, and the vertices of S_2 be a copy of the vertices of S_1. We

draw edges between $u \in S_0$ and $X \in S_1$ if and only if $u \in X$, between $X \in S_1$ and $Y \in S_2$ if and only if $X = Y$, and between $Y \in S_2$ and $v \in S_0$ if and only if $v \in Y$.

Let $h > 3$. Let S_0 be k vertices, labelled 1 to k. Define S_1 to be all nonempty and not full subsets of $[k]$. With edges from $u \in S_0$ to $X \in S_1$ if and only if $u \in X$. Then S_2 is a copy of S_1 with edges from $X \in S_1$ to $Y \in S_2$ if and only if $X = Y$. S_3 will have k vertices, again labelled from 1 to k, with edges (Y, v) from S_2 to S_3 if and only if $v \notin Y$. For each $j \leq h - 3$ odd, the vertex set of S_j is k vertices labelled 1 to k, and S_{j+1} will have vertices corresponding to subsets of $[k]$ with edges from $\ell \in S_j$ to $Z \in S_{j+1}$ if and only if $\ell \notin Z$ and edges $Z \in S_{j+1}$ to $m \in S_{j+2}$ if and only if $m \notin Z$. For the final independent sets, we follow that S_{h-2} is a set of size k labeled from 1 to k, create S_{h-1} as all subsets of $[k]$, with have edges from $u \in S_{h-2}$ to $X \in S_{h-1}$ if and only if $u \in X$, and finally from $Y \in S_{h-1}$ to $v \in S_0$ if and only if $v \notin Y$. See Figure 3.5 for an example of $D_{5,3}$.

For any independent dominating set I, we claim that $|S_0 \cap I| = 1$. Suppose for contradiction that $|S_0 \cap I| \geq 2$ without loss of generality we may assume that $S_0 \cap I \supseteq \{1, 2\}$. Then $S_1 \cap I$ is contains all sets which contain neither 1 nor 2. Then we have that $S_3 \cap D$ is all sets which contain either 1 or 2, in particular, both the set 1 and 2 are in the dominating set, and in S_4 1, and 2 dominate S_5 since 1 does not contain 2 and 2 does not contain 1. Therefore $S_5 \cap D = \emptyset$. A contradiction with Lemma 5. Indeed, $|S_0 \cap D| = 1$.

Since all vertices of S_0 are the same up to isomorphism, and every independent set must have nonempty intersection with the dominating set, we may assume that $S_0 \cap I = 1$. Therefore in S_1, only vertices not containing 1 can be in the dominating set. Hence in every vertex in $S_3 \cap I$ contains a 1. Therefore $S_4 \cap D$ must contain 1, and $S_5 \cap D$ must contain only vertices which have a 1. So $S_4 \cap D = S_{4+2j} \cap D = 1$ and $S_3 \cap D = S_{3+2j} \cap D$ is all subsets which contain 1 for all j such that $3 + 2j \leq h - 2$. At S_{h-2} we have edges from a k set to subsets by inclusion, hence $S_{h-1} \cap D$ is all subsets not containing 1. But these subsets all point to 1 which is assumed to be in the set, a contradiction with D being independent. Therefore no independent dominating set exists. $\square$

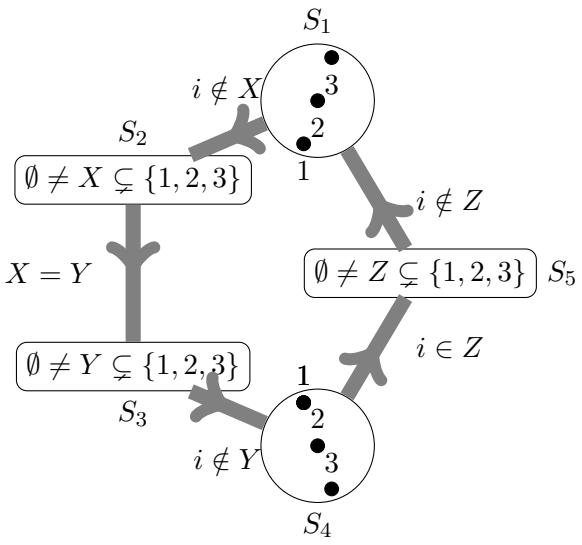

Figure 3.3 The Digraph $D_{5,3}$.

By altering this construction slightly, we instead get a nontrivial family of graphs with odd period which contain independent dominating sets. This shows that only knowing the period of a graph is not sufficient for determining existence of an independent dominating set.

Lemma 7. *In a directed graph D with odd period h and decomposition into independent sets $S_0, \ldots, S_{h-1}$ such that vertices in S_i are adjacent only to vertices in S_{i+1} with addition modulo h, an independent dominating set I is defined entirely by $S_i \cap I$ for any $0 \leq i \leq h-1$.*

Proof. Suppose that we have a digraph D with period h decomposed as in the statement, and we have $S_i \cap I = X$ for some $X \subseteq V(D)$. Notice that the only vertices which can dominate S_{i+1} are vertices of S_i or vertices of S_{i+1}. Therefore, any vertex in $S_{i+1} \setminus N^+(X) \in I$. Hence we have determined $S_{i+1} \cap I = S_{i+1} \setminus N^+(X)$. By the same argument we can now construct $S_{i+2} \cap I$, and taking one step at a time S_{i+k} for any $1 \leq k$. □

Theorem 14. *For each h, there exists an infinite family of graphs $\mathcal{G}$ with period h such that for all $G \in \mathcal{G}$, G has an independent dominating set.*

Proof. We follow the construction in Theorem 6, but instead draw edges by from $u \in S_{h-2}$ to S_{h-1} if and only if $u \notin S$. We note then that the independent dominating set defined by $S_0 \cap D = 1$

is an independent dominating set. In particular, we see that from S_3 onward, we alternate $S_i \cap D$ between the set 1, and the set of all subsets not including one based on parity. □

3.6 Conclusion

In this chapter, we expanded on independent domination theory in directed graphs by providing a generalization of several of Cary, Cary, and Prabhu's original results, by showing that directed graphs with even period have independent dominating sets and allowing anti-parallel edges. We prove that for certain classes of graphs, the existence of independent dominating sets is in P, and provide an exponential time algorithm for the class of graphs with odd period greater than 1. We finally provided constructions of graphs that show that the directed analogue of Vizing's Conjecture for independent dominating sets does not hold.

We determined after this research that independent domination in directed graphs has been of interest in the field of computer science and has a long history. As such, we direct the reader to a survey by Boros and Gurvich [3]. With this discovery, we have found that the existence of independent sets in graphs with even period was proven by Richardson [36]. Additionally, Chvátal proved that determining the existence of an independent dominating set is NP-complete [8] in general but special classes of graphs have been a rich area of study since this result. We note that some results of this research are still best known, including the time complexity result for finding an independent dominating set in graphs with period greater than 1. Additionally, it appears that since kernels were not introduced with respect to independent domination, the discussion of Vizing's Conjecture maintains relevancy, and provides an important avenue for continued research.

There are many significant questions which arise from this research. An important direction of study for the independent domination in directed graphs is the idomatic number. We wonder also under what restrictions an analogue of Vizing's Conjecture that might hold, for example forcing that all graphs and their Cartesian products contain independent dominating sets. Finding classes of graphs which do not satisfy Vizing's Conjecture despite the existence of independent dominating

sets would be a very important result, or determining other additional conditions on the structure of directed graphs under which we can prove Vizing's Conjecture is a rich area for study.

As Cary, Cary, and Prabhu suggest, studying how the reversal or addition of a single edge can alter the idomatic number, which is of interest because of the application of independent domination in ad-hoc networks.

CHAPTER 4. SPLIT DOMINATION IN DIRECTED GRAPHS

4.1 Introduction

A set S will be called a *split dominating* set of G if S is dominating and $V \setminus S$ is disconnected. This was defined in 1997 by Kulli and Janakiram [29]. Continuing in the style of Chapter 3, we will be taking this well studied domination parameter and considering it in the context of directed graphs. First, we must translate the question into digraph terminology. As defined in Chapter 3, for a digraph $D = (V, A)$, $S \subseteq V$ is said to be *dominating* if for every $u \in V$, either $u \in D$ or there exists some $v \in D$ such that $(v, x) \in D$. The hurdle in translating split domination into directed graphs is determining the meaning of disconnected. Fortunately this ambiguity in connectivity of directed graphs is well studied, and we provide several definitions below to understand different concepts of connectedness.

As defined in Chapter 3, a directed graph is said to be *strongly connected* if between any two vertices $u, v \in V(G)$ there exists both a directed path from u to v and a directed path from v to u. A directed graph is said to be *unilaterally connected* if between any two vertices $u, v \in V(G)$ there exists either a directed path from u to v or a directed path from v to u. A digraph is called *strictly unilaterally connected* if it is unilaterally connected but not strongly connected. A directed graph is said to be *weakly connected* if the underlying graph is connected. A digraph is called *strictly weakly connected* if it is weakly connected but not unilaterally connected.

Due to these three distinct notions of connectivity, we see that there is some ambiguity in defining the split domination number of a digraph. We will now instead consider split domination sequences $SDS(G) = (s_1, s_2, s_3)$ where s_1 is the size of a smallest dominating set whose removal makes G unilaterally connected, s_2 the size of a smallest dominating set whose removal makes G weakly connected, and s_3 the size of a smallest dominating set whose removal makes G disconnected. We define the strict split domination sequences $SSDS(G) = (s_1, s_2, s_3)$ analogously, where instead

require strictly unilaterally connected and strictly weakly connected graphs after vertex removal. For each of these parameters, we seek to determine which sequences are possible split domination sequences and iterated split domination sequences.

As an aside, we recall that sequences in this vein have been studied in the past. Namely the domination sequence $(\gamma(G), i(G), ir(G), \alpha(G))$ was studied by Cockayne and Mynhardt [9] where $\gamma(G)$ is the domination number, $i(G)$ is the independent domination number, $ir(G)$ is the irredundance number, and $\alpha(G)$ is the independence number. One of the premier theorems involving this sequence is from Cockayne and Myndhart which states that barring a few pathological examples, all weakly increasing positive integer sequences are domination sequences. In this chapter, we seek the same type of result.

4.2 Strict Split Domination Sequences

In this section we seek to answer the following question:

Question 6. *For what positive integer sequences $s = (s_1, s_2, s_3)$ is s a strict split domination sequence?*

We will first a quick observation which rules out a pathological case.

Observation 3. *No strongly connected oriented graph contains a dominating vertex.*

We notice that by the above observation, there do not exist any strict split domination sequences which have $s_1 = 1$.

To approach this problem, we provide a construction of a directed graph which takes care of many of the possible strict split domination sequences. Consider the following construction, $D_{r,s,t}$. We begin with a bowtie graph and turn the triangles into directed triangles. We blow up the central vertex into a tournament on t vertices. In one of the four remaining vertices, we blow up into a tournament on s vertices, and we blow up the adjacent vertex into an empty graph on r vertices. Blow up the remaining two vertices of the bowtie into complete graphs on rst vertices. See Figure 4.1 for the bowtie structure and $D_{r,s,t}$.

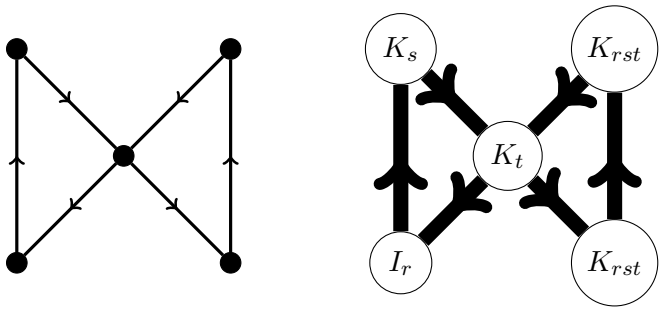

Figure 4.1 The Bowtie and $D_{r,s,t}$

Claim 7. *If $S \subseteq D_{r,s,t}$ has $V \setminus S$ is not strongly connected, then at least one full blow up is a subset of S.*

Proof. By construction, if there exists a vertex from each blow up, we may follow the bowtie structure through the blow ups to reach any other vertex. □

Claim 8. *The size of a minimum dominating set whose removal leaves $D_{r,s,t}$ disconnected is $t+2$.*

Proof. By construction, notice that if the center K_t is removed, the graph is disconnected, and this set dominates I_r and one K_{rst}. It remains to take only two more vertices one in I_r and one in K_{rst}. So there exists a set S whose removal leaves the graph disconnected of size $t+2$.

It remains to show that any disconnecting set is of size at least $t+2$. Clearly if K_{rst} is removed, then we have removed more than $t+2$ vertices, so either K_s or I_r are removed. In either case, the graph is still not disconnected, and we must remove another blob. Say the other of the two is removed, and the graph is still not disconnected. Must remove a full extra blob since inside the blobs are complete graphs, so add at least t vertices to the set, so it is too big. □

Claim 9. *The size of a minimum dominating set whose removal leaves $D_{r,s,t}$ strictly weakly connected is $s+2$.*

Proof. We first show that there exists such a set of size $s+3$. By removing the blob K_s, we note that any two vertices in I_r have no directed path between them, so as long as $r \geq 2$, indeed the graph is

weakly connected. To dominate we may take one vertex in K_t and one in the K_{rst} dominated by K_t. This is a dominating set such a set as long as $t \geq 2$.

We now seek to show that this is indeed the minimum size of such a set. Note that all of K_t cannot be removed, lest the graph be disconnected, and removing an entire K_{rst} is too large. To break strong connectivity we must remove I_r, which is r vertices. Now by the bowtie structure between any two vertices in different blobs there exists a directed path between them. Between any two vertices in the same blob, since each reamining blob is an oriented K_m for some m, there is an edge between the vertices, so more vertices must be removed to leave weak connectivity. Notice that to break either of the above arguments, we must remove an entire blob, that is at least s more vertices must be removed (since K_t cannot be removed without leaving disconnected unless we also remove s). So at least $s + 2$ vertices must be removed. □

Claim 10. *The size of a minimum dominating set whose removal leaves $D_{r,s,t}$ strictly unilaterally connected is $r + 3$.*

Proof. Similar to other cases, removing I_r, a vertex from each of K_t and both K_{rst} is dominating. Also the graph is unilaterally connected since each remaining blob is a complete graph, and the remaining directed paw is unilaterally connected.

Note once again we cannot remove K_t, and K_{rst} is too big, so we consider removing K_s. We see that the graph is not unilaterally connected since each pair of vertices in I_r does not have a directed path between them. Therefore all but one vertex must be removed from I_r, but then we have removed $r - 1 + s$ vertices. Note that $s \geq 2$, so we have removed at least $r + 1$ vertices, but this set is not dominating, so we must remove at least two more vertices from $K_t \cup 2K_{rst}$ to be dominating, removing at least $r + 3$ vertices, as desired. □

Theorem 15. *All integer sequences $\{(s_1, s_2, s_3) | s_1 \geq 5$ and $s_2, s_3 \geq 4\}$ are strict split dominating sequences.*

Proof. By the claims above, $D_{r,s,t}$ is a graph which has strict split domination sequence $\{r, s, t\}$ for all $r \geq 5, s \geq 4$, and $t \geq 4$. □

4.3 Conclusion

In this chapter we build on the study of split domination in directed graphs which has largely been studied in the context of special families of graphs in works similar to that of Factor, Langley, and Merz [5, 17]. We introduced the strict split dominating sequence and we provided a partial answer to the following question:

Question 7. *What integer sequences* (s_1, s_2, s_3) *are strict split dominating sequences?*

A confirmation or refutation of the following conjecture would require only a few special cases to be handled, namely constructions for $r < 4$, $s < 3$, or $t < 3$ with the other two parameters left arbitrary. This is the most natural conclusion to the question above, and would be interesting to pursue. Hopefully there exists some more efficient construction which takes care of all cases (including those answered by this chapter) but multiple different constructions may be necessary.

Conjecture 4. *All integer sequences* $\{(s_1, s_2, s_3) | s_1, s_2, s_3 > 1\}$ *are strict split domination sequences.*

We also note that the definition of strict split domination sequence is not the only one that would make sense depending on our applications in a communication network. We may, for example be interested only in reducing the level of connectivity below a certain threshold, rather than precisely maintain a certain connectivity. Namely, we define the weak *split domination sequence* of a strongly connected directed graph G as the sequence $\{s_1, s_2, s_3\}$ where s_1 is the minimum number of vertices that must be removed to make D not strongly connected, s_2 the minimum number of vertices that must be removed to make D not unilaterally connected, s_3 the minimum number of vertices that must be removed to make D not weakly connected. There is a quick observation to be made about weak split domination sequences.

Observation 4. *All weak split domination sequences are weakly increasing.*

Another potentially valuable variation of this concept would be an iterated version of the same procedure. We may view the iteration a few ways. One perspective is that the vertex removal occurs

in stages, beginning with removing a smallest vertex set to make the graph unilaterally connected, then from the remaining graph removing a vertex set to leave a weakly connected graph, then finally removing vertices to create a disconnected graph. The other view of this is to create a sequence of vertex sets S_1, S_2, S_3 such that $S_1 \subseteq S_2 \subseteq S_3$ and $D \setminus S_1$ is unilaterally connected, $D \setminus S_2$ is weakly connected, and $D \setminus S_3$ is disconnected. With this second view, we may be interested in finding the smallest set at each step, or we may be interested in finding the smallest sum $|S_1| + |S_2| + |S_3|$.

CHAPTER 5. FLAG ALGEBRAS AND INDUCIBILITY OF NETS

This chapter will serve as a light introduction to the Flag Algebra method, and in doing so we will prove the following theorem:

Theorem 16. *For graphs on 6^k vertices, the unique maximizer of the density of nets is the iterated blow up of a net.*

Recall that this research began as an attempt to find a fractalizer. We will begin the discussion with a short observation that the net is not a fractalizer, then continue to prove Theorem 16 to show that intuitively the net is "close to" a fractalizer. The techniques used in this paper are similar to those used in papers by Balogh, Hu, Lidický, and Pfender [2] in which they confirmed a conjecture of Pippenger and Golumbic [33] on which graphs maximize the density of C_5. There has been other recent interest in determining the density maximizers for various small graphs, particularly on graphs on less than or equal to five vertices [16, 26]. Some of the constraints of determining maximizers are largely due to the limits of flag algebras. In the next section we will discuss the method of flag algebras, and some of the limitations, as well as provide an outline of how the proof of Theorem 16 will proceed.

5.1 Introduction

Recall that a fractalizer is a graph G such that the unique maximizer of density of G on graphs of order n is a balanced iterated blow up of G for all n. Fox, Huang, and Lee [19] proved that almost all graphs are fractalizers using random graphs, but the only known constructions of fractalizers are the empty graphs and complete graphs, both trivial cases. It has been shown that no other graphs on at most five vertices are fractalizers. For this reason, we begin the search among graphs on 6 vertices with the net graph. Our first observation will unfortunately end the hopes that the net is

a fractalizer, but we will continue to prove that in some way, the net does behave like a fractalizer once n is sufficiently large.

For the remainder of this chapter, we will focus on the net, and will refer to the vertices in the triangle as either inner vertices or triangle vertices. We will refer to the vertices of degree one as either outer vertices or pendant vertices.

Observation 5. *The net is not a fractalizer.*

Proof. We construct a graph G starting with K_4 and for each $v \in V(K_4)$ we add one vertex p_v that is only adjacent to v. We claim that G has the same number of induced nets as a balanced blow up of the net.

First we count the number of induced nets in G. To construct the inner triangle of the net, we must choose three vertices from the induced K_4, giving $\binom{4}{3}$ options. Then each pendant vertex of the net is entirely determined by the vertices chosen from K_4. This gives 4 induced nets.

We now seek to compare this number to one of the balanced iterated blow ups of the net on 8 vertices. This means that we may choose exactly two vertices two vertices to duplicate, and for ease of counting, we will choose to duplicate two distinct pendant vertices, to create the graph G' see Figure 5.1. We note that there is only one triangle in G', so it must be chosen to induce a net. One pendant vertex is forced by the triangle vertex with only one remaining unchosen vertex, and the remaining two triangle vertices must get a pendant and each has two choices giving 4 unique nets. Therefore G and G' have the same number of induced nets.

It remains to see that G is not a balanced iterated blow up of the net. We note that when duplicating vertices from the net, the K_4 can only be created by duplicating a triangle vertex, but we see that none of the vertices outside of the K_4 have two neighbors in the K_4. Hence G is not a balanced iterated blow up of the net, and we have shown that the net is not a fractalizer. □

We will now begin our discussion of the Flag Algebra method to better understand why the standard applications of the method would not be able to determine that the net is not a fractalizer. The Flag Algebra method was developed by Razborov [34] in 2007. It has since been used for

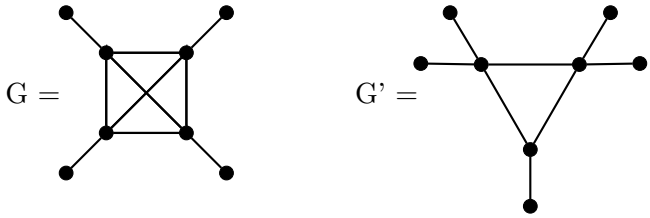

Figure 5.1 The Graphs G and G'

many applications including (but not limited to), discrete geometry, graph colorings, permutations, hypergraphs, and Ramsey theory [39, 28, 35]. At its heart, the Flag Algebra machinery is a tool for helping to solve extremal combinatorics problems. To see some of these results applications and a survey of the method, we refer the reader to a paper of Razborov [35]. The aforementioned results of Balogh, Hu, Lidický, and Pfender are also based on the method of flag algebras. Our primary goal now is to define the algebras $\mathcal{A}^\sigma$ for which flag algebra gets its name. Much of the following discussion follows the descriptions of flag algebra due to Razborov [34] and Volec [41].

First we build to define an algebra $\mathcal{A}$ which allows us to define addition and multiplication functions on graphs so that we can calculate linear combinations of graphs. Let $\mathcal{F}_\ell$ be the set of all distinct graphs on ℓ vertices, and $\mathcal{F}$ to be the set of all distinct graphs. We will treat $\mathcal{F}_\ell$ as though it has been given some ordering, but the ordering is arbitrary. To formalize the intuition that densities of subgraphs H in a graph G can be found by randomly selecting $|V(H)|$ vertices from G and checking if they induce a graph isomorphic to H we introduce a few more definitions. Let $p(H, G)$ be the probability that $|V(H)|$ randomly chosen vertices of G induce a graph isomorphic to G. We note that in a graph G on n vertices, if we choose a graph H on $k \leq n$ vertices we see that the probability of randomly selecting k vertices gives a graph isomorphic to H can be calculated in two ways. Either the value $p(H, G)$ could be calculated or we could calculate $\sum_{H' \in F_k} p(H, H')p(H', G)$ for some $|V(H)| \leq k \leq |V(G)|$. One interpretation of this second calculation is first we pick a graph of an intermediate size, and then sample from the smaller graph to find a copy of H. To capture this intuitive notion of finding the same value, we introduce the subspace $\mathcal{K}$ of $\mathbb{R}\mathcal{F}$ as the subspace

generated by all linear combinations of the form

$$H - \sum_{H' \in \mathcal{F}_{|V(H)|}} p(H, H') \cdot H'$$

for all H, H' in $\mathcal{F}$. Notice that this expression only looks at graphs on one more vertex than our H, but using linear combinations we may fill in any gaps for larger graphs. With this definition, we can factor out $\mathcal{K}$ in $\mathbb{R}\mathcal{F}$ to sensibly define equivalence, calling this class corresponding to the zero of $\mathbb{R}\mathcal{F}$. This equivalence formalizes the intuition above, that we can find the probability of sampling a graph through taking intermediary samples. Let this factored space be called $\mathcal{A}$.

We seek now to create an algebra out of $\mathcal{A}$ by defining addition, multiplication by a real number, and multiplication of two elements of $\mathcal{A}$. Our multiplication by a real number is naturally defined by distributing it to all terms of any linear combination, and addition similarly as addition modulo $\mathcal{K}$. It remains to define a product of two elements of $\mathcal{A}$. The intuition behind the product is relatively natural, that is the product of two graphs $H_1 \times H_2$ can be calculated in a graph G as the probability of selecting $|V(H_1)|$ vertices which are isomorphic to H_1 and simultaneously selecting $|V(H_2)|$ vertices which are isomorphic to H_2 such that the vertex sets are disjoint which we will write as $p(H_1, H_2; G)$. To write this multiplication formally, we define

$$H_1 \times H_2 = \sum_{H \in F_{|V(H_1)|+|V(H_2)|}} p(H_1, H_2; H) H.$$

This captures our intuition if we think of H in the equation as the probability of selecting H from some large graph G. We are now equipped with notions of all of the necessary additions and multiplications, and as proven by Razborov [34] these definitions extend uniquely to operations on the algebra $\mathcal{A}$.

We now seek to define the desired algebra $\mathcal{A}^\sigma$. A labelled graph is a graph with its vertices assigned unique integers. We will refer to these labelled graphs as *types*, often denoted σ. The idea behind types is to fix certain vertices that must appear in the sampled set when sampling vertices for subgraphs and force all other vertices to be chosen from the non-labelled vertices. When drawing labelled vertices in types, we will denote them with ■. For example, we may now count the degree of a vertex in a graph by using these types. We now define analogously to how $\mathcal{A}$

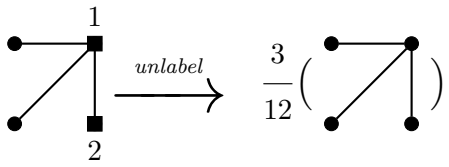

Figure 5.2 An Example of a Type and Unlabeling

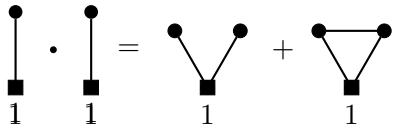

Figure 5.3 A Multiplication of Two Types

was constructed the set of graphs $\mathcal{F}^\sigma$, linear combinations of graphs $\mathcal{K}^\sigma$, and algebra $\mathcal{A}^\sigma$ and with the similar operations. The one peculiarity that arises is how multiplication of two graphs in $\mathcal{A}^\sigma$. In Figure 5.3, we see how the multiplication of two edges with one labelled vertex is calculated. Intuitively we can consider this multiplication as first ensuring that the labeled vertices are placed in a way that induces the type, and then finding the probability of sampling the two graphs as before.

Our goal with constructing these labelled graphs was to create inequalities regarding nonlabelled graph densities. We now mention how to convert back from a labelled graph to an unlabelled through a process called *averaging* or *unlabelling*. We can unlabel graphs by taking advantage of the following idea. On one hand we may count the total number of subgraphs isomorphic to a given graph by using $\mathcal{A}$. On the other hand we may calculate it by using $\mathcal{A}^\sigma$ by first fixing σ in a host graph, then summing across all possible vertices that could have been chosen for σ. With this observation, we can set up equalities between elements of $\mathcal{A}^\sigma$ and $\mathcal{A}$, as demonstrated in Figure 5.3.

Finally, we will be using the positive semidefinite method in each of our proofs. This is a method for creating inequalities which must be true in extremal examples given some conditions. The flag algebra method can be used with the open source software Flagmatic [18]. The construction of

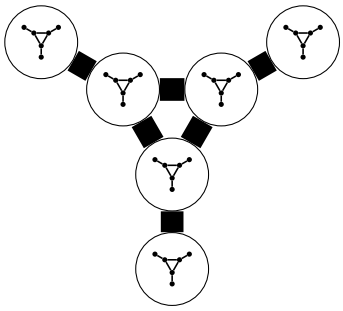

Figure 5.4 The Once Iterated Blow Up of the net $N^{1\times}$

inequalities using the positive semidefinite method has been largely automated, and an in depth discussion on the method can be found in the Ph.D thesis of Volec [41].

5.2 Nets

The following chapter is joint work with Michael Phillips. Recall that a fractalizer is a graph whose induced density is maximized only be iterated blow ups of itself, but in Chapter 1 we proved that the net is not an example of such a graph. In this section we will prove that in some sense, the net is close to a fractalizer. We will prove the following theorem:

Theorem 17. *For $k \geq 1$, among all graphs of size 6^k the unique maximizer of density of nets is the balanced blow up of the net.*

Our theorem largely follows the method of proof introduced by Balogh, Hu, Lidický, and Pfender [2], with some alterations that will be noted as we arrive at them. The basic idea of the proof is as follows. We first use Flagmatic and the positive semidefinite method to determine an upper bound on the possible density of nets in any graph. Since the conjectured extremal construction is iterated, Flagmatic will give us an imperfect bound (that is, the bound will be slightly larger than our construction). To lower this bound, we can instead check linear inequalities in $\mathcal{A}$ utilizing larger graphs which contain the net. Using bound from these larger graphs, we can employ stability arguments involving discrete optimization methods to determine that certain structures must (or must not) exist in any extremal graph. This will allow us to narrow down (for large graphs) the

potential extremal examples to the desired construction. In particular, we will use information on the density of two classes of graphs, $\mathcal{N}_3$ and $\mathcal{N}_{22}$. $\mathcal{N}_3$ is the class of all graphs in which a vertex of the net has been duplicated twice, which is to say that two new vertices are added to a net with the same adjacencies as a vertex already in the net. This class includes all possibilities of edges between these three vertices, and all possible vertices that can be duplicated (up to isomorphism). $\mathcal{N}_{22}$ is defined similarly, but with two distinct vertices being duplicated. Using an inequality involving these two classes, we can show that the top level structure of the graph must be close to a net. We then continue with several arguments to show with stability arguments that if an extremal graph has such a structure close to a net, the structure must agree with a blow up of a net exactly on the top level. From there, we use discrete optimization to show that each of these top level pieces must be balanced. From there we note that any extremal example which which matches the iterated blow up of the net on the top level must indeed be the iterated blow up.

5.3 Proof of Theorem 16

As discussed above, we begin with the basic flag algebra method to obtain the following result. From now on, we will refer to the density of the net as N and the density of the classes $\mathcal{N}_3$ and $\mathcal{N}_{22}$ as N_3 and N_{22} respectively.

Claim 11. *There exists n_0 such that every extremal graph on at least n_0 vertices satisfies that $N \geq \frac{24}{1555}$ and*

$$4N_3 - 14.97N_{22} > .00071788399. \tag{5.1}$$

Proof. This claim is a consequence of the plain flag algebra method. We ran Flagmatic on 8 vertices which verified $N \geq \frac{24}{1555}$ with certificate.

For the second inequality we minimize the difference $4N_3 - 15.96N_{22}$ subject to the constraint that $N > d_{N'}(N)$ where N' is the limit of an infinitely iterated blow up of the net. We add this constraint since we know that any extremal example must have at least as many nets as our conjectured graph. In particular for large enough graphs, the iterated blow up has a vanishing

proportion of nets in its innermost layers, so we anticipate that the limit would give a reasonable bound for flag algebras to use. □

To know what we are shooting for, it might be nice just to calculate the density of nets in the iterated blow up. We will count the number of nets. To find a net, one must either select one vertex from each part, or select all vertices from the same part. We have that

$$d(N)\binom{n}{6} = \left(\frac{n}{6}\right)^6 + 6d(N)\binom{\frac{n}{6}}{6} \xrightarrow{n\to\infty} d(N) = \frac{6!}{6^6 - 6}. \tag{5.2}$$

To compare with the second inequality, we would like to look at N_3 and N_{22} in the iterated blow up. To do so, we use the following calculation

$$N_{22}\binom{n}{8} = \binom{6}{2}\binom{\frac{n}{6}}{2}^2\left(\frac{n}{6}\right)^4 + 6N_{22}\binom{\frac{n}{6}}{8} \xrightarrow{n\to\infty} N_{22} = \frac{8! * 15}{4(6^8 - 6)} \tag{5.3}$$

$$d(N_3)\binom{n}{8} = 6\binom{\frac{n}{6}}{3}\left(\frac{n}{6}\right)^5 + 6d(N_3)\binom{\frac{n}{6}}{8} \xrightarrow{n\to\infty} N_3 = \frac{8!}{(6^8 - 6)} \tag{5.4}$$

In particular, we have that in our desired limit object, $4N_{22} - 15N_3 = 0$ and

$$4N_{22} - 14.97N_3 \approx .00084019254. \tag{5.5}$$

We now introduce some notation to work toward our stability results. In an extremal example G, let $\mathcal{N}$ be the set of induced nets in G. For any induced net H in G we will define $N_{22}(H)$ and $N_3(H)$ to be the number of copies of N_{22} containing H and N_3 containing H respectively. We will now find the "base" of our structure, by picking a net to be the backbone of the structure that we will show is like an iterated blow up. That is, among all nets in $\mathcal{N}$ we pick one net H such that for all $H' \in \mathcal{N} \setminus H$, $N_{22}(H) - 4.99N_3(H) \geq N_{22}(H) - 4.99N_3(H)$. This will allow us to take advantage of Claim 11. To push to our iterated construction we start to create sets of vertices which match, to some extent, the structure of H. Label the triangle in H as $i_1i_2i_3o_1o_2o_3$ where $i_1i_2i_3$ induces a triangle and $i_jo_j \in E(G)$. Call $\{i_1, i_2, i_3, o_1, o_2, o_3\}$ *skeleton vertices*. We now define sets of vertices, called *blobs* I_1, I_2, I_3, O_1, O_2, and O_3, which act like the vertices of H in the sense that

$$I_1 = \{v \in v(G) | (H \cup v) \setminus i_1 \cong H\}.$$

We define the remaining sets similarly, and note that we may think about this as creating sets of all vertices which are isomorphic to the vertices in H with respect to H. We say that a pair of vertices w_1w_2 which intersects two distinct blobs is *funky* if the skeleton vertices are adjacent but w_1w_2 is a non-edge or vice versa. Formally, this means that $(H \setminus \{v_1, v_2\}) \cup \{w_1, w_2\} \ncong H$ where v_1, v_2 are the skeleton vertices corresponding to the blobs that w_1 and w_2 are in respectively. We will think about this as each funky pair destroying a potential copy of N_{22} in the subgraph induced by vertices in blobs.

Claim 12. *In any graph maximizing the induced density of nets, the following inequalities are satisfied:*

$$0.16579160 < I_i, O_i < 0.16754174, 1 \leq i \leq 3, \tag{5.6}$$

$$x_0 < 0.00165262197319, \tag{5.7}$$

$$f < .00000276. \tag{5.8}$$

Proof. We use Lagrange multipliers and some symmetry arguments to simplify the search space. We may now take advantage of Claim 11 to set up several quadratic programs, in which we will solve the following problems:

- Maximize the proportion of the trash vertices, x_0 (those that do not match the skeleton net)
- Maximize the number of funky pairs f
- Maximize the proportion of vertices in a skeleton blob $i_1, i_2, i_3, o_1, o_2, o_3$
- Minimize the proportion of vertices in a skeleton blob $i_1, i_2, i_3, o_1, o_2, o_3$.

The constraints for each of the programs are that $\sum_{i=1}^{3}(i_i + o_i) = 1$ and the equation from Claim 11. We will demonstrate how to show the lower bound on the proportion of vertices in a skeleton blob. As such, we will be solving the following quadratic program

$$P = \begin{cases} \text{minimize } i_1 \\ \text{subject to } \sum_{i=1}^{3}(i_i + o_i) + x_0 = 1 \\ 2\sum_{k,j\subseteq[3]}(i_k o_j + o_k i_j) - 2f - 4.98\sum_{1\leq\ell\leq 3}(i_\ell^2 + o_\ell^2) > .00071788399 \\ i_k, o_k, f \geq 0 \text{ for all } k \in 1, 2, 3 \end{cases}$$

To simplify this computationally, we notice that in our constraints, there is no distinction between outer and inner blobs. From now on, we will use only one variable x_1 to denote the blob that we are interested in and we can limit our search to find a solution to the program instead where all other blobs are given equal weighting, namely $\frac{1}{5}(1 - x_1 - x_0)$. It is clear to see that since f appears only in the second constraint with a negative coefficient, if there exists a feasible solution with $f > 0$, we may find another feasible solution by setting $f = 0$ since this will increase the left hand side of the constraint. By factoring out x_1 from all terms in the left hand side, we will see also that we have a sum of differences of blob sizes squared, which is minimized when the blobs are equally sized. These two factors allow us to make these simplifications and solve the simpler quadratic program that follows:

$$P' = \begin{cases} \text{minimize } x_1 \\ \text{subject to } x_1 + 5y + x_0 = 1 \\ 5x_1 y + 10y^2 - 2.0 * f - 4.99x_1^2 - 4.99 \cdot 5y^2 > .00071788399 \\ x_1, x_0, y, f \geq 0 \end{cases}$$

This was solved using Sage [38] and we provide the code required in Appendix A.

From this we can derive an upper bound on the funky degree of a vertex in $\cup_{i=1}^{3}(I_i \cup O_i)$. In particular, we have that the funky degree of a vertex in a skeleton blob is bounded above by $1 - (1 - 4.99)x_{min}$ where x_{min} is the lower bound on skeleton blobs found above. □

In the following claim, we are now forced to slightly deviate from the strategies of Balogh, Hu, Lidický, and Pfender. Where they can take advantage of symmetry and C_5 being self-complementary, the net has fewer advantageous properties. To deal with this, we instead give an iterated argument that certain types of funky pairs cannot exist, and use that information to lower the complexity of work for the remaining possible funky pairs. While this is a little longer, it allows us to overcome the fact that some funky pairs do seem to be in a "large" number of nets, but that each pair of this type relied on funky pairs which must be uncommon. Let us identify skeleton blobs as the following six sets of vertices $I1, I2, I3, O1, O2, O3$ where $I1, I2, I3$ are the skeleton blobs corresponding to the triangle vertices of the chosen net and $O1, O2, O3$ corresponding to the pendant vertices such that $O1$ is the skeleton blob corresponding to the pendant of the skeleton net vertex in $I1$, and $O2$ and $O3$ defined similarly as the pendants of the skeleton net vertices in $I2$ and $I3$ respectively. That is, the skeleton net has edges between all vertices starting with I and edges between Oj and Ik vertices if and only if $i = k$.

Claim 13. *There are no funky pairs in* $\cup_{i=1}^{3}(I_i \cup O_i)$.

Proof. Let $uv = \{u, v\}$ be a funky pair in G. We will compare the number of nets in G to the number of nets in G' where G' is a copy of G with the only difference being that $\{u, v\}$ is not funky. We see that in G' any set S of 4 vertices in the blobs not containing u and v induces a net containing u and v unless there is funky pair other than uv in $S \cup uv$. Since we have not yet chosen which blobs u and v are in, to preserve generality we will denote the blobs containing the remaining four vertices as X_i, X_j, X_k, and X_ℓ, with the proportions of these blobs in G' being denoted x_i, x_j, x_k, x_ℓ respectively. Therefore with appropriate choice of i, j, k, and ℓ, we have at least

$$x_i x_j x_k x_\ell n^4 - (d_f(u) + d_f(v)) x_i x_j x_k n^4 - f x_i x_j n^4 = ((x_\ell - ((d_f(u) + d_f(v))))x_k - f) x_i x_j n^4 \quad (5.9)$$

$$\geq ((x_{min} - 2d_f)x_{min} - f) x_{min}^2 n^4 \quad (5.10)$$

$$\geq 0.00069 n^4 \quad (5.11)$$

nets containing uv in G'.

We now seek to count the maximum possible number of nets containing $\{u, v\}$ in G. We first see that there are at most $(x_0/6)n^4$ nets containing uv and a vertex from the trash blob. There are at most $(f/2)n^4$ nets containing uv and another funky pair which does not intersect with uv. Additionally, there are at most $(d_f(u) + d_f(v))/2n^4$ nets containing uv and two vertices which are in a funky pair with at least one of u or v. We now count the nets in which uv is the only funky pair.

First we note a small structure claim that will help simplify the arguments. Let N be a net containing uv such that uv is the only funky pair in G. There exists a path on four vertices in N with no funky pairs. (Recall, if this path was in four different blobs, it would be called blob-induced). Indeed, regardless of which two vertices are chosen from the net, there exists a path of length four containing at most one of those vertices.

We claim that either the P_4 is blob-induced, or contained in exactly one blob. Suppose that the P_4 is not blob induced. Then, there exists a blob which contains at least two vertices of the P_4. We note that this pair has the same neighborhood in the P_4 except in the same blob, but every neighborhood must be unique so all vertices are in the same blob.

Suppose that the P_4 is blob-induced with vertices in $O1, I1, I2$, and $O2$. We seek to place the final pendant vertex, p. Note that p must have degree 0 to the P_4 and can be in at most one funky pair, so p cannot be placed in an inner blob since each inner blob is expected to be adjacent to at least two of the P_4 blobs. Similarly we note that if the triangle vertex, t, corresponding to p is placed in $I3$, p cannot be placed in $O1$ or $O2$, but also p cannot be placed in $O3$, else there would be no funky pairs. Therefore $V(N) \cap I3 = \emptyset$ and $p \in O1 \cup O2$. In either case, we see that p is in a funky pair with one inner vertex. We see that no matter where we place t, to induce a net, t must be in a funky pair, a contradiction with N having at most one funky pair. Hence, there exist no funky nets containing uv and a blob-induced P_4 which intersects four blobs.

Since the P_4 is not blob induced and at least one of the pendants in the P_4 is in no funky pairs in N, so no remaining vertices are blob distance 1 from the P_4. Similarly, there is a triangle vertex with no funky pairs in N, so the final triangle vertex must be within blob distance 1, and therefore

in the same blob. Finally, it remains to place the last pendant to have exactly one funky degree. There are at most $(x_{max}n)^4/12$ ways to place vertices in this way.

Finally, it remains only to count the nets in which there exists exactly one vertex w which is in a funky pair with u, v, or both.

We first count nets involving a funky pair between two inner blobs in order to show that there are no funky pairs in $I1 \cup I2 \cup I3$.

Claim 14. *There exist no funky pairs in* $I1 \cup I2 \cup I3$.

Proof. We will assume that $u \in I1$ and $v \in I2$ and note that since we replace all of our counts with x_{max} and x_{min} which are independent of which blobs we started with, we will have counted the nets for any pair of inner blobs.

Suppose first that w is in a funky pair with both u and v. We now try to place w.

Let $w \in O1 \cup O2$. Suppose, without loss of generality that u is in the inner blob corresponding to w. Then in any net, w and u have a mutual non-neighbor x which must lie in $O3$ Since there is a path from x to u, there exists a vertex in y in $I3$. But y has degree 3 and its neighbor set contains no edges, so we are unable to create a net. So there are no nets with $w \in O1 \cup O2$.

Let $w \in O3$. We see that w must be a triangle vertex, and exactly one of u and v is a triangle vertex. The only mutual neighbors of this pair of triangle vertices are in $I3$. But any vertex in $I3$ would create a C_4, which is not a subgraph of the net. Hence there are no such nets.

Let $w \in I3$. Since $\{u, v, w\}$ is independent, we know that at most one of these vertices is a triangle vertex. Then each remaining triangle vertex must be blob distance at most one from the remaining vertices, so they must be in inner blobs. Also, they must be in different blobs, else we create a C_4 a subgraph. There are three ways to place these triangle vertices, and then the final vertex must be a pendant of the triangle vertex in $\{u, w, v\}$ so it's location is forced to be the outer blob corresponding to it. This creates at most $3d_f x_{max}^3 n^4$ nets.

We now count the nets in which w is in only one funky pair. We will assume that w is in a funky pair with u, and by symmetry we will also obtain a count for w in a funky pair with v. Since

$uv \notin E(G)$, v must be in a P_4 containing no funky pairs. Either the P_4 is entirely in the blob with v or it is blob-induced.

If the P_4 is in one blob, then u is a triangle vertex and v and w are pendant vertices. The final vertex must be the pendant of u and can be either in the outer blob corresponding to u's blob, or in $I2$. This gives at most $(\frac{1}{6} + \frac{1}{2})d_f x_{max}^3 n^4$ nets with the P_4 in one blob.

If the P_4 is blob-induced, v is a triangle vertex, and u is a pendant. We note that the pendant of v must be in $O2$, and the second internal vertex (hence triangle vertex, t_2) in the P_4 is either in $I1$ or $I3$. In the latter case, we have that $w \in I3$, triangle vertex for $u \in I1$, and the final pendant in $O3$. Otherwise, we have $w \in O1$, the triangle vertex of w in $I1$ and the triangle vertex for u in $I3$. Each of these independently adds at most $(4/3)d_f x_{max}^3 n^4$ nets, so accounting for both cases as well as switching the roles of u and v, we have a total of at most $(16/3)d_f x_{max} n^4$ nets.

We have now counted the total number of nets that a funky pair in inner blobs can be in as bounded above by $(3 + 16/3)d_f x_{max}^3 n^4$ nets. So by Claim 12 uv is in at most

$$x_{max}^4/12 + (3 + 16/3)d_f x_{max}^3 n^4 + x_0/6x_{max}^3 + d_f^2 x_{max}^3 + f/2x_{max}^2 < .00069n^4$$

nets, a contradiction with Equation 5.11. □

We now inspect the next type of funky pair that could appear, in hopes to use the information we have gained from Claim 14. We will turn to pairs of vertices in corresponding inner and outer blobs.

Claim 15. *There exist no funky pairs in $Oj \cup Ij$ for each $j \in 1, 2, 3$.*

Proof. We will assume similarly to the previous argument without loss of generality that $u \in O1$ and $v \in I1$.

We begin once again by assuming first that the vertex w is in a funky pair with both u and v. By the Claim 14, we know that $w \notin I2 \cup I3$, hence $w \in O2 \cup O3$. In these positions, u and w have no mutual neighbors (which are not funky pairs) so so u must be the pendant vertex of w since w has degree at least 2. Also v is a triangle vertex, which forces the location of the final

triangle vertex to be the inner blob corresponding to the outer blob containing w. Therefore the final pendant vertex must be placed in the same blob as w. This yields at most $2d_f x_{max}^3 n^4$ nets.

We now count the number of nets containing $\{w, u, v\}$ where w is in a funky pair only with v. Again, since no funky pairs exist in the inner blobs, $w \in O2 \cup O3$. There must exist a P_4 containing no funky pairs that includes u but not v. If this P_4 is entirely contained in $O1$, w is in $O1$, else v has degree 4. Also, u and w are pendants, and v is a triangle vertex missing its pendant still. This final pendant may be placed in any of $O1, I2$ or $I3$. These constructions yield a maximum of $(\frac{1}{6} + 2 \cdot \frac{1}{2}) d_f x_{max}^3 n^4$ nets. If the P_4 is blob induced, u is the pendant of a vertex other than v in $I1$, and it intersects exactly one of $I2$ and $I3$. Therefore v is adjacent to this second triangle vertex, and must itself be a triangle vertex, with w being its pendant. Therefore w must be in the outer blob corresponding to the inner blob with no vertex. These constructions yield a maximum of $d_f x_{max}^3 n^4$ nets. Therefore, in the case that w is only in a funky pair with v, we have at most $\frac{19}{6} d_f x_{max}^3 n^4$ nets.

We now count the nets containing $\{w, u, v\}$ such that w is in a funky pair only with u. Notice that any such net will contain a P_4 which has no funky pairs and contains v. If the P_4 is in $I1$, we see that both v and w are pendants in the P_4, and u must be a triangle vertex without a pendant placed so far. We see that u's pendant can be placed only in $I1$, giving at most $\frac{1}{6} d_f x_{max}^3 n^4$ nets of this form. If the P_4 is blob-induced, then v is a triangle vertex, and its pendant must be in $O1$. Also u must be a pendant with triangle vertex being w and the triangle must be blob induced, so there are at most $2d_f x_{max}^3 n^4$ of this type. In total if w is in a funky pair only with u, there are at most $\frac{13}{6} d_f x_{max}^3 n^4$ nets containing $\{u, v, w\}$.

In total, we have at most $(\frac{19}{6} + \frac{13}{6} + 2) d_f x_{max}^3 n^4$ nets containing uv. Via that same contradiction as before, we find that by Claim 12 uv is in less than $.00069n^4$ nets, a contradiction with Equation 5.11. □

The next case that we will deal with is to show there are no funky pairs between a pair of one outer and one inner blob which are not corresponding.

Claim 16. *There exist no funky pairs in $Oi \cup Ij$ where $i, j \in \{1, 2, 3\}$ and $i \neq j$.*

Proof. We will proceed in the same manner of the last two claims. Without loss of generality, assume $u \in O1$ and $v \in I2$.

We once again begin by counting the nets in which w is in a funky pair with both u and v. By Claims 14 and 15, $w \in O3$. Hence, the triangle of any net containing these vertices is exactly $\{w, u, v\}$ and the pendants of u and w must be in $O1$ and $O3$ respectively. Finally, the pendant of v can be in either $O2$ or $I2$, giving a maximum of $2d_f x_{max}^3 n^4$ nets.

We now count the nets where w is in a funky pair only with u. By Claim 15 we have that $w \notin I1$. Using the previous claims, one can find that there are no nets where $w \in O2$, and at most $d_f x_{max}^3 n^4$ when $w \in I3$. If $w \in I2$, the third neighbor of u must be in $O1$, since a vertex in $I1$ would force w to be a triangle vertex whose pendant cannot be placed. Therefore v and w are triangle vertices, and their pendants must also be placed in $I2$. There are at most $\frac{1}{2} d_f x_{max}^3 n^4$ nets of this type. If $w \in O3$, we note that $I3$ must be empty else we create a C_4. Therefore w is the pendant vertex of u and there is a triangle vertex in $I1$. The pendant for v must be in $O2$, and the final pendant must be placed in $O1$ giving at most $d_f x_{max}^3 n^4$. Therefore, in the case that w is in a funky pair only with u, there are at most $\frac{5}{2} d_f x_{max}^3 n^4$ nets.

To finish the count, we now determine the maximum number of nets where w is in a funky pair only with v. By Claims 14 and 15, $w \in O1 \cup O3$. Suppose first that $w \in O1$. We note that there are no vertices in $I1$, else we find a C_4 subgraph. Therefore, the P_4 which does not contain v and contains no funky pairs is contained entirely in $O1$. Then v's pendant can be placed anywhere in $O2 \cup I2 \cup I3$ giving a maximum of $\frac{3}{2} d_f x_{max}^3 n^4$ nets. We now see that if $w \in O3$, either u or w is in the triangle, but not both and in either case the pendant of the triangle vertex must share the same blob as the triangle vertex, and a third triangle vertex cannot be placed to create a net. Therefore there are at most $\frac{3}{2} d_f x_{max}^3 n^4$ nets of this kind.

Similarly to the above contradictions, we have that uv is in less than $.00069n^4$ nets, a contradiction with Equation 5.11. □

Finally, we can move to the last case of funky pairs, between two outer blobs.

Claim 17. *There are no funky pairs in* $O1 \cup O2 \cup O3$.

Proof. We will follow the same pattern as the previous claims. Suppose without loss of generality that $u \in O1$ and $v \in O2$.

If w is in funky pairs with both u and v, by the previous claims it must lie in $O3$, creating the triangle $\{w, u, v\}$ and there is at most one vertex in an inner blob. There are at most $4d_f x_{max}^3 n^4$ nets of this type.

Suppose w is in a funky pair only with u. We see that the third vertex adjacent to u must be in $O1$ or $I1$, so w must be a neighbor of v. Therefore, $w \in O2$ as well as the pendants of v and w. This gives a maximum number of $2d_f x_{max}^3 n^4$ nets. To account for w in a funky pair only with v, by symmetry we may simply double this number, yielding a total of at most $4d_f x_{max}^3 n^4$ of these types.

Once again, by Claim 12, uv is in less than $.00069n^4$ nets, a contradiction with Equation 5.11. □

Combining Claims 14, 15, 16, and 17, we have demonstrated that no funky pairs exist between any pair of blobs, as desired.

□

We now seek to show that $X_0 = \emptyset$. We begin this process by determining a lower bound for the funky degree of a vertex in X_0 using the extremality of our graph G, by counting the number of nets it must be in versus how many nets it would be in if it were placed in one of our assigned blobs.

Claim 18. *For each $x \in X_0$ if x is added to a skeleton blob, $d_f(x) \geq .024467627389$.*

Proof. We may assume now that all funky pairs contain a specific vertex, x. Let xw be a funky pair. We note that there are six possible ways to place xw. For each case, we get a trivial bound on the number of nets containing both x and another vertex in the trash as $x_0/6$. We will take advantage of the fact that x is in all funky pairs. In particular, each induced net that does not involve an additional trash vertex contains a path on 4 vertices with no funky pairs which we will call a blob-induced path. The arguments are similar to the previous claim, so we provide one

sample argument here (and the rest in an appendix) to find the lower bound for the funky degree of x.

We suppose that both x and w are in distinct outer blobs. If x is acting as a pendant in a net containing xw, then w is adjacent to two vertices and non-adjacent to two vertices in the blob-induced path, hence the path must be placed in the same blob as w since blob-induced P_4 must follow the skeleton's edges as proven in the previous claim. This yields an upper bound of $\binom{x_{max}}{4}$ nets containing xw in a net where x is a pendant. We now seek to count the number of nets where x is a triangle vertex. Suppose that w is a pendant. Then we must place the blob-induced path such that no vertices are adjacent to w. There is only one way to place the vertices in different blobs to achieve this, giving $d_f(x)x_{max}^3$ total nets. Placing all vertices in a single blob which can be non-adjacent to w gives $4\binom{d_f(x)}{2}\binom{x_{max}}{2} + \binom{x_{max}}{4}$ nets. Finally we are left to deal with the case the both x and w are triangle vertices, so we suppose that w is a triangle vertex. Since the blob-induced path containing w and the final triangle vertex follows expected blob structure or is in one blob, the blob-induced path is in the same blob with v. This produces $2d_f(x)x_{max}\binom{x_{max}}{2} + \binom{d_f(x)}{2}\binom{x_{max}}{2} + d_f(x)\binom{x_{max}}{3}$ potential nets.

In total we have that at most $x_0/6 + \frac{1}{12}x_{max}^4 + \frac{7}{6}d_f(x)x_{max}^3 + \frac{3}{2}d^2x_{max}^2 + dx_{max}^3$ nets containing xw in G.

We now consider the number of nets that the pair xw would be in if it followed blob structure, and we call this altered graph G'. We may assume without loss of generality that $x \in O_1$ and $w \in O_2$ since all of our bounds will be replacing every blob size with either maximum (or minimum) blob sizes in our counts for bounds. Therefore, xw is in at least $x_{min}^4 - d_f(x)x_{max}^3$ nets in G'.

Since G is extremal, the number of nets in G is greater than or equal to the number of nets in G'. Hence

$$x_{min}^3(x_{min} - d_f(x)) \leq \frac{1}{6}x_0 + \frac{1}{12}x_{max}^4 + \frac{13}{6}d_f(x)x_{max}^3 + \frac{3}{2}d_f(x)^2x_{max}^2$$

and we have that $d_f(x) \geq 0.0261504452636$.

Following similar argumentation and counting, we arrive at Table 5.3 for our six possibilities.

Therefore, we have that $d_f(x) \geq .024467627389$. □

Table 5.1 Minimum Funky Degrees

x	w	Upper Bound for Nets	$d_f(x)$
O_1	O_2	$\frac{1}{6}x_0 + \frac{1}{12}x_{max}^4 + \frac{13}{6}d_f(x)x_{max}^3 + \frac{3}{2}d_f(x)^2x_{max}^2$	0.0261504452636
O_1	I_1	$\frac{1}{6}x_0 + \frac{1}{4}d_f(x)x_{max}^3 + 2d_f(x)^2x_{max}^2 + \frac{1}{6}d_f(x)^3x_{max}$	0.0338086457944
I_1	O_1	$\frac{1}{6}x_0 + \frac{19}{6}d_f(x)x_{max}^3 + \frac{1}{4}d_f(x)^2x_{max}^2 + \frac{1}{6}d_f(x)^3x_{max}$	0.0244676473884
O_1	I_2	$\frac{1}{6}x_0 + \frac{1}{12}x_{max}^4 + 2d_f(x)x_{max}^3 + \frac{1}{4}d_f(x)^2x_{max}^2$	0.0288442703582
I_2	O_1	$\frac{1}{6}x_0 + \frac{1}{24}x_{max}^4 + \frac{5}{2}d_f(x)x_{max}^3 + \frac{3}{2}d_f(x)^2x_{max}^2$	0.0257091074498
I_1	I_2	$\frac{1}{6}x_0 + \frac{5}{2}d_f(x)x_{max}^3 + \frac{1}{4}d_f(x)^2x_{max}^2$	0.0290639784666

Claim 19. *Every vertex in the extremal graph G is contained in at least $(24/1555 + o(1))\binom{n}{5} \approx .00012861736n^5$ nets.*

Proof. Let N^u be the number of nets containing a vertex u of G. Since G is extremal, it must contain at least as many nets as the iterated blow up, namely $(24/1555 + o(1))\binom{n}{6}$ nets. From this, we have that the average number of nets that a vertex is contained in in G is

$$\bar{N} = \frac{\sum_{v \in V(G)} N^v}{|V(G)|} \geq \frac{6}{n}(24/1555 + o(1))\binom{n}{6} \geq (24/1555 + o(1))\binom{n}{5}.$$

Let $u, v \in V(G)$ and N^{uv} be the number of nets in G containing both u and v. We note that $N^{uv} \leq \binom{n-2}{4} = O(n^4)$. Consider the graph G' constructed by removing v and duplicating u. Since G was extremal, we have that

$$0 \geq N(G') - N(G) \geq N^u - N^v - N^{uv} \geq N^u - N^v - O(n^4). \tag{5.12}$$

Since there exists some vertex $x \in V(G)$ in at least $(24/1555 + o(1))\binom{n}{5}$, we consider replacing a vertex in the least number of nets in G with a duplicate of x. In particular from the Equation 5.12, we see that that unless y was in $(24/1555 + o(1))\binom{n}{5}$ nets, we obtain a contradiction, hence all vertices must be in at least $(24/1555 + o(1))\binom{n}{5}$ nets as desired. □

Claim 20. *The set X_0 is empty.*

Proof. The primary technique used in this proof is brute force. In particular, we create a mesh the possible sizes of X_0 as a proportion of the entire graph, bound the derivative of the number of nets that the graph can contain, and run some code to determine that there do not exist vertices in any extremal example with sufficiently high funky degree to be in X_0. See the code in Appendix B.

Suppose that there exists some vertex $x \in X_0$ such that $d_f(x) > 0$ for each blob that x could be placed in. Let $X_1 = I1, X_2 = I2, X_3 = I3, X_4 = O1, X_5 = O2$, and $X_6 = O3$. Let $a_i n$ be the number of neighbors of x in X_i and $b_i n$ be the number of non-neighbors of x in X_i for each $i \in \{1, 2, 3, 4, 5, 6\}$. Normalizing by n^5 we will let A be the number of nets containing x and vertices in 5 distinct other blobs, B be the number of nets containing x and 5 vertices in the same blob, and C be the number of nets containing x and any other non-trash vertices not already counted. Finally, let D be the (normalized) number of nets containing x and any trash vertices.

We have that:

$$A \leq b_2b_3a_4b_5b_6 + b_1b_3b_4a_5b_6 + b_1b_2b_4b_5a_6 + a_1b_2b_3a_5a_6 + b_1a_2b_3a_4a_6 + b_1b_2a_3a_4a_5,$$

$$B \leq \sum_{i=1}^{6} \left(\frac{a_i^3 b_i^2}{3!2!} + \frac{a_i b_i^4}{1!4!} \right), \text{ and}$$

$$C \leq \frac{1}{2!2!} \sum_{i=1}^{3} a_i^2 b_i^2 \left(-a_i - a_{i+3} + \sum_{1 \leq j \leq 6} a_j \right) + \frac{1}{2!2!} \sum_{i=4}^{6} a_i^2 b_i^2 (a_1 + a_2 + a_3 - a_{i-3}).$$

We seek to show the following program is bounded above by
$(24/1555)/5! = 1/7775 \approx 0.000128617363$:

$$(P)\begin{cases} \text{maximize} & A+B+C+D \\ \text{subject to} & \sum_{i=0}^{6}(a_i+b_i)=1 \\ & (x_{min}) \le a_i+b_i \le (x_{max}) \text{ for } i \in [6], \\ & a_0+b_0 \le x_0, \\ & a_2+a_3+b_4+a_5+a_6 \ge 0.0433316, \\ & a_1+a_3+a_4+b_5+a_5 \ge 0.0433316, \\ & a_1+a_2+a_4+a_5+b_6 \ge 0.0433316, \\ & b_1+a_2+a_3+b_5+b_6 \ge 0.0322447, \\ & a_1+b_2+a_3+b_4+b_6 \ge 0.0322447, \\ & a_1+a_2+b_3+b_4+b_5 \ge 0.0322447, \\ & a_i, b_i \ge 0 \text{ for } i \in \{0,1,\ldots,6\}. \end{cases}$$

However, we do not have a form for D. To alleviate this problem, we will "place" trash vertices into every blob simultaneously pretending that x is both adjacent and non-adjacent to every vertex

in X_0. So we relax the above problem the following:

$$(P') \begin{cases} \text{maximize} & f = A + B + C \\ \text{subject to} & \sum_{i=1}^{6}(a_i + b_i) = 1 \\ & (x_{min}) \leq a_i + b_i \leq (x_{max} + x_0) \text{ for } i \in [6], \\ & a_2 + a_3 + b_4 + a_5 + a_6 \geq 0.0433316, \\ & a_1 + a_3 + a_4 + b_5 + a_5 \geq 0.0433316, \\ & a_1 + a_2 + a_4 + a_5 + b_6 \geq 0.0433316, \\ & b_1 + a_2 + a_3 + b_5 + b_6 \geq 0.0322447, \\ & a_1 + b_2 + a_3 + b_4 + b_6 \geq 0.0322447, \\ & a_1 + a_2 + b_3 + b_4 + b_5 \geq 0.0322447, \\ & a_i, b_i \geq 0 \text{ for } i \in \{0, 1, \ldots, 6\}. \end{cases}$$

We will discretize the space of possible solutions to (P'), determine the the value of the objective function at the center of each cell, and use a bound on the gradient to show that the function is bounded above by 0.0001275 in each cell. If the global bound on the gradient is not sufficient in bounding the optimization function, we generate a bound on the gradient within the cell, and if necessary, refine the discretization within the cell. For every a_i and b_i, we check $s + 1 = 51$ equally spaced values between 0 and $x_{max} + x_0$ that include the boundaries. By this, we have a grid of s^{12} boxes where every feasible solution of (P'), and hence of (P), is in one of the boxes.

To determine a global bound on the gradient, we find the partial derivatives of f:

$$\frac{\partial f}{\partial a_1} = b_2b_3a_5a_6 + \frac{1}{24}b_1^4 + \frac{1}{4}a_1^2b_1^2 + \frac{1}{2}(a_1b_1^2)(a_2 + a_3 + a_5 + a_6) + \frac{1}{4}(a_2^2b_2^2 + a_3^2b_3^2 + a_5^2b_5^2 + a_6^2b_6^2)$$
$$\frac{\partial f}{\partial a_4} = b_2b_3b_5b_6 + b_1a_2b_3a_6 + b_1b_2a_3a_5 + \frac{1}{24}b_4^4 + \frac{1}{4}a_4^2b_4^2 + \frac{1}{4}(a_2^2b_2^2 + a_3^2b_3^2) + \frac{1}{2}(a_4b_4^2)(a_2 + a_3)$$
$$\frac{\partial f}{\partial b_1} = b_3b_4a_5b_6 + b_2b_4b_5a_6 + a_2b_3a_4a_6 + b_2a_3a_4a_5 + \frac{1}{6}a_1^3b_1 + \frac{1}{6}a_1b_1^3 + \frac{1}{2}a_1^2b_1(a_2 + a_3 + a_5 + a_6)$$
$$\frac{\partial f}{\partial b_4} = b_1b_3a_5b_6 + b_1b_2b_5a_6 + \frac{1}{6}a_4^3b_4 + \frac{1}{6}a_4b_4^3 + \frac{1}{2}a_4^2b_4(a_2 + a_3).$$

We bound each of these partials by $\frac{4}{3}(x_{max}+x_0)^4$. Since the portion of the partial contributed by A can be bounded above by $(x_{max}+x_0)^4$ in each case, and therefore the rest of the partial can be bounded by $(1/3)(x_{max}+x_0)^3$. One should note that the partial taken with respect to b_1 is the closest to meeting our bound, while the others are closer to $\frac{7}{6}(x_{max}+x_0)^3$.

As each 12-dimensional cell has side-length $1/s$, the objective function can exceed the value at its center by at most $12 \cdot \frac{1/2}{2} \cdot \frac{4}{3}(x_{max}+x_0)^4$. The local bounds on the gradient are obtained in our algorithm are achieved by substituting $(a_i + (1/t)/2)$ and $(b_i + (1/t)/2)$ into each partial, where t is the side-length of the current cell which may or may not be refined, and we simply take the steepest direction of ascent as our bound to replace $\frac{4}{3}(x_{max}+x_0)^4$.

Using $s = 50$, we successfully bounded the objective function below 0.0001275. With this bound, we have that the maximum number of nets that x can be contained in is less than $.00012861736n^5$, a contradiction with Claim 19.

As the number of nets containing x is bounded away from the average, we have contradicted the existence of a trash vertex which cannot be placed in any blob without creating funky pairs. Therefore, there exists a blob in which x matches the expected edge structure perfectly.

We have proven that every trash vertex can be placed into one of the six blobs without creating any funky pairs with vertices originally in $X_1 \cup \cdots \cup X_6$. Therefore, we simply add each trash vertex into its corresponding blob. At worst, we may have funky pairs involving trash vertices in different blobs, but we simply apply Claim 13 noting that our bounds on $d_f(x)$ and f are even more strict.

□

We have now established that on the top layer, the blobs are all roughly balanced, and there are no vertices in X_0. Furthermore, we have that there are no funky pairs, so the only nets in any extremal graph are between vertices of one of the two forms: either all are in the same blob, or all are in different blobs.

At long last, it remains only to show that the blobs are balanced. This will perfectly establish the outer layer structure of the graph into looking like the iterated blow up of the net for any extremal graph that is sufficiently large. We will prove this using a contradiction, assuming that there are

two blobs with a size difference of two or more vertices. We will use the fact that the density of nets in a graph is monotone increasing, and denote the maximum density of nets in a graph on n vertices as $N(n)$. Let f be a bijection from $\{I_1, I_2, I_3, O_1, O_2, O_3\}$ to $\{X_1, X_2, X_3, X_4, X_5, X_6\}$. Additionally, we introduce notation $y_i := |f^{-1}(X_i)|$.

Claim 21. *For large enough n, $|X_i| - |X_j| \leq 1$.*

Proof. Since our bijection was chosen arbitrarily, it is sufficient to prove the claim for X_1 and X_2. Suppose for contradiction that $y_1 - y_2 \geq 2$. Let $v \in X_1$ where v is in the least number of nets contained in X_1 among vertices in X_1, and $w \in X_2$ where w is in the most number of nets contained in X_2 among vertices in X_2. Since G is assumed to be extremal, we have that $N^v + N^{vw} - N^2 \geq 0$ else we would be able to increase the number of nets by removing v and duplicating w. So we have by monotonicity of $N(n)/\binom{n}{6}$ that

$$\frac{24}{1555} + o(1) \geq \frac{N(y_1)}{\binom{y_1}{6}} \geq \frac{N(y_2)}{\binom{y_2}{6}} \geq \frac{24}{1555} - o(1)$$

Since $y_1 - y_2 \geq 2$,

$$\begin{aligned}
N^v + N^{vw} - N^u &\leq \frac{N(y_1)}{y_1} + y_2y_3y_4y_5y_6 + y_3y_4y_5y_6 - \frac{N(y_2)}{y_2} - y_1y_3y_4y_5y_6 \\
&= \frac{y_2N(y_1) - y_1N(y_2)}{y_1y_2} + (y_2 + 1 - y_1)y_3y_4y_5y_6 \\
&\leq (\frac{24}{1555} + o(1))\frac{y_2\binom{y_1}{6} - y_1\binom{y_2}{6}}{y_1y_2} + (y_2 + 1 - y_1)y_3y_4y_5y_6 \\
&\leq (\frac{24}{1555} + o(1))\frac{1}{6!}(y_1^5 - y_2^5) + (y_2 + 1 - y_1)y_3y_4y_5y_6 \\
&= (\frac{24}{1555} + o(1))\frac{1}{6!}(y_1 - y_2)(y_1^4 + y_1^3y_2 + y_1^2y_2^2 + y_1y_2^3 + y_2^4) - (y_1 - y_2)y_3y_4y_5y_6 + y_3y_4y_5y_6 \\
&= (y_1 - y_2)\left((\frac{24}{1555 \cdot 6!} + o(1))\frac{6n^4}{6^4} - \frac{n^4}{6^4} + o(n^4)\right) + \frac{1 + o(1)}{6^4}n^4 \\
&= (y_1 - y_2)\left(\frac{24}{1555}\frac{6}{6^4}n^4 + o(n^4) - \frac{n^4}{6^4} + o(n^4)\right) + \frac{1 + o(1)}{6^4}n^4 \\
&\leq 2\left(\frac{24}{1555}\frac{6}{6^4}n^4 - \frac{n^4}{6^4} + o(n^4)\right) + \frac{1 + o(1)}{6^4}n^4 \\
&= \frac{48}{1555}\frac{6}{6^4}n^4 - \frac{n^4}{6^4} + o(n^4) \\
&< 0
\end{aligned}$$

which is a contradiction with the extremality of G. Therefore, for any X_i, X_j, $|X_i| - |X_j| \leq 1$ as desired. □

The culmination of all of these claims is the following theorem, which we will use to prove Theorem 16. To more easily state the theorem, we will call a graph G *net partitionable* if $V(G)$ can be partitioned into six parts $X_1, X_2, X_3, X_4, X_5, X_6$ of sizes $x_1, x_2, x_3, x_4, x_5, x_6$ such that there exists a function f from $\{X_1, X_2, X_3, X_4, X_5, X_6\}$ to $\{O_1, O_2, O_3, I_1, I_2, I_3\}$ such that for each $u \in X_i$ and $v \in X_j$, $uv \in E(G)$ if and only if $f(X_i)$ and $f(X_j)$ have the same second index, or $f(X_i), f(X_j) \in \{I_1, I_2, I_3\}$.

Theorem 18. *There exist n_0 such that for all $n \geq n_0$*

$$N(n) = x_1x_2x_3x_4x_5x_6 + N(x_1) + N(x_2) + N(x_3) + N(x_4) + N(x_5) + N(x_6)$$

where $\sum_{i=1}^{6} x_i = 1$ and all $x_i - x_j \leq 1$ for all $i, j \in \{1, 2, 3, 4, 5, 6\}$. Moreover, if G has $N(n)$ induced nets, then G is net partitionable.

We will now complete the proof of Theorem 16 by use of a minimal counterexample and a contradiction argument.

Proof of Theorem 16. Clearly on 6 vertices, the unique maximizer of the net is the net, so the case $k = 1$ is proven. Suppose that there exists a graph G on 6^k vertices with $N(G) = N(6^k)$ not isomorphic to the $(k-1)$ iterated blow up of the net, where $k \geq 2$ is the smallest such integer. Let n_0 be from the statement of Theorem 18.

Suppose G is net paritionable. Since G has 6^k vertices, each set in the net partition must have 6^{k-1} vertices. By the minimality of k, we have that each of these sets maximizes the number nets by being isomorphic to the $(k-2)$-iterated blow up of the net. In particular, G is isomorphic to the $(k-1)$-iterated blow up of the net, a contradiction, so G is not net partitionable.

Let H be an extremal graph on $6^\ell > n_0$ vertices. We create the graph G_1 by replacing each vertex of the $(k-1)$-iterated blow up, $N^{k\times}$ of the net with a copy of H, where each copy shares adjacencies with other copies according to the adjacency of their corresponding vertices in the iterated blow up.

By applying Theorem 18 ℓ times, we see that this is indeed an extremal graph on $6^{k+\ell}$ vertices with $N(G_1) = 6^k N(H) + (6^\ell)^6 N(N^{k\times})$ nets. By running the same process as above, starting with G and replacing each vertex with a copy of H, we get a graph with $N(G_2) = 6^k N(H) + (6^\ell)^6 N(G) \geq N(G_1)$ nets. Therefore G_2 is also extremal. Since $|V(G_2)| = 6^{k+\ell} > n_0$, G_2 is net partitionable by Theorem 18. Note that two vertices of H are placed in the same set in the net partition if and only if their adjacency patterns match on at least $\frac{2}{6}$ of the remaining vertices. Hence, for any copy of H, H' ,two vertices in H' must be in the same set in the net partition. In particular, using the net partition of G_2, we have constructed a net partition of G, a contradiction. Therefore, there is no $k \geq 2$ such that there exists a G on 6^k vertices not isomorphic to an iterated blow up of the net that is extremal. □

5.4 Conclusion

This chapter provides a brief introduction to flag algebra, as well as an application of this method to an extremal problem in graph theory. flag algebra are relatively new but are widely used and for a more in depth study, we refer the reader to Razborov [34]. You can find another nice introductory explanation as well as a more in depth discussion of the positive semidefinite method, in the thesis of Jan Volec [41].

While we hoped originally that the net might be a fractalizer, we prove that this was not the case. To build on this, we proved that in some way, the net is close to a fractalizer with Theorem 16. We employed and altered techniques of Balogh, Hu, Lidický, and Pfender [2] by modifying the stability arguments to allow for more precise bounds on the number of edges which do not follow iterated blow up structures. This method can be applied in many cases where there are not as many symmetries as in the case of the cycle on five vertices that they proved. This idea of treating particularly poorly suited examples can be extended to any part of the stability argument. For example if one wants to show that there are no trash vertices, one could prove that none with a certain range of funky degrees cannot exist, and remove them in stages. One nice potential improvement to the methods of this proof would be to avoid the brute force search used in Claim

20. While it has been reasonably effective for small graphs, larger graphs would be significantly hampered by a search like this. Improvements here would be aesthetically pleasing, but are unlikely to widely expand the range of graphs which can be analyzed using these methods, as the positive semidefinite method with flag algebras limit the size of graphs which can be inspected as well. This idea may be valuable as we continue the search for a fractalizer among small graphs. In particular, an interesting set of graphs to explore to find a fractalizer would be *identity graphs* which are graphs that have a trivial automorphism group. Note that the only known fractalizers have maximum sized automorphism groups. It may be the case that with very few automorphisms, we are left with fewer possible maximizers on a small number of vertices like 8. Furthermore, through random graphs Erdős and Reyni prove that almost all graphs have trivial automorphism group [15]. Perhaps, this is an important property underlying the use of random graphs in the proof of Fox, Huang, and Lee [19]. There are 8 such graphs on six vertices and interestingly on 7 vertices there is exactly one self-dual identity graph which might allow for some simplification in the adaptation of the arguments used in this chapter.

The most obvious remaining question to follow up on this research is to find a nontrivial fractalizer. This would be the most interesting path to pursue, but there are other questions that follow. Our result holds only for large graphs, but it is possible to prove for all graph sizes. That is, we may hope to prove that there are only two maximizers for the net. Namely, if G is a graph which maximizes the number of induced nets, either G iterated blow ups or G is a K_4 with each vertex having a pendant edge. There has been work in this direction by Lidický et al. on C_5 and they have been kind enough to share their manuscript. This may illuminate more general methods to prove these types of results even in graphs which are not as symmetric as C_5.

If one were to try to find a fractalizer, it is known that no graph such graph exists on at most 5 vertices. One could continue to search among graphs of size 6 up to potentially 8 using flag algebra methods as the ones above, but small graphs may contain too many symmetries to avoid sporadic maximizers. It may be easier to construct a large graph which has some nice properties like the random graph which can be demonstrated to be a fractalizer.

BIBLIOGRAPHY

[1] Robert B. Allan and Renu Laskar. "On domination and independent domination numbers of a graph". In: *Discrete Math.* 23.2 (1978), pp. 73–76. ISSN: 0012-365X. DOI: 10.1016/0012-365X(78)90105-X. URL: https://doi.org/10.1016/0012-365X(78)90105-X.

[2] József Balogh et al. "Maximum density of induced 5-cycle is achieved by an iterated blow-up of 5-cycle". In: *European J. Combin.* 52.part A (2016), pp. 47–58. ISSN: 0195-6698. DOI: 10.1016/j.ejc.2015.08.006. URL: https://doi.org/10.1016/j.ejc.2015.08.006.

[3] E. Boros and V. Gurvich. "Perfect graphs, kernels, and cores of cooperative games". In: *Discrete Math.* 306.19-20 (2006), pp. 2336–2354. ISSN: 0012-365X. DOI: 10.1016/j.disc.2005.12.031. URL: https://doi.org/10.1016/j.disc.2005.12.031.

[4] Boštjan Brešar et al. "Vizing's conjecture: a survey and recent results". In: *J. Graph Theory* 69.1 (2012), pp. 46–76. ISSN: 0364-9024. DOI: 10.1002/jgt.20565. URL: https://doi.org/10.1002/jgt.20565.

[5] David E. Brown et al. "Cycle extendability and split domination in tournaments". In: *Congr. Numer.* 230 (2018), pp. 239–247. ISSN: 0384-9864.

[6] Yair Caro and Michael A. Henning. "Directed domination in oriented graphs". In: *Discrete Appl. Math.* 160.7-8 (2012), pp. 1053–1063. ISSN: 0166-218X. DOI: 10.1016/j.dam.2011.12.027. URL: https://doi.org/10.1016/j.dam.2011.12.027.

[7] Michael Cary, Jonathan Cary, and Savari Prabhu. *Independent Domination in Directed Graphs*. 2019. eprint: arXiv:1909.05121.

[8] Vašek Chvátal. "On the computational complexity of finding a kernel". In: *Report CRM-300, Centre de Recherches Mathématiques, Université de Montréal* 592 (1973).

[9] E. J. Cockayne and C. M. Mynhardt. "The sequence of upper and lower domination, independence and irredundance numbers of a graph". In: *Discrete Math.* 122.1-3 (1993), pp. 89–102. ISSN: 0012-365X. DOI: 10.1016/0012-365X(93)90288-5. URL: https://doi.org/10.1016/0012-365X(93)90288-5.

[10] D. G. Corneil and Y. Perl. "Clustering and domination in perfect graphs". In: *Discrete Appl. Math.* 9.1 (1984), pp. 27–39. ISSN: 0166-218X. DOI: 10.1016/0166-218X(84)90088-X. URL: https://doi.org/10.1016/0166-218X(84)90088-X.

[11] C. F. De Jaenisch. *Traité des Applications de l'Analyse Mathématique au Jue des Échecs*. 1862.

[12] W. Duckworth and N. C. Wormald. "Minimum independent dominating sets of random cubic graphs". In: *Random Structures Algorithms* 21.2 (2002), pp. 147–161. ISSN: 1042-9832. DOI: 10.1002/rsa.10047. URL: https://doi.org/10.1002/rsa.10047.

[13] Jean E. Dunbar et al. "Broadcasts in graphs". In: *Discrete Appl. Math.* 154.1 (2006), pp. 59–75. ISSN: 0166-218X. DOI: 10.1016/j.dam.2005.07.009. URL: https://doi.org/10.1016/j.dam.2005.07.009.

[14] P. Erdős. "On a problem in graph theory". In: *Math. Gaz.* 47 (1963), pp. 220–223. ISSN: 0025-5572. DOI: 10.2307/3613396. URL: https://doi.org/10.2307/3613396.

[15] P. Erdős and A. Rényi. "Asymmetric graphs". In: *Acta Math. Acad. Sci. Hungar.* 14 (1963), pp. 295–315. ISSN: 0001-5954. DOI: 10.1007/BF01895716. URL: https://doi.org/10.1007/BF01895716.

[16] Chaim Even-Zohar and Nati Linial. "A note on the inducibility of 4-vertex graphs". In: *Graphs Combin.* 31.5 (2015), pp. 1367–1380. ISSN: 0911-0119. DOI: 10.1007/s00373-014-1475-4. URL: https://doi.org/10.1007/s00373-014-1475-4.

[17] Kim A. S. Factor and Sarah K. Merz. "Split domination in digraphs". In: *Congr. Numer.* 229 (2017), pp. 275–283. ISSN: 0384-9864.

[18] E. R. Vaughan. *Flagmatic (Version 2.0)*. `https://flagmatic.org`. 2020.

[19] J. Fox, H. Huang, and C. Lee. "A solution to the inducibility problem for almost all graph". In: *Manuscript* ().

[20] Georg Ferdinand Frobenius. *Ueber Matrizen aus nicht negativen Elementen*. Königliche Akademie der Wissenschaften, 1912.

[21] Michael R. Garey and David S. Johnson. *Computers and intractability*. Freeman, 1979.

[22] Wayne Goddard and Michael A. Henning. "Nordhaus-Gaddum bounds for independent domination". In: *Discrete Math.* 268.1-3 (2003), pp. 299–302. ISSN: 0012-365X. DOI: 10.1016/S0012-365X(03)00032-3. URL: https://doi.org/10.1016/S0012-365X(03)00032-3.

[23] Wayne Goddard and Michael A. Henning. "Independent domination in graphs: a survey and recent results". In: *Discrete Math.* 313.7 (2013), pp. 839–854. ISSN: 0012-365X. DOI: 10.1016/j.disc.2012.11.031. URL: https://doi.org/10.1016/j.disc.2012.11.031.

[24] Wayne Goddard et al. "On the independent domination number of regular graphs". In: *Ann. Comb.* 16.4 (2012), pp. 719–732. ISSN: 0218-0006. DOI: 10.1007/s00026-012-0155-4. URL: https://doi.org/10.1007/s00026-012-0155-4.

[25] Christopher D. Godsil and Gordon F. Royle. *Algebraic graph theory*. Springer, 2010.

[26] James Hirst. "The inducibility of graphs on four vertices". In: *J. Graph Theory* 75.3 (2014), pp. 231–243. ISSN: 0364-9024. DOI: 10.1002/jgt.21733. URL: https://doi.org/10.1002/jgt.21733.

[27] J. P. Jarvis and D. R. Shier. *Graph-Theoretic Analysis of Finite Markov Chains*. 1996.

[28] Daniel Kráľ, Lukáš Mach, and Jean-Sébastien Sereni. "A new lower bound based on Gromov's method of selecting heavily covered points". In: *Discrete Comput. Geom.* 48.2 (2012), pp. 487–498. ISSN: 0179-5376. DOI: 10.1007/s00454-012-9419-3. URL: https://doi.org/10.1007/s00454-012-9419-3.

[29] Veerabhadrappa R. Kulli and Bidarahalli Janakiram. "The split domination number of a graph". In: vol. 32. Dedicated to the memory of Paul Erdős (Riverdale, NY, 1996). 1997, pp. 16–19.

[30] Peter Che Bor Lam, Wai Chee Shiu, and Liang Sun. "On independent domination number of regular graphs". In: *Discrete Math.* 202.1-3 (1999), pp. 135–144. ISSN: 0012-365X. DOI: 10.1016/S0012-365X(98)00350-1. URL: https://doi.org/10.1016/S0012-365X(98)00350-1.

[31] David F. Manlove. "On the algorithmic complexity of twelve covering and independence parameters of graphs". In: *Discrete Appl. Math.* 91.1-3 (1999), pp. 155–175. ISSN: 0166-218X. DOI: 10.1016/S0166-218X(98)00147-4. URL: https://doi.org/10.1016/S0166-218X(98)00147-4.

[32] Dömötör Pálvölgyi and András Gyárfás. "Domination in transitive colorings of tournaments". In: *J. Combin. Theory Ser. B* 107 (2014), pp. 1–11. ISSN: 0095-8956. DOI: 10.1016/j.jctb.2014.02.001. URL: https://doi.org/10.1016/j.jctb.2014.02.001.

[33] Nicholas Pippenger and Martin Charles Golumbic. "The inducibility of graphs". In: *J. Combinatorial Theory Ser. B* 19.3 (1975), pp. 189–203. ISSN: 0095-8956. DOI: 10.1016/0095-8956(75)90084-2. URL: https://doi.org/10.1016/0095-8956(75)90084-2.

[34] Alexander A. Razborov. "Flag algebras". In: *J. Symbolic Logic* 72.4 (2007), pp. 1239–1282. ISSN: 0022-4812. DOI: 10.2178/jsl/1203350785. URL: https://doi.org/10.2178/jsl/1203350785.

[35] Alexander A. Razborov. "Flag algebras: an interim report". In: *The mathematics of Paul Erdős. II*. Springer, New York, 2013, pp. 207–232. DOI: 10.1007/978-1-4614-7254-4_16. URL: https://doi.org/10.1007/978-1-4614-7254-4_16.

[36] Moses Richardson. "Solutions of Irreflexive Relations". In: *Annals of Mathematics* 58.3 (1953), pp. 573–590. ISSN: 0003486X. URL: http://www.jstor.org/stable/1969755.

[37] M. Rosenfeld. "Independent sets in regular graphs". In: *Israel J. Math.* 2 (1964), pp. 262–272. ISSN: 0021-2172. DOI: 10.1007/BF02759743. URL: https://doi.org/10.1007/BF02759743.

[38] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 6.3)*. https://www.sagemath.org. 2020.

[39] Jakub Sliačan and Walter Stromquist. "Improving bounds on packing densities of 4-point permutations". In: *Discrete Math. Theor. Comput. Sci.* 19.2 (2017), Paper No. 3, 18. DOI: 10.1109/mcse.2017.21. URL: https://doi.org/10.1109/mcse.2017.21.

[40] V. G. Vizing. "The cartesian product of graphs". In: *Vyčisl. Sistemy No.* 9 (1963), pp. 30–43. ISSN: 0568-661X.

[41] J. Volec. "Analytic Methods in Combinatorics". In: *Ph.D Thesis* (2014). URL: http://wrap.warwick.ac.uk/71063/.

[42] M. Yannakakis and F. Gavril. "Edge dominating sets in graphs". In: *SIAM J. Appl. Math.* 38.3 (1980), pp. 364–372. ISSN: 0036-1399. DOI: 10.1137/0138030. URL: https://doi.org/10.1137/0138030.

APPENDIX A. CODE USED FOR CLAIM 12

Find below the code that was used to verify that computations for Claim 12. I would like to thank Bernard Lidický for providing the base code which was used to prove the results of [2], which was easily modified to help solve the problem on nets.

Code

```
import sys
from sage.all import *

# names of variables
vs = var('x1 x2 x3 x4 x5 x6 x0 f lc ls') #  lc ... lambda constraint

# function to optimize
#objective=(x1)
objective=(f)
#objective=(x0)



# extra constraints
addon=",x2==x3,x2==x4,x2==x5,x2==x6"



notfixedsolution = 0  # number of cases where the solution is not just one point
maxf = 0 # maximum value of f found so far in the process
```

```
maxi = 0 # i for which the maximum is reached
maxsolution = [] # contains the best solution so far
minf = 10000 # minimum value of f found so far in the process
mini = 0 # i for which the minimum is reached
minsolution = [] # contains the best solution so far

# We generate number i = 0...1023 and look at its binary representation
#(transformed to array).
# The array has 10 entries and each of them corresponds to one constraint being equality.
# entry 0 means free and 1 means =. The order in the field is
#
# ID 0  1   2   3   4   5
#   lc l1l l1u l2l l2u lfl
#
# Note that we actually do not use l1l l1u l2l l2u lfl -
#they are l1l stands for 'lambda x_1 lower bound',...
$We rather made a substitution directly.



# APMonitor solution

constrcnt = 9 # number of constraints
programs =  pow(2,constrcnt)
binformat = "{0:0"+str(constrcnt)+"b}" # makes something like "{0:012b}"
#avalue = 4.98
#flagbound = 0.0014380452
avalue = 4.99
```

```
flagbound = .00071788399
#avalue = 4.96
#flagbound = .0028783717



for i in range(0,programs):
active = [ int(z) for z in list(binformat.format(i)) ] # creating {0,1} array

# if any of the x_1 == 0, we are doomed anyway
if active[1] == 1 or active[2] == 1 or active[3] == 1 or
active[4] == 1 or active[5] == 1 or active[6] == 1:
continue

# some quick kills.... lower and upper bounds cannot be at the same time
# Building L, where L is the Lagrangian.
#Note that we ALWAYS include all parts of the Lagrangian in L
# and when we actually use it,
#we may disable parts of it by setting say l1 = 0 and not taking
# the partial derivative according to l1 in to the system of equations.
Lstr = "L =  ("+str(objective)+" + lc*(2.0*(x1*x2+x1*x3+x1*x4+x1*x5+x1*x6+x2*x3+
x2*x4+x2*x5+x2*x6+x3*x4+x3*x5+x3*x6+x4*x5+x4*x6+x5*x6)
- 2.0*f - avalue*(x1*x1 + x2*x2 + x3*x3 + x4*x4 + x5*x5 + x6*x6) -
flagbound/(0.0154342*28)) + ls*(x1+x2+x3+x4+x5+x6+x0-1))"
# Now we make the system of equations used in \nabla L
# the following are always there
eqs = "[ L.diff(ls)==0"
# for lambdas we decide if we include them in the gradient or not.
```

```
#The first column
# corresponds to not including (means not equality)
#and the second column means including
if active[0] == 0:
eqs += ",lc == 0";
else:
eqs += ",L.diff(lc) == 0";
if active[1] == 0:
eqs += ",L.diff(x1) == 0"
else:
eqs += ",x1 == 0"
if active[2] == 0:
eqs += ",L.diff(x2) == 0"
else:
eqs += ",x2 == 0"
if active[3] == 0:
eqs += ",L.diff(x3) == 0"
else:
eqs += ",x3 == 0"
if active[4] == 0:
eqs += ",L.diff(x4) == 0"
else:
eqs += ",x4 == 0"
if active[5] == 0:
eqs += ",L.diff(x5) == 0"
else:
eqs += ",x5 == 0"
```

```
if active[6] == 0:
eqs += ",L.diff(x6) == 0"
else:
eqs += ",x6 == 0"
if active[7] == 0:
eqs += ",L.diff(x0) == 0"
else:
eqs += ",x0 == 0"
if active[8] == 0:
eqs += ",L.diff(f) == 0"
else:
eqs += ",f == 0"
eqs += addon+"]"
# Now we have all the equations from gradient in eqs
#so we create a command for solving them
command="solution=solve("+eqs+",(x1,x2,x3,x4,x5,x6,x0,f,lc,ls), solution_dict=True)"
exec Lstr # this should actually be fine do do just once at the beginning
exec command # solving the equations - solution is in variable solution
# Now we check how many solutions we have
if len(solution) == 0:
pass
print i,active,"No solution!!"
#if len(solution) == 1:
#print i,active,len(solution)
for solutionID in range(len(solution)):
# print solutionID
# Now we check if all constraints are satisfied. We verify constraints in the form
```

```
#  g(x) <= 0
constraintsg = [
-(x1), -(x2), -(x3), -(x4), -(x5), -(x6), -(f), -(x0), # r2 >= 0 kind
-(2.0*(x1*x2+x1*x3+x1*x4+x1*x5+x1*x6+x2*x3+x2*x4
+x2*x5+x2*x6+x3*x4+x3*x5+x3*x6+x4*x5+x4*x6+x5*x6)
- 2.0*f - avalue*(x1*x1 + x2*x2 + x3*x3 +
x4*x4 + x5*x5 + x6*x6) - flagbound/(0.0154342*28))
]
feasiblesolution = True
for g in constraintsg:  # test all constraints
try:
value = g.subs(solution[solutionID]) # substituting to g
float(value) # try if it is a number (and no some free variables)
#print g,float(value)
if value > 0.00001: # constraint g <= 0 violated (epsilon tolerance needed)
print i,active,"Violates constraint ",g,"<= 0 evaluated as",float(value),"<=0"
feasiblesolution = False
break
except TypeError:
print i,active,"Not a point solution for constraint"
print solution
notfixedsolution = notfixedsolution + 2
#feasiblesolution = False
#break
pass
if feasiblesolution == False:
continue # do not process the thing further if constraints violated
```

```
# Finally, we try to evaluate f
value = objective.subs(solution[solutionID]) # substituting to f
try:
#float(value)
#valf = float(value)
# if there were free variables, just evaluate them as zeros...
valf=float(value.subs({value.arguments()[x]:0 for x in range(0,len(value.arguments()))}))
# evaluate it
if maxf < valf: # test if we got a better new solution
maxf = valf
maxsolution=solution[solutionID]
maxi = i
if minf > valf: # test if we got a better new solution
minf = valf
minsolution=solution[solutionID]
mini = i
print i,active,"Got value",valf,"min is",minf,"max is",maxf
#print solution[solutionID]
except TypeError:
# this happens if value is not a float - means not unique solution
notfixedsolution = notfixedsolution + 1
print i,active,"Not a float!! cnt: ", notfixedsolution,len(value.arguments())

# final wrap up
print "Summary:"
print 'Number of not floats is',notfixedsolution
print 'mainimim is for mini=',mini,
```

```
'with value minf=',minf,'and solution maxsolution=',minsolution
print 'maximum is for maxi=',maxi,
'with value maxf=',maxf,'and solution maxsolution=',maxsolution



print
print 'Minimum solution: ',minf,'<=',objective
for myvar in vs:
try:
print myvar,'=',float(minsolution[myvar])
except TypeError: # this happens if value is not a float
print myvar,'=', minsolution[myvar]



print
print 'Maximum solution: ',maxf,'>=',objective
for myvar in vs:
try:
print myvar,'=',float(maxsolution[myvar])
except TypeError: # this happens if value is not a float
print myvar,'=',maxsolution[myvar]
```

APPENDIX B. CODE FOR CLAIM 20

Below is the code that was written to create the mesh to prove Claim 20. Many thanks to Michael Phillips for the production of this code and his many ideas on how best to optimize this mesh program. The base of this code was once again provided by Bernard Lidický as it was used for [2], and was modified for the purpose of the net.

Code

```
/*

This solves the program (P') from Claim 8.

Usage:
 g++ mesh-opt.cpp -Wall -O3 -o mesh-opt && ./mesh-opt

If you have OpenMP, you could use
 g++ mesh-opt.cpp -Wall -O3 -o mesh-opt -fopenmp && ./mesh-opt

 */

#include <iostream>
#include <fstream>
#include <sstream>
#include <istream>
#include <vector>
```

```
#include <utility>
#include <assert.h>
#include <cstring>
#include <algorithm>
#include <cstdarg>
#include <cmath>
#include <limits>


using namespace std;


//const double max21 = 0.21;
const int steps = 100; // Number of steps for each Xi in the recursion
(you may also use 200)
//const double stepsize = max21/steps;

const double max1685 = 0.1685;
const double stepsize = max1685/steps;


// const double Xmax = max21 + 2*stepsize; // Upper bound for Xi
const double Xmax = max1685 + 2*stepsize;

//const double funkyDegree = 0.079;
//const double funkyDegree = 0.0808;
```

```
const double funkyDegreeOut = 0.0402; // Check this number
const double funkyDegreeIn = 0.0177; // Check this number

// const int funkyDegreeInt = (funkyDegree*steps)/Xmax - 4;
// the - 4 is to cover all rounding errors
const int funkyDegreeOutInt = (funkyDegreeOut*steps)/Xmax - 4;
// Not sure if 4 needs to change, probs to 5 maybe?
const int funkyDegreeInInt = (funkyDegreeIn*steps)/Xmax - 4;

// const double extra =  5.0*0.21/2.0*0.001/steps;
const double extra = 3*max1685*max1685*max1685*max1685*max1685/steps;

int ext[7];

int main(int argc, char *argv[]) {
double max_total =  0;

  //#pragma omp parallel for ordered schedule(dynamic)
  // pick a[1] to be the smallest
for (int a1 = 0; a1 <= steps; a1++) {
    double max = 0;
    int a[7];
    int b[7];

    // Assuming smallest part is a pendant blob
    //a[1] = a1;
    //for (a[2] = a[1]; a[2] <= steps; a[2]++)
```

```
  //for (a[3] = a[1]; a[3] <= steps; a[3]++)
//for (a[4] = a[1]; a[4] <= steps; a[4]++)
//for (a[5] = a[1]; a[5] <= steps; a[5]++)
//for (a[6] = a[1]; a[6] <= steps; a[6]++)

// Assuming smallest part is a triangle blob
  a[4] = a1;
  for (a[2] = a[4]; a[2] <= steps; a[2]++)
  for (a[3] = a[4]; a[3] <= steps; a[3]++)
  for (a[1] = a[4]; a[1] <= steps; a[1]++)
  for (a[5] = a[4]; a[5] <= steps; a[5]++)
  for (a[6] = a[4]; a[6] <= steps; a[6]++) {
// for (int i = 1; i <= 5; ++i) b[i] = steps - a[i];
  for (int i=1; i <= 6; ++i) b[i] = steps - a[i];

// Kill entires with too few funky edges
    if ((a[2]+a[3]+b[4]+a[5]+a[6] < funkyDegreeOutInt) ||
        (a[1]+a[3]+a[4]+b[5]+a[6] < funkyDegreeOutInt) ||
        (a[1]+a[2]+a[4]+a[5]+b[6] < funkyDegreeOutInt) ||
        (b[1]+a[2]+a[3]+b[5]+b[6] < funkyDegreeInInt) ||
        (a[1]+b[2]+a[3]+b[4]+b[6] < funkyDegreeInInt) ||
        (a[1]+a[2]+b[3]+b[4]+b[5] < funkyDegreeInInt)) continue;
            //(b[2]+b[5]+a[3]+a[4] < funkyDegreeInt) ||
            //(b[1]+b[3]+a[4]+a[5] < funkyDegreeInt) ||
            //(b[2]+b[4]+a[1]+a[5] < funkyDegreeInt) ||
            //(b[3]+b[5]+a[1]+a[2] < funkyDegreeInt) ||
            //(b[4]+b[1]+a[2]+a[3] < funkyDegreeInt)
```

```
// int valueInt = a[2]*a[5]*b[3]*b[4] + a[1]*a[3]*b[4]*b[5] +
a[2]*a[4]*b[1]*b[5] + a[3]*a[5]*b[1]*b[2] + a[4]*a[1]*b[2]*b[3] +
0.25*(a[1]*a[1]*b[1]*b[1] + a[2]*a[2]*b[2]*b[2] +
a[3]*a[3]*b[3]*b[3]+ a[4]*a[4]*b[4]*b[4] +
a[5]*a[5]*b[5]*b[5]);
    // double value = valueInt*Xmax*Xmax*Xmax*Xmax/(double)(steps*steps*steps*steps);

    int valueInt = b[2]*b[3]*a[4]*b[5]*b[6] + b[1]*b[3]*b[4]*a[5]*b[6] +
    b[1]*b[2]*b[4]*b[5]*a[6] + a[1]*b[2]*b[3]*a[5]*a[6] +
    b[1]*a[2]*b[3]*a[4]*a[6] + b[1]*b[2]*a[3]*a[4]*a[5];
    // Here, I'm only including the first 6 terms because HOPEFULLY
    //that's all we need to bound using this method

    //double value = valueInt*Xmax*Xmax*Xmax*Xmax*Xmax/
    (double)(steps*steps*steps*steps*steps);
    // Here, I'm assuming that including an extra factor of Xmax/steps is correct
    double value = valueInt*(Xmax/(double)steps)*(Xmax/(double)steps)*
    (Xmax/(double)steps)*(Xmax/(double)steps)*(Xmax/(double)steps);

if (max < value) {
max = value;
  cout << "Max+extra=" << max+extra << " Max=" << max << " " << a[1] << " " << a[2]
  <<  " " << a[3] <<  " " << a[4] <<  " " << a[5] << " " << a[6] << endl;
  for (int i=1; i<=6; i++) ext[i] = a[i];
}
}
  //#pragma omp ordered {
```

```
    //  cerr << "Done " << a1 << "/" << steps << endl;
    //  if (max > max_total) {
    //    max_total = max;
    //    cout << "Max(P)+extra=" << max+extra << " Max(P)=" << max <<
    " Construction: " << ext << endl;
    //  }
    //}
  }

  cout << "Discretization of (P'')  is bounded by " << max_total << endl;
cout << "Discretization of (P'') is bounded by " << max_total + extra << endl;
cout << "  Construction: (" << ext[1] <<"," << ext[2] << "," << ext[3] << "," << ext[4]
<< "," << ext[5] << "," << ext[6] << ")" << endl;

return 0;}
```