Some Results in Extremal Combinatorics

Rahil Baber

Department of Mathematics, UCL

A thesis submitted for the degree of
Doctor of Philosophy

Supervisor: Dr. John Talbot

March 2011

Declaration

I, Rahil Baber, confirm that the work presented in this thesis is my own, except for those parts stated in the following Authorship page. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

.....

Authorship

Chapter 1 is joint work with J. Robert Johnson and John Talbot. Chapter 2 Sections 2 and 3 are joint work with John Talbot. The remaining work is my own.

Abstract

In Chapter 1 we determine the minimal density of triangles in a tripartite graph with prescribed edge densities. This extends work of Bondy, Shen, Thomassé and Thomassen characterizing those edge densities guaranteeing the existence of a triangle in a tripartite graph. We also determine those edge densities guaranteeing a copy of a triangle or C_5 in a tripartite graph.

In Chapter 2 we describe Razborov's flag algebra method and apply this to Erdős' jumping hypergraph problem to find the first non-trivial regions of jumps. We also use Razborov's method to prove five new sharp Turán densities, by looking at six vertex 3-graphs which are edge minimal and not 2-colourable.

We extend Razborov's method to hypercubes. This allows us to significantly improve the upper bound given by Thomason and Wagner on the number of edges in a C_4-free subgraph of the hypercube. We also show that the vertex Turán density of a 3-cube with a single vertex removed is precisely 3/4.

In Chapter 3 we look at problems for intersecting families of sets on graphs. We give a new bound for the size of G-intersecting families on a cycle, disproving a conjecture of Johnson and Talbot. We also extend this result to cross-intersecting families and to powers of cycles.

Finally in Chapter 4 we disprove a conjecture of Hurlbert and Kamat that the largest trivial intersecting family of independent r-sets from the vertex set of a tree is centred on a leaf.

Contents

Chapter 1

The Minimal Density of Triangles in Tripartite Graphs

1.1 Introduction

Extremal questions for triangles in graphs have a very long history. The first such result, Mantel's theorem [26], tells us that a graph with n vertices and more than $n^2/4$ edges must contain at least one triangle.

For graphs with more than $n^2/4$ edges it is natural to pose a quantitative question: what is the minimum number of triangles in a graph with a given number of edges? In this direction Razborov [30] recently determined that for a fixed $\rho \in [0,1]$, (asymptotically) the minimal possible density of triangles in a graph with edge density ρ is

$$\frac{(t-1)\left(t-2\sqrt{t(t-\rho(t+1))}\right)\left(t+\sqrt{t(t-\rho(t+1))}\right)^2}{t^2(t+1)^2},$$

where $t = \lfloor 1/(1-\rho) \rfloor$. This density is achieved by a complete $(t+1)$-partite graph with t roughly equally sized classes and one smaller class. Razborov's result [30] was the cumulation of decades of contributions on this question due to Bollobás [3], Erdős [9], Lovász and Simonovits [25], and Fisher [15].

Recently Bondy, Shen, Thomassé, and Thomassen [4] considered the very natural question of when a tripartite graph with prescribed edge densities must contain a triangle. (A tripartite graph is a graph $G = (V, E)$ for which there exists a partition of its vertices into three vertex classes such that all edges go

between classes. The edge density between a pair of vertex classes X, Y is simply the proportion of edges present between the two classes: $|E(X,Y)|/|X||Y|$.)

Bondy, Shen, Thomassé, and Thomassen [4] characterized those triples of edge densities guaranteeing a triangle in a tripartite graph. As a special case they showed that a tripartite graph must contain a triangle if the density of edges between each pair of classes is greater than $1/\varphi = 0.618\ldots$ (this comes from considering an optimally blown-up 5-cycle). A precise statement of their full result can be found in the next section.

The main aim of this chapter is to prove a quantitative result which extends the theorem of Bondy, Shen, Thomassé, and Thomassen in the same way that Razborov's result extends Mantel's theorem.

The remainder of the chapter is organised as follows. Formal definitions and the main result are given in the next section. Our main result splits into two rather different cases and the following two sections contain their proofs. We then determine those edge densities guaranteeing a triangle or 5-cycle in a tripartite graph. We finish with some conjectures and open problems.

The proof of our main result relies on a computer search. The C++ source code for this computation is included in Appendix A and can also be found in the accompanying CD-ROM.

1.2 Definitions and results

A tripartite graph is a graph $G = (V, E)$ for which there exists a partition of its vertices into three independent sets. Throughout, whenever we consider a tripartite graph we will implicitly assume that a fixed tripartition $V = A \dot\cup B \dot\cup C$ is given.

A *weighted tripartite graph* (G, w) is a tripartite graph $G = (V, E)$ together with a *weighting* $w : V \to [0, 1]$ satisfying

$$\sum_{a \in A} w(a) = \sum_{b \in B} w(b) = \sum_{c \in C} w(c) = 1.$$

We will see shortly that there is a very natural way of representing tripartite graphs as weighted tripartite graphs and vice versa. We can manipulate a weighted tripartite graph (whilst still retaining the underlying structure) by modifying the weights of its vertices. This will prove to be very useful, and so

rather than work with tripartite graphs we will work with weighted tripartite graphs instead.

For a weighted tripartite graph (G, w), the weight of an edge $xy \in E(G)$ is $w(xy) = w(x)w(y)$. The *edge densities* of (G, w) are

$$\alpha(G, w) = \sum_{bc \in E(B,C)} w(bc), \quad \beta(G, w) = \sum_{ac \in E(A,C)} w(ac), \quad \gamma(G, w) = \sum_{ab \in E(A,B)} w(ab).$$

We denote the set of all weighted tripartite graphs by **Tri**. For $\alpha, \beta, \gamma \in [0, 1]$ we let $\mathbf{Tri}(\alpha, \beta, \gamma)$ denote the set of all weighted tripartite graphs with edge densities $\alpha(G, w) = \alpha$, $\beta(G, w) = \beta$, $\gamma(G, w) = \gamma$.

Let $(G, w) \in \mathbf{Tri}$. A *triangle* in G is a set of three vertices, $a \in A, b \in B, c \in C$, such that $ab, ac, bc \in E(G)$. We denote the set of all triangles in G by $T(G)$. The weight of a triangle $xyz \in T(G)$ is $w(xyz) = w(x)w(y)w(z)$. The *triangle density* of $(G, w) \in \mathbf{Tri}$ is

$$t(G, w) = \sum_{abc \in T(G)} w(abc).$$

Note that with the obvious definitions of edge and triangle densities for simple tripartite graphs any such graph can be converted into a weighted tripartite graph with the same edge and triangle densities by setting the vertex weights to be $1/|A|, 1/|B|, 1/|C|$ for vertices in classes A, B, C respectively.

Also, any weighted tripartite graph with rational weights can be converted into a simple tripartite graph with the same edge and triangle densities by taking a suitable blow-up. To be precise, choose an integer n so that $nw(v)$ is an integer for all vertices v. Then replace each vertex of weight x with nx new vertices which are clones of the old vertex, i.e. we join a pair of vertices in the new graph if and only if the pair of vertices they arise from are adjacent in the weighted graph.

We are interested in how small the triangle density of a weighted tripartite graph with prescribed edge densities can be. Formally we wish to determine the following function. For $\alpha, \beta, \gamma \in [0, 1]$ let

$$T_{\min}(\alpha, \beta, \gamma) = \min_{(G,w) \in \mathbf{Tri}(\alpha,\beta,\gamma)} t(G, w).$$

It follows from later results that this function is well-defined (in particular

Lemmas 1.4.1 and 1.4.11).

Bondy, Shen, Thomassé and Thomassen [4] proved the following sharp Turán-type result. If $(G, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ and $(\alpha, \beta, \gamma) \in R$, where

$$R = \{(\alpha, \beta, \gamma) \in [0,1]^3 : \alpha\beta + \gamma > 1, \alpha\gamma + \beta > 1, \beta\gamma + \alpha > 1\},$$

then G must contain a triangle.

Theorem 1.2.1 (Bondy, Shen, Thomassé and Thomassen [4])**.**

$$T_{\min}(\alpha, \beta, \gamma) = 0 \iff (\alpha, \beta, \gamma) \in [0,1]^3 \setminus R.$$

In particular, $T_{\min}(d, d, d) = 0$ if and only if $d \leq 0.618\ldots$ (the positive root of the quadratic $x^2 + x - 1 = 0$).

The following simple lemma shows that $T_{\min}(\alpha, \beta, \gamma)$ gives an asymptotic answer to the question of how many triangles a simple (unweighted) tripartite graph with given edge densities must have.

Lemma 1.2.2.

(i) If G is a simple tripartite graph with edge densities α, β, γ then it has triangle density at least $T_{\min}(\alpha, \beta, \gamma)$.

(ii) For rational α, β, γ, if $(H, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ then for all $\epsilon > 0$ there is a simple tripartite graph G with edge densities α, β, γ and triangle density at most $t(H, w) + \epsilon$.

Proof. Part (i) is immediate since any tripartite graph can be transformed into a weighted tripartite graph by weighting vertices uniformly in each vertex class as described above.

For part (ii) let w' be a rational weighting of H such that if the edge densities of (H, w') are α', β', γ' we have $|\alpha - \alpha'|, |\beta - \beta'|, |\gamma - \gamma'|, |t(H, w) - t(H, w')| < \frac{1}{4}\epsilon$. We can do this since for a given H the edge and triangle densities are continuous functions of the vertex weights. Now choose an integer n so that $nw'(v)$ is an integer for all vertices v, and $n^2|\alpha - \alpha'|, n^2|\beta - \beta'|, n^2|\gamma - \gamma'|$ are all integers. Blow up H by replacing each vertex v with $nw'(v)$ cloned vertices to form a simple graph G' with n vertices in each class. Finally add or remove at most $\frac{3}{4}\epsilon n^2$ edges from G' to form a graph G with edge densities α, β, γ. This

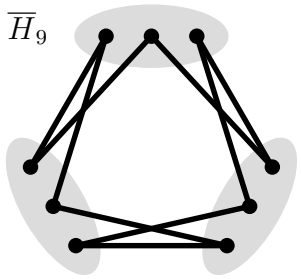

Figure 1.1: The tripartite complement of the graph H_9.

creates at most $\frac{3}{4}\epsilon n^3$ new triangles and so the triangle density of G is at most $t(H, w') + \frac{3}{4}\epsilon < t(H, w) + \epsilon$. □

Our main result (Theorem 1.2.3) determines the minimal density of triangles in a weighted tripartite graph with prescribed edge densities.

In order to simplify diagrams we make the following definition. The *tripartite complement* of a tripartite graph G is the graph obtained by deleting the edges of G from the complete tripartite graph on the same vertex classes as G. We will denote this by $\overline{G}$.

Theorem 1.2.3. *Let H_9 be the graph whose tripartite complement is given in Figure 1.1. For any $(\alpha, \beta, \gamma) \in R$ there exists a weighting w of H_9 such that $(H_9, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ and $t(H_9, w) = T_{\min}(\alpha, \beta, \gamma)$.*

This theorem combined with Lemma 1.2.2 shows that a suitable blow-up of H_9 has asymptotically the minimum density of triangles for given edge densities. Comparing this to Razborov's result [30] it is quite remarkable that only the graph H_9 is needed to describe the extremal examples.

There are two distinct cases to consider in the proof of Theorem 1.2.3, depending on the values of α, β, γ. Let

$$\Delta(\alpha, \beta, \gamma) = \alpha^2 + \beta^2 + \gamma^2 - 2\alpha\beta - 2\alpha\gamma - 2\beta\gamma + 4\alpha\beta\gamma.$$

We partition R into two regions: R_1 and R_2 where

$$R_1 = \{(\alpha, \beta, \gamma) \in R : \Delta(\alpha, \beta, \gamma) \geq 0\}$$

and $R_2 = R \setminus R_1$. For R_1 we have the following result.

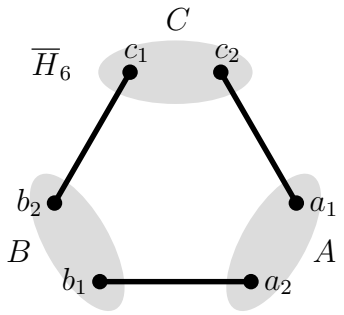

Figure 1.2: The tripartite complement of the graph H_6.

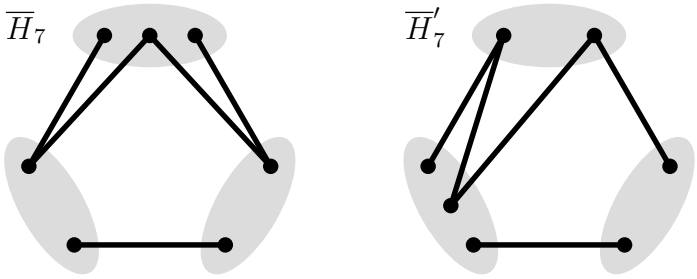

Figure 1.3: The tripartite complements of the graphs H_7 and H'_7.

Theorem 1.2.4. *If $(\alpha, \beta, \gamma) \in R_1$ and H_6 is the graph whose tripartite complement is given in Figure 1.2 then there exists a weighting w such that $(H_6, w) \in$ **Tri**(α, β, γ), and for any such w*

$$T_{\min}(\alpha, \beta, \gamma) = t(H_6, w) = \alpha + \beta + \gamma - 2.$$

Let $(G, w) \in$ **Tri**(α, β, γ). If $t(G, w) = T_{\min}(\alpha, \beta, \gamma)$ then (G, w) is said to be *extremal*. If there does not exist $(G', w') \in$ **Tri**(α, β, γ) with $t(G', w') = t(G, w)$ and $|V(G')| < |V(G)|$ then (G, w) is said to be *vertex minimal*. The tripartite graphs G and H with specified tripartitions are *strongly-isomorphic* if there is a graph isomorphism $f : G \to H$ such that the image of each vertex class in G is a vertex class in H.

Theorem 1.2.5. *If $(\alpha, \beta, \gamma) \in R_2$ and $(G, w) \in$ **Tri**(α, β, γ) is extremal and vertex minimal then G is strongly-isomorphic to H_7, H'_7, or H_9 (see Figure 1.3).*

Proof of Theorem 1.2.3. The graphs H_6, H_7 and H'_7 are induced subgraphs of

H_9 hence Theorems 1.2.4 and 1.2.5 imply Theorem 1.2.3. □

We conjecture that in fact the extremal graph is always an appropriate weighting of H_7. This would also give a simple formula for $T_{\min}(\alpha, \beta, \gamma)$. See Section 1.6 for details.

1.3 Proof of Theorem 1.2.4 (the region R_1)

We will begin by proving a lower bound for the density of triangles by double counting edges.

Lemma 1.3.1. *For any $\alpha, \beta, \gamma \in [0,1]$ and $(G, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ we have*

$$t(G, w) \geq \alpha + \beta + \gamma - 2.$$

Proof. Define

$$\mathbf{1}_{xy} = \begin{cases} 1, & \text{if } xy \in E(G), \\ 0, & \text{otherwise,} \end{cases} \qquad \mathbf{1}_{xyz} = \begin{cases} 1, & \text{if } xyz \in T(G), \\ 0, & \text{otherwise.} \end{cases}$$

Given $abc \in A \times B \times C$, the number of edges present between these three vertices is at most two unless abc forms a triangle. Hence

$$\sum_{abc \in A \times B \times C} w(abc)(\mathbf{1}_{bc} + \mathbf{1}_{ac} + \mathbf{1}_{ab}) \leq \sum_{abc \in A \times B \times C} w(abc)(2 + \mathbf{1}_{abc}). \tag{1.1}$$

The LHS of (1.1) sums to $\alpha + \beta + \gamma$, while the RHS is $2 + t(G, w)$. Therefore $t(G, w) \geq \alpha + \beta + \gamma - 2$. □

Lemma 1.3.2. *If w is a weighting of H_6 satisfying $(H_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ then*

$$t(H_6, w) = \alpha + \beta + \gamma - 2 = T_{\min}(\alpha, \beta, \gamma).$$

For ease of notation the weight associated with a vertex is indicated with a hat above the label, for example $w(b_1)$ is represented as $\hat{b}_1$.

Proof. Consider a general weighting of H_6 with vertices as labelled in Figure 1.2. We know $\hat{a}_2 = 1 - \hat{a}_1, \hat{b}_2 = 1 - \hat{b}_1$ and $\hat{c}_2 = 1 - \hat{c}_1$ since the sum of the

weights of the vertices in a class add up to one. Hence we can express the densities in terms of only $\hat{a}_1, \hat{b}_1$, and $\hat{c}_1$. The edge densities of H_6 are

$$\alpha = 1 - \hat{c}_1 + \hat{b}_1\hat{c}_1, \quad \beta = 1 - \hat{a}_1 + \hat{a}_1\hat{c}_1, \quad \gamma = 1 - \hat{b}_1 + \hat{a}_1\hat{b}_1.$$

The triangle density is given by

$$\begin{aligned} t(H_6, w) &= \hat{a}_1\hat{b}_1\hat{c}_1 + (1 - \hat{a}_1)(1 - \hat{b}_1)(1 - \hat{c}_1) \\ &= 1 - \hat{a}_1 - \hat{b}_1 - \hat{c}_1 + \hat{a}_1\hat{b}_1 + \hat{a}_1\hat{c}_1 + \hat{b}_1\hat{c}_1 \\ &= \alpha + \beta + \gamma - 2. \end{aligned}$$

By Lemma 1.3.1 we have $t(H_6, w) = T_{\min}(\alpha, \beta, \gamma)$. □

We now need to determine for which $(\alpha, \beta, \gamma) \in R$ a weighting w exists such that $(H_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$.

Lemma 1.3.3.

(i) If $(\alpha, \beta, \gamma) \in R$ then $\alpha, \beta, \gamma > 0$.

(ii) If $(\alpha, \beta, \gamma) \in R_2$ then $0 < \alpha, \beta, \gamma < 1$.

Proof. If $(\alpha, \beta, \gamma) \in R$ and $\alpha = 0$ then, since $\alpha\beta + \gamma > 1$, we have $\gamma > 1$, a contradiction. Similarly $\beta, \gamma > 0$.

If $(\alpha, \beta, \gamma) \in R_2$ then $R_2 \subseteq R$ implies that $\alpha, \beta, \gamma > 0$. If $\alpha = 1$ then $\Delta(\alpha, \beta, \gamma) = \Delta(1, \beta, \gamma) = (1 - \beta - \gamma)^2 \geq 0$. But $(\alpha, \beta, \gamma) \in R_2$ implies that $\Delta(\alpha, \beta, \gamma) < 0$, a contradiction. Similarly $\beta, \gamma < 1$. □

Lemma 1.3.4. *For $(\alpha, \beta, \gamma) \in R$ there exists a weighting w of H_6 such that $(H_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ if and only if $(\alpha, \beta, \gamma) \in R_1$.*

Proof. If $(\alpha, \beta, \gamma) \in R$ then Lemma 1.3.3 (i) implies that $\alpha, \beta, \gamma \neq 0$. First we will prove that if $(\alpha, \beta, \gamma) \in R$ and there exists a weighting w such that $(H_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$, then $(\alpha, \beta, \gamma) \in R_1$.

Let us label the vertices of H_6 as in Figure 1.2. Suppose w is weighting of H_6 such that $(H_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$. The edge densities in terms of $\hat{a}_1, \hat{b}_1, \hat{c}_1$ are

$$\alpha = 1 - \hat{c}_1 + \hat{b}_1\hat{c}_1, \tag{1.2}$$

$$\beta = 1 - \hat{a}_1 + \hat{a}_1\hat{c}_1, \tag{1.3}$$

$$\gamma = 1 - \hat{b}_1 + \hat{a}_1\hat{b}_1. \tag{1.4}$$

Case 1: One of α, β, γ equals 1.

By Lemma 1.3.3 (ii) $(\alpha, \beta, \gamma) \notin R_2$. Hence $(\alpha, \beta, \gamma) \in R$ implies $(\alpha, \beta, \gamma) \in R_1$.

Case 2: $\alpha, \beta, \gamma \neq 1$.

Since $\alpha, \beta, \gamma \neq 1$ we have $\hat{a}_1, \hat{b}_1, \hat{c}_1 \neq 0, 1$. Rearranging (1.4) and (1.3) we can write $\hat{b}_1$ and $\hat{c}_1$ in terms of $\hat{a}_1$,

$$\hat{b}_1 = \frac{1-\gamma}{1-\hat{a}_1}, \tag{1.5}$$

$$\hat{c}_1 = \frac{\hat{a}_1 + \beta - 1}{\hat{a}_1}. \tag{1.6}$$

Substituting into (1.2) and simplifying gives

$$\alpha\hat{a}_1^2 + (-\alpha + \beta - \gamma)\hat{a}_1 + \gamma - \beta\gamma = 0.$$

Hence

$$\hat{a}_1 = \frac{\alpha - \beta + \gamma \pm \sqrt{\Delta(\alpha, \beta, \gamma)}}{2\alpha}, \tag{1.7}$$

substituting back into (1.5) and (1.6) gives

$$\hat{b}_1 = \frac{\alpha + \beta - \gamma \pm \sqrt{\Delta(\alpha, \beta, \gamma)}}{2\beta}, \tag{1.8}$$

$$\hat{c}_1 = \frac{-\alpha + \beta + \gamma \pm \sqrt{\Delta(\alpha, \beta, \gamma)}}{2\gamma}. \tag{1.9}$$

By the definition of a weighting we have $\hat{a}_1, \hat{b}_1, \hat{c}_1 \in \mathbb{R}$, hence $\Delta(\alpha, \beta, \gamma) \geq 0$, and so $(\alpha, \beta, \gamma) \in R_1$.

Next we will show that if $(\alpha, \beta, \gamma) \in R_1$ then there exists a weighting w such that $(H_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$.

Case 1: One of α, β, γ equals 1.

Without loss of generality suppose $\alpha = 1$. Since $(1, \beta, \gamma) \in R_1 \subseteq R$ we have $\beta + \gamma > 1$. It is easy to check that $\hat{a}_1 = \gamma, \hat{b}_1 = 1, \hat{c}_1 = (\beta + \gamma - 1)/\gamma$ satisfy (1.2),(1.3),(1.4) and $\hat{a}_1, \hat{b}_1, \hat{c}_1 \in [0, 1]$ when $\beta + \gamma > 1$. This is enough to define a weighting w of H_6.

Case 2: $\alpha, \beta, \gamma \neq 1$.

Since $\Delta(\alpha, \beta, \gamma) \geq 0$, we may define $\hat{a}_1, \hat{b}_1, \hat{c}_1 \in \mathbb{R}$ by (1.7), (1.8), (1.9), taking the positive square root in each case. Due to the way $\hat{a}_1, \hat{b}_1, \hat{c}_1$ were constructed

they satisfy (1.2), (1.3), (1.4). Hence if $\hat{a}_1, \hat{b}_1, \hat{c}_1$ form a weighting w we will have $(H_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$. We need only prove $\hat{a}_1, \hat{b}_1, \hat{c}_1 \in (0, 1)$.

We will prove $\hat{a}_1 \in (0, 1)$, the proofs of $\hat{b}_1, \hat{c}_1 \in (0, 1)$ follow similarly. If $0 < \alpha - \beta + \gamma$ then $0 < \hat{a}_1$ because $\hat{a}_1$ is the positive square root version of (1.7). Now $(\alpha, \beta, \gamma) \in R$ implies $0 < \alpha\beta + \gamma - 1 < \alpha + \gamma - \beta$, and consequently $0 < \hat{a}_1$. By (1.7) if $\sqrt{\Delta(\alpha, \beta, \gamma)} < \alpha + \beta - \gamma$ then $\hat{a}_1 < 1$. Again $(\alpha, \beta, \gamma) \in R$ implies that $0 < \alpha\gamma + \beta - 1 < \alpha + \beta - \gamma$. Hence if we can show $\Delta(\alpha, \beta, \gamma) < (\alpha + \beta - \gamma)^2$ we will be done. Expanding and simplifying yields $0 < 4\alpha\beta(1 - \gamma)$ which is true because $\alpha, \beta, \gamma \in (0, 1)$. □

Proof of Theorem 1.2.4. The result follows immediately from Lemma 1.3.2 and 1.3.4. □

1.4 Proof of Theorem 1.2.5 (the region R_2)

We will begin by introducing a new type of graph in Section 1.4.1 which will allow us to develop a series of conditions that extremal vertex minimal examples must satisfy. In Section 1.4.2 we outline an algorithm that allows us to utilize the results of Section 1.4.1 to search for the extremal vertex minimal graphs in a finite time. This algorithm produces fourteen possible graphs. In Section 1.4.3 we eliminate those not strongly-isomorphic to H_7, H_7' and H_9 by analysing each of them in turn.

1.4.1 Properties

Our proof strategy for Theorem 1.2.5 is to establish various properties any extremal and vertex minimal weighted tripartite graph must satisfy. To prove these properties we introduce a new type of tripartite graph.

A *doubly-weighted tripartite graph* (G, w, p) is a weighted tripartite graph $(G, w) \in \mathbf{Tri}$ together with a function $p : E(G) \to (0, 1]$. We denote the set of all doubly-weighted tripartite graphs by $\mathbf{DTri}$. If $(G, w, p) \in \mathbf{DTri}$ then the *weight* of an edge $xy \in E(G)$ is defined to be

$$\lambda(xy) = w(xy)p(xy).$$

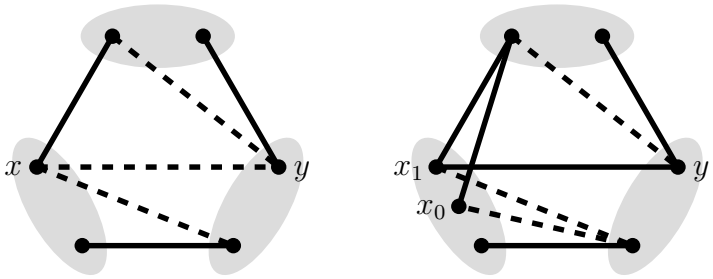

Figure 1.4: An example of (G, w, p) and $\mathsf{Split}(G, w, p, x, y)$. Partial edges are represented by dotted lines, the solid lines are edges which p maps to 1.

The *edge density* between a pair of vertex classes X and Y is

$$\sum_{xy \in E(X,Y)} \lambda(xy).$$

The *triangle density* is defined as

$$t(G, w, p) = \sum_{abc \in T(G)} p(ab)p(ac)p(bc)w(abc).$$

Any $(G, w) \in \mathbf{Tri}$ may be converted into a doubly-weighted tripartite graph (G, w, p) with the same triangle and edge densities by adding the function $p : E(G) \to (0, 1]$, $p(e) = 1$ for all $e \in E(G)$. Our next result allows us to do the reverse and convert a doubly-weighted tripartite graph into a weighted tripartite graph, leaving triangle and edge densities unchanged.

Lemma 1.4.1. *Given* $(G, w, p) \in \mathbf{DTri}$ *there exists* $(G', w') \in \mathbf{Tri}$ *with the same triangle and edge densities.*

For $(G, w, p) \in \mathbf{DTri}$ we will say that $e \in E(G)$ is a *partial edge* if $p(e) < 1$. To prove Lemma 1.4.1 we need a process to eliminate partial edges without affecting any of the densities.

For a graph G and vertex $v \in V(G)$ let $\Gamma^G(v)$ denote the neighbourhood of v in G. When no confusion can arise we write this simply as $\Gamma(v)$. Given a tripartite graph G with a vertex class X and $v \in V(G)$ we write $\Gamma^G_X(v) = \Gamma^G(v) \cap X$. Again when no confusion can arise we write this simply as $\Gamma_X(v)$.

Algorithm 1.4.2 (Split)**.** The algorithm Split takes as input $(G, w, p) \in \mathbf{DTri}$

and an ordered pair of vertices (x, y), such that xy is a partial edge. Its output, $\mathsf{Split}(G, w, p, x, y)$, is a doubly-weighted tripartite graph, which no longer contains the partial edge xy. If $(G', w', p') = \mathsf{Split}(G, w, p, x, y)$ then G', w', p' are formed as follows:

- Construct G' from G by replacing the vertex x by two new vertices x_0 and x_1 that lie in the same vertex class as x. Add edges from x_0, x_1 so that $\Gamma^{G'}(x_0) = \Gamma^G(x) \setminus \{y\}$ and $\Gamma^{G'}(x_1) = \Gamma^G(x)$.
- Set $w'(x_0) = w(x)(1 - p(xy))$ and $w'(x_1) = w(x)p(xy)$. Let $w'(v) = w(v)$ for all $v \in V(G) \setminus \{x\}$.
- Set $p'(x_0v) = p'(x_1v) = p(xv)$ for all $v \in \Gamma^G(x) \setminus \{y\}$, and $p'(x_1y) = 1$. Let $p'(uv) = p(uv)$ for all $uv \in E(G)$ such that $u, v \neq x$.

Note that in $\mathsf{Split}(G, w, p, y, x)$ (the result of applying Split to (G, w, p) and (y, x)) the vertex y would have been "split" into two new vertices rather than x. It also does not contain the partial edge xy. So if we wish to remove the partial edge xy we can choose between $\mathsf{Split}(G, w, p, x, y)$ and $\mathsf{Split}(G, w, p, y, x)$.

Figure 1.4 shows an example application of Split with "before" and "after" pictures of (G, w, p) and $\mathsf{Split}(G, w, p, x, y)$.

Lemma 1.4.3. *For any $(G, w, p) \in \mathbf{DTri}$ and xy a partial edge, $(G', w', p') = \mathsf{Split}(G, w, p, x, y)$ has the same triangle and edge densities as (G, w, p).*

Proof. Without loss of generality let us assume $x \in A$ and $y \in B$. We will prove the result by calculating the difference in densities between (G', w', p') and (G, w, p) and showing them to be zero. The change in the edge density between classes A and B is

$$w(y)(w'(x_1)p'(x_1y) - w(x)p(xy)) + \sum_{v \in \Gamma_B^G(x) \setminus \{y\}} w(v)(w'(x_0)p'(x_0v) + w'(x_1)p'(x_1v) - w(x)p(xv))$$

which is zero. Similarly the change in density between classes A and C is zero. There is no change in the density between classes B and C since the algorithm Split leaves this part of the graph untouched. The change in the triangle density

is

$$\sum_{xuv\in T(G),u\in B\setminus\{y\},v\in C}\Big(w'(x_0)+w'(x_1)-w(x)\Big)w(u)w(v)p(xu)p(xv)p(uv)+$$
$$\sum_{xyv\in T(G),v\in C}\Big(w'(x_1)p'(x_1y)-w(x)p(xy)\Big)w(y)w(v)p(xv)p(yv),$$

which is zero, hence the triangle and edge densities do not change. □

Proof of Lemma 1.4.1. Given $(G,w,p)\in\mathbf{DTri}$, if $p(e)=1$ for all $e\in E(G)$ then the weighted tripartite graph (G,w) will have the same densities as the doubly-weighted tripartite graph.

Suppose (G,w,p) contains a partial edge av, with $a\in A$. We can remove this partial edge by replacing (G,w,p) by $\mathsf{Split}(G,w,p,a,v)$. Unfortunately this may introduce new partial edges. However, we can show that by repeated applications of Split we will eventually remove all partial edges. Consider

$$Z(G,w,p)=\sum_{v\in A}3^{d_z(v)},$$

where

$$d_z(v)=|\{u\in V(G):uv\in E(G),p(uv)\neq 1\}|.$$

If $(G',w',p')=\mathsf{Split}(G,w,p,a,v)$ then $Z(G',w',p')<Z(G,w,p)$. This is because Split replaces vertex a with the vertices a_0 and a_1, and so Z changes by

$$\begin{aligned}3^{d_z(a_0)}+3^{d_z(a_1)}-3^{d_z(a)}&=3^{d_z(a)-1}+3^{d_z(a)-1}-3^{d_z(a)}\\&=-3^{d_z(a)-1}.\end{aligned}$$

Since Z is integral and is bounded below (by zero for instance), repeatedly applying Split will eventually remove all partial edges incident with A. Note that doing this will not have created any new partial edges between classes B and C.

We can repeat this process on the partial edges leaving B, to get rid of the remaining partial edges. Let us call the resulting doubly-weighted tripartite graph (G'',w'',p''). Since we created (G'',w'',p'') only by applying Split, by Lemma 1.4.3, (G,w,p) and (G'',w'',p'') must have the same edge and triangle

densities. Since (G'', w'', p'') has no partial edges, $p''(e) = 1$ for all $e \in E(G'')$, consequently (G'', w'') has the same edge and triangle densities as (G'', w'', p'') and therefore (G, w, p). □

Since we can convert easily between weighted and doubly-weighted tripartite graphs, it is useful to know when there exist doubly-weighted tripartite graphs with the same edge densities but with smaller triangle densities. Let (G, w, p) be a doubly-weighted tripartite graph. By carefully modifying p we can adjust the weights of edges whilst not affecting the edge densities and potentially decreasing the triangle density. Our next result lists a series of conditions under which this could occur.

Let G be a tripartite graph with vertex classes A, B, C. For $a \in A, b \in B$ define

$$C_{ab} = \{c \in C : ac, bc \in E(G)\}.$$

Lemma 1.4.4. *If $(G, w, p) \in$ **DTri** satisfies conditions $(i) - (iv)$ below, then there exists $(G', w, p') \in$ **DTri** with the same edge densities as (G, w, p) but $t(G', w, p') < t(G, w, p)$.*

(i) $w(v) > 0$ *for all* $v \in V(G)$,

(ii) $p(e) = 1$ *for all* $e \in E(A, C) \cup E(B, C)$,

(iii) there exist, not necessarily distinct, vertices $a_0, a_1 \in A$, $b_0, b_1 \in B$ *such that* $a_1b_1 \in E(G)$ *and either* $a_0b_0 \notin E(G)$ *or* $p(a_0b_0) < 1$,

(iv) $$\sum_{c \in C_{a_0b_0}} w(c) < \sum_{c \in C_{a_1b_1}} w(c).$$

Corollary 1.4.5. *Let $(G, w) \in$ **Tri**. If there exist, not necessarily distinct, vertices $a_0, a_1 \in A$, $b_0, b_1 \in B$ such that $a_0b_0 \notin E(G)$, $a_1b_1 \in E(G)$ and $C_{a_0b_0}$ is a proper subset of $C_{a_1b_1}$ then (G, w) is either not extremal or not vertex minimal.*

Proof of Corollary 1.4.5. We will prove that if (G, w) is vertex minimal then it is not extremal by applying Lemma 1.4.4.

Let (G, w) be vertex minimal, so $w(v) > 0$ for all $v \in V(G)$. We can add the function p which maps all edges of G to 1 to create $(G, w, p) \in$ **DTri**. Now (G, w, p) has the same triangle and edge densities as (G, w). By Lemma 1.4.1 it is enough to show that there exists $(G', w', p') \in$ **DTri** with the same edge densities as (G, w, p) but a smaller density of triangles. Note that conditions

$(i) - (iii)$ in the statement of Lemma 1.4.4 hold for (G, w, p). Thus Lemma 1.4.4 will provide such a (G', w', p') if we can show that

$$\sum_{c \in C_{a_0 b_0}} w(c) < \sum_{c \in C_{a_1 b_1}} w(c).$$

Let $u \in C_{a_1 b_1} \setminus C_{a_0 b_0}$. Since (G, w) is vertex minimal $w(u) > 0$. Hence

$$\sum_{c \in C_{a_1 b_1}} w(c) - \sum_{c \in C_{a_0 b_0}} w(c) = \sum_{c \in C_{a_1 b_1} \setminus C_{a_0 b_0}} w(c) \geq w(u) > 0.$$

In which case all the conditions of Lemma 1.4.4 are satisfied, and (G, w) is not extremal. □

Proof of Lemma 1.4.4. If $a_0 b_0 \notin E(G)$ let G' be the graph produced from G by adding the edge $a_0 b_0$. If $a_0 b_0 \in E(G)$ then let $G' = G$. Define $p' : E(G') \to (0, 1]$ by $p'(e) = p(e)$ for $e \in E(G') \setminus \{a_0 b_0, a_1 b_1\}$ and

$$p'(a_0 b_0) = \begin{cases} \dfrac{\delta}{w(a_0) w(b_0)}, & \text{if } a_0 b_0 \notin E(G), \\ p(a_0 b_0) + \dfrac{\delta}{w(a_0) w(b_0)}, & \text{if } a_0 b_0 \in E(G), \end{cases}$$
$$p'(a_1 b_1) = p(a_1 b_1) - \frac{\delta}{w(a_1) w(b_1)},$$

where $\delta > 0$ is chosen sufficiently small so that $p'(a_0 b_0), p'(a_1 b_1) \in (0, 1)$.

The weights and edges have not changed between classes A, C and B, C. Consequently the corresponding edge densities will be the same in (G, w, p) and (G', w, p'). However, the edge density between class A and B, and the triangle densities may have changed. The difference in the A, B edge density between (G', w, p') and (G, w, p) is

$$w(a_0) w(b_0) \frac{\delta}{w(a_0) w(b_0)} - w(a_1) w(b_1) \frac{\delta}{w(a_1) w(b_1)} = 0.$$

The change in triangle density is

$$\sum_{a_0 b_0 c \in T(G')} w(a_0) w(b_0) w(c) \frac{\delta}{w(a_0) w(b_0)} - \sum_{a_1 b_1 c \in T(G')} w(a_1) w(b_1) w(c) \frac{\delta}{w(a_1) w(b_1)}$$

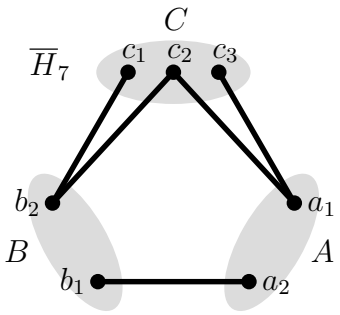

Figure 1.5: The tripartite complement of the graph H_7.

which simplifies to

$$\delta \left(\sum_{a_0 b_0 c \in T(G')} w(c) - \sum_{a_1 b_1 c \in T(G')} w(c) \right) < 0.$$

Where the final inequality follows from condition (iv).

Hence the density of triangles in (G', w, p') is smaller than that in (G, w, p), but the edge densities are the same in both. □

Lemma 1.4.6. *Consider the graph H_7 whose tripartite complement is given in Figure 1.5. If $(\alpha, \beta, \gamma) \in R_2$ then there exists a weighting w such that $(H_7, w) \in$ **Tri**(α, β, γ) and $t(H_7, w) = 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2$. Furthermore $t(H_7, w) \leq t(H_7, w')$ for all weightings w', such that $(H_7, w') \in$ **Tri**(α, β, γ).*

Proof. If $(\alpha, \beta, \gamma) \in R_2$ then, by Lemma 1.3.3 (ii), we know that $0 < \alpha, \beta, \gamma < 1$. Consider a general weighting of H_7, with vertices labelled as in Figure 1.5. If such a weighting of H_7 has edge densities α, β, γ then $\alpha, \beta, \gamma < 1$ implies that $\hat{a}_1 \neq 0, 1, \gamma$. Now given α, β, γ and $\hat{a}_1 \neq 0, 1, \gamma$ we have enough information to deduce the rest of the weights of the vertices. (Note that this may not be an actual weighting since some of these values may lie outside of $[0, 1]$.)

$$\hat{a}_2 = 1 - \hat{a}_1, \qquad \hat{b}_1 = \frac{1-\gamma}{1-\hat{a}_1}, \qquad \hat{b}_2 = \frac{\gamma - \hat{a}_1}{1-\hat{a}_1},$$

$$\hat{c}_1 = 1 - \frac{1-\beta}{\hat{a}_1}, \qquad \hat{c}_3 = 1 - \frac{(1-\alpha)(1-\hat{a}_1)}{\gamma - \hat{a}_1},$$

$$\hat{c}_2 = \frac{1-\beta}{\hat{a}_1} + \frac{(1-\alpha)(1-\hat{a}_1)}{\gamma - \hat{a}_1} - 1,$$

which have been deduced from

$$1 = \hat{a}_1 + \hat{a}_2, \qquad 1 - \gamma = \hat{a}_2\hat{b}_1, \qquad 1 = \hat{b}_1 + \hat{b}_2,$$
$$1 - \beta = (1 - \hat{c}_1)\hat{a}_1, \qquad 1 - \alpha = (1 - \hat{c}_3)\hat{b}_2, \qquad 1 = \hat{c}_1 + \hat{c}_2 + \hat{c}_3,$$

respectively. There are two triangles in H_7, with weights $\hat{a}_1\hat{b}_1\hat{c}_1$ and $\hat{a}_2\hat{b}_2\hat{c}_3$, hence the triangle density is

$$\hat{a}_1\left(\frac{1-\gamma}{1-\hat{a}_1}\right)\left(1-\frac{1-\beta}{\hat{a}_1}\right) + (1-\hat{a}_1)\left(\frac{\gamma-\hat{a}_1}{1-\hat{a}_1}\right)\left(1-\frac{(1-\alpha)(1-\hat{a}_1)}{\gamma-\hat{a}_1}\right)$$

which simplifies to

$$2\gamma - 2 + \frac{\beta(1-\gamma)}{1-\hat{a}_1} + \alpha(1-\hat{a}_1).$$

This expression is minimized when $1 - \hat{a}_1 = \sqrt{\beta(1-\gamma)/\alpha}$, and consequently we obtain the desired triangle density of $2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2$. We now must show that the vertex weights implied by $\hat{a}_1 = 1 - \sqrt{\beta(1-\gamma)/\alpha}$ all lie in $[0,1]$ and that $\hat{a}_1 \neq \gamma, 0, 1$. Since the sum of the weights in each class equals 1, in order to show that all weights lie in $[0,1]$ it is sufficient to show that they are all non-negative.

If $\hat{a}_1 = \gamma$ then $1 - \gamma = \sqrt{\beta(1-\gamma)/\alpha}$, which rearranges to $\alpha\gamma + \beta - \alpha = 0$. However, $\alpha\gamma + \beta - \alpha > \alpha\gamma + \beta - 1 > 0$ (as $(\alpha, \beta, \gamma) \in R_2 \subseteq R$), hence $\hat{a}_1 \neq \gamma$. $1 - \hat{a}_1$ is clearly positive, proving that $0 < \hat{a}_2$ and $\hat{a}_1 \neq 1$. Showing $0 < \hat{a}_1$ is equivalent to proving $\sqrt{\beta(1-\gamma)/\alpha} < 1$ which is true if $0 < \beta\gamma + \alpha - \beta$, and this holds because $\beta\gamma + \alpha - \beta > \beta\gamma + \alpha - 1 > 0$. Since $\hat{b}_2$ equals $1 - \sqrt{\alpha(1-\gamma)/\beta}$, a similar argument shows that $\hat{b}_1, \hat{b}_2 > 0$. It is also straightforward to show that $\hat{c}_1, \hat{c}_3 > 0$, but showing $\hat{c}_2 > 0$ requires more work . Using $\hat{c}_1 + \hat{c}_2 + \hat{c}_3 = 1$, $\hat{c}_1 = 1 - (1-\beta)/\hat{a}_1$, and $\hat{c}_3 = 1 - (1-\alpha)/\hat{b}_2$ we obtain

$$\hat{c}_2 = -1 + \frac{(1-\beta)\hat{b}_2 + (1-\alpha)\hat{a}_1}{\hat{a}_1\hat{b}_2}.$$

Hence $\hat{c}_2 > 0$ if and only if

$$\hat{a}_1\hat{b}_2 < (1-\beta)\hat{b}_2 + (1-\alpha)\hat{a}_1.$$

Substituting $\hat{a}_1 = 1 - \sqrt{\beta(1-\gamma)/\alpha}$ and $\hat{b}_2 = 1 - \sqrt{\alpha(1-\gamma)/\beta}$ yields

$$\alpha + \beta - \gamma < 2\sqrt{\alpha\beta(1-\gamma)}.$$

Now $\alpha + \beta - \gamma > \alpha\gamma + \beta - 1 > 0$, hence $0 < \hat{c}_2$ if and only if $(\alpha + \beta - \gamma)^2 < 4\alpha\beta(1-\gamma)$. Collecting all the terms onto the left hand side shows that we require $\Delta(\alpha, \beta, \gamma) < 0$, which we have from the fact that $(\alpha, \beta, \gamma) \in R_2$. □

Lemma 1.4.7. *For any* $(\alpha, \beta, \gamma) \in R_2$,

$$T_{\min}(\alpha, \beta, \gamma) < \min\{\alpha\beta + \gamma - 1, \alpha\gamma + \beta - 1, \beta\gamma + \alpha - 1\}.$$

Proof of Lemma 1.4.7. Without loss of generality let us assume that

$$\alpha\beta + \gamma - 1 = \min\{\alpha\beta + \gamma - 1, \alpha\gamma + \beta - 1, \beta\gamma + \alpha - 1\}.$$

By Lemma 1.4.6 we know that for any $(\alpha, \beta, \gamma) \in R_2$ there exists a weighting w such that $(H_7, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ and $t(H_7, w) = 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2$. Hence $T_{\min}(\alpha, \beta, \gamma) \leq 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2$. If $\alpha\beta + \gamma - 1 \leq 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2$ then

$$\alpha\beta + 1 - \gamma \leq 2\sqrt{\alpha\beta(1-\gamma)}.$$

Squaring and rearranging yields

$$(\alpha\beta + \gamma - 1)^2 \leq 0.$$

Since $(\alpha, \beta, \gamma) \in R_2 \subseteq R$ we know $\alpha\beta + \gamma - 1 > 0$ holds true, hence we have a contradiction. □

Lemma 1.4.8. *Let* $(\alpha, \beta, \gamma) \in R_2$. *If* $(G, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ *is extremal, then* $|A|, |B|, |C| \geq 2$.

To prove Lemma 1.4.8 we will require the following algorithm.

Algorithm 1.4.9 (Merge)**.** The algorithm Merge takes as input $(G, w, p) \in \mathbf{DTri}$, and two distinct vertices $x_1, x_2 \in X$, where X is one of the vertex classes of G. The vertices x_1, x_2, must satisfy, for some vertex class $Y \neq X$, $\Gamma_Y(x_1) = \Gamma_Y(x_2)$, $w(x_1) + w(x_2) > 0$ and $p(x_1 y) = p(x_2 y) = 1$ for all $y \in \Gamma_Y(x_1)$. The output of the algorithm is represented by $\mathsf{Merge}(G, w, p, x_1, x_2)$ and is a doubly-weighted tripartite graph in which x_1, x_2 have been replaced by a single new

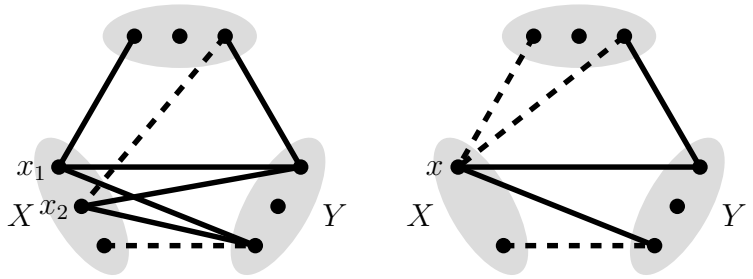

Figure 1.6: An example of (G, w, p) and $\mathsf{Merge}(G, w, p, x_1, x_2)$. Partial edges are represented by dotted lines, the solid lines are edges which p maps to 1

vertex: x. For convenience let us write $(G', w', p') = \mathsf{Merge}(G, w, p, x_1, x_2)$. Now G', w', p' are formed as follows:

- Construct G' from G by replacing the vertices x_1, x_2 by a new vertex x in X. Add edges from x so that $\Gamma^{G'}(x) = \Gamma^G(x_1) \cup \Gamma^G(x_2)$.
- Set $w'(x) = w(x_1) + w(x_2)$. Let $w'(v) = w(v)$ for all $v \in V(G') \setminus \{x\}$.
- For $u, v \in V(G') \setminus \{x\}$ and $uv \in E(G')$, let $p'(uv) = p(uv)$. For $xv \in E(G')$ set

$$p'(xv) = \begin{cases} w(x_1)p(x_1v)/w'(x), & \text{if } x_1v \in E(G),\ x_2v \notin E(G), \\ w(x_2)p(x_2v)/w'(x), & \text{if } x_1v \notin E(G),\ x_2v \in E(G), \\ (w(x_1)p(x_1v) + w(x_2)p(x_2v))/w'(x), & \text{if } x_1v \in E(G),\ x_2v \in E(G). \end{cases}$$

Observe that for $y \in Y$ we have $xy \in E(G')$ if and only if $x_1y, x_2y \in E(G)$ and in this case $p(xy) = 1$. It is easy to check that the edge and triangle densities of (G, w, p) and (G', w', p') are the same.

Proof of Lemma 1.4.8. Suppose (G, w) is extremal and without loss of generality vertex class $C = \{c\}$ contains exactly one vertex. We can assume $w(v) \neq 0$ for all $v \in V(G)$, as any vertices with weight zero can be removed without affecting any of the densities. Create a doubly-weighted tripartite graph (G, w, p) with the same densities as (G, w) by setting $p(e) = 1$ for all $e \in E(G)$. We will show that the triangle density of (G, w, p) is at least $\alpha\beta+\gamma-1$ and consequently, by Lemma 1.4.7, (G, w) is not extremal.

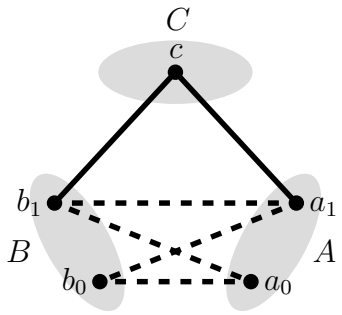

Figure 1.7: A graph with $|C| = 1$ after merging vertices in A and B. The dotted lines represent edges that may or may not be in the graph.

Since $(\alpha, \beta, \gamma) \in R_2$, by Lemma 1.3.3 (ii) we have $\beta \neq 0, 1$. Moreover since $C = \{c\}$ we know that there must exist a vertex in A whose neighbourhood in C is empty and another whose neighbourhood in C is $\{c\}$. We can replace all vertices $a \in A$ satisfying $\Gamma_C(a) = \emptyset$ by a single vertex a_0 via repeated applications of the Merge algorithm on pairs of such vertices. Similarly we can replace all vertices with $\Gamma_C(a) = \{c\}$ by a single vertex a_1. Having done this we obtain a doubly-weighted graph in which $A = \{a_0, a_1\}$, a_1c is an edge, and a_0c is a non-edge. Note the edges and weights between B and C remain unchanged but we may have modified the edges and weights between A and B.

By a similar argument we can reduce B to two vertices b_0, b_1, with b_1c an edge and b_0c a non-edge. Let us call this doubly weighted graph (G', w', p'), and note it has the same densities as (G, w, p) and hence (G, w). By construction we have

$$a_0c, b_0c \notin E(G'), \quad a_1c, b_1c \in E(G'), \quad p'(a_1c) = p'(b_1c) = 1,$$

see Figure 1.7.

We now have enough information to determine the weights of all of the vertices:

$$w'(c) = 1, \quad w'(a_1) = \beta, \quad w'(a_0) = 1 - \beta, \quad w'(b_1) = \alpha, \quad w'(b_0) = 1 - \alpha.$$

The only information we are lacking about (G', w', p') is which edges are present in $E(A, B)$ and what their weights are. However, since $(\alpha, \beta, \gamma) \in R$, Theorem 1.2.1 implies that G' contains a triangle. Hence $a_1b_1 \in E(A, B)$. Since $C_{a_1b_1} =$

$\{c\}$ and $C_{a_0b_0} = C_{a_0b_1} = C_{a_1b_0} = \emptyset$, Lemma 1.4.4 tells us that (G, w) will not be extremal unless a_0b_0, a_0b_1, a_1b_0 are all edges which p' maps to 1.

Now, a_1b_1c is the only triangle in the doubly-weighted tripartite graph, hence the triangle density is $w'(a_1)w'(b_1)p'(a_1b_1) = \lambda(a_1b_1)$ (as $w'(c), p'(a_1c), p'(b_1c)$ are all 1). By the definition of edge density in a doubly-weighted tripartite graph, we have

$$\begin{aligned}\gamma &= \lambda(a_0b_0) + \lambda(a_0b_1) + \lambda(a_1b_0) + \lambda(a_1b_1) \\ &= (1-\alpha)(1-\beta) + \alpha(1-\beta) + (1-\alpha)\beta + t(G', w', p') \\ &= 1 - \alpha\beta + t(G', w', p')\end{aligned}$$

Hence the triangle density is $\alpha\beta + \gamma - 1$, which by Lemma 1.4.7 and Lemma 1.4.1 implies that (G, w) is not extremal. □

Lemma 1.4.10. *If $(\alpha, \beta, \gamma) \in R_2$, $(G, w) \in$ **Tri**(α, β, γ) and for all $a_1, a_2 \in A$, $\Gamma_C(a_1) = \Gamma_C(a_2)$ then (G, w) is not extremal.*

Proof. If there exist any vertices with weight zero, we can remove them without affecting the densities. Convert the resulting weighted tripartite graph into a doubly-weighted tripartite graph and reduce A down to a single vertex, by repeated applications of Merge on the vertices in A. Any partial edges that appear will lie in $E(A, B)$.

Now repeatedly apply Split choosing to replace vertices in B rather than A, until no more partial edges remain. Consequently we have modified the weighted graph into a new weighted graph with the same densities and now $|A| = 1$. By Lemma 1.4.8 we know this is not extremal and hence (G, w) was not extremal. □

Our next lemma is an adaptation of a convexity argument by Bondy, Shen, Thomassé and Thomassen (see proof of Theorem 3 [4]). By exploiting the freedom we have to change the weights of vertices we can reduce the number of vertices in each class to at most three. Consequently there are only a finite number of graphs we need to consider.

Lemma 1.4.11. *If $(G, w) \in$ **Tri** is extremal and vertex minimal, then $|A|, |B|, |C| \leq 3$*

Again we introduce an algorithm to prove this lemma.

Algorithm 1.4.12 (Reduce)**.** The algorithm Reduce takes as input $(G, w) \in$ **Tri** and a vertex class X of G, satisfying $|X| > 3$. Its output, represented by $\mathsf{Reduce}(G, w, X)$, is a weighted tripartite graph, which has the same edge densities as (G, w), but with $|X| \leq 3$, and triangle density at most that of (G, w).

To help explain the algorithm we will suppose $X = A$, (the other choices of X work similarly). For each vertex $a_i \in A$ let

$$\beta_i = \sum_{c \in \Gamma_C(a_i)} w(c), \quad \gamma_i = \sum_{b \in \Gamma_B(a_i)} w(b), \quad t_i = \sum_{bc \in E(B,C), a_i bc \in T(G)} w(bc).$$

By definition

$$\beta = \sum_{i=1}^{|A|} w(a_i)\beta_i, \quad \gamma = \sum_{i=1}^{|A|} w(a_i)\gamma_i, \quad t(G, w) = \sum_{i=1}^{|A|} w(a_i)t_i.$$

Consider the convex hull

$$P = \left\{ \sum_{i=1}^{|A|} x_i(\beta_i, \gamma_i, t_i) : \sum_{i=1}^{|A|} x_i = 1 \text{ and } x_i \geq 0 \right\}.$$

Setting $x_i = w(a_i)$ shows that $(\beta, \gamma, t(G, w))$ lies in P. By varying the values of the x_i we can decrease the value of $t(G, w)$ to t' such that (β, γ, t') lies on the boundary of P. Moreover, by triangulating the facet of P containing (β, γ, t'), we can express (β, γ, t') as a convex combination of at most three elements of $\{(\beta_i, \gamma_i, t_i) : 1 \leq i \leq |A|\}$. Consequently we can write

$$(\beta, \gamma, t') = \sum_{i=1}^{|A|} x_i(\beta_i, \gamma_i, t_i)$$

where $\sum x_i = 1$ and at most three of the x_i are positive, the rest are zero. Now define a new weighting w' for G by $w'(a_i) = x_i$, $w'(v) = w(v)$ for $v \in V(G) \setminus A$. The weighted tripartite graph (G, w') has the same edge densities as (G, w) and a new triangle density t' satisfying $t' \leq t(G, w)$. Furthermore we can remove the zero weighted vertices from A so that $|A| \leq 3$ and the densities are unchanged.

Proof of Lemma 1.4.11. Suppose (G, w) is extremal and vertex minimal with, without loss of generality, $|A| > 3$. Now, using Algorithm 1.4.12, $\mathsf{Reduce}(G, w, A)$

has the same densities as (G, w) (since (G, w) is extremal), but it has fewer vertices, contradicting the vertex minimality of (G, w). □

Lemma 1.4.13. *Let (G, w) be a weighted tripartite graph. If there exist distinct vertices $a_1, a_2 \in A$ with $\Gamma_C(a_1) = \Gamma_C(a_2)$ and $|B| = 3$, then (G, w) is not extremal or not vertex minimal.*

Proof. Convert (G, w) into a doubly-weighted tripartite graph and replace a_1, a_2 with a vertex a by applying Merge (we may assume $w(a_1)+w(a_2) > 0$ by vertex minimality of (G, w)). Now A has reduced in size by one. If there are partial edges they will lie between classes A and B. Use the Split algorithm to remove them, choosing to replace vertices in B rather than A. Now convert the doubly-weighted graph back into a weighted graph. This weighted graph will have the same densities as (G, w), A has one less vertex, and $|B| \geq 3$. If $|B| = 3$ then this weighted graph is of smaller order than (G, w). If $|B| > 3$ we can use Reduce to modify the weights of vertices in B, such that at most three of them have a non-zero weight. Simply remove all vertices with zero weight and the resulting graph will be of smaller order than G, contradicting vertex minimality. □

Lemma 1.4.14. *Consider a weighted graph (G, w). If there exist distinct vertices $a_1, a_2 \in A$ with $\Gamma(a_1) = \Gamma(a_2)$ then (G, w) is not vertex minimal.*

Proof. Remove vertex a_2 and increase the weight of a_1 by $w(a_2)$. The resulting weighted graph has the same densities as (G, w). □

Lemma 1.4.15. *Given a tripartite graph G with $|A| = 3$, not necessarily distinct, vertices $a_0, a_1 \in A$, $b_0, b_1 \in B$ such that $a_0b_0 \notin E(G), a_1b_1 \in E(G)$ and $C_{a_0b_0} = C_{a_1b_1}$, construct two graphs G_1, G_2 as follows:*

- *Let $G_1' = G - a_1b_1$. Construct G_1 from G_1' by adding a new vertex a_2 to A and adding edges incident to a_2 so that $\Gamma^{G_1}(a_2) = \Gamma^{G_1'}(a_0) \cup \{b_0\}$.*
- *Let $G_2' = G + a_0b_0$. Construct G_2 from G_2' by adding a new vertex a_2 to A and adding edges incident to a_2 so that $\Gamma^{G_2}(a_2) = \Gamma^{G_2'}(a_1) \setminus \{b_1\}$.*

Note that in G_1 and G_2 we have $|A| = 4$. Let $\mathcal{H}$ denote the family of eight graphs constructed from G_1 or G_2 by deleting a single vertex from A.

If (G, w) is extremal and vertex minimal then there exists $H \in \mathcal{H}$ and a weighting w' of H, such that (H, w') has the same edge densities as (G, w) and is also extremal and vertex minimal.

Proof. Our proof will involve first showing that there exists a weighting w'' of G_1, G_2 such that either (G_1, w'') or (G_2, w'') have the same densities as (G, w).

Form a doubly-weighted graph (G, w, p) with $p(e) = 1$ for all $e \in E(G)$. Since $C_{a_0b_0} = C_{a_1b_1}$, if we add the edge a_0b_0 to G we can move weight from edge a_1b_1 to a_0b_0, by modifying $p(a_1b_1)$ and $p(a_0b_0)$, whilst keeping the edge and triangle densities constant. If we move as much weight as we can from a_1b_1 to a_0b_0, one of two things must happen. Either we manage to make $p(a_0b_0) = 1$ before $p(a_1b_1)$ reaches zero, or $p(a_1b_1)$ reaches zero (so we remove edge a_1b_1) and $p(a_0b_0) \leq 1$. In either case we have at most one partial edge either a_1b_1 or a_0b_0. We can remove the partial edge by an application of the Split algorithm, introducing an extra vertex into class A. The two possible resulting graphs are G_2, G_1 respectively. Hence there exists a weighting w'' such that either (G_1, w'') or (G_2, w'') have the same densities as (G, w).

Without loss of generality let us assume (G_1, w'') has the same densities as (G, w). Since $|A| = 4$ for G_1, applying the Reduce algorithm will remove at least one vertex from A to create a doubly-weighted graph, say (H, w'), with the same edge densities and possibly a smaller triangle density. However, since $t(G, w) = t(G_1, w'') \geq t(H, w')$ and (G, w) is extremal, we must have $t(G, w) = t(H, w')$, implying (H, w') is extremal. We can also conclude, by the vertex minimality of (G, w), that H is formed from G_1 by removing exactly one vertex from A. □

1.4.2 Search for extremal examples

We have now developed a number of important conditions that any vertex minimal extremal examples must satisfy. These will, eventually, allow us to conduct an exhaustive search for such graphs (with the aid of a computer). This will then leave us with a small number of possible extremal graphs which we will deal with by hand.

Recall that the tripartite graphs G and H (as always with specified tripartitions) are *strongly-isomorphic* if there is a graph isomorphism $f : G \to H$ such that the image of each vertex class in G is a vertex class in H.

It turns out that if we can eliminate graphs that are strongly-isomorphic to two particular examples: F_7 and F_9 (see Figure 1.8), then our computer search will be able to eliminate many more possible extremal vertex minimal examples, and thus reduce the amount of work we will finally need to do by hand.

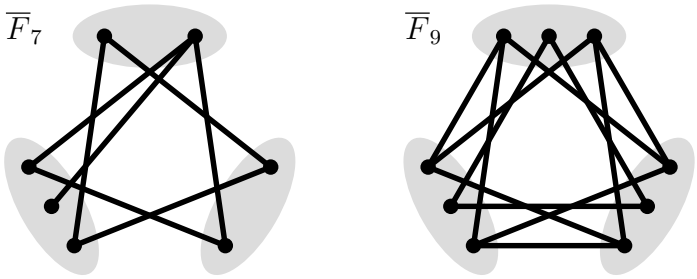

Figure 1.8: The tripartite complements of the graphs F_7 and F_9.

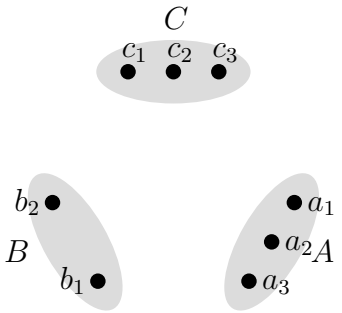

Figure 1.9: Canonical labelling of vertices and vertex classes.

For ease of notation we will henceforth implicitly label the vertices and vertex classes of all figures as in Figure 1.9. Indices of vertices start at one and increase clockwise. Recall that the weight associated with a vertex is indicated with a hat above the label, for example $w(b_1)$ is represented as $\hat{b}_1$.

Lemma 1.4.16. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that $(F_7, w) \in$ **Tri**(α, β, γ), (F_7, w) is either not extremal or not vertex minimal.*

To prove Lemma 1.4.16, we first need to prove the following result about the graph F_6 given in Figure 1.10.

Lemma 1.4.17. *For any $\alpha, \beta, \gamma \in [0, 1]$ and weighting w satisfying $(F_6, w) \in$ **Tri**(α, β, γ) we have*

$$t(F_6, w) \geq \min\{\alpha\beta + \gamma - 1, \alpha\gamma + \beta - 1, \beta\gamma + \alpha - 1\}. \tag{1.10}$$

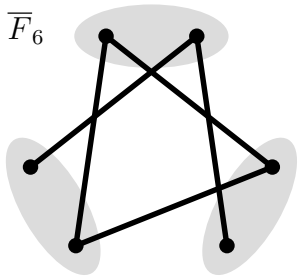

Figure 1.10: The tripartite complement of the graph F_6.

Proof. Suppose (1.10) fails to hold. Since F_6 contains only one triangle: $a_2b_2c_1$, and using the fact that $\hat{a}_2 = 1 - \hat{a}_1$, $\hat{b}_2 = 1 - \hat{b}_1$, $\hat{c}_2 = 1 - \hat{c}_1$, we have

$$t(F_6, w) = (1 - \hat{a}_1)(1 - \hat{b}_1)\hat{c}_1 \tag{1.11}$$

$$\alpha = \hat{b}_1(1 - \hat{c}_1) + (1 - \hat{b}_1)\hat{c}_1 \tag{1.12}$$

$$\beta = \hat{a}_1(1 - \hat{c}_1) + (1 - \hat{a}_1)\hat{c}_1 \tag{1.13}$$

$$\gamma = 1 - \hat{a}_1\hat{b}_1, \tag{1.14}$$

Substitute (1.11), (1.12), (1.13), (1.14), into $t(F_6, w) < \alpha\beta + \gamma - 1$ and rearrange to obtain

$$(1 - 2\hat{a}_1)(1 - 2\hat{b}_1)(1 - \hat{c}_1)\hat{c}_1 + \hat{a}_1\hat{b}_1\hat{c}_1 < 0. \tag{1.15}$$

This implies (since $\hat{c}_1, 1 - \hat{c}_1, \hat{a}_1, \hat{b}_1 \geq 0$) that $0 < 1 - 2\hat{a}_1$ or $0 < 1 - 2\hat{b}_1$ (if $1 - 2\hat{a}_1 \leq 0$ and $1 - 2\hat{b}_1 \leq 0$ then the LHS of (1.15) would be non-negative).

If $0 < 1 - 2\hat{a}_1$ is true then substitute (1.11), (1.12), (1.13), (1.14), into $t(F_6, w) < \alpha\gamma + \beta - 1$ and rearrange to obtain

$$\hat{a}_1\hat{b}_1\hat{c}_1(2 - \hat{b}_1) + \hat{a}_1(1 - \hat{b}_1)^2(1 - \hat{c}_1) + (1 - 2\hat{a}_1)(1 - \hat{b}_1)(1 - \hat{c}_1) < 0.$$

But each term in the LHS is strictly non-negative so we have a contradiction.

If instead $0 < 1 - 2\hat{b}_1$ holds then looking at $t(F_6, w) < \beta\gamma + \alpha - 1$ yields

$$\hat{a}_1\hat{b}_1\hat{c}_1(2 - \hat{a}_1) + \hat{b}_1(1 - \hat{a}_1)^2(1 - \hat{c}_1) + (1 - 2\hat{b}_1)(1 - \hat{a}_1)(1 - \hat{c}_1) < 0,$$

which is similarly false. □

Proof of Lemma 1.4.16. Suppose (F_7, w) is extremal and vertex minimal. We may assume $w(v) \in (0,1)$ for all $v \in V(F_7)$. If $t(F_7, w) \geq \alpha\beta + \gamma - 1$ then by Lemma 1.4.7 (F_7, w) is not extremal, so we may assume that

$$t(F_7, w) < \alpha\beta + \gamma - 1, \tag{1.16}$$

and similarly

$$t(F_7, w) < \alpha\gamma + \beta - 1, \tag{1.17}$$
$$t(F_7, w) < \beta\gamma + \alpha - 1. \tag{1.18}$$

Consider moving all the weight from b_3 to b_2 to create the following weighting w' of F_7 defined formally as $w'(v) = w(v)$ for all $v \in V(G) \setminus \{b_2, b_3\}$, $w'(b_2) = w(b_2) + w(b_3)$, and $w'(b_3) = 0$. Changing the weighting from w to w' does not change the edge density between A and C, or B and C, but it may have increased the edge density between A and B and the triangle density. Let us call the new edge density, between A and B, γ'. Its value can be expressed in terms of the old weights and densities

$$\gamma' = \gamma + \hat{a}_2\hat{b}_3.$$

Similarly

$$t(F_7, w') = t(F_7, w) + \hat{a}_2\hat{b}_3\hat{c}_1.$$

If we can show that

$$t(F_7, w') < \alpha\beta + \gamma' - 1, \tag{1.19}$$
$$t(F_7, w') < \alpha\gamma' + \beta - 1, \tag{1.20}$$
$$t(F_7, w') < \beta\gamma' + \alpha - 1, \tag{1.21}$$

all hold then, since $w'(b_3) = 0$, we could remove b_3 from F_7 leaving all densities unchanged, and the resulting graph would be strongly-isomorphic to F_6. This contradicts Lemma 1.4.17, hence our assumption that (F_7, w) is extremal and vertex minimal must be false.

First let us show that (1.19) holds. Consider

$$\begin{aligned}\alpha\beta + \gamma' - 1 - t(F_7, w') &= \alpha\beta + (\gamma + \hat{a}_2\hat{b}_3) - 1 - (t(F_7, w) + \hat{a}_2\hat{b}_3\hat{c}_1) \\ &= \alpha\beta + \gamma - 1 - t(F_7, w) + \hat{a}_2\hat{b}_3(1 - \hat{c}_1) \\ &> 0.\end{aligned}$$

The inequality holds because $\alpha\beta+\gamma-1-t(F_7, w) > 0$ by (1.16) and $\hat{a}_2, \hat{b}_3, \hat{c}_1 \in (0, 1)$.

To prove (1.20) we look at

$$\begin{aligned}\alpha\gamma' + \beta - 1 - t(F_7, w') &= \alpha(\gamma + \hat{a}_2\hat{b}_3) + \beta - 1 - (t(F_7, w) + \hat{a}_2\hat{b}_3\hat{c}_1) \\ &= \alpha\gamma + \beta - 1 - t(F_7, w) + \hat{a}_2\hat{b}_3(\alpha - \hat{c}_1).\end{aligned}$$

We know $\alpha\gamma + \beta - 1 - t(F_7, w) > 0$ by (1.17), and $\hat{a}_2, \hat{b}_3 > 0$, so all we have to do is show that $\alpha - \hat{c}_1 \geq 0$. By definition α is the sum of the weighted edges between B and C, hence

$$\begin{aligned}\alpha &= (\hat{b}_2 + \hat{b}_3)\hat{c}_1 + \hat{b}_1\hat{c}_2 \\ &= (1 - \hat{b}_1)\hat{c}_1 + \hat{b}_1(1 - \hat{c}_1).\end{aligned}$$

Therefore

$$\begin{aligned}\alpha - \hat{c}_1 &= (1 - \hat{b}_1)\hat{c}_1 + \hat{b}_1(1 - \hat{c}_1) - \hat{c}_1 \\ &= \hat{b}_1(1 - 2\hat{c}_1).\end{aligned}$$

Since $\hat{b}_1$ is greater than zero, we require $\hat{c}_1 \leq 1/2$.

Consider $C_{a_1b_1} = \{c_2\}$ and $C_{a_2b_2} = \{c_1\}$. Construct $(F_7, w, p) \in \mathbf{DTri}$ by setting $p(e) = 1$ for all edges of F_7. If $\hat{c}_2 < \hat{c}_1$ then, by Lemma 1.4.4, we know we can achieve a smaller triangle density. Therefore $\hat{c}_1 \leq \hat{c}_2$ must hold, or equivalently $\hat{c}_1 \leq 1/2$ (as $\hat{c}_1 + \hat{c}_2 = 1$).

Similarly to prove (1.21) consider

$$\beta\gamma' + \alpha - 1 - t(F_7, w') = \beta\gamma + \alpha - 1 - t(F_7, w) + \hat{a}_2\hat{b}_3(\beta - \hat{c}_1).$$

By (1.18) we need only show $\beta-\hat{c}_1 \geq 0$, which is true because $\beta-\hat{c}_1 = \hat{a}_1(1-2\hat{c}_1)$, $\hat{a}_1 > 0$, and $\hat{c}_1 \leq 1/2$. □

Lemma 1.4.18. *For all weightings w such that $(F_9, w) \in$ **Tri**, (F_9, w) is either not extremal or not vertex minimal.*

Proof. Let us assume that (F_9, w) is extremal and vertex minimal, in which case $w(v) \neq 0$ for all $v \in V(F_9)$. Construct $(F_{10}, w') \in$ **Tri** from (F_9, w) as follows:

- Create F_{10} from F_9 by removing the edge a_3c_1. Add a new vertex into C, labelled c_4, and add in edges so that $\Gamma^{F_{10}}(c_4) = \Gamma_B^{F_9}(c_1) \cup A$.
- Set $w'(v) = w(v)$ for all $v \in V(F_{10}) \setminus \{c_1, c_4\}$. Let

$$w'(c_1) = \frac{w(a_1)w(c_1)}{w(a_1) + w(a_3)}, \qquad \text{and} \qquad w'(c_4) = \frac{w(a_3)w(c_1)}{w(a_1) + w(a_3)}.$$

The edge density between A and B remains unchanged and it is easy to check that the density between B and C also hasn't changed. The change in edge density between A and C is

$$w(a_2)w'(c_1) + w'(c_4) - w(a_2)w(c_1) - w(a_3)w(c_1) = 0.$$

The triangles in F_9 are $a_1b_3c_2$, $a_2b_1c_3$, $a_3b_2c_1$ and the triangles in F_{10} are $a_1b_3c_2$, $a_2b_1c_3$, $a_1b_2c_4$, $a_3b_2c_4$. Hence the change in triangle density between (F_9, w) and (F_{10}, w') is

$$\big(w(a_1) + w(a_3)\big)w(b_2)w'(c_4) - w(a_3)w(b_2)w(c_1) = 0.$$

Therefore (F_9, w) and (F_{10}, w') have the same triangle and edge densities.

Note that $\Gamma_C^{F_{10}}(a_1) = \Gamma_C^{F_{10}}(a_3) = \{c_2, c_4\}$. Since $|C| = 4$ we can apply the Reduce algorithm to class C in F_{10}, and the resultant output $(F'', w'') \in$ **Tri** has the same edge densities and the same triangle density (because (F_9, w) is extremal). Moreover $|V(F'')| = |V(F_9)|$ (as (F_9, w) is vertex minimal) and $\Gamma_C^{F''}(a_1) = \Gamma_C^{F''}(a_3)$. Hence we can apply Lemma 1.4.13 to (F'', w''), showing that it is either not extremal or not vertex minimal and so the same must be true of (F_9, w). □

Our goal is to produce a list of all tripartite graphs G for which there exists a weighting w such that $(G, w) \in$ **Tri**(α, β, γ) is extremal and vertex minimal for some $(\alpha, \beta, \gamma) \in R_2$. With this aim in mind we have developed a number of

results that allow us to show (G, w) is not extremal or not vertex minimal by simply examining G, *irrespective of the weighting* w.

By Lemmas 1.4.8 and 1.4.11 we need only consider tripartite graphs G in which all vertex classes contain either two or three vertices. This reduces the problem to a finite search. However, tripartite graphs with $|A| = |B| = |C| = 3$ can contain twenty-seven possible edges, so naively there are at least $2^{27} \approx 100,000,000$ graphs to consider. We can decrease the possible number of graphs by looking at only those that contain triangles, since otherwise $(\alpha, \beta, \gamma) \notin R$ by Theorem 1.2.1. By Lemma 1.4.14 we know that if G has a class containing a pair of vertices with identical neighbours then it is not vertex minimal (because we can move all the weight from one vertex to the other). Similarly the more technical results given in Corollary 1.4.5, Lemmas 1.4.10, 1.4.13, 1.4.15, 1.4.16, and 1.4.18 can also be used to eliminate graphs without knowledge of the vertex weights. Tripartite graphs that are strongly-isomorphic to graphs eliminated by these results will also not be extremal or not vertex minimal, and so may also be discarded.

Unfortunately applying Corollary 1.4.5, Lemmas 1.4.10, 1.4.13, 1.4.14, 1.4.15, 1.4.16, 1.4.18 and Theorem 1.2.1 to over $100,000,000$ tripartite graphs would take too long to perform by hand, but can easily be done by computer. A C++ implementation is given in Appendix A. This algorithm produces a list of possible extremal vertex minimal tripartite graphs in R_2, which are equivalent up to strong-isomorphism to the fourteen graphs, given in Figure 1.11. To decrease the number further we will have to check each of these graphs by hand.

1.4.3 Specific graphs

To complete the proof of Theorem 1.2.5 we need to eliminate the eleven graphs found by the computer search, other than H_7, H_7' and H_9. (In the list of fourteen graphs these are G_8, G_7 and G_{13} respectively.)

To be precise we will show that for each G_i, $1 \leq i \leq 14$, $i \neq 7, 8, 13$, if $(\alpha, \beta, \gamma) \in R_2$ then there does not exist a weighting w such that $(G_i, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ and (G_i, w) is both extremal and vertex minimal. Note that the proof of Lemma 1.4.29 which eliminates G_{14} is particularly interesting.

Lemma 1.4.19. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that $(G_1, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$, (G_1, w) is not extremal.*

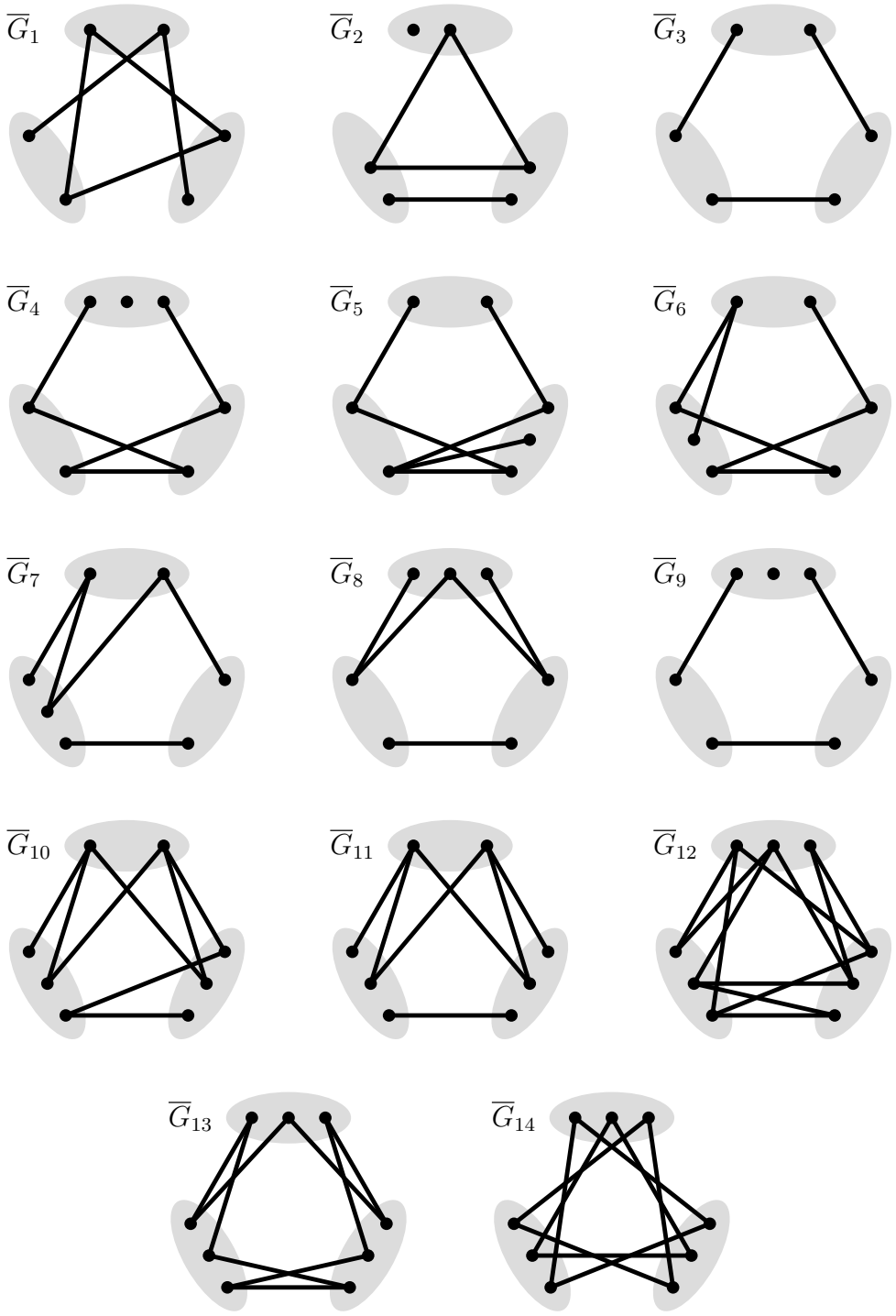

Figure 1.11: The tripartite complements of the graphs produced by the computer search.

Proof. G_1 is strongly-isomorphic to F_6. Hence Lemma 1.4.17 and Lemma 1.4.7 imply (G_1, w) is not extremal. □

Lemma 1.4.20. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that $(G_2, w) \in$ **Tri**(α, β, γ), (G_2, w) is not extremal.*

Proof. Suppose (G_2, w) is extremal, by Lemma 1.4.7 we must have $t(G_2, w) < \alpha\beta + \gamma - 1$. The edge and triangle densities are given by

$$\begin{aligned} \alpha &= \hat{b}_1 + (1 - \hat{b}_1)\hat{c}_1, \\ \beta &= 1 - \hat{a}_1 + \hat{a}_1\hat{c}_1, \\ \gamma &= \hat{a}_1\hat{b}_1 + (1 - \hat{a}_1)(1 - \hat{b}_1), \\ t(G_2, w) &= \hat{a}_1\hat{b}_1\hat{c}_1 + (1 - \hat{a}_1)(1 - \hat{b}_1)\hat{c}_1. \end{aligned}$$

Substituting into $t(G_2, w) < \alpha\beta + \gamma - 1$ and simplifying yields

$$\hat{a}_1(1 - \hat{b}_1)(1 - \hat{c}_1)(1 + \hat{c}_1) < 0$$

which is false. □

Lemma 1.4.21. *For $(\alpha, \beta, \gamma) \in R_2$ there exist no weightings w of G_3 such that $(G_3, w) \in$ **Tri**(α, β, γ).*

Proof. G_3 is strongly-isomorphic to H_6. Hence the result follows immediately from Lemma 1.3.4. □

Lemma 1.4.22. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that $(G_4, w) \in$ **Tri**(α, β, γ), (G_4, w) is either not extremal or not vertex minimal.*

Proof. Let us assume $(G_4, w) \in$ **Tri**(α, β, γ) is vertex minimal, and so $w(v) \neq 0$ for all $v \in V(G_4)$. By Lemma 1.3.3 (ii) we also have $\alpha, \beta, \gamma \neq 0, 1$. The densities in terms of the vertex weights $\hat{a}_1, \hat{b}_2, \hat{c}_1$, and $\hat{c}_3$, are as follows,

$$\begin{aligned} \alpha &= 1 - \hat{b}_2\hat{c}_1, \\ \beta &= 1 - \hat{a}_1\hat{c}_3, \\ \gamma &= \hat{a}_1\hat{b}_2, \\ t(G_4, w) &= \hat{a}_1\hat{b}_2(1 - \hat{c}_1 - \hat{c}_3). \end{aligned}$$

We can use these equations to write $\hat{a}_1, \hat{b}_2, \hat{c}_3$, and $t(G_4, w)$ in terms of $\hat{c}_1$,

$$\hat{b}_2 = \frac{1-\alpha}{\hat{c}_1}, \tag{1.22}$$

$$\hat{a}_1 = \frac{\gamma\hat{c}_1}{1-\alpha}, \tag{1.23}$$

$$\hat{c}_3 = \frac{(1-\alpha)(1-\beta)}{\gamma\hat{c}_1},$$

$$t(G_4, w) = \gamma - \gamma\hat{c}_1 - \frac{(1-\alpha)(1-\beta)}{\hat{c}_1}. \tag{1.24}$$

From (1.24) we can deduce that $t(G_4, w)$ will be minimized when $\hat{c}_1$ is as large or as small as possible, because the second derivative with respect to $\hat{c}_1$ is negative. Since $\hat{b}_2 \leq 1$ and $\hat{a}_1 \leq 1$, (1.22) and (1.23) imply that $\hat{c}_1 \in [1-\alpha, (1-\alpha)/\gamma]$.

Substituting $\hat{c}_1 = 1-\alpha$ into (1.24) gives $t(G_4, w) = \alpha\gamma + \beta - 1$. Substituting $\hat{c}_1 = (1-\alpha)/\gamma$ into (1.24) gives $t(G_4, w) = \beta\gamma + \alpha - 1$. Hence for $\hat{c}_1 \in [1-\alpha, (1-\alpha)/\gamma]$ we have

$$t(G_4, w) \geq \min\{\alpha\gamma + \beta - 1, \beta\gamma + \alpha - 1\}.$$

Lemma 1.4.7 therefore tells us that (G_4, w) can not be extremal. □

Lemma 1.4.23. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that $(G_5, w) \in$ **Tri**(α, β, γ), (G_5, w) is either not extremal or not vertex minimal.*

Proof. Suppose $(\alpha, \beta, \gamma) \in R_2$ and $(G_5, w) \in$ **Tri**(α, β, γ). We will show that there exists a weighting w' of G_4 such that $(G_4, w') \in$ **Tri**(α, β, γ) and $t(G_4, w') = t(G_5, w)$. Since $|V(G_4)| = |V(G_5)|$, Lemma 1.4.22 implies that (G_5, w) is either not extremal or not vertex minimal.

Suppose (G_5, w) is vertex minimal, in which case we may assume $w(v) > 0$ for all $v \in V(G_5)$. To prove there exists (G_4, w') with the same densities as (G_5, w), note that $\Gamma_B(a_1) = \Gamma_B(a_2)$ in G_5. Hence we can modify G_5 by applying Merge on a_1, a_2 labelling the resulting merged vertex by a. This creates one partial edge ac_2. Apply Split on this edge, to remove it, choosing to replace the vertex c_2. The resulting weighted tripartite graph has the same densities as (G_5, w) and it is easy to check that it is strongly-isomorphic to G_4. □

Lemma 1.4.24. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that $(G_6, w) \in$ **Tri**(α, β, γ), (G_6, w) is either not extremal or not vertex minimal.*

Proof. Suppose $(\alpha, \beta, \gamma) \in R_2$ and $(G_6, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$. We will show that there exists a weighting w' of G_5 such that $(G_5, w') \in \mathbf{Tri}(\alpha, \beta, \gamma)$ and $t(G_5, w') = t(G_6, w)$. Since $|V(G_5)| = |V(G_6)|$, Lemma 1.4.23 implies that (G_6, w) is either not extremal or not vertex minimal.

Suppose (G_6, w) is vertex minimal, in which case we may assume $w(v) > 0$ for all $v \in V(G_6)$. To prove there exists (G_5, w') with the same densities as (G_6, w), note that $\Gamma_C(b_2) = \Gamma_C(b_3)$ in G_6. Hence we can modify G_6 by applying Merge on b_2, b_3 labelling the resulting merged vertex b. This creates one partial edge a_2b. Apply Split on that edge, to remove it, choosing to replace the vertex a_2. The resulting weighted tripartite graph has the same densities as (G_6, w) and it is easy to check that it is strongly-isomorphic to G_5. □

Lemma 1.4.25. *For $(\alpha, \beta, \gamma) \in R_2$ there exist no weightings w of G_9 such that $(G_9, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$.*

Proof. Suppose $(G_9, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ for $(\alpha, \beta, \gamma) \in R_2$. If $w(c_2) = 0$ then removing c_2 leaves G_9 strongly-isomorphic to H_6. Hence we get a contradiction from Lemma 1.3.4. If $w(c_1) = 0$ or $w(b_2) = 0$ then $\alpha = 1$, and $(1, \beta, \gamma) \notin R_2$ by Lemma 1.3.3 (ii). Similarly we can show all other vertices must have a non-zero weight. We will get a contradiction by showing that $\Delta(\alpha, \beta, \gamma) \geq 0$ and hence $(\alpha, \beta, \gamma) \notin R_2$.

Consider a new weighting w' given by $w'(v) = w(v)$ for all $v \in V(G_9) \setminus \{c_1, c_2\}$, $w'(c_1) = w(c_1) + w(c_2)$, and $w'(c_2) = 0$. For convenience let us write $\alpha' = \alpha(G_9, w')$ (note that $\beta(G_9, w') = \beta$ and $\gamma(G_9, w') = \gamma$). Since $w'(c_2) = 0$ we could remove it from G_9 without changing any densities and the resulting weighted tripartite graph would be strongly-isomorphic to H_6, let w'' be the corresponding weighting. Since $w(v) \neq 0$ for all $v \in V(G_9)$ we know $w''(v) \neq 0$ for all $v \in V(H_6)$, and consequently $t(H_6, w'') > 0$. Lemma 1.3.2 tells us that $T_{\min}(\alpha', \beta, \gamma) = t(H_6, w'') > 0$, therefore by Theorem 1.2.1 we have $(\alpha', \beta, \gamma) \in R$. Moreover, Lemma 1.3.4 implies that $\Delta(\alpha', \beta, \gamma) \geq 0$.

Since $\alpha' = 1 - w'(b_2)w'(c_1) = 1 - w(b_2)w(c_1) - w(b_2)w(c_2)$, we have

$$\alpha' = \alpha - w(b_2)w(c_2).$$

Hence we can write $\alpha = \alpha' + \epsilon$, where $\epsilon = w(b_2)w(c_2) > 0$. Consider

$$\begin{aligned}\Delta(\alpha, \beta, \gamma) &= \Delta(\alpha' + \epsilon, \beta, \gamma) \\ &= \Delta(\alpha', \beta, \gamma) + 2\epsilon\alpha' + \epsilon^2 - 2\epsilon\beta - 2\epsilon\gamma + 4\epsilon\beta\gamma \\ &= \Delta(\alpha', \beta, \gamma) + \epsilon^2 + 2\epsilon(\alpha' + \beta + \gamma - 2) + 4\epsilon(1-\beta)(1-\gamma) \\ &= \Delta(\alpha', \beta, \gamma) + \epsilon^2 + 2\epsilon t(H_6, w'') + 4\epsilon(1-\beta)(1-\gamma).\end{aligned}$$

Since each term is non-negative we have $\Delta(\alpha, \beta, \gamma) \geq 0$. Therefore $(\alpha, \beta, \gamma) \notin R_2$, a contradiction. □

Lemma 1.4.26. *For all weightings w such that $(G_{10}, w) \in$ **Tri**, (G_{10}, w) is either not extremal or not vertex minimal.*

Proof. Suppose (G_{10}, w) is extremal and vertex minimal, hence $w(v) \neq 0$ for all $v \in V(G_{10})$. Convert (G_{10}, w) into a doubly-weighted tripartite graph by adding the function p which maps all edges to 1. Applying Merge on (G_{10}, w, p) and b_2, b_3, results in only one partial edge being created bc_2 (where b is the vertex replacing b_2, b_3). We can apply Split on that edge choosing to replace the vertex c_2, and then revert back to a weighted graph (G'_{10}, w') say. Now (G'_{10}, w') has the same densities as (G_{10}, w) but G'_{10} has $|B| = 2$ and $|C| = 3$. Moreover, G_{10} has the property that $\Gamma_B(a_1) = \Gamma_B(a_3)$, and this is also true in G'_{10}. Hence applying Lemma 1.4.13 to (G'_{10}, w') and a_1, a_3, we see that (G'_{10}, w') is not extremal or not vertex minimal. Since $|V(G'_{10})| = |V(G_{10})|$ the same is true of (G_{10}, w). □

Lemma 1.4.27. *For all weightings w such that $(G_{11}, w) \in$ **Tri**, (G_{11}, w) is either not extremal or not vertex minimal.*

Proof. The proof is almost identical to that of Lemma 1.4.26. The only difference being at the end, where now we have $\Gamma_B(a_1) = \Gamma_B(a_2)$ holding true, and so we apply Lemma 1.4.13 to vertices a_1 and a_2 instead. □

Lemma 1.4.28. *For all weightings w such that $(G_{12}, w) \in$ **Tri**, (G_{12}, w) is either not extremal or not vertex minimal.*

Proof. Suppose (G_{12}, w) is vertex minimal, so $w(v) > 0$ for all $v \in V(G_{12})$. Of the three statements $\hat{a}_1 \leq \hat{a}_2$, $\hat{b}_1 \leq \hat{b}_2$, $\hat{c}_1 \leq \hat{c}_2$, at least two must be true or at least two must be false. Without loss of generality let us suppose that $\hat{a}_1 \leq \hat{a}_2$, $\hat{b}_1 \leq \hat{b}_2$ are both true.

The densities of (G_{12}, w) are given by

$$\begin{aligned} t(G_{12}, w) &= \hat{a}_3\hat{b}_3\hat{c}_3, \\ \alpha &= \hat{b}_1\hat{c}_2 + \hat{b}_2\hat{c}_1 + \hat{c}_3, \\ \beta &= \hat{a}_1\hat{c}_2 + \hat{a}_2\hat{c}_1 + \hat{a}_3, \\ \gamma &= \hat{a}_1\hat{b}_2 + \hat{a}_2\hat{b}_1 + \hat{b}_3. \end{aligned}$$

Consider the doubly-weighted tripartite graph (G_{12}, w, p) where p maps all edges to 1. It has the same densities as (G_{12}, w). If we move a sufficiently small amount of weight $\delta > 0$ from vertex c_2 to c_1, α and β increase. By decreasing $p(b_3c_3)$ and $p(a_3c_3)$ respectively we can keep all edge densities unchanged. More precisely set

$$p(b_3c_3) = 1 - \delta(\hat{b}_2 - \hat{b}_1)/\hat{b}_3\hat{c}_3, \qquad p(a_3c_3) = 1 - \delta(\hat{a}_2 - \hat{a}_1)/\hat{a}_3\hat{c}_3.$$

If $\hat{a}_1 = \hat{a}_2$ and $\hat{b}_1 = \hat{b}_2$, then increasing the weight of c_1 to $\hat{c}_1 + \hat{c}_2$ and removing c_2 will result in a weighted tripartite graph with the same densities as (G_{12}, w) but with fewer vertices. Hence we know that $p(b_3c_3) < 1$ or $p(a_3c_3) < 1$. Consequently we now have a doubly-weighted tripartite graph with the same edge densities as (G_{12}, w) but a strictly smaller triangle density. Hence by Lemma 1.4.1 (G_{12}, w), is not extremal.

Suppose now that two of the statements $\hat{a}_1 \leq \hat{a}_2$, $\hat{b}_1 \leq \hat{b}_2$, $\hat{c}_1 \leq \hat{c}_2$, are false, for example $\hat{a}_1 > \hat{a}_2$ and $\hat{b}_1 > \hat{b}_2$. We can repeat the above argument, this time moving weight from c_1 to c_2, again constructing a doubly-weighted tripartite graph with the same edge densities but a smaller triangle density. □

Lemma 1.4.29. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that $(G_{14}, w) \in$ **Tri**(α, β, γ), (G_{14}, w) is either not extremal or not vertex minimal.*

Proof. Suppose (G_{14}, w) is extremal and vertex minimal, so $w(v) > 0$ for all $v \in V(G_{14})$. Consider the doubly-weighted tripartite graph (G_{14}, w, p), where p maps all edges to 1. Applying Lemma 1.4.4 to (G_{14}, w, p) on the non-edge a_1b_1 and the edge a_3b_2 tells us that in order to be extremal

$$\sum_{c \in C_{a_1b_1}} w(c) \geq \sum_{c \in C_{a_3b_2}} w(c)$$

must hold. Since $C_{a_1b_1} = \{c_2, c_3\}$ and $C_{a_3b_2} = \{c_1\}$ we must have $\hat{c}_2 + \hat{c}_3 \geq \hat{c}_1$

or equivalently $1 - 2\hat{c}_1 \geq 0$ (using the fact that $\hat{c}_1 + \hat{c}_2 + \hat{c}_3 = 1$). Similarly we can show that $1 - 2\hat{c}_2 \geq 0$ by looking at a_2b_2, a_1b_3, and $1 - 2\hat{c}_3 \geq 0$ by taking a_3b_3, a_2b_1. By symmetry we must have $1 - 2w(v) \geq 0$ for all $v \in V(G_{14})$. Note that the function w' defined by $w'(v) = 1 - 2w(v)$ for all $v \in V(G_{14})$ provides a valid weighting of G_{14}, as $w'(v) \geq 0$ for all $v \in V(G_{14})$ and the sum of the weights in a class, X say, is

$$\begin{aligned}\sum_{v \in X} w'(v) &= \sum_{v \in X} (1 - 2w(v)) \\ &= |X| - 2\sum_{v \in X} w(v) \\ &= |X| - 2 \\ &= 1\end{aligned}$$

because every class in G_{14} has size three.

Recall that $\overline{G}_{14}$ is the tripartite complement of the graph G_{14}. Consider the weighted tripartite graph $(\overline{G}_{14}, w')$, with edge densities

$$\alpha(\overline{G}_{14}, w') = \alpha', \quad \beta(\overline{G}_{14}, w') = \beta', \quad \gamma(\overline{G}_{14}, w') = \gamma'.$$

We can write down α' in terms of α.

$$\begin{aligned}\alpha' &= (1 - 2\hat{b}_1)(1 - 2\hat{c}_1) + (1 - 2\hat{b}_2)(1 - 2\hat{c}_2) + (1 - 2\hat{b}_3)(1 - 2\hat{c}_3) \\ &= 3 - 2(\hat{b}_1 + \hat{b}_2 + \hat{b}_3) - 2(\hat{c}_1 + \hat{c}_2 + \hat{c}_3) + 4(\hat{b}_1\hat{c}_1 + \hat{b}_2\hat{c}_2 + \hat{b}_3\hat{c}_3) \\ &= 3 - 4(1 - \hat{b}_1\hat{c}_1 - \hat{b}_2\hat{c}_2 - \hat{b}_3\hat{c}_3) \\ &= 3 - 4\alpha,\end{aligned}$$

similarly $\beta' = 3 - 4\beta$, and $\gamma' = 3 - 4\gamma$. Next let us write $t(\overline{G}_{14}, w')$ in terms of $t(G_{14}, w)$,

$$
\begin{aligned}
t(\overline{G}_{14}, w') &= (1-2\hat{a}_1)(1-2\hat{b}_1)(1-2\hat{c}_1) + (1-2\hat{a}_2)(1-2\hat{b}_2)(1-2\hat{c}_2) + \\
&\quad (1-2\hat{a}_3)(1-2\hat{b}_3)(1-2\hat{c}_3) \\
&= 3 - 2(\hat{a}_1+\hat{a}_2+\hat{a}_3) - 2(\hat{b}_1+\hat{b}_2+\hat{b}_3) - 2(\hat{c}_1+\hat{c}_2+\hat{c}_3) + \\
&\quad 4(\hat{a}_1\hat{b}_1 + \hat{a}_1\hat{c}_1 + \hat{b}_1\hat{c}_1 + \hat{a}_2\hat{b}_2 + \hat{a}_2\hat{c}_2 + \hat{b}_2\hat{c}_2 + \hat{a}_3\hat{b}_3 + \hat{a}_3\hat{c}_3 + \hat{b}_3\hat{c}_3) - \\
&\quad 8(\hat{a}_1\hat{b}_1\hat{c}_1 + \hat{a}_2\hat{b}_2\hat{c}_2 + \hat{a}_3\hat{b}_3\hat{c}_3) \\
&= 1 + 4(\hat{a}_1\hat{b}_1 + \hat{a}_1\hat{c}_1 + \hat{b}_1\hat{c}_1 + \hat{a}_2\hat{b}_2 + \hat{a}_2\hat{c}_2 + \hat{b}_2\hat{c}_2 + \hat{a}_3\hat{b}_3 + \hat{a}_3\hat{c}_3 + \hat{b}_3\hat{c}_3 - \\
&\quad 2\hat{a}_1\hat{b}_1\hat{c}_1 - 2\hat{a}_2\hat{b}_2\hat{c}_2 - 2\hat{a}_3\hat{b}_3\hat{c}_3 - \hat{a}_1 - \hat{a}_2 - \hat{a}_3) \\
&= 1 + 4((1-\hat{a}_1)\hat{b}_1\hat{c}_1 + (1-\hat{a}_2)\hat{b}_2\hat{c}_2 + (1-\hat{a}_3)\hat{b}_3\hat{c}_3 - \\
&\quad \hat{a}_1(1-\hat{b}_1)(1-\hat{c}_1) - \hat{a}_2(1-\hat{b}_2)(1-\hat{c}_2) - \hat{a}_3(1-\hat{b}_3)(1-\hat{c}_3)) \\
&= 1 + 4((\hat{a}_2+\hat{a}_3)\hat{b}_1\hat{c}_1 + (\hat{a}_1+\hat{a}_3)\hat{b}_2\hat{c}_2 + (\hat{a}_1+\hat{a}_2)\hat{b}_3\hat{c}_3 - \\
&\quad \hat{a}_1(\hat{b}_2+\hat{b}_3)(\hat{c}_2+\hat{c}_3) - \hat{a}_2(\hat{b}_1+\hat{b}_3)(\hat{c}_1+\hat{c}_3) - \hat{a}_3(\hat{b}_1+\hat{b}_2)(\hat{c}_1+\hat{c}_2)) \\
&= 1 + 4(-\hat{a}_1\hat{b}_2\hat{c}_3 - \hat{a}_1\hat{b}_3\hat{c}_2 - \hat{a}_2\hat{b}_1\hat{c}_3 - \hat{a}_2\hat{b}_3\hat{c}_1 - \hat{a}_3\hat{b}_1\hat{c}_2 - \hat{a}_3\hat{b}_2\hat{c}_1) \\
&= 1 - 4t(G_{14}, w).
\end{aligned}
$$

Without loss of generality suppose $\alpha' \leq \beta' \leq \gamma'$. Since (G_{14}, w) is extremal by Lemma 1.4.6 we have

$$t(G_{14}, w) \leq 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2.$$

Rewriting in terms of $\alpha', \beta', \gamma', t(\overline{G}_{14}, w')$ gives

$$3 + 2\gamma' - t(\overline{G}_{14}, w') \leq \sqrt{(3-\alpha')(3-\beta')(1+\gamma')}.$$

Note that in any weighted tripartite graph the triangle density is bounded above by all of the edge densities, thus $t(\overline{G}_{14}, w') \leq \alpha'$, and so

$$3 + 2\gamma' - \alpha' \leq \sqrt{(3-\alpha')(3-\beta')(1+\gamma')}.$$

Squaring both sides and rearranging yields

$$\alpha'^2 + \gamma'(4\gamma' - \alpha'\beta') + \gamma'(3\beta' - \alpha') + 3(\gamma' - \alpha') + \beta'(3-\alpha') \leq 0.$$

Each term is non-negative (because $0 \leq \alpha' \leq \beta' \leq \gamma' \leq 1$), and so the only

way this can be true is if $\alpha' = \beta' = \gamma' = 0$. Hence $\alpha = \beta = \gamma = 3/4$, but such values do not lie in R_2 due to the fact that $\Delta(3/4, 3/4, 3/4) = 0$. Thus we have a contradiction and our assumption that (G_{14}, w) is extremal and vertex minimal must be false. □

Proof of Theorem 1.2.5. Our computer search tells us that the only possible extremal and vertex minimal tripartite graphs are strongly-isomorphic to those given in Figure 1.11. Given $(\alpha, \beta, \gamma) \in R_2$ for all weightings w, (G_1, w), (G_2, w), (G_3, w), (G_4, w), (G_5, w), (G_6, w), (G_9, w), (G_{10}, w), (G_{11}, w), (G_{12}, w), (G_{14}, w) are either not extremal, not vertex minimal, or do not lie in $\mathbf{Tri}(\alpha, \beta, \gamma)$ by Lemmas 1.4.19 to 1.4.29 respectively. This just leaves G_7, G_8, and G_{13} which are strongly-isomorphic to H_7', H_7 and H_9 respectively. □

1.5 Guaranteeing a triangle or a 5-cycle

Bondy, Shen, Thomassé, and Thomassen [4] showed that any tripartite graph with edge densities α, β, γ must contain a triangle if $(\alpha, \beta, \gamma) \in R$, where

$$R = \{(\alpha, \beta, \gamma) \in [0,1]^3 : \alpha\beta + \gamma > 1, \alpha\gamma + \beta > 1, \beta\gamma + \alpha > 1\}.$$

By setting $\alpha = \beta = \gamma$ we see that if the edge densities between each pair of classes in a tripartite graph is greater than $1/\varphi = 0.618\ldots$ then the graph must contain a triangle. A triangle-free tripartite graph with edge densities 0.618 is possible by taking an appropriate blow-up of the 5-cycle. Hence a natural question to ask is: if our tripartite graph has no triangles or 5-cycles how large can the minimum density between pairs of classes be?

It is easy to show that blow-ups of triangle and 5-cycle-free graphs are still triangle and 5-cycle-free. Hence any results for weighted tripartite graphs will hold for tripartite graphs. Using weighted tripartite graphs we can characterize those edge densities that guarantee the existence of a triangle or 5-cycle.

Theorem 1.5.1. *If $(G, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ and $(\alpha, \beta, \gamma) \in R'$ then G contains a triangle or 5-cycle, where*

$$R' = \{(\alpha, \beta, \gamma) \in [0,1]^3 : 2\alpha\beta - \alpha - \beta + \gamma > 0,\ \ 2\alpha\gamma - \alpha - \gamma + \beta > 0,\ \ 2\beta\gamma - \beta - \gamma + \alpha > 0\}.$$

If $(\alpha, \beta, \gamma) \in [0,1]^3 \setminus R'$*, then there exists* $(G, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ *such that* G *does not contain a triangle or* 5*-cycle.*

Again by setting $\alpha = \beta = \gamma$ we get the following simple corollary.

Corollary 1.5.2. *If* G *is a tripartite graph with edge densities* $\alpha, \beta, \gamma > 1/2$ *then* G *contains a triangle or* 5*-cycle.*

Note that the value of $1/2$ is best possible due to the following construction. Take a tripartite graph with two vertices in each class, colour one vertex red and the other blue. Add edges between every blue and red vertex (provided they do not lie in the same class). This graph is bipartite (as seen by its red-blue-colouring) and so cannot contain a triangle or 5-cycle. Also it is easy to check that the edge density is $1/2$ between every pair of classes.

A simple extension of this result is that every k-partite graph with $k \geq 3$ and pairwise edge density greater than $1/2$, contains a triangle or 5-cycle. This is trivially implied by Corollary 1.5.2. An edge density of $1/2$ is again best possible. This can be seen by considering a k-partite graph with two vertices in each class one coloured red and the other blue, and whose edges are precisely those between the red and blue vertices.

To prove Theorem 1.5.1 we need the following modified version of the Reduce algorithm, which we will call Reduce2. This appears as part of the proof of Theorem 3 in [4].

Algorithm 1.5.3 (Reduce2)**.** The algorithm Reduce2 takes as input $(G, w) \in \mathbf{Tri}$ and a vertex class X of G, satisfying $|X| > 2$. Its output, represented by Reduce2(G, w, X), is a weighted tripartite graph with $|X| \leq 2$, and edge densities at least as big as those of (G, w). Furthermore if G is triangle and 5-cycle-free so is the underlying tripartite graph of Reduce2(G, w, X).

To help explain the algorithm we will suppose $X = A$, (the other choices of X work similarly). For each vertex $a_i \in A$ let

$$\beta_i = \sum_{c \in \Gamma_C(a_i)} w(c), \quad \gamma_i = \sum_{b \in \Gamma_B(a_i)} w(b).$$

By definition

$$\beta = \sum_{i=1}^{|A|} w(a_i)\beta_i, \quad \gamma = \sum_{i=1}^{|A|} w(a_i)\gamma_i.$$

Consider the convex hull

$$P = \left\{ \sum_{i=1}^{|A|} x_i(\beta_i, \gamma_i) : \sum_{i=1}^{|A|} x_i = 1 \text{ and } x_i \geq 0 \right\}.$$

Setting $x_i = w(a_i)$ shows that (β, γ) lies in P. By varying the values of the x_i we can increase the value of β to β' such that (β', γ) lies on the boundary of P. Consequently we can express (β', γ) as a convex combination of at most two elements of $\{(\beta_i, \gamma_i) : 1 \leq i \leq |A|\}$. Hence we can write

$$(\beta', \gamma) = \sum_{i=1}^{|A|} x_i(\beta_i, \gamma_i)$$

where $\sum x_i = 1$ and at most two of the x_i are positive, the rest are zero. Now define a new weighting w' for G by $w'(a_i) = x_i$, $w'(v) = w(v)$ for $v \in V(G) \setminus A$. The weighted tripartite graph (G, w') has edge densities at least as big as those of (G, w). Furthermore we can remove the zero weighted vertices from A so that $|A| \leq 2$ (this does not affect the densities). Since the underlying graph is G with some vertices removed, if G is triangle and 5-cycle-free so is $\mathsf{Reduce2}(G, w, X)$.

The edge densities of a weighted tripartite graph may increase after an application of the Reduce2 algorithm, consequently we will require the following lemma.

Lemma 1.5.4. *If $(\alpha, \beta, \gamma) \in R'$ and $\alpha', \beta', \gamma' \in [0, 1]$ satisfy $\alpha' \geq \alpha$, $\beta' \geq \beta$, $\gamma' \geq \gamma$ then $(\alpha', \beta', \gamma') \in R'$.*

Proof. It is enough to show that $(\alpha, \beta, \gamma) \in R'$ implies $(\alpha', \beta, \gamma) \in R'$. The same argument can be used to show that $(\alpha', \beta, \gamma) \in R'$ implies $(\alpha', \beta', \gamma) \in R'$, and that $(\alpha', \beta', \gamma) \in R'$ implies $(\alpha', \beta', \gamma') \in R'$. To prove that (α', β, γ) lies in R' we need to show α', β, γ satisfy

$$2\alpha'\beta - \alpha' - \beta + \gamma > 0, \;\; 2\alpha'\gamma - \alpha' - \gamma + \beta > 0, \;\; 2\beta\gamma - \beta - \gamma + \alpha' > 0.$$

We will begin by showing $2\alpha'\beta - \alpha' - \beta + \gamma > 0$. Since $(\alpha, \beta, \gamma) \in R'$ we know that

$$0 < (2\alpha\beta - \alpha - \beta + \gamma) + (2\alpha\gamma - \alpha - \gamma + \beta) = 2\alpha(\beta + \gamma - 1),$$

hence $\beta+\gamma-1>0$. Consider the following rearrangement of $2\alpha\beta-\alpha-\beta+\gamma$:

$$\alpha(2\beta-1)-\beta+\gamma. \tag{1.25}$$

If $2\beta-1\geq 0$ then increasing α to α' will not cause a decrease in (1.25), and therefore $2\alpha'\beta-\alpha'-\beta+\gamma>0$. If $2\beta-1<0$ then (1.25) will decrease, in the worst case (when $\alpha'=1$) to $\beta+\gamma-1$. However, we have shown that $\beta+\gamma-1>0$ and so $2\alpha'\beta-\alpha'-\beta+\gamma>0$ still holds.

We omit the proof of $2\alpha'\gamma-\alpha'-\gamma+\beta>0$ as it follows an almost identical argument. Proving $2\beta\gamma-\beta-\gamma+\alpha'>0$ is trivial given that $\alpha'\geq\alpha$ and $(\alpha,\beta,\gamma)\in R'$ implies $2\beta\gamma-\beta-\gamma+\alpha>0$. Hence (α',β,γ) lies in R'. □

Proof of Theorem 1.5.1. First we will show that if $(G,w)\in\mathbf{Tri}(\alpha,\beta,\gamma)$ and $(\alpha,\beta,\gamma)\in R'$ then G contains a triangle or 5-cycle. Suppose there exists a counterexample $(G,w)\in\mathbf{Tri}(\alpha,\beta,\gamma)$ with G triangle and 5-cycle-free, and $(\alpha,\beta,\gamma)\in R'$. If G has more than two vertices in any class we can apply Reduce2 to ensure that G has at most two vertices in each class. Lemma 1.5.4 tells us that even if the Reduce2 increases the edge densities we will still be left with a counterexample. If G has one vertex in a class, we can add a vertex and give it a weight of zero (this will not affect the edge densities). Hence we may assume that G has precisely two vertices in each class.

Since G has six vertices and does not contain a triangle or 5-cycle, it does not contain an odd cycle and so must be 2-colourable. Given a 2-colouring of G we can assume that all edges between different coloured vertices are present (unless such an edge lies in one of the vertex classes). Adding in such edges cannot create a triangle or 5-cycle and can only increase the edge densities (which does not pose a problem due to Lemma 1.5.4). Finding all edge maximal 2-colourable tripartite graphs with two vertices in each class, is simple and easily doable by hand. There are six such graphs (up to strong-isomorphism) and they are given in Figure 1.12, we will refer to them as $S_1,\ldots,S_6$.

Due to the symmetric nature of R' we do not need to consider all the graphs strongly-isomorphic to $S_1,\ldots,S_6$. It is enough to show that for any weighting w and any S_i $(i=1,\ldots,6)$ the edge densities cannot lie in R'. We will continue to use the canonical labelling as illustrated in Figure 1.9 to refer to the classes, vertices, and weights of $S_1,\ldots,S_6$.

For all weightings w, (S_1,w), (S_2,w), and (S_3,w) have $\gamma=0$. It is easy to

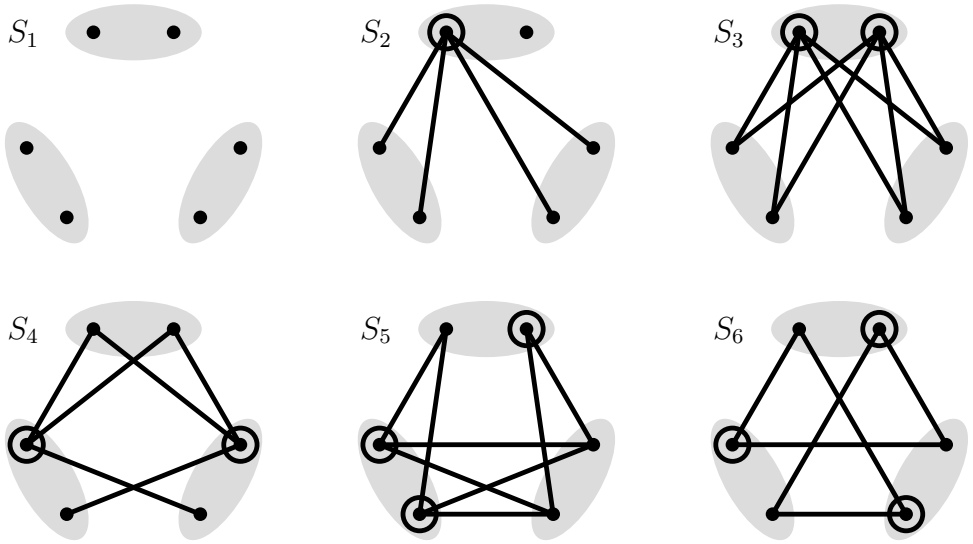

Figure 1.12: All edge maximal 2-colourable tripartite graphs, with two vertices in each class, up to strong-isomorphism. The colour classes, red-blue say, are highlighted by circles around the red coloured vertices, and no circles around the blue vertices.

check that $(\alpha, \beta, 0) \in R'$ implies $-\alpha+\beta > 0$ and $-\beta+\alpha > 0$, therefore (S_1, w), (S_2, w), and (S_3, w) do not have densities which lie in R'.

For S_4, $\alpha = \hat{b}_2, \beta = \hat{a}_1$, and $\gamma = \hat{b}_2(1-\hat{a}_1)+\hat{a}_1(1-\hat{b}_2) = \alpha+\beta-2\alpha\beta$. Hence $2\alpha\beta - \alpha - \beta + \gamma = 0$ and consequently $(\alpha, \beta, \gamma) \notin R'$.

For S_5, $\alpha = \hat{c}_1, \beta = 1 - \hat{c}_1 = 1 - \alpha$, and $\gamma = 1$. Substituting $\gamma = 1$ and $\beta = 1 - \alpha$ into $2\alpha\gamma - \alpha - \gamma + \beta$ shows that it is 0, proving the densities do not lie in R'.

The last graph we have to consider is S_6. Its densities are

$$\begin{aligned}
\alpha &= \hat{b}_1(1 - \hat{c}_1) + \hat{c}_1(1 - \hat{b}_1), \\
\beta &= \hat{a}_1(1 - \hat{c}_1) + \hat{c}_1(1 - \hat{a}_1), \\
\gamma &= \hat{a}_1(1 - \hat{b}_1) + \hat{b}_1(1 - \hat{a}_1).
\end{aligned}$$

By the pigeonhole principle two of $1-2\hat{a}_1$, $1-2\hat{b}_1$, $1-2\hat{c}_1$ must be negative or two must be non-negative. Without loss of generality suppose $(1-2\hat{a}_1)(1-2\hat{b}_1) \geq 0$.

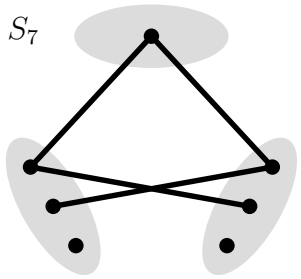

Figure 1.13: The tripartite graph S_7 that can always be weighted to achieve any edge densities satisfying $2\alpha\beta - \alpha - \beta + \gamma \leq 0$.

This is enough to show $(\alpha, \beta, \gamma) \notin R'$, as

$$2\alpha\beta - \alpha - \beta + \gamma = -2(1 - 2\hat{a}_1)(1 - 2\hat{b}_1)\hat{c}_1(1 - \hat{c}_1) \leq 0.$$

Hence there does not exist a triangle and 5-cycle-free $(G, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ with $(\alpha, \beta, \gamma) \in R'$. Next we will show that the region R' is best possible by proving for any $(\alpha, \beta, \gamma) \in [0, 1]^3 \setminus R'$ there exists $(G, w) \in \mathbf{Tri}(\alpha, \beta, \gamma)$ which is triangle and 5-cycle-free.

If $(\alpha, \beta, \gamma) \in [0, 1]^3 \setminus R'$, then one of the three inequalities

$$2\alpha\beta - \alpha - \beta + \gamma > 0, \;\; 2\alpha\gamma - \alpha - \gamma + \beta > 0, \;\; 2\beta\gamma - \beta - \gamma + \alpha > 0,$$

must not hold. Without loss of generality suppose $2\alpha\beta - \alpha - \beta + \gamma \leq 0$. The tripartite graph S_7 given in Figure 1.13 can be weighted to achieve any edge densities satisfying $2\alpha\beta - \alpha - \beta + \gamma \leq 0$. Specifically the weighting w defined as follows,

$$\begin{aligned}
&w(a_1) = \beta, && w(a_2) = \lambda(1 - \beta), && w(a_3) = (1 - \lambda)(1 - \beta),\\
&w(b_1) = (1 - \lambda)(1 - \alpha), && w(b_2) = \lambda(1 - \alpha), && w(b_3) = \alpha,\\
&w(c_1) = 1,
\end{aligned}$$

where $\lambda = 0$ if $\alpha + \beta - 2\alpha\beta = 0$ and $\lambda = \gamma/(\alpha + \beta - 2\alpha\beta)$ otherwise. It is easy to check that w is a valid weighting. □

1.6 Conjectures

The following conjecture, if true, would allow us to write $T_{\min}(\alpha, \beta, \gamma)$ as a simple expression for all values of $\alpha, \beta, \gamma \in [0, 1]$.

Conjecture 1.6.1. *For $\gamma \leq \alpha, \beta$,*

$$T_{\min}(\alpha, \beta, \gamma) = \begin{cases} 0, & \text{if } (\alpha, \beta, \gamma) \in [0,1]^3 \setminus R, \\ 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2, & \text{if } (\alpha, \beta, \gamma) \in R_2, \\ \alpha + \beta + \gamma - 2, & \text{otherwise.} \end{cases}$$

To prove Conjecture 1.6.1 it is sufficient to prove the subsequent conjecture.

Conjecture 1.6.2. *If $(\alpha, \beta, \gamma) \in R_2$ then for all weightings w such that (H_9, w) $\in$ **Tri**(α, β, γ), (H_9, w) is either not extremal or not vertex minimal.*

Theorem 1.6.3. *Conjecture 1.6.2 implies Conjecture 1.6.1.*

Proof. Theorems 1.2.1 and 1.2.4 tell us when $T_{\min}(\alpha, \beta, \gamma) = 0$ and $\alpha+\beta+\gamma-2$ respectively. By Theorem 1.2.5 and Conjecture 1.6.2 we know that the only extremal tripartite graphs we have to consider are H_7 and H_7'. Let us show that H_7' can do no better than H_7.

Let $(\alpha, \beta, \gamma) \in R_2$ and $(H_7', w') \in$ **Tri**(α, β, γ). We need to show there exists a weighting w for H_7 so that (H_7, w) has the same densities as (H_7', w'). Note that $\Gamma_A(b_2) = \Gamma_A(b_3)$ in H_7' and $w'(b_2) + w'(b_3) > 0$ (otherwise $\alpha = 1$ which can not occur according to Lemma 1.3.3 (ii)). Hence we can modify H_7' by applying Merge on b_2, b_3 labelling the resulting merged vertex b. This creates one partial edge bc_2. Apply Split to this edge to remove it, choosing to replace the vertex c_2. The resulting weighted tripartite graph has the same densities as (H_7', w') and it is easy to check that it is strongly-isomorphic to H_7.

Therefore when $(\alpha, \beta, \gamma) \in R_2$ we need only consider the graphs strongly-isomorphic to H_7, and by Lemma 1.4.6 we get $T_{\min}(\alpha, \beta, \gamma)$ is equal to

$$\min\{2\sqrt{\alpha\beta(1-\gamma)}+2\gamma-2, 2\sqrt{\alpha\gamma(1-\beta)}+2\beta-2, 2\sqrt{\beta\gamma(1-\alpha)}+2\alpha-2\}.$$

To finish the proof let us show that $\gamma \leq \beta$ if and only if

$$2\sqrt{\alpha\gamma(1-\beta)} + 2\beta - 2 \geq 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2.$$

We can prove a similar result for $\gamma \leq \alpha$. For ease of notation let $d_1 = 2\sqrt{\alpha\gamma(1-\beta)} + 2\beta - 2$ and $d_2 = \alpha + \beta + \gamma - 2$. So we have

$$\begin{aligned}
& & d_1 &\geq 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2 \\
\iff & & d_1 + 2(1-\gamma) &\geq 2\sqrt{\alpha\beta(1-\gamma)} \\
\iff & & (d_1 + 2(1-\gamma))^2 &\geq 4\alpha\beta(1-\gamma) \\
& & &= (d_2 + 2(1-\gamma))^2 - \Delta(\alpha,\beta,\gamma) \\
\iff & & d_1^2 + 4d_1(1-\gamma) &\geq d_2^2 + 4d_2(1-\gamma) - \Delta(\alpha,\beta,\gamma) \\
\iff & d_1^2 + 4d_1 - d_2^2 - 4d_2 + \Delta(\alpha,\beta,\gamma) &\geq 4\gamma(d_1 - d_2)
\end{aligned}$$

By Lemma 1.3.1 we know $d_1 - d_2 \geq 0$. It is easy to check that $d_1 - d_2 = 0$ implies $\Delta(\alpha,\beta,\gamma) \geq 0$ which is not true, since $(\alpha,\beta,\gamma) \in R_2$. Consequently we have

$$d_1 \geq 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2 \iff \frac{d_1^2 + 4d_1 - d_2^2 - 4d_2 + \Delta(\alpha,\beta,\gamma)}{4(d_1 - d_2)} \geq \gamma.$$

Substituting $d_1 = 2\sqrt{\alpha\gamma(1-\beta)} + 2\beta - 2$ and $d_2 = \alpha + \beta + \gamma - 2$ into

$$\frac{d_1^2 + 4d_1 - d_2^2 - 4d_2 + \Delta(\alpha,\beta,\gamma)}{4(d_1 - d_2)}$$

shows that it simplifies to β. Thus

$$2\sqrt{\alpha\gamma(1-\beta)} + 2\beta - 2 \geq 2\sqrt{\alpha\beta(1-\gamma)} + 2\gamma - 2 \iff \beta \geq \gamma.$$

□

In this chapter we have dealt primarily with triangles in tripartite graphs, but a natural generalisation is to consider at what edge density a 4-partite graph must contain a K_4 (the complete graph on four vertices). Nagy [28] makes the following conjecture.

Conjecture 1.6.4 (Nagy [28])**.** *If the edge density between every pair of classes in a 4-partite graph is greater than* $0.7722\ldots$ *then the graph must contain a* K_4*.*

Chapter 2

Turán Densities and Razborov's Flag Algebras

2.1 Introduction

An *r-uniform hypergraph* (or *r-graph* for short) is a pair $F = (V(F), E(F))$ where $V(F)$ is a set of *vertices* and $E(F)$ is a family of r-subsets of $V(F)$ called *edges*. So a 2-graph is a simple graph. For ease of notation we often identify an r-graph F with its edge set. The *density* of an r-graph F is

$$d(F) = \frac{|E(F)|}{\binom{n}{r}}.$$

Given a family of r-graphs $\mathcal{F}$ we say that an r-graph H is *$\mathcal{F}$-free* if H does not contain a subgraph isomorphic to any member of $\mathcal{F}$. For any integer $n \geq 1$ we define the *Turán number* of $\mathcal{F}$ to be

$$\mathrm{ex}(n, \mathcal{F}) = \max\{|E(H)| : H \text{ is } \mathcal{F}\text{-free}, \ |V(H)| = n\}.$$

The *Turán density* of $\mathcal{F}$ is defined to be the following limit (a simple averaging argument shows that it always exists)

$$\pi(\mathcal{F}) = \lim_{n\to\infty} \frac{\mathrm{ex}(n, \mathcal{F})}{\binom{n}{r}}.$$

In this chapter we describe the recent work of Razborov [31] on flag algebras that introduces a new technique that drastically improves our ability to compute

(and approximate) Turán densities of hypergraphs. We outline the necessary background in the next section but emphasize that the reader should consult Razborov [29] and [31] for a full description of his work.

We apply Razborov's method to Erdős' jumping hypergraph problem to find the first non-trivial regions of jumps (Theorem 2.3.1 and Corollary 2.3.5). We then use it to determine the exact Turán densities of five hypergraphs (Theorem 2.4.1). We finish by extending the method to Turán type problems in the hypercube, proving a number of new results. These include a significantly smaller bound on the Turán density of 4-cycle-free subgraphs of hypercubes (a problem posed by Erdős), and a new exact Turán density for hypercubes (Theorems 2.5.1 and 2.5.2 respectively).

2.2 Computing Turán densities via flag algebras

2.2.1 Razborov's method

Let $\mathcal{F}$ be a family of r-graphs whose Turán density we wish to compute (or at least approximate). Razborov [31], describes a method for attacking this problem that can be thought of as a general application of Cauchy–Schwarz using the information given by small $\mathcal{F}$-free r-graphs.

Let $\mathcal{H}$ be the family of all $\mathcal{F}$-free r-graphs of order l, up to isomorphism. If l is sufficiently small we can explicitly determine $\mathcal{H}$ (by computer search if necessary).

For $H \in \mathcal{H}$ and a large $\mathcal{F}$-free r-graph G, we define $p(H;G)$ to be the probability that a random l-set from $V(G)$ induces a subgraph isomorphic to H. Trivially, the density of G is equal to the probability that a random r-set from $V(G)$ forms an edge in G. Thus, averaging over l-sets in $V(G)$, we can express the density of G as

$$d(G) = \sum_{H \in \mathcal{H}} d(H)p(H;G), \tag{2.1}$$

and hence $d(G) \leq \max_{H \in \mathcal{H}} d(H)$.

This "averaging" bound on $d(G)$ is in general rather poor: clearly it could only be sharp if all subgraphs of G of order l are as dense as possible. It also

fails to consider how different subgraphs of G can overlap. Razborov's flag algebras method allows us to make use of the information given by examining overlapping subgraphs of G to give far stronger bounds.

A *flag*, $F = (G_F, \theta)$, is an r-graph G_F together with an injective map $\theta : [s] \to V(G_F)$. If θ is bijective (and so $|V(G_F)| = s$) we call the flag a *type*. For ease of notation given a flag $F = (G_F, \theta)$ we define its order $|F|$ to be $|V(G_F)|$.

Given a type σ we call a flag $F = (G_F, \theta)$ a *σ-flag* if the induced labelled subgraph of G_F given by θ is σ. A flag $F = (G_F, \theta)$ is *admissible* if G_F is $\mathcal{F}$-free.

Fix a type σ and an integer $m \leq (l + |\sigma|)/2$. (The bound on m ensures that an l-vertex r-graph can contain two m-vertex subgraphs overlapping in $|\sigma|$ vertices.) Let $\mathcal{F}^\sigma_m$ be the set of all admissible σ-flags of order m, up to isomorphism. Let Θ be the set of all injective functions from $[|\sigma|]$ to $V(G)$. Given $F \in \mathcal{F}^\sigma_m$ and $\theta \in \Theta$ we define $p(F, \theta; G)$ to be the probability that an m-set V' chosen uniformly at random from $V(G)$ subject to $\mathrm{im}(\theta) \subseteq V'$, induces a σ-flag $(G[V'], \theta)$ that is isomorphic to F.

If $F_a, F_b \in \mathcal{F}^\sigma_m$ and $\theta \in \Theta$ then $p(F_a, \theta; G)p(F_b, \theta; G)$ is the probability that two m-sets $V_a, V_b \subseteq V(G)$, chosen independently at random subject to $\mathrm{im}(\theta) \subseteq V_a \cap V_b$, induce σ-flags $(G[V_a], \theta)$, $(G[V_b], \theta)$ that are isomorphic to F_a, F_b respectively. We define a related probability, $p(F_a, F_b, \theta; G)$, to be the probability that if we choose a random m-set $V_a \subseteq V(G)$, subject to $\mathrm{im}(\theta) \subseteq V_a$ and then choose a random m-set $V_b \subseteq V(G)$ such that $V_a \cap V_b = \mathrm{im}(\theta)$ then $(G[V_a], \theta)$, $(G[V_b], \theta)$ are isomorphic to F_a, F_b respectively. Note that the difference between $p(F_a, \theta; G)p(F_b, \theta; G)$ and $p(F_a, F_b, \theta; G)$ is due to the effect of sampling *with* or *without* replacement. When G is large this difference will be negligible, as the following lemma tells us. (This is a very special case of Lemma 2.3 in [29].)

Lemma 2.2.1 (Razborov [29])**.** *For any $F_a, F_b \in \mathcal{F}^\sigma_m$, and $\theta \in \Theta$,*

$$p(F_a, \theta; G)p(F_b, \theta; G) = p(F_a, F_b, \theta; G) + o(1),$$

where the $o(1)$ term tends to zero as $|V(G)|$ tends to infinity.

Proof. Choose random m-sets $V_a, V_b \subseteq V(G)$, independently, subject to $\mathrm{im}(\theta) \subseteq V_a \cap V_b$. Let E be the event that $V_a \cap V_b = \mathrm{im}(\theta)$. Then

$$p(F_a, F_b, \theta; G)\mathbf{P}[E] \leq p(F_a, \theta; G)p(F_b, \theta; G) \leq p(F_a, F_b, \theta; G)\mathbf{P}[E] + \mathbf{P}[\bar{E}].$$

If $|V(G)| = n$ then

$$\mathbf{P}[E] = \frac{\binom{n-|\sigma|}{m-|\sigma|}\binom{n-m}{m-|\sigma|}}{\binom{n-|\sigma|}{m-|\sigma|}^2} = 1 - o(1).$$

□

Averaging over a uniformly random choice of $\theta \in \Theta$ we have

$$\mathbf{E}_{\theta\in\Theta}\left[p(F_a, \theta; G)p(F_b, \theta; G)\right] = \mathbf{E}_{\theta\in\Theta}\left[p(F_a, F_b, \theta; G)\right] + o(1). \tag{2.2}$$

Note that this expectation can be computed by averaging over l-vertex subgraphs of G. For an l-vertex subgraph $H \in \mathcal{H}$ let Θ_H be the set of all injective maps $\theta : [|\sigma|] \to V(H)$. Recall that, for $H \in \mathcal{H}$, $p(H; G)$ is the probability that a random l-set from $V(G)$ induces a subgraph isomorphic to H. Thus,

$$\mathbf{E}_{\theta\in\Theta}\left[p(F_a, F_b, \theta; G)\right] = \sum_{H\in\mathcal{H}} \mathbf{E}_{\theta\in\Theta_H}\left[p(F_a, F_b, \theta; H)\right] p(H; G). \tag{2.3}$$

Consider a positive semidefinite matrix $Q = (q_{ab})$ of dimension $|\mathcal{F}_m^\sigma|$. For $\theta \in \Theta$ define $\mathbf{p}_\theta = (p(F, \theta; G) : F \in \mathcal{F}_m^\sigma)$. Using (2.2), (2.3) and linearity of expectation we have

$$\mathbf{E}_{\theta\in\Theta}[\mathbf{p}_\theta^T Q \mathbf{p}_\theta] = \sum_{F_a, F_b\in\mathcal{F}_m^\sigma} \sum_{H\in\mathcal{H}} q_{ab}\mathbf{E}_{\theta\in\Theta_H}[p(F_a, F_b, \theta; H)]p(H; G) + o(1). \tag{2.4}$$

For $H \in \mathcal{H}$ define the coefficient of $p(H; G)$ in (2.4) by

$$c_H(\sigma, m, Q) = \sum_{F_a, F_b\in\mathcal{F}_m^\sigma} q_{ab}\mathbf{E}_{\theta\in\Theta_H}[p(F_a, F_b, \theta; H)]. \tag{2.5}$$

Suppose we have t choices of (σ_i, m_i, Q_i), where each σ_i is a type, each $m_i \le (l + |\sigma_i|)/2$ is an integer, and each Q_i is a positive semidefinite matrix of dimension $|\mathcal{F}_{m_i}^{\sigma_i}|$. For $H \in \mathcal{H}$ define

$$c_H = \sum_{i=1}^{t} c_H(\sigma_i, m_i, Q_i).$$

Note that c_H is independent of G.

Since each Q_i is positive semidefinite (2.4) implies that

$$\sum_{H \in \mathcal{H}} c_H p(H; G) + o(1) \geq 0.$$

Thus, using (2.1), we have

$$d(G) \leq \sum_{H \in \mathcal{H}} (d(H) + c_H) p(H; G) + o(1).$$

Hence the Turán density satisfies

$$\pi(\mathcal{F}) \leq \max_{H \in \mathcal{H}} (d(H) + c_H). \tag{2.6}$$

Since the c_H may be negative, for an appropriate choice of the (σ_i, m_i, Q_i), this bound may be significantly better than the trivial averaging bound given by (2.1).

Note that we now have a semidefinite programming problem: given any particular choice of the (σ_i, m_i) find positive semidefinite matrices Q_i so as to minimize the bound for $\pi(\mathcal{F})$ given by (2.6).

2.2.2 An example

We now illustrate Razborov's method with a simple example. Let $K_4^- = \{123, 124, 134\}$. We will reprove de Caen's [7] bound: $\pi(K_4^-) \leq 1/3$.

Let $l = 4$, so $\mathcal{H}$ consists of all K_4^--free 3-graphs of order four, up to isomorphism. There are three such 3-graphs which we will refer to as H_0, H_1, and H_2, they have 0, 1, and 2 edges respectively (this is enough information to uniquely identify them). We will use a single type: $\sigma = (G_\sigma, \theta)$ where $V(G_\sigma) = [2]$, $E(G_\sigma) = \emptyset$ and $\theta(x) = x$. Taking $m = 3$, there are only two admissible σ-flags of order three up to isomorphism: F_0 and F_1, containing no edges and one edge respectively.

In order to calculate the coefficients c_H we need to compute $\mathbf{E}_{\theta \in \Theta_H}[p(F_a, F_b, \theta; H)]$, for each $H \in \{H_0, H_1, H_2\}$ and each pair $F_a, F_b \in \{F_0, F_1\}$. Their values are given in the following table.

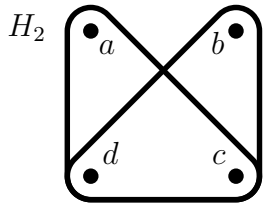

Figure 2.1: The 3-graph H_2, with vertices labelled a, b, c, d. Its two edges are acd and bcd.

	H_0	H_1	H_2
F_0, F_0	1	1/2	1/6
F_0, F_1	0	1/4	1/3
F_1, F_1	0	0	1/6

As an example of how these values are computed consider $\mathbf{E}_{\theta\in\Theta_{H_2}}[p(F_0, F_1, \theta; H_2)]$. This is the probability that a random choice of $\theta \in \Theta_{H_2}$ and 3-sets $V_0, V_1 \subset V(H_2)$ such that $V_0 \cap V_1 = \mathrm{im}(\theta)$, induce σ-flags $(H_2[V_0], \theta), (H_2[V_1], \theta)$ that are isomorphic to F_0, F_1 respectively. A random of choice of $\theta \in \Theta_{H_2}$ is equivalent to picking a random ordered pair of vertices (u, v) from H_2, and setting $\theta(1) = u$ and $\theta(2) = v$. To form the random 3-sets V_0, V_1 we pick the remaining two vertices of $V(H_2) \setminus \{u, v\}$ randomly in the order x, y and set $V_0 = \{u, v, x\}, V_1 = \{u, v, y\}$. The σ-flags $(H_2[V_0], \theta), (H_2[V_1], \theta)$ are isomorphic to F_0, F_1 if and only if $V_0 \notin E(H_2)$ and $V_1 \in E(H_2)$ respectively. Consequently $\mathbf{E}_{\theta\in\Theta_{H_2}}[p(F_0, F_1, \theta; H_2)]$ is the probability that a random permutation (u, v, x, y) of $V(H_2)$ satisfies $\{u, v, x\} \notin E(H_2)$ and $\{u, v, y\} \in E(H_2)$. Of the twenty-four permutations of $V(H_2) = \{a, b, c, d\}$, see Figure 2.1, the following eight have this property:

$$\begin{array}{llll} (a,c,b,d), & (a,d,b,c), & (b,c,a,d), & (b,d,a,c), \\ (c,a,b,d), & (d,a,b,c), & (c,b,a,d), & (d,b,a,c). \end{array}$$

Hence $\mathbf{E}_{\theta\in\Theta_{H_2}}[p(F_0, F_1, \theta; H_2)] = 8/24 = 1/3$.

We now need to find a positive semidefinite matrix

$$Q = \begin{pmatrix} q_{00} & q_{01} \\ q_{01} & q_{11} \end{pmatrix},$$

to minimize the bound given by (2.6). Note that

$$\begin{aligned} c_{H_0} &= q_{00}, \\ c_{H_1} &= \frac{1}{2}q_{00} + \frac{1}{2}q_{01}, \\ c_{H_2} &= \frac{1}{6}q_{00} + \frac{2}{3}q_{01} + \frac{1}{6}q_{11}. \end{aligned}$$

The bound on $\pi(K_4^-)$ given by (2.6) is now

$$\pi(K_4^-) \leq \max\left\{q_{00}, \quad \frac{q_{00}}{2} + \frac{q_{01}}{2} + \frac{1}{4}, \quad \frac{q_{00}}{6} + \frac{2q_{01}}{3} + \frac{q_{11}}{6} + \frac{1}{2}\right\}.$$

This can be expressed as a semidefinite programming problem. The solution to which is

$$Q = \frac{1}{3}\begin{pmatrix} 1 & -2 \\ -2 & 4 \end{pmatrix}.$$

Consequently $\pi(K_4^-) \leq \max\{1/3, 1/12, 1/3\} = 1/3$.

2.2.3 Solving the semidefinite program

Razborov's method as outlined above reduces the problem of computing an upper bound on a Turán density to solving a semidefinite programming problem. In practice this may be computationally difficult. Razborov [31] describes a number of ways that this problem can be simplified so as to make the computation more tractable. We outline one of these ideas below, which we made use of in our work.

For a type σ and the collection of all admissible σ-flags of order m, $\mathcal{F}_m^\sigma$ define $\mathbb{R}\mathcal{F}_m^\sigma$ to be the real vector space of formal linear combinations of σ-flags of order m. Let $\mathcal{H}$ be the collection of all admissible r-graphs of order l.

Let us introduce Razborov's $[\![\cdot]\!]_\sigma$ notation (which will make our expressions easier to read). Define $[\![\cdot]\!]_\sigma : \mathbb{R}\mathcal{F}_m^\sigma \times \mathbb{R}\mathcal{F}_m^\sigma \to \mathbb{R}^{|\mathcal{H}|}$, by

$$[\![F_a F_b]\!]_\sigma = (\mathbf{E}_{\theta\in\Theta_H}[p(F_a, F_b, \theta; H)] : H \in \mathcal{H}),$$

for $F_a, F_b \in \mathcal{F}_m^\sigma$ and extend to be bilinear.

For a positive semidefinite matrix Q and $\mathbf{p} = (F : F \in \mathcal{F}_m^\sigma)$, the vector of

all admissible σ-flags (in an arbitrary but fixed order), we have

$$[\![\mathbf{p}^T Q \mathbf{p}]\!]_\sigma = (c_H(\sigma, m, Q) : H \in \mathcal{H}),$$

where the c_H are as defined in (2.5).

Razborov [31] describes a natural change of basis for $\mathbb{R}\mathcal{F}_m^\sigma$. The important property (in terms of reducing the computational complexity of the associated semidefinite program) is that the new basis is of the form $\mathcal{B} = \mathcal{B}^+ \dot\cup \mathcal{B}^-$ and for all $B^+ \in \mathcal{B}^+$ and $B^- \in \mathcal{B}^-$ we have $[\![B^+B^-]\!]_\sigma = \mathbf{0}$. Thus in our new basis the corresponding semidefinite program has a solution Q' which is a block diagonal matrix with two blocks: of sizes $|\mathcal{B}^+|$ and $|\mathcal{B}^-|$ respectively. Since the best algorithms for solving semidefinite programs scale like the square of the size of block matrices this change of basis can potentially simplify our computation significantly.

For a type $\sigma = (G_\sigma, \theta_\sigma)$ we construct the basis $\mathcal{B}$ as follows. First construct Γ_σ, the automorphism group of σ, whose elements are bijective maps $\alpha : [|\sigma|] \to [|\sigma|]$ such that $(G_\sigma, \theta_\sigma\alpha)$ is isomorphic to σ. The elements of Γ_σ act on σ-flags in an obvious way: for $\alpha \in \Gamma_\sigma$ and σ-flag $F = (G_F, \theta_F)$ we define $F\alpha$ to be the σ-flag $(G_F, \theta_F\alpha)$. Define subspaces

$$\mathbb{R}\mathcal{F}_m^{\sigma+} = \{L \in \mathbb{R}\mathcal{F}_m^\sigma : L\alpha = L \ \forall \alpha \in \Gamma_\sigma\}$$

and

$$\mathbb{R}\mathcal{F}_m^{\sigma-} = \{L \in \mathbb{R}\mathcal{F}_m^\sigma : \sum_{\alpha\in\Gamma_\sigma} L\alpha = \mathbf{0}\}.$$

Below we describe how to find bases $\mathcal{B}^+, \mathcal{B}^-$ for these subspaces. By the construction of these bases it will be clear that $\mathbb{R}\mathcal{F}_m^\sigma = \mathbb{R}\mathcal{F}_m^{\sigma+} \oplus \mathbb{R}\mathcal{F}_m^{\sigma-}$. Finally we will verify that for all $B^+ \in \mathcal{B}^+$ and $B^- \in \mathcal{B}^-$ we have $[\![B^+B^-]\!]_\sigma = \mathbf{0}$.

We start with the canonical basis for $\mathbb{R}\mathcal{F}_m^\sigma$ given by $\mathcal{F}_m^\sigma = \{F_1, F_2, \ldots, F_t\}$. For each $F_i \in \mathcal{F}_m^\sigma$ define the orbit of F_i under Γ_σ by

$$F_i\Gamma_\sigma = \{F\alpha : \alpha \in \Gamma_\sigma\}.$$

Any two orbits are either equal or disjoint. Suppose there are u distinct orbits: $O_1, \ldots, O_u$. For $i \in [u]$ let $B_i^+ = \sum_{F\in O_i} F$. Then $\mathcal{B}^+ = \{B_1^+, \ldots, B_u^+\}$ is easily seen to be a basis for $\mathbb{R}\mathcal{F}_m^{\sigma+}$. Moreover if $O_i = \{F_{i_1}, \ldots, F_{i_q}\}$ then $F_{i_1} - F_{i_z} \in$

$\mathbb{R}\mathcal{F}_m^{\sigma-}$ for $2 \le z \le q$ and the union of all such vectors forms a basis $\mathcal{B}^-$ for $\mathbb{R}\mathcal{F}_m^{\sigma-}$.

We now need to check that if $B^+ \in \mathcal{B}^+$ and $B^- \in \mathcal{B}^-$ then $[\![B^+B^-]\!]_\sigma = \mathbf{0}$. If $B^- \in \mathcal{B}^-$ then by construction $B^- = F_b\alpha - F_b$ for some $F_b \in \mathcal{F}_m^\sigma$ and $\alpha \in \Gamma_\sigma$. Moreover $B^+\alpha = B^+$. Hence, by linearity,

$$[\![B^+B^-]\!]_\sigma = [\![B^+(F_b\alpha - F_b)]\!]_\sigma = [\![(B^+\alpha)(F_b\alpha) - B^+F_b]\!]_\sigma.$$

We observe that for any $F_a \in \mathcal{F}_m^\sigma$

$$\begin{aligned}[\![(F_a\alpha)(F_b\alpha)]\!]_\sigma &= (\mathbf{E}_{\theta\in\Theta_H}[p(F_a, F_b, \theta\alpha^{-1}; H)] : H \in \mathcal{H}) \\ &= (\mathbf{E}_{\theta\in\Theta_H\alpha^{-1}}[p(F_a, F_b, \theta; H)] : H \in \mathcal{H})\end{aligned}$$

where $\Theta_H\alpha^{-1} = \{\theta\alpha^{-1} : \theta \in \Theta_H\}$. Since $\Theta_H\alpha^{-1} = \Theta_H$ we must have $[\![(F_a\alpha)(F_b\alpha)]\!]_\sigma = [\![F_aF_b]\!]_\sigma$. Thus, since $B^+ = F_{a_1} + F_{a_2} + \cdots + F_{a_s}$, we have $[\![(B^+\alpha)(F_b\alpha) - B^+F_b]\!]_\sigma = \mathbf{0}$, and hence $[\![B^+B^-]\!]_\sigma = \mathbf{0}$.

2.3 Hypergraphs do jump

We say that $\alpha \in [0, 1)$ is a *jump* for an integer $r \ge 2$ if there exists $c(\alpha) > 0$ such that for all $\epsilon > 0$ and all $t \ge 1$ there exists $n_0(\alpha, \epsilon, t)$ such that any r-graph with $n \ge n_0(\alpha, \epsilon, t)$ vertices and at least $(\alpha + \epsilon)\binom{n}{r}$ edges contains a subgraph on t vertices with at least $(\alpha + c)\binom{t}{r}$ edges.

The Erdős–Stone–Simonovits theorem [13], [14] implies that for $r = 2$ every $\alpha \in [0, 1)$ is a jump. Erdős [10] showed that for all $r \ge 3$, every $\alpha \in [0, r!/r^r)$ is a jump. Note that $r!/r^r$ is the asymptotic density of an equally partitioned r-partite r-graph. Erdős went on to make his famous "jumping constant conjecture" that for all $r \ge 3$, every $\alpha \in [0, 1)$ is a jump. Frankl and Rödl [18] disproved this conjecture by giving a sequence of values of non-jumps for all $r \ge 3$. More recently a number of authors have given more examples of non-jumps for each $r \ge 3$ in the interval $[5r!/2r^r, 1)$ (see [17] for example). However nothing was previously known regarding the location of jumps or non-jumps in the interval $[r!/r^r, 5r!/2r^r)$ for any $r \ge 3$.

We give the first examples of jumps for any $r \ge 3$ in the interval $[r!/r^r, 1)$.

Theorem 2.3.1. *If $\alpha \in [0.2299, 0.2316)$ then α is a jump for $r = 3$.*

In order to explain our proof we require some definitions and a theorem of Frankl and Rödl [18].

Let F be an r-graph with vertex set $[n] = \{1, 2, \ldots, n\}$ and edge set $E(F)$. Define

$$S_n = \{(x_1, \ldots, x_n) \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1, x_i \geq 0\}.$$

For $x \in S_n$ let

$$\lambda(F, x) = \sum_{\{i_1, i_2, \ldots, i_r\} \in E(F)} r! x_{i_1} x_{i_2} \cdots x_{i_r}.$$

The *Lagrangian* of F is defined to be

$$\lambda(F) = \max_{x \in S_n} \lambda(F, x).$$

We say that α is *threshold* for $\mathcal{F}$ if $\pi(\mathcal{F}) \leq \alpha$.

Theorem 2.3.2 (Frankl and Rödl [18])**.** *The following are equivalent:*

(i) α *is a jump for* r*.*

(ii) α *is threshold for a finite family* $\mathcal{F}$ *of* r*-graphs satisfying*

$$\min_{F \in \mathcal{F}} \lambda(F) > \alpha.$$

We can prove (i) implies (ii), by considering the family

$$\mathcal{F} = \{F : |V(F)| = t, d(F) \geq \alpha + c/2\}$$

where t is some sufficiently large integer. Erdős [10] tells us that the Turán density of a hypergraph is the same as that of its "blow-ups" which proves (ii) implies (i).

Let F_r be the r-graph consisting of a single edge. Since any $\alpha \in [0, 1)$ is threshold for F_r and $\lambda(F_r) = r!/r^r$, Theorem 2.3.2 trivially implies Erdős' result [10] that for each $r \geq 3$, every $\alpha \in [0, r!/r^r)$ is a jump for r.

The original version of Erdős' jumping constant conjecture asserted that $r!/r^r$ is a jump for every $r \geq 3$. This fascinating problem is still open, even for $r = 3$. Erdős speculated [10] that $3!/3^3 = 2/9$ was threshold for the following

family of 3-graphs $\mathcal{F}^* = \{F_1, F_2, F_3\}$, where

$$F_1 = \{123, 124, 134\}, F_2 = \{123, 124, 125, 345\}, F_3 = \{123, 124, 235, 145, 345\}.$$

It is straightforward to check that $\lambda(F_1) = 8/27$, $\lambda(F_2) = \frac{189+15\sqrt{5}}{961}$ and $\lambda(F_3) = 6/25$. Since $\min_{1\leq i\leq 3} \lambda(F_i) = \lambda(F_2) > 2/9$, if $2/9$ were threshold for $\mathcal{F}^*$ then Theorem 2.3.2 would imply $2/9$ is a jump for $r = 3$.

Unfortunately Erdős' suggestion is incorrect: $2/9$ is not threshold for $\mathcal{F}^*$. There exist seven vertex 3-graphs that are $\mathcal{F}^*$-free with Lagrangians greater than $2/9$. By taking appropriate "blow-ups" of such 3-graphs we find that $\pi(\mathcal{F}^*) > 2/9$. (To be precise we could take blow-ups of F_4, defined below, to show that $\pi(\mathcal{F}^*) \geq 0.2319$.) However Erdős' idea suggests a natural approach to proving that $2/9$ is a jump for $r = 3$. Let $\mathcal{F}'$ be a family of 3-graphs containing F_1, F_2, F_3 with the property that $\min_{F\in\mathcal{F}'} \lambda(F) > 2/9$. If we can show that $2/9$ is threshold for $\mathcal{F}'$ then (by Theorem 2.3.2) $2/9$ is a jump for $r = 3$.

A search of all 3-graphs with at most seven vertices yields the following two additional 3-graphs which we can add to $\mathcal{F}'$

$$F_4 = \{123, 135, 145, 245, 126, 246, 346, 356, 237, 147, 347, 257, 167\},$$

$$F_5 = \{123, 124, 135, 145, 236, 346, 256, 456, 247, 347, 257, 357, 167\}.$$

It is easy to check that $\lambda(F_4) \geq 0.2319 > \lambda(F_2)$ (to see this set $x_1 = x_2 = x_3 = 0.164, x_4 = 0.154, x_5 = x_6 = x_7 = 0.118$) and $\lambda(F_5) \geq \lambda(F_2)$ (set $\mu = \frac{18-3\sqrt{5}}{31}$, $x_1 = x_6 = x_7 = \mu/3, x_2 = x_3 = x_4 = x_5 = (1-\mu)/4$).

We can now ask: is it true that $2/9$ is threshold for $\mathcal{F}' = \{F_1, F_2, F_3, F_4, F_5\}$? Unfortunately this is still false, there exist 3-graphs on eight vertices avoiding all members of $\mathcal{F}'$ and with Lagrangians greater than $2/9$. By taking appropriate "blow-ups" of such 3-graphs we can show that $\pi(\mathcal{F}') > 2/9$. Moreover, by considering eight vertex 3-graphs, numerical evidence suggests that if $2/9$ is a jump then the size of the jump is extremely small: $c(2/9) \leq 0.00009254$.

However, although $2/9$ is not threshold for $\mathcal{F}'$ we can show the following upper bound on the Turán density of $\mathcal{F}'$.

Lemma 2.3.3. *The Turán density of $\mathcal{F}'$ satisfies $\pi(\mathcal{F}') \leq 0.2299$.*

Since $0.2299 < \min_{F\in\mathcal{F}'} \lambda(F) = \lambda(F_2) = 0.2316$, Theorem 2.3.1 is an immediate corollary of Lemma 2.3.3 and Theorem 2.3.2.

Proof. To prove $\pi(\mathcal{F}') \leq 0.2299$, we use Razborov's flag algebras method as outlined in Section 2.2. We set $l = 7$, so $\mathcal{H}$ consists of all seven vertex 3-graphs that do not contain any $F \in \mathcal{F}'$, up to isomorphism. There are 4042 such 3-graphs, which are explicitly determined by the C++ program DensityBounder, see Appendix B. To calculate the coefficients c_H we take six choices of (σ_i, m_i, Q_i). The types are $\sigma_i = ((V_i, E_i), \theta_i)$, where

$$\begin{aligned} V_1 &= [1], & E_1 &= \emptyset, \\ V_2 &= [3], & E_2 &= \emptyset, \\ V_3 &= [3], & E_3 &= \{123\}, \\ V_4 &= [5], & E_4 &= \{123, 124, 135\}, \\ V_5 &= [5], & E_5 &= \{123, 124, 345\}, \\ V_6 &= [5], & E_6 &= \{123, 124, 135, 245\}, \end{aligned}$$

and $\theta_i : [|V_i|] \to V_i$, maps $x \mapsto x$. Ideally we would use all types of size at most $l - 2 = 5$, however this yields a computationally intractable semidefinite program. Our actual choice was made by experiment, in each case taking the value of $m_i = \lfloor (7 + |\sigma_i|)/2 \rfloor$. DensityBounder determines the positive semidefinite matrices Q_i by creating a semidefinite programming problem. Several implementations of semidefinite program solvers exist. We chose the CSDP library [5] to solve the problem. The CSDP library uses floating point arithmetic which may introduce rounding errors. DensityBounder takes the output of the CSDP program and uses it to construct the Q_i (removing any rounding errors). Our results can however be verified without needing to solve a semidefinite program: DensityBounder can load pre-computed matrices Q_i from the file `HypergraphsDoJump.soln` which can be found on the accompanying CD-ROM, see Appendix B.

For each $H \in \mathcal{H}$, $d(H)$ and c_H are calculated by DensityBounder which it then uses to show that 0.2299 is an upper bound for $\pi(\mathcal{F}')$. Note that although floating point operations are used by the semidefinite program solver, our final computer proof consists of positive semidefinite matrices with rational coefficients and our proof can be verified using only integer operations, thus there is no issue of numerical accuracy. □

The program DensityBounder can be used to calculate upper bounds on the

Turán density of other families of 3-graphs. The conjectured value of $\pi(K_4^-)$ is $2/7 = 0.2857$. Razborov [31] showed that $\pi(K_4^-) \leq 0.2978$. Using Density-Bounder we obtain a new upper bound of 0.2871 by taking $l = 7$ and considering the following four types $\sigma_i = ((V_i, E_i), \theta_i)$ with the given values of m_i (in each case θ_i is the identity map):

$$\begin{aligned} V_1 &= [3], \quad E_1 = \emptyset, & m_1 &= 5, \\ V_2 &= [3], \quad E_2 = \{123\}, & m_2 &= 5, \\ V_3 &= [4], \quad E_3 = \{123\}, & m_3 &= 5, \\ V_4 &= [5], \quad E_4 = \{123, 124, 125\}, & m_4 &= 6. \end{aligned}$$

As before the positive semidefinite matrices Q_i are determined by solving a semidefinite programming problem.

Theorem 2.3.4. *Let K_4^- be the 3-graph on four vertices with three edges. The Turán density of K_4^- satisfies*

$$0.2857\ldots = \frac{2}{7} \leq \pi(K_4^-) \leq 0.2871.$$

As with our main result our computations can be verified without any floating point operations so there is no issue of numerical accuracy in these results. Theorem 2.3.4 yields a second new interval of jumps for $r = 3$.

Corollary 2.3.5. *If $\alpha \in [0.2871, 8/27)$ then α is a jump for $r = 3$.*

Proof. Since $\lambda(K_4^-) = 8/27$, this follows directly from Theorem 2.3.4 and Theorem 2.3.2. □

2.3.1 Open problems

We have shown that $[0.2299, 0.2316)$ is an interval of jumps for $r = 3$. If we were able to compute $\pi(\mathcal{F}')$ precisely we could quite possibly extend this interval below 0.2299. However, as noted in the introduction, we know that $\pi(\mathcal{F}') > 2/9$ so our approach could never resolve the most important open question in this area: is $2/9$ a jump?

Indeed the question of whether $2/9$ is a jump for $r = 3$ seems remarkably difficult to resolve. If $2/9$ is a jump then the size of this jump is very small and so to give a proof along the same lines as the proof of Theorem 2.3.1 would appear to require a very precise approximation of the Turán density of some

unknown family of 3-graphs. On the other hand the only current technique for showing a value is *not* a jump is to follow the method of Frankl and Rödl [18], but this trivially fails for 2/9 (or indeed $r!/r^r$ for any $r \geq 3$).

Another obvious open problem is to compute $\pi(K_4^-)$ exactly. It is likely that improvements over our bound of 0.2871 could be made by applying Razborov's method with larger flags.

2.4 Some exact Turán densities

2.4.1 Introduction

Goldwasser [19] proved that $\pi(S_1) = 3/4$, where

$$S_1 = \{234, 125, 135, 145, 126, 136, 146, 256, 356, 456\}.$$

It is easy to prove $\pi(S_1) \geq 3/4$ by noting that S_1 is not 2-colourable (i.e. no matter how we colour the vertices with two colours we cannot avoid having a monochromatic edge). Hence S_1 does not appear as a subgraph in a complete bipartite 3-graph (a 3-graph with two vertex classes where the only edges missing are those that would lie entirely in one vertex class). If we choose the vertex classes as equally as possible the complete bipartite 3-graph has an asymptotic density of 3/4, therefore $\pi(S_1) \geq 3/4$.

A natural question to ask is what other 3-graphs have a Turán density of 3/4 and a complete bipartite construction as an extremal example. With this in mind we compiled a list of all non-2-colourable 3-graphs on six vertices which are edge minimal. There are six such graphs up to isomorphism, one of which is S_1. Using Razborov's flag algebra method we will prove five of these graphs have a Turán density of 3/4 (reproving $\pi(S_1) = 3/4$). The remaining graph is $K_5^{(3)}$ with an extra isolated vertex. We were not able to calculate its Turán density exactly but we can prove $\pi(K_5^{(3)}) \leq 0.76954$ which beats the previously best known bound of $649/816 = 0.79534$ due to Markström [27].

Theorem 2.4.1. $\pi(S_1) = \pi(S_2) = \pi(S_3) = \pi(S_4) = \pi(S_5) = 3/4$, *where*

$$S_1 = \{234, 125, 135, 145, 126, 136, 146, 256, 356, 456\},$$
$$S_2 = \{123, 134, 145, 156, 162, 235, 346, 452, 563, 624\},$$
$$S_3 = \{234, 235, 145, 126, 136, 246, 346, 256, 356, 456\},$$
$$S_4 = \{234, 135, 145, 245, 126, 146, 346, 256, 356, 456\},$$
$$S_5 = \{234, 235, 145, 345, 136, 246, 346, 256, 356, 456\}.$$

Interestingly Frankl and Füredi [16] showed that a blow-up of S_2 has the maximum density of any 3-graph in which any four vertices span exactly zero or two edges.

Proof. A lower bound of 3/4 can be proved for all five graphs by considering a bipartite construction.

To prove the upper bounds of 3/4 we use Razborov's flag algebra method as described in the previous sections. One of the matrices involved has over 2000 entries and as such we exclude the details of the calculations. The files `S1.txt`, ..., `S5.txt` (see Appendix C) contain all the necessary information such as the types, flags, matrices, and the basis the matrices are written in. The calculations are too long to do by hand so we provide the program DensityChecker (see Appendix C). It reads in the text files and based on the information the files contain, the program calculates $\mathcal{H}$, checks the matrices are positive semidefinite, and calculates the upper bounds. The calculations are done entirely using integers so no rounding errors can occur. □

Theorem 2.4.2. $3/4 \leq \pi(K_5^{(3)}) \leq 0.76954$.

Proof. The lower bound can be shown by considering a bipartite construction. The upper bound was proved using Razborov's flag algebras. The specific data can be found on the CD-ROM in the file `K5.txt`. The program DensityChecker (see Appendix C) can be used to verify the bound is 0.76954. □

In the next section we describe more of Razborov's method. In particular we explain how to turn an approximate solution returned by a computer into an exact solution which precisely determines the Turán density.

2.4.2 Making approximate solutions exact

In order to bound the Turán density of a family of graphs $\mathcal{F}$, we use programs which employ floating point arithmetic to efficiently solve the underlying semidefinite programming problem. However floating point arithmetic introduces small rounding errors and so the solutions these solvers return are in fact only approximate solutions. Razborov [31] explains how to make an approximate solution precise using information contained in the (conjectured) extremal examples. We will spend the rest of this section describing this method, and extending it to cover the case when the extremal examples are unknown or do not provide sufficient information to make the solution exact.

Our aim is to perturb the entries of the matrices (given by a close approximate solution) in such a way that we do not violate the condition that the matrices are positive semidefinite and that $c_H + d(H) \leq \pi(\mathcal{F})$ holds for all $H \in \mathcal{H}$. In general there will be many exact solutions, which is precisely why such a move from an approximate to an exact solution is possible. However, all the exact solutions share certain properties. It will be important to identify these properties if we are to have any chance of modifying the approximate solution to an exact one.

The sharpness property: We require $c_H + d(H) \leq \pi(\mathcal{F})$ for all $H \in \mathcal{H}$ in an exact solution. Let us call those values of $c_H + d(H)$ which will equal $\pi(\mathcal{F})$ in the exact solution the *sharp* values, and those which will be strictly less than $\pi(\mathcal{F})$ *non-sharp*. As we will see certain $c_H + d(H)$ must be sharp regardless of the exact solution we move to. Hence it is important to identify precisely which values of $c_H + d(H)$ will be sharp in our exact solution.

The positive definite property: If an $n \times n$ matrix is positive definite with minimum eigenvalue $\lambda_{min} > 0$, then changing all entries by less than λ_{min}/n will result in a matrix which is still positive definite. Therefore if the matrices given by the approximate solution were all *positive definite* we could use this fact to change the approximate solution slightly, without affecting the positive definite property of the matrices.

We can make the matrices given by the approximate solution positive definite by modifying the bases, see Section 2.4.2.2. Once this is done our aim is to make a small change to the approximate solution which leaves the matrices

positive definite and makes the solution exact. If successful we ultimately end up with an exact solution whose matrices are all positive definite in the modified bases. Hence the key idea is to find a modification of the bases such that the matrices in the *exact* solution will be positive definite. Providing the approximate and exact solutions are close, the matrices in the approximate solution will automatically be positive definite under these new bases.

Next we will discuss how we determine these properties and utilize them.

2.4.2.1 The sharpness property

If the approximate and exact solution are sufficiently close to each other then the non-sharp values will be less than $\pi(\mathcal{F})$ in the approximate solution. Furthermore we can assume that only a small change in the matrix entries of the approximate solution is needed to make the sharp values equal $\pi(\mathcal{F})$. If this change is small enough then the non-sharp values will not have changed sufficiently to be larger than $\pi(\mathcal{F})$, (and the matrices will still remain positive definite). Hence we only need worry about identifying the sharp values and making them $\pi(\mathcal{F})$.

We can identify some sharp values by considering extremal examples. Any H which appears with positive probability in any extremal example must necessarily have $c_H + d(H) = \pi(\mathcal{F})$ in an exact solution.

Lemma 2.4.3. *Given $\{G_n\}$ a sequence of $\mathcal{F}$-free graphs of increasing order, such that $d(G_n) \to \pi(\mathcal{F})$, and $p(H; G_n)$ converges for all $H \in \mathcal{H}$. Let us define $p(H; G_\infty)$ to be the value $p(H; G_n)$ converges to. Then for any exact solution, $p(H; G_\infty) > 0$ implies $c_H + d(H) = \pi(\mathcal{F})$.*

Proof. We know

$$d(G_n) \leq \sum_{H \in \mathcal{H}} (c_H + d(H)) p(H; G_n) + o(1).$$

Taking the limit as n tends to infinity, we get

$$\pi(\mathcal{F}) \leq \sum_{H \in \mathcal{H}} (c_H + d(H)) p(H; G_\infty).$$

In an exact solution $c_H + d(H) \leq \pi(\mathcal{F})$ for all $H \in \mathcal{H}$. It is easy to check

$p(H; G_\infty) \geq 0$ for all $H \in \mathcal{H}$ and that $\sum_{H \in \mathcal{H}} p(H; G_\infty) = 1$. Consequently

$$\sum_{H \in \mathcal{H}} (c_H + d(H)) p(H; G_\infty) \leq \pi(\mathcal{F})$$

with equality only when $c_H + d(H) = \pi(\mathcal{F})$ for all $H \in \mathcal{H}$ with $p(H; G_\infty) \neq 0$. □

In fact, assuming that the sharp values were given precisely by considering the subgraphs which appear in the bipartite construction was enough for the proofs of $\pi(S_i) = 3/4$ for $i = 1, \ldots, 5$. In general there may be sharp values which cannot be determined in this way, for example there may be extremal constructions of which we are unaware. In such cases we can often determine which values are likely to be sharp by considering which values are very close to $\pi(\mathcal{F})$ in the approximate solution.

Once the sharp values have been determined, or guessed, we need to modify the matrix entries to make them equal to $\pi(\mathcal{F})$. The outline of the procedure to move to an exact solution is given below.

- First we change the bases of the matrices as described in Section 2.4.2.2. This ensures after we modify the matrix entries the matrices remain positive definite.
- Next we take the entries of the matrices (ignoring the entries below the main diagonals) and form a vector which represents the approximate solution.
- We then solve a linear system of equations to find all such vectors that represent solutions in which all the sharp values equal $\pi(\mathcal{F})$ (irrespective of whether the non-sharp values are less than $\pi(\mathcal{F})$).
- We now have a linear space of solutions and we wish to find the point in this space which is closest to the point given by the approximate solution. We choose to define closeness via the Euclidean metric as this allows us to find the closest point by solving another linear system of equations.
- Once we have the closest point we can convert it back into a series of matrices, which will have the property that the sharp values are $\pi(\mathcal{F})$ and it is a small perturbation from the approximate solution. (Rather

than take the closest point often we take a point close to the closest point in order to produce matrices with simpler entries.)

- After we make this modification to the approximate solution we need to check the change wasn't so large that the non-sharp values are now larger than $\pi(\mathcal{F})$ or that the matrices are not positive semidefinite. If the change is too large, then it indicates either the approximate solution is too inaccurate, or the choice of sharp values was incorrect.

2.4.2.2 The positive definite property

In order for us to make small changes to the approximate solution to get to an exact solution it is necessary that we modify the bases the matrices are written in, so that the matrices appear positive definite in the exact solution.

Suppose we are given an exact solution. We can rewrite the bases of the matrices so that the matrices in the exact solution become diagonal, with the eigenvalues down the diagonal. In order to make the exact solution's matrices positive definite we simply remove the bases elements corresponding to the zeros in the diagonals, and remove the respective rows and columns from the matrices. (It is easy to check that doing this will not affect the bound the exact solution gives.) If we rewrite the matrices in the approximate solution in the new bases, and remove the same basis elements, we may affect the bound the approximate solution gives. However, providing the approximate solution was close to the exact solution the change will be small. Another consequence of the closeness of the exact and approximate solutions is that the new matrices in the approximate solution will now automatically be positive definite.

In general we do not need to find bases that diagonalize the matrices, doing so would require us to find all the eigenvectors of the matrices. It is sufficient to just determine the 0-eigenvectors (the eigenvectors whose corresponding eigenvalue is zero). We can modify the bases so that the 0-eigenvectors do not lie in the space the bases span, and this is enough to ensure that the matrices will be positive definite. We can determine some of these 0-eigenvectors by considering extremal examples.

Given $\{G_n\}$ a sequence of $\mathcal{F}$-free graphs of increasing order, such that $d(G_n) \to \pi(\mathcal{F})$, and $p(H; G_n) \to p(H; G_\infty)$ for all $H \in \mathcal{H}$. Then for any

exact solution

$$\sum_{H \in \mathcal{H}} (c_H + d(H)) p(H; G_\infty) = \pi(\mathcal{F}).$$

and since $\sum d(H)p(H;G_\infty) = \pi(\mathcal{F})$, we have $\sum c_H p(H;G_\infty) = 0$. This implies terms of the form $\mathbf{E}_{\theta \in \Theta}[\mathbf{p}_\theta^T Q \mathbf{p}_\theta]$ tend to zero. Generally rather than $\mathbf{p}_\theta = (p(F, \theta; G) : F \in \mathcal{F}_m^\sigma)$ we often have a more complicated basis as outlined in Section 2.2.3. Let us call this basis $\mathbf{b}_{\theta,n}$ and its elements consist of linear combinations of $p(F, \theta; G_n)$ terms, where $F \in \mathcal{F}_m^\sigma$, $\theta \in \Theta_n$, and Θ_n is the set of all injective functions from $[|\sigma|]$ to $V(G_n)$. Informally, if with positive probability a random $\mathbf{b}_{\theta,\infty}$ looks like $\mathbf{b}$ in an extremal example then $\mathbf{b}$ is a 0-eigenvector of Q.

Lemma 2.4.4. *Given a vector $\mathbf{b} \neq \mathbf{0}$ and a function $\epsilon(n) > 0$ that tends to zero. If the proportion of $\theta \in \Theta_n$ which satisfies $\|\mathbf{b} - \mathbf{b}_{\theta,n}\|_\infty \leq \epsilon(n)$ converges to a strictly positive quantity, then $\mathbf{b}$ is an eigenvector with eigenvalue zero for Q in an exact solution.*

Proof. In an exact solution $\mathbf{E}_{\theta \in \Theta_n}[\mathbf{b}_{\theta,n}^T Q \mathbf{b}_{\theta,n}]$ tends to zero. If $\mathbf{b}^T Q \mathbf{b} > 0$, then $\mathbf{E}_{\theta \in \Theta_n}[\mathbf{b}_{\theta,n}^T Q \mathbf{b}_{\theta,n}]$ must necessarily tend to a positive quantity. Hence we must have $\mathbf{b}^T Q \mathbf{b} = 0$. Since Q is a real symmetric matrix, it can be written in the form $R^T D R$ where R is an orthogonal matrix and D is a diagonal matrix whose diagonal elements are the eigenvalues of Q. Since Q is positive semidefinite, all its eigenvalues are non-negative and so Q can be written as $R^T M^T M R$, where M is a diagonal matrix whose diagonal elements are the square root of those in D. Hence $\mathbf{b}^T Q \mathbf{b} = (MR\mathbf{b})^T (MR\mathbf{b}) = 0$ which implies $MR\mathbf{b} = \mathbf{0}$ and thus $Q\mathbf{b} = \mathbf{0}$. □

Considering such $\mathbf{b}$ in the bipartite extremal example was enough to find the 0-eigenvectors for the matrices when proving $\pi(S_i) = 3/4$ for $i = 1, \ldots, 5$. However, some 0-eigenvectors may not be deducible this way, for example we may not know all the extremal examples. In such cases we can again look at the approximate solution to help us guess what the 0-eigenvectors should be. We outline the procedure below:

1. First we take the matrix Q (from our approximate solution) and calculate its eigenvalues and corresponding eigenvectors.

2. We assume the eigenvalues close to zero would be zero in an exact solution. Let us call the corresponding eigenvectors the *approximate vectors*. The

space spanned by the approximate vectors we assume is an approximation of the space spanned by the 0-eigenvectors in the exact solution.

3. We take one of our approximate vectors and find its component which has the largest absolute value, say it is in position i.

4. We then divide all its components by the ith component.

5. Next we subtract multiples of this vector from all the other approximate vectors, so that the ith component is zero in all other approximate vectors.

6. We repeat steps 3-5 for the other approximate vectors. This gives us a new basis for the approximated space.

7. Making the assumption that the 0-eigenvectors have simple rational components allows us to guess them from the modified approximate vectors.

We give the following example to illustrate this algorithm, and the process of making a matrix positive definite given the 0-eigenvectors. Let us take the matrix

$$Q = \begin{pmatrix} 0.5228 & -0.1845 & 0.0923 & 0.4612 \\ -0.1845 & 1.1070 & -0.5535 & 0.1846 \\ 0.0923 & -0.5535 & 0.2767 & -0.0920 \\ 0.4612 & 0.1846 & -0.0920 & 0.5227 \end{pmatrix}.$$

Its eigenvalues are $1.4452, 0.9840, 0.0002$, and -0.0002. We will assume that the eigenvalues 0.0002 and -0.0002 would correspond to the zero eigenvalues in an exact solution. Their corresponding eigenvectors (the approximate vectors) are

$$\mathbf{v}_1 = \begin{pmatrix} -0.5534 \\ 0.1210 \\ 0.6108 \\ 0.5532 \end{pmatrix}, \mathbf{v}_2 = \begin{pmatrix} 0.4153 \\ 0.4684 \\ 0.6599 \\ -0.4156 \end{pmatrix}.$$

Applying steps 3-5 to $\mathbf{v}_1$ leaves the approximate vectors as

$$\mathbf{v}_1 = \begin{pmatrix} -0.9060 \\ 0.1981 \\ 1 \\ 0.9057 \end{pmatrix}, \mathbf{v}_2 = \begin{pmatrix} 1.0132 \\ 0.3377 \\ 0 \\ -1.0133 \end{pmatrix}.$$

Repeating steps 3-5 for $\mathbf{v}_2$, gives

$$\mathbf{v}_1 = \begin{pmatrix} -0.0004 \\ 0.4999 \\ 1 \\ 0 \end{pmatrix}, \mathbf{v}_2 = \begin{pmatrix} -0.9999 \\ -0.3333 \\ 0 \\ 1 \end{pmatrix}.$$

Assuming the components are simple rational values we would guess the 0-eigenvectors to be

$$\mathbf{v}_1 = \begin{pmatrix} 0 \\ 1/2 \\ 1 \\ 0 \end{pmatrix}, \mathbf{v}_2 = \begin{pmatrix} -1 \\ -1/3 \\ 0 \\ 1 \end{pmatrix}.$$

Now we will make the matrix positive definite by modifying the basis. By combining $\mathbf{v}_1, \mathbf{v}_2$ with $\mathbf{v}_3 = (1, 0, 0, 0)^T$ and $\mathbf{v}_4 = (0, 1, 0, 0)^T$ (two independent vectors) we can form a new basis. Let R be a matrix representing the new basis i.e.

$$R = \begin{pmatrix} 0 & -1 & 1 & 0 \\ 1/2 & -1/3 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad \text{and hence} \quad R^{-1} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & -1/2 & 1/3 \end{pmatrix}.$$

We can rewrite $Q = (R^{-1})^T Q' R^{-1}$, where

$$Q' = R^T Q R = \begin{pmatrix} -0.0001 & 0.0003 & 0.0001 & 0 \\ 0.0003 & 0.0001 & -0.0001 & 0.0001 \\ 0.0001 & -0.0001 & 0.5228 & -0.1845 \\ 0 & 0.0001 & -0.1845 & 1.1070 \end{pmatrix}.$$

In order to make the matrix in the exact solution positive definite we would remove columns $1, 2$ and rows $1, 2$ from Q' as these are associated with the 0-eigenvectors $\mathbf{v}_1, \mathbf{v}_2$. Hence

$$Q \approx \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & -1/2 \\ 1 & 1/3 \end{pmatrix} \begin{pmatrix} 0.5228 & -0.1845 \\ -0.1845 & 1.1070 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & -1/2 & 1/3 \end{pmatrix}.$$

The 2 by 2 matrix in the above expression is positive definite (as intended), and modifying its entries would hopefully lead to an exact solution. (The 4 by 2 and 2 by 4 matrices represent the change in basis.)

2.4.3 Open problems

We have shown $\pi(S_1) = \pi(S_2) = \pi(S_3) = \pi(S_4) = \pi(S_5) = 3/4$, however we were not able to calculate $\pi(K_5^{(3)})$ precisely. An obvious open problem is to show $\pi(K_5^{(3)}) = 3/4$, which may be possible to prove using the method we have described in this section and the use of larger flags.

The central problem in this area is Turán's conjecture that $\pi(K_4^{(3)}) = 5/9$. Razborov [31] managed to prove $\pi(K_4^{(3)}) \leq 0.561666$, which is very close to the conjectured value. This bound may also be improved through the use of larger flags.

2.5 Hypercube results

2.5.1 Introduction

In the following sections we will extend Razborov's method from hypergraphs to hypercubes. An n-dimensional hypercube $\mathcal{Q}_n$ is a 2-graph with 2^n vertices. Setting $V(\mathcal{Q}_n) = \{0, 1, \ldots, 2^n - 1\}$ we can define $E(\mathcal{Q}_n)$ as follows: $v_1v_2 \in E(\mathcal{Q}_n)$ if and only if v_1 differs from v_2 by precisely one digit in their binary representations. For example $E(\mathcal{Q}_2) = \{01, 02, 13, 23\}$. It is easy to see that the binary representations of the vertices indicate the coordinates of the vertices of a unit hypercube in $\mathbb{R}^n$. Let us also define the *layers* of a hypercube which will be useful later. *Layer* m of $\mathcal{Q}_n$ consists of all vertices in $V(\mathcal{Q}_n)$ which have

m digits that are one in their binary representations. For example in $\mathcal{Q}_3$, layer $0 = \{0\}$, layer $1 = \{1, 2, 4\}$, layer $2 = \{3, 5, 6\}$, and layer $3 = \{7\}$.

We will consider two different types of Turán problems involving hypercubes. In the first type we will be interested in the following question: given a forbidden family of graphs $\mathcal{F}$, what is the maximum number of edges an $\mathcal{F}$-free subgraph of $\mathcal{Q}_n$ can have? We are particularly interested in the limit of the maximum *hypercube edge density* as n tends to infinity, where we define the *hypercube edge density* of a subgraph G of $\mathcal{Q}_n$ to be $|E(G)|/|E(\mathcal{Q}_n)|$. We will refer to the limit as the *edge Turán density* $\pi_e(\mathcal{F})$, a simple averaging argument shows it always exists.

Motivation to study the edge Turán density comes from Erdős [11] who conjectured that $\pi_e(\mathcal{Q}_2) = 1/2$. It is easily seen that $\pi_e(\mathcal{Q}_2) \geq 1/2$ by taking $\mathcal{Q}_n$ and removing those edges that have one vertex in layer $2r-1$ and the other in layer $2r$ for each r. Such subgraphs of $\mathcal{Q}_n$ are $\mathcal{Q}_2$-free and contain exactly half the edges. The densest known constructions which are $\mathcal{Q}_2$-free are given by Brass, Harborth, and Nienborg [6], and they have a hypercube edge density of approximately $(1 + 1/\sqrt{n})/2$. Chung [8] showed that $\pi_e(\mathcal{Q}_2) \leq (2 + \sqrt{13})/9 = 0.62284$, her argument was extended by Thomason and Wagner [32] using a computer, to get the currently best known bound of 0.62256. By extending Razborov's flag algebra technique to hypercubes we will prove a significantly smaller upper bound of 0.60680. Chung [8] also considered the edge Turán density of 6-cycles, and proved $1/4 \leq \pi_e(C_6) \leq \sqrt{2} - 1 = 0.41421$. We will improve the upper bound to 0.37550.

The second type of hypercube Turán problem we will look at is very similar to the first but focuses on the density of vertices rather than edges. In the second type we are interested in the following question: given a forbidden family of graphs $\mathcal{F}$, what is the maximum number of vertices an $\mathcal{F}$-free *induced* subgraph of $\mathcal{Q}_n$ can have? We are particularly interested in the limit of the maximum *hypercube vertex density* as n tends to infinity, where we define the *hypercube vertex density* of an induced subgraph G of $\mathcal{Q}_n$ to be $|V(G)|/|V(\mathcal{Q}_n)|$. We will refer to the limit as the *vertex Turán density* $\pi_v(\mathcal{F})$, again a simple averaging argument shows it always exists.

The analogous problem to Erdős' conjecture is calculating $\pi_v(\mathcal{Q}_2)$. E.A. Kostochka [24] and independently Johnson and Entringer [23] showed that $\pi_v(\mathcal{Q}_2) = 2/3$. Johnson and Talbot [22] proved that $\pi_v(R_1) = 2/3$, where

R_1 is the graph formed by removing vertices 0 and 1 from $\mathcal{Q}_3$. By extending Razborov's flag algebra method we will prove $\pi_v(R_2) = 2/3$, where R_2 is the graph formed by removing a single vertex from $\mathcal{Q}_3$. The value of $\pi_v(\mathcal{Q}_3)$, however, still remains undetermined. A lower bound of $3/4$ is easily achieved by considering the induced subgraphs of $\mathcal{Q}_n$ formed by removing all vertices in layers that are a multiple of four (i.e. layers $0, 4, 8, \ldots$). Although we could not show $\pi_v(\mathcal{Q}_3) \leq 3/4$ we will prove $\pi_v(\mathcal{Q}_3) \leq 0.76900$. We will also show that $1/2 \leq \pi_v(C_6) \leq 0.53111$.

2.5.2 Edge Turán density

Calculating the edge Turán density involves looking at subgraphs of hypercubes. However, the structure of the hypercubes may not be retained by the subgraphs. This structure will prove to be useful and will simplify definitions later. Hence rather than work directly with subgraphs we will instead use edge-coloured hypercubes that represent subgraphs of $\mathcal{Q}_n$. In particular we will colour the edges red and blue. The subgraph a red-blue edge-coloured hypercube represents can be constructed by removing those edges that are coloured red and keeping those edges that are blue. Erdős' conjecture [11] that $\pi_e(\mathcal{Q}_2) = 1/2$ comes from asking what is the maximum number of edges a $\mathcal{Q}_2$-free subgraph of $\mathcal{Q}_n$ can have. It should be clear that this is equivalent to asking what is the maximum number of blue edges an edge-coloured $\mathcal{Q}_n$ can have such that it does not contain a blue $\mathcal{Q}_2$. Therefore the problem of calculating $\pi_e(\mathcal{Q}_2)$, and $\pi_e(\mathcal{F})$ in general, can be translated into a problem involving forbidding edge-coloured hypercubes in an edge-coloured $\mathcal{Q}_n$. We will define the equivalent notion of edge Turán density for edge-coloured hypercubes shortly, but first we need some definitions.

We will use the notation $(n, \kappa)_e$ to represent an edge-coloured $\mathcal{Q}_n$, where $\kappa : E(\mathcal{Q}_n) \to \{\text{red, blue}\}$. We define $V(F)$ and $E(F)$ for an edge-coloured hypercube $F = (n, \kappa)_e$ to be $V(\mathcal{Q}_n)$ and $E(\mathcal{Q}_n)$ respectively. Consider two edge-coloured hypercubes $F_1 = (n_1, \kappa_1)_e$, and $F_2 = (n_2, \kappa_2)_e$. We say F_1 is *isomorphic* to F_2 if there exists a bijection $f : V(F_1) \to V(F_2)$ such that for all $v_1v_2 \in E(F_1)$, $f(v_1)f(v_2) \in E(F_2)$ and $\kappa_1(v_1v_2) = \kappa_2(f(v_1)f(v_2))$. We say F_1 is a *subcube* of F_2 if there exists an injection $g : V(F_1) \to V(F_2)$ such that for all $v_1v_2 \in E(F_1)$, $g(v_1)g(v_2) \in E(F_2)$ and if $\kappa_1(v_1v_2) = \text{blue}$ then $\kappa_2(g(v_1)g(v_2)) = \text{blue}$.

The *edge density* of $F = (n, \kappa)_e$ is

$$d_e(F) = \frac{|\{v_1v_2 \in E(F) : \kappa(v_1v_2) = \text{blue}\}|}{|E(F)|}.$$

Note that this is analogous to the hypercube edge density defined in Section 2.5.1. Given a family of coloured hypercubes $\mathcal{F}$, we say H, a coloured hypercube, is $\mathcal{F}$*-free* if H does not contain a subcube isomorphic to any member of $\mathcal{F}$. The *coloured edge Turán density* of $\mathcal{F}$ is defined to be the following limit (a simple averaging argument shows that it always exists)

$$\pi_{ce}(\mathcal{F}) = \lim_{n\to\infty} \max_{\kappa}\{d_e(H) : H = (n, \kappa)_e \text{ and is } \mathcal{F}\text{-free}\}.$$

Given these definitions it should be easy to see that $\pi_e(\mathcal{Q}_2) = \pi_{ce}(B)$ where B is a $\mathcal{Q}_2$ with all four of its edges coloured blue. We are also interested in $\pi_e(C_6)$. It is not to difficult to show that all 6-cycles in $\mathcal{Q}_n$ lie within a $\mathcal{Q}_3$ subgraph. There are two distinct 6-cycles in a $\mathcal{Q}_3$ up to isomorphism, their edge sets are $E_1 = \{01, 13, 32, 26, 64, 40\}$, and $E_2 = \{51, 13, 32, 26, 64, 45\}$. Let B_1 be a $\mathcal{Q}_3$ with those edges in E_1 coloured blue and the remaining edges coloured red. Similarly let B_2 be a $\mathcal{Q}_3$ with those edges in E_2 coloured blue and the remaining edges coloured red. Hence forbidding a blue edged 6-cycle in an edge-coloured hypercube is equivalent to requiring that it is B_1 and B_2-free. Therefore $\pi_e(C_6) = \pi_{ce}(B_1, B_2)$. By extending Razborov's flag algebra method to edge-coloured hypercubes, we will be able to prove the following bounds on $\pi_{ce}(B)$ and $\pi_{ce}(B_1, B_2)$.

Theorem 2.5.1. $\pi_e(\mathcal{Q}_2) = \pi_{ce}(B) \leq 0.60680$ *and* $\pi_e(C_6) = \pi_{ce}(B_1, B_2) \leq 0.37550$.

In the remainder of this section we will describe how to apply Razborov's flag algebra method to hypercubes.

Let $\mathcal{F}$ be a family of coloured hypercubes whose coloured edge Turán density we wish to compute (or at least approximate). Let $\mathcal{H}$ be the family of all $\mathcal{F}$-free edge-coloured hypercubes of dimension l, up to isomorphism. If l is sufficiently small we can explicitly determine $\mathcal{H}$ (by computer search if necessary). For $H \in \mathcal{H}$ and a large $\mathcal{F}$-free coloured hypercube G, we define $p(H; G)$ to be the probability that a random hypercube of dimension l from G induces a coloured subcube isomorphic to H.

Trivially, the edge density of G is equal to the probability that a random $\mathcal{Q}_1$ (a single edge) from G is coloured blue. Thus, averaging over hypercubes of dimension l in G, we can express the edge density of G as

$$d_e(G) = \sum_{H \in \mathcal{H}} d_e(H) p(H; G),$$

and hence $\pi_{ce}(\mathcal{F}) \leq \max_{H \in \mathcal{H}} d_e(H)$. This "averaging" bound can be improved upon by using flags and types just as we did in the hypergraph case.

For edge-coloured hypercubes we define flags and types as follows. A *flag*, $F = (G_F, \theta)$, is an edge-coloured hypercube G_F together with an injective map $\theta : \{0, 1, \ldots, 2^s - 1\} \to V(G_F)$ such that $\theta(i)\theta(j) \in E(G_F)$ if and only if i and j differ by precisely one digit in their binary representations (i.e. θ induces a canonically labelled hypercube). If θ is bijective (and so $|V(G_F)| = 2^s$) we call the flag a *type*. For ease of notation given a flag $F = (G_F, \theta)$ we define its dimension $dim(F)$ to be the dimension of the hypercube underlying G_F. Given a type σ we call a flag $F = (G_F, \theta)$ a *σ-flag* if the induced labelled and coloured subcube of G_F given by θ is σ.

Fix a type σ and an integer $m \leq (l + dim(\sigma))/2$. (The bound on m ensures that an l-dimensional hypercube can contain two m-dimensional subcubes overlapping in a dimension $dim(\sigma)$ hypercube.) Let $\mathcal{F}^\sigma_m$ be the set of all $\mathcal{F}$-free σ-flags of dimension m, up to isomorphism. Let Θ be the set of all injective functions from $\{0, 1, \ldots, 2^{dim(\sigma)} - 1\}$ to $V(G)$, that result in a canonically labelled hypercube. Given $F \in \mathcal{F}^\sigma_m$ and $\theta \in \Theta$ we define $p(F, \theta; G)$ to be the probability that an m-dimensional coloured hypercube R chosen uniformly at random from G subject to $\mathrm{im}(\theta) \subseteq V(R)$, induces a σ-flag (R, θ) that is isomorphic to F.

If $F_a, F_b \in \mathcal{F}^\sigma_m$ and $\theta \in \Theta$ then $p(F_a, \theta; G)p(F_b, \theta; G)$ is the probability that two m-dimensional coloured hypercubes R_a, R_b chosen independently at random from G subject to $\mathrm{im}(\theta) \subseteq V(R_a) \cap V(R_b)$, induce σ-flags (R_a, θ), (R_b, θ) that are isomorphic to F_a, F_b respectively. We define the related probability, $p(F_a, F_b, \theta; G)$, to be the probability that two m-dimensional coloured hypercubes R_a, R_b chosen independently at random from G subject to $\mathrm{im}(\theta) = V(R_a) \cap V(R_b)$, induce σ-flags (R_a, θ), (R_b, θ) that are isomorphic to F_a, F_b respectively. It is easy to show that $p(F_a, \theta; G)p(F_b, \theta; G) = p(F_a, F_b, \theta; G) + o(1)$ where the $o(1)$ term vanishes as $|V(G)|$ tends to infinity.

For $H \in \mathcal{H}$ let Θ_H be the set of all injective maps $\theta : \{0, 1, \ldots, 2^{dim(\sigma)} - 1\} \to$

$V(H)$ which induces a canonically labelled hypercube. Just as in Section 2.2.1 we have

$$\mathbf{E}_{\theta\in\Theta}\left[p(F_a,\theta;G)p(F_b,\theta;G)\right] = \sum_{H\in\mathcal{H}} \mathbf{E}_{\theta\in\Theta_H}\left[p(F_a,F_b,\theta;H)\right] p(H;G) + o(1),$$

by averaging over l-dimensional subcubes of G. The remainder of the argument is identical to that in Section 2.2.1. By considering positive semidefinite matrices Q we can use terms of the form $\mathbf{E}_{\theta\in\Theta}[\mathbf{p}_\theta^T Q \mathbf{p}_\theta]$ to improve our bound on $\pi_{ce}(\mathcal{F})$. Choosing the optimal matrices Q can again be posed as a semidefinite programming problem. The ideas presented in Sections 2.2.3 and 2.4 similarly extend from hypergraphs to hypercubes.

Proof of Theorem 2.5.1. All the necessary data (types, flags, matrices, etc.) needed to prove $\pi_{ce}(B) \leq 0.60680$ can be found on the CD-ROM in the file `B.txt`. The calculation that converts this data into an upper bound is too long to do by hand and so we provide the program HypercubeEdgeDensityChecker to verify our claim (see Appendix D). Similarly the data needed to prove $\pi_{ce}(B_1, B_2) \leq 0.37550$ can be found in the file `B1B2.txt` on the CD-ROM. □

2.5.3 Vertex Turán density

When we looked at the edge Turán density problem we found that rather than working directly with subgraphs of hypercubes it was simpler to use edge-coloured hypercubes instead. Similarly when calculating the vertex Turán density we will use red-blue vertex-coloured hypercubes to represent induced subgraphs of hypercubes. The induced subgraph a vertex-coloured hypercube represents can be constructed by removing precisely those vertices which are red.

We will use the notation $(n,\kappa)_v$ to represent a vertex-coloured $\mathcal{Q}_n$, where $\kappa : V(\mathcal{Q}_n) \to \{\text{red, blue}\}$. We define $V(F)$ and $E(F)$ for a vertex-coloured hypercube $F = (n,\kappa)_v$ to be $V(\mathcal{Q}_n)$ and $E(\mathcal{Q}_n)$ respectively. Consider two vertex-coloured hypercubes $F_1 = (n_1,\kappa_1)_v$, and $F_2 = (n_2,\kappa_2)_v$. We say F_1 is *isomorphic* to F_2 if there exists a bijection $f : V(F_1) \to V(F_2)$ such that for all $v_1v_2 \in E(F_1)$, $f(v_1)f(v_2) \in E(F_2)$ and for all $v \in V(F_1)$, $\kappa_1(v) = \kappa_2(f(v))$. We say F_1 is a *subcube* of F_2 if there exists an injection $g : V(F_1) \to V(F_2)$

such that for all $v_1v_2 \in E(F_1)$, $g(v_1)g(v_2) \in E(F_2)$ and for all $v \in V(F_1)$ if $\kappa_1(v) = \text{blue}$ then $\kappa_2(g(v)) = \text{blue}$.

The *vertex density* of $F = (n, \kappa)_v$ is

$$d_v(F) = \frac{|\{v \in V(F) : \kappa(v) = \text{blue}\}|}{|V(F)|}.$$

Note that this is analogous to the hypercube vertex density defined in Section 2.5.1. Given a family of vertex-coloured hypercubes $\mathcal{F}$, we say H, a vertex-coloured hypercube, is *$\mathcal{F}$-free* if H does not contain a subcube isomorphic to any member of $\mathcal{F}$. The *coloured vertex Turán density* of $\mathcal{F}$ is defined to be the following limit (a simple averaging argument shows that it always exists)

$$\pi_{cv}(\mathcal{F}) = \lim_{n\to\infty} \max_{\kappa}\{d_v(H) : H = (n, \kappa)_v \text{ and is } \mathcal{F}\text{-free}\}.$$

Given these definitions it should be easy to see that $\pi_v(\mathcal{Q}_3) = \pi_{cv}(B_3)$, where B_3 is a $\mathcal{Q}_3$ with all its vertices coloured blue. It is also not hard to show that forbidding R_2 in $\mathcal{Q}_n$ is equivalent to asking that a vertex-coloured hypercube is B_3^--free, where B_3^- is a $\mathcal{Q}_3$ with vertex 7 coloured red and the remaining vertices coloured blue. Hence $\pi_v(R_2) = \pi_{cv}(B_3^-)$. As discussed in Section 2.5.2 all 6-cycles in $\mathcal{Q}_n$ lie within a $\mathcal{Q}_3$ subgraph, and there are only two distinct 6-cycles in a $\mathcal{Q}_3$ up to isomorphism. These two 6-cycles can be represented by B_4 and B_5, two vertex-coloured hypercubes of dimension three. Specifically vertices 5 and 7 are coloured red in B_4 and vertices 0 and 7 are red in B_5 (the remaining vertices are blue). Therefore $\pi_v(C_6) = \pi_{cv}(B_4, B_5)$. By extending Razborov's method to vertex-coloured hypercubes we will be able to prove the following result.

Theorem 2.5.2. *The following all hold:*

(i) $\pi_v(R_2) = \pi_{cv}(B_3^-) = 2/3$,

(ii) $3/4 \leq \pi_v(\mathcal{Q}_3) = \pi_{cv}(B_3) \leq 0.76900$,

(iii) $1/2 \leq \pi_v(C_6) = \pi_{cv}(B_4, B_5) \leq 0.53111$.

We omit the details of extending Razborov's technique to vertex-coloured hypercubes; it is virtually identical to the extension described in Section 2.5.2, with the term "vertex-colouring" replacing the term "edge-colouring".

Proof of Theorem 2.5.2. The lower bound of $2/3$ for $\pi_v(R_2)$ is easily proved by considering the induced subgraph of $\mathcal{Q}_n$ formed by removing every third layer of vertices. Similarly $\pi_v(\mathcal{Q}_3) \geq 3/4$ and $\pi_v(C_6) \geq 1/2$ can be proved by looking at $\mathcal{Q}_n$ with every fourth layer removed and every second layer removed respectively.

The upper bounds were calculated using Razborov's flag algebra method extended to vertex-coloured hypercubes. Specific data can be found in the `HypercubeVertexDensityChecker` folder on the CD-ROM. The calculations required to turn the data into upper bounds are too long to do by hand and so we provide the program HypercubeVertexDensityChecker to verify our claims (see Appendix D).

It is worth noting that the layered extremal example for $\pi_v(R_2) = 2/3$ did not provide enough information to remove the floating point errors. Instead we used the methods described in Section 2.4 to get the information directly from an approximate solution. □

2.5.4 Open problems

The study of Turán problems in hypercubes is largely motivated by Erdős' conjecture that $\pi_e(\mathcal{Q}_2) = 1/2$. This is perhaps the most interesting question in the area, and still remains open. We have provided improvements on the bounds of various edge and vertex Turán densities but were only able to calculate $\pi_v(R_2)$ exactly. Improving the bounds further to get exact results in any of the problems discussed would be of interest. However, due to the exponential nature of hypercubes this may prove to be computationally intractable to do via our modified applications of Razborov's method.

Chapter 3

Almost Intersecting Families that are Almost Everything

3.1 Introduction

An *intersecting family* is a family of sets $\mathcal{F}$, such that for all $A, B \in \mathcal{F}$ we have $A \cap B \neq \emptyset$. For $a, b \in [n] = \{1, 2, 3, \ldots, n\}$ we define $d(a, b)$ to be the shortest distance from a to b in the usual cyclic ordering of $[n]$. A family of sets $\mathcal{F} \subset \mathcal{P}([n])$ is *almost intersecting* if $A, B \in \mathcal{F}$ implies there exists $a \in A$, $b \in B$ such that $d(a, b) \leq 1$. This is equivalent to the definition of *G-intersecting* given by Bohman, Frieze, Ruszinkó, and Thoma in [1], when G is an n-cycle labelled $1, \ldots, n$ in the usual way. In general given a graph G with vertex set V, we say a family of sets $\mathcal{F} \subset \mathcal{P}(V)$ is G-intersecting, if $A, B \in \mathcal{F}$ implies there exists $a \in A$ and $b \in B$ such that $a = b$ or a is adjacent to b in G.

We are particularly interested in answering the following question: how large can an almost intersecting family be if it is composed entirely of k-sets (sets of size k), for some fixed k? This is motivated by the Erdős–Ko–Rado theorem [12] which tells us how large an *intersecting* family of k-sets from $[n]$ can be.

Theorem 3.1.1 (Erdős–Ko–Rado [12])**.** *If $\mathcal{F}$ is an intersecting family of k-sets from $[n]$ and $k \leq n/2$ then $|\mathcal{F}| \leq \binom{n-1}{k-1}$.*

Note that if $k > n/2$ then every pair of k-sets intersect, so the family of all k-sets is an intersecting family. When $k \leq n/2$ we can achieve an intersecting family of size $\binom{n-1}{k-1}$ by choosing all k-sets that contain a fixed element. We can do something similar when looking at almost intersecting families. Instead of

fixing one element let us fix two adjacent elements in the n-cycle, say 2 and 3. We can construct an almost intersecting family by choosing all k-sets that contain either 2 or 3. The size of the family can be improved upon by also including all sets that contain both 1 and 4 to create the almost intersecting family

$$\mathcal{A}^* = \{A \in [n] : 2 \in A,\ 3 \in A,\ \text{or } \{1,4\} \subseteq A\}.$$

Bohman, Frieze, Ruszinkó, and Thoma [1] conjecture that for large values of n, the size of the largest almost intersecting family is $|\mathcal{A}^*|$ when k/n is small, and almost $\binom{n}{k}$ when k/n is large.

Conjecture 3.1.2 (Bohman, Frieze, Ruszinkó, and Thoma [1])**.** *Let $\alpha(n,k)$ be the size of the largest almost intersecting family in $[n]^{(k)}$. There exists $\gamma > 0$ such that, for any fixed $\epsilon > 0$,*

$$\alpha(n,k) = \begin{cases} |\mathcal{A}^*|, & \text{if } k \leq (\gamma - \epsilon)n, \\ (1 - o(1))\binom{n}{k}, & \text{if } k \geq (\gamma + \epsilon)n, \end{cases}$$

where the $o(1)$ term tends to zero as n tends to infinity.

Note the similarity of this conjecture with the behaviour of intersecting families. Bohman, Frieze, Ruszinkó, and Thoma [1] proved $\alpha(n,k) = |\mathcal{A}^*|$ when $k = O(n^{1/4})$, and $\alpha(n,k) = (1 - o(1))\binom{n}{k}$ when $k > 0.318n$. Bohman and Martin [2] improved the bound for $\alpha(n,k) = |\mathcal{A}^*|$ to $k = O(\sqrt{n})$. Johnson and Talbot [21], showed that $\alpha(n,k) = (1-o(1))\binom{n}{k}$ when $k > 0.266n$. Furthermore they conjectured that $0.266\ldots$ (the root of a quartic equation) was the threshold value γ.

We show in the next section that the bound for attaining almost all k-sets can be lowered from 0.266 to 1/4, disproving the conjecture of Johnson and Talbot. We then generalize our argument to G-intersecting families, where G is the p-th power of a cycle. Finally we extend our result to cross-intersecting families.

3.2 A bound on large almost intersecting families

In this section we will prove our main result: that we can construct an almost intersecting family of k-sets that is almost everything when $k/n > 1/4$. Throughout this section we will assume that k varies with n.

Theorem 3.2.1. *Let $\alpha(n,k)$ be the size of the largest almost intersecting family, of k-sets, on the n-cycle. For any fixed $\epsilon > 0$, if $k/n \geq 1/4 + \epsilon$ then*

$$\alpha(n,k) = (1 - o(1))\binom{n}{k}.$$

Before we start the proof we need to make a few definitions and prove a few lemmas. We begin by defining *runs* and *gaps* and then we will write the property of almost intersection in a way which makes use of these concepts.

Definition 3.2.2 (Extended Neighbourhood)**.** *Given a set A we define its* extended neighbourhood $N(A)$*, to be the union of A and its neighbours. Therefore two sets A and B almost intersect if and only if $A \cap N(B) \neq \emptyset$.*

Definition 3.2.3 (Run)**.** *We define a* run *to be a set of consecutive points that are in the extended neighbourhood of a set. Thus the extended neighbourhood of a set is the union of its runs. Furthermore let us define a t-*run *to be a run of size t which is not a proper subset of a larger run. See Figure 3.1.*

Definition 3.2.4 (Gap)**.** *We define a* gap *to be a set of consecutive points that do not lie in a given set. Note that the complement of a set is the union of its gaps. Additionally we define a t-*gap *to be a gap of size t which is not a proper subset of a larger gap. See Figure 3.1.*

Lemma 3.2.5. *Two sets A and B almost intersect if and only if the intersection of their respective extended neighbourhoods contain two adjacent points.*

Proof. Throughout this proof we will refer to the two neighbours of any given vertex v from the cycle as $v-1$ and $v+1$.

Let us first prove that if A and B almost intersect then $N(A) \cap N(B)$ contains two adjacent points. If $A \cap B \neq \emptyset$, then there must exist a $v \in A \cap B$. This implies $N(A) \cap N(B)$ contains three consecutive vertices, namely $v-1$, v, and

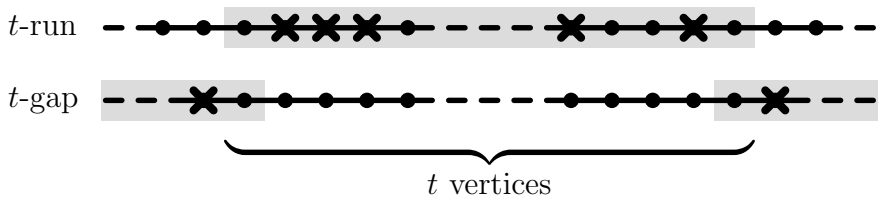

Figure 3.1: Sections of a set on the n-cycle showing a t-run and a t-gap. The crosses indicate vertices that are members of the set. The shaded rectangles enclose vertices that are members of the extended neighbourhood.

$v+1$. If A and B almost intersect but $A \cap B = \emptyset$, then there exists a vertex $v \in A$ and $w \in B$ such that $w = v - 1$ or $v + 1$. In either case $v, w \in N(A) \cap N(B)$, and hence the intersection contains two adjacent points.

Now let us prove that if $N(A) \cap N(B)$ contains two adjacent points then A and B almost intersect. Let the two adjacent points be v and $v + 1$. If $v \in A$ then A and B almost intersect as $v \in N(B)$. Similarly $v + 1 \in A$ implies A and B almost intersect, as does $v \in B$, or $v + 1 \in B$. The only case left to consider is if v and $v + 1$ do not lie in A or B. In such a situation the only way v can lie in $N(A)$ is if $v - 1 \in A$. Similarly $v - 1$ must lie in B, and hence A and B intersect. □

We say that a t-run from a set A *lies* in an r-gap from a set B if the t vertices of the run in A are a subset of the r vertices of the gap in B.

Lemma 3.2.6. *Two sets A and B do not almost intersect if and only if every run in A lies in a gap in B.*

Proof. If every run in A lies in a gap in B then the intersection of the extended neighbourhoods of A and B will only consist of points that are at the boundary of runs in A and B. If A and B almost intersect then the intersection of the extended neighbourhoods contain two adjacent vertices by Lemma 3.2.5. These adjacent vertices must both be at boundaries of runs in A which is impossible.

If A and B do not almost intersect then no point in B can lie in the extended neighbourhood of A. Consequently no point in B corresponds with a point in a run of A. Hence all runs of A lie in gaps of B. □

In order to prove we can get a large almost intersecting family, we look at a typical k-set (on an n-cycle) and show that it has a $(t + 3)$-run and no gaps of

size t or greater, where

$$t = 6\left\lceil -\frac{\log n}{2\log\left(3x(1-x)^2\right)}\right\rceil,$$

and $x = k/n$. By Lemma 3.2.6 any two such sets must almost intersect which will be enough to prove Theorem 3.2.1.

Lemma 3.2.7 (A typical set does not contain a gap of size t or greater)**.** *For any fixed $\epsilon > 0$, if $\frac{1}{4} + \epsilon \leq x \leq \frac{1}{2}$ then $(1 - o(1))\binom{n}{k}$ of k-sets on an n-cycle do not contain a gap of size t or greater.*

Lemma 3.2.8 (A typical set contains a $(t+3)$-run)**.** *For any fixed $\epsilon > 0$, if $\frac{1}{4} + \epsilon \leq x \leq \frac{1}{2}$ then $(1 - o(1))\binom{n}{k}$ of k-sets, on an n-cycle, contain a $(t+3)$-run.*

Proof of Theorem 3.2.1. When $k > n/2$ by the pigeonhole principle any two k-sets will intersect, so $\alpha(n,k) = \binom{n}{k}$. Consequently all we have to prove is that for all fixed $\epsilon > 0$ if $1/4 + \epsilon \leq x \leq 1/2$ then $\alpha(n,k) = (1 - o(1))\binom{n}{k}$. Lemmas 3.2.7 and 3.2.8 tell us that $(1 - o(1))\binom{n}{k}$ of the k-sets contain a $(t+3)$-run and no gaps of size t or greater. Hence by Lemma 3.2.6 any two sets from this family must almost intersect. □

The proofs of Lemmas 3.2.7 and 3.2.8 require us to write various binomials as proportions of $\binom{n}{k}$. Consequently the following lemma will be very useful.

Lemma 3.2.9. *Let f, g be integers that vary with n. If $x = k/n$ satisfies $\epsilon \leq x \leq 1 - \epsilon$ for some fixed $\epsilon > 0$, $f = o(\sqrt{n})$, and $g = o(\sqrt{n})$ then as n tends to infinity we have*

$$\frac{\binom{n-f}{k-g}}{\binom{n}{k}} \sim x^g (1-x)^{f-g}.$$

Proof. First let us rewrite the left hand side using factorials

$$\frac{\binom{n-f}{k-g}}{\binom{n}{k}} = \frac{(n-f)!}{n!}\,\frac{(xn)!}{(xn-g)!}\,\frac{(n-xn)!}{(n-xn-f+g)!}.$$

Stirling's formula tells us that $m! \sim \sqrt{2\pi m}\, m^m e^{-m}$, hence

$$\frac{\binom{n-f}{k-g}}{\binom{n}{k}} \sim \sqrt{\left(\frac{n-f}{n}\right)\left(\frac{xn}{xn-g}\right)\left(\frac{n-xn}{n-xn-f+g}\right)} \times \frac{(n-f)^{n-f}}{n^n}\,\frac{(xn)^{xn}}{(xn-g)^{xn-g}}\,\frac{(n-xn)^{n-xn}}{(n-xn-f+g)^{n-xn-f+g}}.$$

The term inside the square root tends to 1 as $n \to \infty$, because $f = o(\sqrt{n})$, $g = o(\sqrt{n})$, and $\epsilon \le x \le 1-\epsilon$. The remaining terms can be rearranged to give

$$\frac{\binom{n-f}{k-g}}{\binom{n}{k}} \sim \left(1-\frac{f}{n}\right)^{n-f}\left(1-\frac{g}{xn}\right)^{-xn+g}\left(1-\frac{f-g}{(1-x)n}\right)^{-(1-x)n+f-g} x^g(1-x)^{f-g}.$$

Next let us show that $(1-\frac{f}{n})^{n-f} \sim e^{-f}$. This is true if and only if $(1-\frac{f}{n})^{n-f}e^f \to 1$, which is in itself true if and only if $n(1-\frac{f}{n})\log(1-\frac{f}{n}) + f \to 0$. We can use the Taylor expansion for $\log(1-t)$ to show

$$n\log\left(1-\frac{f}{n}\right) + f = -\frac{f^2}{2n} - \frac{f^3}{3n^2} - \frac{f^4}{4n^3} - \cdots.$$

Note that $|f|^2/n + |f|^3/n^2 + |f|^4/n^3 + \cdots$ has a larger magnitude and is a geometric series that sums to

$$\frac{|f|^2}{n}\left(1-\frac{|f|}{n}\right)^{-1}.$$

The sum therefore tends to 0 because $f = o(\sqrt{n})$. Consequently we know

$$\left(n\log\left(1-\frac{f}{n}\right)+f\right)\left(1-\frac{f}{n}\right) = n\left(1-\frac{f}{n}\right)\log\left(1-\frac{f}{n}\right) + f - \frac{f^2}{n} \to 0,$$

and since $f^2/n \to 0$ (because $f = o(\sqrt{n})$) we have the desired statement that $n(1-\frac{f}{n})\log(1-\frac{f}{n}) + f \to 0$.

We can similarly show that $(1-\frac{g}{xn})^{-xn+g} \sim e^g$ and $(1-\frac{f-g}{(1-x)n})^{-(1-x)n+f-g} \sim e^{f-g}$. Therefore $\binom{n-f}{k-g}/\binom{n}{k} \sim e^{-f}e^g e^{f-g}x^g(1-x)^{f-g} = x^g(1-x)^{f-g}$. □

We will use Lemma 3.2.9 to prove that a typical set does not contain a gap of size t or greater.

Proof of Lemma 3.2.7. Any set that contains a gap of size t or greater will have

t consecutive points that are not in the set. An overestimate of the number of such sets is $n\binom{n-t}{k}$, which as a proportion of all k-sets is

$$\frac{n\binom{n-t}{k}}{\binom{n}{k}} \sim n(1-x)^t$$

for large n, by Lemma 3.2.9. We want to show that this value is $o(1)$, or equivalently as $n \to \infty$, we have $n(1-x)^t \to 0$.

Observe that $n(1-x)^t \to 0$ if and only if $\log n + t \log(1-x) \to -\infty$. We can substitute in

$$t = 6\left\lceil -\frac{\log n}{2\log(3x(1-x)^2)} \right\rceil = -\frac{3\log n}{\log(3x(1-x)^2)} + \delta$$

where δ is some real number between 0 and 6 (depending on n). Hence

$$\begin{aligned}\log n + t\log(1-x) &= \log n + \left(-\frac{3\log n}{\log(3x(1-x)^2)} + \delta\right)\log(1-x) \\ &= \left(1 - \frac{3\log(1-x)}{\log(3x(1-x)^2)}\right)\log n + \delta\log(1-x)\end{aligned}$$

Note that $\delta\log(1-x)$ is bounded so we can ignore any contribution it makes. Define

$$f(x) = 1 - \frac{3\log(1-x)}{\log(3x(1-x)^2)}.$$

If we can show $f(x)$ is smaller than some fixed negative constant then we will be done. A graph of $f(x)$ is given in Figure 3.2, it shows that $f(1/4) = 0$ and as x increases $f(x)$ decreases at least until $x = 1/2$. Since we are given $1/4 + \epsilon \leq x \leq 1/2$, we can deduce

$$0 = f(\tfrac{1}{4}) > f(\tfrac{1}{4} + \epsilon) \geq f(x).$$

Hence $f(x)$ is smaller than the fixed negative constant $f(\frac{1}{4} + \epsilon)$ as required.

To make the proof completely rigorous we need to show the derivative of $f(x)$ is negative in the region $1/4 \leq x \leq 1/2$. It is not hard to compute $f'(x)$,

$$f'(x) = \left(\frac{3}{1-x}\right)\frac{\log(3x(1-x)) + \log(1-x)\left(\frac{1-2x}{x}\right)}{(\log(3x(1-x)^2))^2}.$$

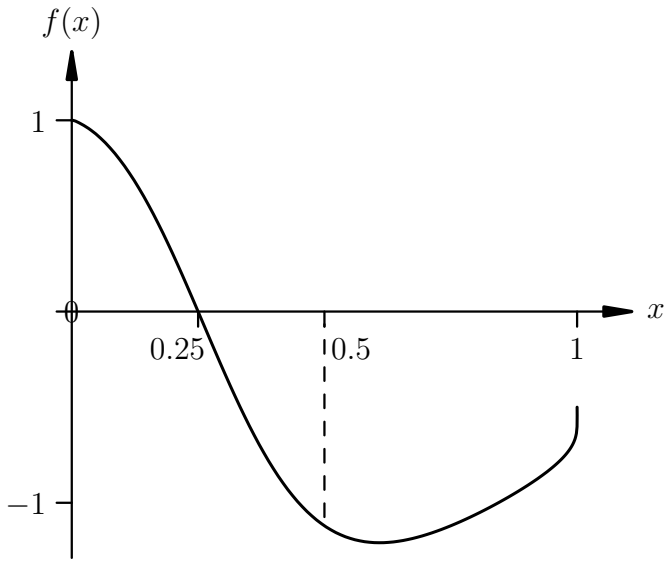

Figure 3.2: Graph of $f(x)$ versus x, illustrating that $f(x)$ is a negative strictly decreasing function when $\frac{1}{4} < x \leq \frac{1}{2}$.

Note that $\frac{3}{1-x}$ and $(\log(3x(1-x)^2))^2$ are positive in the region $1/4 \leq x \leq 1/2$, so the sign of $f'(x)$ is determined completely by the term

$$\log(3x(1-x)) + \log(1-x)\left(\frac{1-2x}{x}\right).$$

$3x(1-x) \leq 3/4$ therefore $\log(3x(1-x)) < 0$, similarly $\log(1-x) < 0$, and $(1-2x)/x \geq 0$, hence $f'(x) < 0$. □

To finish we will prove the harder result that a typical set contains a $(t+3)$-run.

Proof of Lemma 3.2.8. Let us define a specific type of run which we will call a *dominant run*. A dominant run contains exactly $t/2+1$ points from the set. This means that there are exactly $t/2$ spaces between the points from the set. The size of the spaces can be $0, 1$, or 2, see Figure 3.3. (A space of size 3 or greater would cause there to be a gap in the extended neighbourhood.) We further restrict the definition of a dominant run by insisting the number of $0, 1$, and 2 spaces all equal $t/6$. The total number of points that lie in the spaces of a dominant run is $0 \times t/6 + 1 \times t/6 + 2 \times t/6 = t/2$. So the total number of points covered by the spaces in between the points from the set, the points

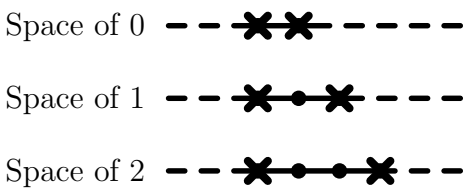

Figure 3.3: Sections of a set on the n-cycle showing spaces of $0, 1$, and 2. The crosses indicate vertices that are members of the set.

from the set themselves, and the points at the boundary is

$$\frac{t}{2} + \left(\frac{t}{2} + 1\right) + 2 = t + 3.$$

So a dominant run is in fact a special type of $(t+3)$-run. We will prove that a typical set contains a $(t+3)$-run by showing that it contains a dominant run.

Under the given definition how many ways can we construct a dominant run? Equivalently we can ask, how many combinations of the $0, 1$, and 2 spaces are there? This is easily calculated to be

$$\frac{\left(\frac{t}{2}\right)!}{\left(\frac{t}{6}\right)!\left(\frac{t}{6}\right)!\left(\frac{t}{6}\right)!},$$

and Stirling's formula tells us that for large t

$$\frac{\left(\frac{t}{2}\right)!}{\left(\frac{t}{6}\right)!\left(\frac{t}{6}\right)!\left(\frac{t}{6}\right)!} \sim \frac{3^{(t+3)/2}}{\pi t}.$$

Using this fact we will now calculate the expected number of dominant runs there are in a random k-set. By calculating the variance and using Chebyshev's inequality we will show that a typical k-set has a dominant run with high probability.

Let us number the points in the cycle from 1 to n and define random variables X_i in the following way

$$X_i = \begin{cases} 1, & \text{if there is a dominant run starting at vertex } i, \\ 0, & \text{otherwise.} \end{cases}$$

Now we can say for a random k-set

$$\mathbf{P}[X_i = 1] \sim \frac{3^{(t+3)/2}}{\pi t} \binom{n-(t+7)}{k-(t/2+1)} \binom{n}{k}^{-1}.$$

The $\binom{n-(t+7)}{k-(t/2+1)}$ term comes from the fact that we fix $t/2+1$ of the points in the set to be one of the $3^{(t+3)/2}/(\pi t)$ arrangements previously described. The remainder of the points in the dominant run are not in the set. Furthermore the points just outside the boundary of the run must not lie in the set to ensure we have a $(t+3)$-run and not a larger run. Hence overall we are fixing $t+7$ points and are free to choose the remaining $k-(t/2+1)$ members of the set from $n-(t+7)$ points.

Lemma 3.2.9 allows us to simplify the binomial parts in the expression of $\mathbf{P}[X_i = 1]$, so for large n

$$\mathbf{P}[X_i = 1] \sim \frac{c}{t}(3x(1-x))^{t/2} \quad \text{where} \quad c = \frac{3\sqrt{3}}{\pi} x(1-x)^6.$$

Note that c is bounded. Define X to be the number of dominant runs in a random k-set,

$$X = \sum_{i=1}^{n} X_i.$$

By the linearity of expectation we have

$$\mathbf{E}[X] = \sum_{i=1}^{n} \mathbf{E}[X_i] \sim n\frac{c}{t}(3x(1-x))^{t/2}.$$

We will eventually show that X is sufficiently concentrated about $\mathbf{E}[X]$ that $\mathbf{P}[X = 0] = o(1)$. For this to occur we require that $\mathbf{E}[X]$ tend to something larger than 1. In fact we will show that $\mathbf{E}[X]$ tends to infinity. By taking logs we see that $\mathbf{E}[X] \to \infty$ if and only if

$$\log n + \log c - \log t + \frac{t}{2}\log\left(3x(1-x)\right) \to \infty.$$

We can substitute in

$$t = -\frac{3\log n}{\log\left(3x(1-x)^2\right)} + \delta$$

where δ is some real number between 0 and 6 (depending on n). Now we have

$$\left(1 - \frac{3\log(3x(1-x))}{2\log(3x(1-x)^2)}\right)\log n - \log t + \log c + \frac{\delta}{2}\log(3x(1-x)).$$

The $\log c$ and $\frac{\delta}{2}\log(3x(1-x))$ terms are bounded so we can ignore any contribution they make. If we can show that the coefficient of $\log n$ is greater than a fixed positive constant then we will be done. This is because n to the power of a fixed positive constant grows much faster than t, which looks like $\log n$. Let us define the function $g(x)$ as

$$g(x) = 1 - \frac{3\log(3x(1-x))}{2\log(3x(1-x)^2)}.$$

Recall that in the proof of Lemma 3.2.7, we defined

$$f(x) = 1 - \frac{3\log(1-x)}{\log(3x(1-x)^2)}.$$

The two functions are similar in form and in fact $g(x) = -\frac{1}{2}f(x)$. This means $g(1/4) = 0$ and $g(x)$ is a strictly increasing function in the range $1/4 \leq x \leq 1/2$. Thus

$$0 = g(\tfrac{1}{4}) < g(\tfrac{1}{4} + \epsilon) \leq g(x)$$

and hence $\mathbf{E}[X] \to \infty$. Note that we have shown that $nc(3x(1-x))^{t/2}$ grows as fast or faster than n to the power of some fixed positive constant, this will be useful later.

In order to apply Chebyshev's inequality we need to calculate the variance of X which we can obtain by looking at $\mathbf{E}[X^2]$. By the linearity of expectation,

$$\mathbf{E}[X^2] = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{E}[X_iX_j] = \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{P}[(X_i = 1) \cap (X_j = 1)].$$

There are $t+7$ consecutive vertices which determine whether a dominant run starts at i ($t+3$ vertices which make up the run, plus an additional two at either boundary to ensure it isn't a larger run). If i and j are at least $t+7$ vertices

apart then it is not hard to see that

$$\begin{aligned}\mathbf{P}[(X_i = 1) \cap (X_j = 1)] &= \left(\frac{(\frac{t}{2})!}{(\frac{t}{6})!\,(\frac{t}{6})!\,(\frac{t}{6})!}\right)^2 \binom{n-2(t+7)}{k-2(t/2+1)}\binom{n}{k}^{-1} \\ &\sim \frac{\mathbf{E}[X]^2}{n^2}.\end{aligned}$$

If i and j are less than $t+7$ vertices apart then we bound the probability as

$$\mathbf{P}[(X_i = 1) \cap (X_j = 1)] \leq \mathbf{P}[X_i = 1] = \frac{\mathbf{E}[X]}{n}.$$

Out of the n^2 choices of i and j, there are $n(2t+13)$ in which i and j are less than $t+7$ vertices apart. Hence asymptotically we have the following bound on $\mathbf{E}[X^2]$

$$\mathbf{E}[X^2] \leq n(2t+13)\left(\frac{\mathbf{E}[X]}{n}\right) + n^2\left(\frac{\mathbf{E}[X]^2}{n^2}\right),$$

and therefore

$$\mathbf{Var}[X] \leq (2t+13)\mathbf{E}[X].$$

Applying Chebyshev's inequality gives

$$\begin{aligned}\mathbf{P}\left[|X - \mathbf{E}[X]| \geq \mathbf{E}[X]^{2/3}\right] &\leq \frac{\mathbf{Var}[X]}{\mathbf{E}[X]^{4/3}} \\ &\leq \frac{2t+13}{\mathbf{E}[X]^{1/3}} \sim \left(\frac{8t^4}{nc(3x(1-x))^{t/2}}\right)^{1/3} \to 0\end{aligned}$$

because as we showed earlier $nc(3x(1-x))^{t/2}$ grows as fast as n to some fixed positive power, whereas t grows like $\log n$. Consequently we can say that $(1 - o(1))\binom{n}{k}$ of the sets satisfy

$$X > \mathbf{E}[X] - \mathbf{E}[X]^{2/3},$$

and when n is large $\mathbf{E}[X] - \mathbf{E}[X]^{2/3} \geq 1$ (in fact it tends to infinity). So a typical set contains at least one dominant run and therefore contains a $(t+3)$-run. $\square$

3.3 Higher powered almost intersecting families

Define $d(a, b)$, the *distance* between vertices a and b on a cycle, to be the number of edges in the shortest path from a to b. In this section we extend our result from cycles to powers of cycles. The p-th power of a cycle is formed by adding edges between any two vertices which are a distance of p or less from each other.

Theorem 3.3.1. *For a fixed power $p \geq 1$, let $\alpha_p(n, k)$ be the size of the largest almost intersecting family, of k-sets, on the p-th power of the n-cycle. For any fixed $\epsilon > 0$, if $\frac{k}{n} \geq \frac{1}{2p+2} + \epsilon$ then*

$$\alpha_p(n, k) = (1 - o(1))\binom{n}{k}.$$

Rather than work directly with powers of cycles, we will work with simple cycles and extend our definition of almost intersection instead. Doing so will make the generalisations clearer. The proof is almost identical to that given in the previous section but there are a few differences. As before we define x to be k/n, and we assume that k varies with n.

Definition 3.3.2 (Extended Neighbourhood of Power p)**.** *Let the* extended neighbourhood of power p *of a set A be defined as*

$$N_p(A) = \bigcup_{\forall a \in A} \{b : d(a, b) \leq p\},$$

i.e. all points that are at most a distance p from some point in A.

Definition 3.3.3 (Almost Intersection of Power p)**.** *Given sets A, B, and a power $p \geq 1$, if $A \cap N_p(B) \neq \emptyset$ then we say A and B* almost intersect*. Equivalently A and B* almost intersect *if and only if there exists some $a \in A$ and some $b \in B$ such that $d(a, b) \leq p$.*

Note that when $p = 1$ we recover our original definitions of almost intersecting sets and extended neighbourhoods.

The definition of a gap remains the same, see Definition 3.2.4. However, the definition of a run changes slightly to incorporate our generalisation of the extended neighbourhood.

Definition 3.3.4 (Run of Power p)**.** *We define a* run of power p *to be a set of consecutive points that are in the extended neighbourhood of power p of a set. Furthermore let us define a* t-run *to be a run of size t which is not a proper subset of a larger run.*

Lemma 3.3.5. *Given a fixed power $p \geq 1$, and $n \geq p+1$ (where n is the size of our cycle), two sets A and B almost intersect if and only if $N_p(A) \cap N_p(B)$ contains at least $p+1$ consecutive points.*

Proof. First let us show that almost intersection implies there are $p+1$ consecutive points in the intersection of the extended neighbourhoods. If A and B almost intersect then there are points $a \in A$ and $b \in B$ such that $d(a,b) \leq p$. By considering $N_p(\{a\}) \cap N_p(\{b\})$ it is easy to see $N_p(A) \cap N_p(B)$ contains at least $p+1$ consecutive points.

Now let us prove that if we have $p+1$ consecutive points in the intersection of the extended neighbourhoods then the sets must almost intersect. Without loss of generality let us assume the $p+1$ points are $C = \{1, \ldots, p+1\}$. If a vertex from A lies in C then it also lies in $N_p(B)$ and so A and B must be almost intersecting. If no vertex from A lies in C then the only way vertex $p+1$ can be in $N_p(A)$ is if a vertex of A lies in $D = \{p+2, \ldots, 2p+1\}$. Similarly if a vertex from B lies in C we have almost intersection otherwise it must contain a vertex in D. However if both A and B contain vertices in D then they must almost intersect as $|D| = p$. □

Lemma 3.3.6. *Given a power p, sets A and B do not almost intersect if and only if every run of power p in A lies in a larger or equal gap in B.*

Proof. If A and B do not almost intersect then no point in B lies in $N_p(A)$ (by the definition of almost intersection). Consequently every point in a run of A corresponds to a point which does not lie in B. Hence runs in A lie in larger or equal gaps in B.

Suppose every run in A lies in a larger or equal gap in B. We will show A and B do not almost intersect by showing that $N_p(A) \cap N_p(B)$ contains at most p consecutive vertices (see Lemma 3.3.5). Note that the extended neighbourhood of a set is the union of its runs, so it is enough to prove the intersection of a run in A and a run in B contains at most p consecutive vertices. The smallest possible run in A is of size $2p+1$ and the largest possible gap in B which lies

entirely within a run of B is of size $2p$. Hence no run in A lies in a gap which lies completely within a run of B. A run in A may lie in a gap which lies partially in a run of B. Such gaps only occur at the boundary of runs in B, and the overlap between the gap and the run in B is precisely p consecutive vertices. This is enough to show that the intersection of a run in A and a run in B contains at most p consecutive vertices and therefore A and B do not almost intersect. □

To prove Theorem 3.3.1 we will show that a typical k-set (on an n-cycle) has a $(t+2p+1)$-run and no gaps of size t or greater, where

$$t = (2p+1)(p+1)\left\lceil -\frac{\log n}{(p+1)\log\left((2p+1)x(1-x)^{2p}\right)}\right\rceil.$$

By Lemma 3.3.6 any two such sets will almost intersect.

Lemma 3.3.7 (A typical set does not contain a gap of size t or greater)**.** *For a fixed power $p \geq 1$, and any fixed $\epsilon > 0$, if $\frac{1}{2p+2} + \epsilon \leq x \leq \frac{1}{p+1}$ then $(1-o(1))\binom{n}{k}$ of k-sets, on an n-cycle, do not contain a gap of size t or greater.*

Lemma 3.3.8 (A typical set contains a $(t+2p+1)$-run)**.** *For a fixed power $p \geq 1$, and any fixed $\epsilon > 0$, if $\frac{1}{2p+2} + \epsilon \leq x \leq \frac{1}{p+1}$ then $(1-o(1))\binom{n}{k}$ of k-sets, on an n-cycle, contain a $(t+2p+1)$-run.*

Proof of Theorem 3.3.1. Rather than work with powers of cycles, we will instead consider the equivalent problem of almost intersection of power p on a simple cycle.

Since almost intersection of power $p-1$ implies almost intersection of power p, by induction it is enough to prove $\alpha_p(n,k) = (1-o(1))\binom{n}{k}$ holds for $\frac{1}{2p+2}+\epsilon \leq x \leq \frac{1}{p+1}$ (as $\frac{1}{p+1} > \frac{1}{2p}$ for $p \geq 2$). Note that the case $p = 1$ is given by Theorem 3.2.1. When x is in this range Lemmas 3.3.7 and 3.3.8 tell us that $(1-o(1))\binom{n}{k}$ of the k-sets contain a $(t+2p+1)$-run and no gaps of size t or greater. Hence by Lemma 3.3.6 any two sets from this family must almost intersect. □

Our proof of Lemma 3.3.7 will require the following lemma.

Lemma 3.3.9. *For an integer $p \geq 1$, and any $0 \leq x \leq 1$ we have*

$$(2p+1)x(1-x)^p < 1.$$

Proof. Define $f(x) = (2p+1)x(1-x)^p$. By considering $f'(x)$ it is easy to show the maximum value of $f(x)$, in the region $0 \leq x \leq 1$, occurs when $x = \frac{1}{p+1}$. So if we can show $f(\frac{1}{p+1}) < 1$ we will be done.

$$\begin{aligned} f\left(\frac{1}{p+1}\right) &= \frac{(2p+1)p^p}{(p+1)^{p+1}} \\ &= \frac{2p^{p+1} + p^p}{p^{p+1} + (p+1)p^p + \binom{p+1}{2}p^{p-1} + \cdots} \\ &= \frac{2p^{p+1} + p^p}{(2p^{p+1} + p^p) + \binom{p+1}{2}p^{p-1} + \cdots} \end{aligned}$$

which is obviously less than 1 provided $p \geq 1$. □

Proof of Lemma 3.3.7. The proof is almost identical to that of Lemma 3.2.7. Any set that contains a gap of size t or greater will have t consecutive points that are not in the set. An overestimate of the number of such sets is $n\binom{n-t}{k}$, which as a proportion of all k-sets is

$$\frac{n\binom{n-t}{k}}{\binom{n}{k}} \sim n(1-x)^t$$

for large n (by Lemma 3.2.9). We want to show that as $n \to \infty$, we have $n(1-x)^t \to 0$. Observe that $n(1-x)^t \to 0$ if and only if $\log n + t\log(1-x) \to -\infty$. We can substitute in

$$\begin{aligned} t &= (2p+1)(p+1)\left\lceil -\frac{\log n}{(p+1)\log\left((2p+1)x(1-x)^{2p}\right)}\right\rceil \\ &= -\frac{(2p+1)\log n}{\log\left((2p+1)x(1-x)^{2p}\right)} + \delta \end{aligned}$$

where δ is some real number between 0 and $(2p+1)(p+1)$ (depending on n). Hence

$$\log n + t\log(1-x) = f(x)\log n + \delta\log(1-x),$$

where

$$f(x) = 1 - \frac{(2p+1)\log(1-x)}{\log\left((2p+1)x(1-x)^{2p}\right)}.$$

It is easy to check that $f(\frac{1}{2p+2}) = 0$, furthermore if f is strictly decreasing in the region $[\frac{1}{2p+2}, \frac{1}{p+1}]$, then

$$0 = f(\tfrac{1}{2p+2}) > f(\tfrac{1}{2p+2} + \epsilon) \geq f(x).$$

Hence $f(x)$ would be smaller than some fixed negative constant, which is enough to prove $f(x)\log n + \delta \log(1-x) \to -\infty$.

To show f is strictly decreasing we look at its derivative,

$$f'(x) = \frac{(2p+1)\left(x\log\left((2p+1)x(1-x)^p\right) + (1-(p+1)x)\log(1-x)\right)}{x(1-x)(\log\left((2p+1)x(1-x)^{2p}\right))^2}.$$

It should be clear that the sign of $f'(x)$ in the region $\frac{1}{2p+2} \leq x \leq \frac{1}{p+1}$ is completely determined by the term

$$x\log\left((2p+1)x(1-x)^p\right) + (1-(p+1)x)\log(1-x).$$

Since $x \leq \frac{1}{p+1}$ we have $(1-(p+1)x)\log(1-x) \leq 0$. From Lemma 3.3.9 we have $\log\left((2p+1)x(1-x)^p\right) < 0$ therefore $x\log\left((2p+1)x(1-x)^p\right) < 0$ and consequently $f'(x) < 0$ as required. □

We finish by proving that a typical set contains a $(t+2p+1)$-run.

Proof of Lemma 3.3.8. Let us extend our definition of a *dominant run* to take into account the power p. A dominant run contains exactly $\frac{t}{p+1}+1$ points from the set. This means that there are exactly $\frac{t}{p+1}$ spaces between the points from the set. The size of the spaces can be $0, 1, 2, \ldots, 2p$. A space of size $2p+1$ or greater will cause there to be a gap in the extended neighbourhood. We further restrict the definition of a dominant run by insisting the number of $0, 1, 2, \ldots, 2p$ spaces are all equal, and hence are $\frac{t}{(2p+1)(p+1)}$. The total number of points that lie in the spaces of a dominant run is

$$(0+1+2+\cdots+2p)\frac{t}{(2p+1)(p+1)} = \frac{pt}{p+1}.$$

Therefore the total number of points covered by the spaces in between the points from the set, the points from the set themselves, and the points at the boundary

is

$$\frac{pt}{p+1} + \left(\frac{t}{p+1} + 1\right) + 2p = t + 2p + 1$$

So a dominant run is in fact a special type of $(t+2p+1)$-run. We will prove that a typical set contains a $(t+2p+1)$-run by showing that it contains a dominant run. The proof is almost identical to that of Lemma 3.2.8.

Let us number the points in the cycle from 1 to n and define random variables X_i in the following way

$$X_i = \begin{cases} 1, & \text{if there is a dominant run starting at vertex } i, \\ 0, & \text{otherwise.} \end{cases}$$

For a random k-set

$$\mathbf{P}[X_i = 1] = \frac{\left(\frac{t}{p+1}\right)!}{\left(\left(\frac{t}{(2p+1)(p+1)}\right)!\right)^{2p+1}} \binom{n-(t+4p+3)}{k-(\frac{t}{p+1}+1)} \binom{n}{k}^{-1}.$$

Stirling's formula and Lemma 3.2.9 allows us to simplify the expression to

$$\mathbf{P}[X_i = 1] \sim \frac{c}{t^p}((2p+1)x(1-x)^p)^{\frac{t}{p+1}},$$

where

$$c = \sqrt{2p+1}\left(\frac{(2p+1)(p+1)}{2\pi}\right)^p x(1-x)^{4p+2}.$$

Note that c is bounded. Define X to be the number of dominant runs in a random k-set. Hence

$$\mathbf{E}[X] = \sum_{i=1}^{n} \mathbf{E}[X_i] \sim n\frac{c}{t^p}((2p+1)x(1-x)^p)^{\frac{t}{p+1}}.$$

As in the proof of Lemma 3.2.8 we require that $\mathbf{E}[X]$ tends to infinity which is true if and only if

$$\log n + \log c - p\log t + \frac{t}{p+1}\log\left((2p+1)x(1-x)^p\right) \to \infty.$$

Substituting in

$$t = -\frac{(2p+1)\log n}{\log\left((2p+1)x(1-x)^{2p}\right)} + \delta,$$

where δ is some real number between 0 and $(2p+1)(p+1)$, gives,

$$g(x)\log n + \log c - p\log t + \frac{\delta}{p+1}\log\left((2p+1)x(1-x)^{p}\right)$$

where

$$g(x) = 1 - \frac{(2p+1)\log\left((2p+1)x(1-x)^{p}\right)}{(p+1)\log\left((2p+1)x(1-x)^{2p}\right)}.$$

It is enough to show that $g(x)$ is greater than some fixed positive constant, which we can prove by observing that $g(x) = -\frac{p}{p+1}f(x)$, where $f(x)$ was defined in the proof of Lemma 3.3.7.

To finish the proof we follow the same argument as that given in the proof of Lemma 3.2.8 to show that asymptotically

$$\mathbf{Var}[X] \leq (2t+8p+5)\mathbf{E}[X],$$

and an application of Chebyshev's inequality tells us that

$$\mathbf{P}\left[|X - \mathbf{E}[X]| \geq \mathbf{E}[X]^{\frac{2}{3}}\right] \to 0,$$

hence $(1-o(1))\binom{n}{k}$ of the k-sets contain a $(t+2p+1)$-run. □

3.4 Almost cross-intersecting families

In this section we generalize the notion of almost intersection in to the domain of cross-intersection. We apply the same idea of runs and gaps to get a bound for almost cross-intersecting families to be almost everything. This bound is more complicated than the previous bounds but it is simple enough that we can write it in a closed form.

To begin with let us describe what we mean by cross-intersection. Given two families of sets $\mathcal{A}$, and $\mathcal{B}$ we say they are cross-intersecting if for all $A \in \mathcal{A}$ and $B \in \mathcal{B}$, $A \cap B \neq \emptyset$. To define almost cross-intersecting, we consider two families $\mathcal{A}, \mathcal{B}$ whose sets are subsets of the vertices of an n-cycle. We say $\mathcal{A}$ and

$\mathcal{B}$ almost cross-intersect if for all $A \in \mathcal{A}$ and $B \in \mathcal{B}$, A and B almost intersect on the n-cycle. We will specifically be interested in when the families consist of sets of a fixed size, i.e. when $\mathcal{A}$ contains only k_1-sets and $\mathcal{B}$ is made up of k_2-sets.

Note that the definitions of extended neighbourhoods, runs, and gaps given in Section 3.2 still hold for the almost cross-intersecting problem. Also Lemma 3.2.6 still applies, which implies that if there is a run in set A which is larger than every gap in set B then A and B must almost intersect. Consequently if for some t every set in $\mathcal{A}$ contains a run of size t or greater, and no set in $\mathcal{B}$ contains a gap of size t or greater then $\mathcal{A}$ and $\mathcal{B}$ must almost cross-intersect. We will use this idea in proving our main result.

Theorem 3.4.1. *Let $x, y \in [0,1]$ be fixed rational values that satisfy $r(x,y) > 0$, where*

$$r(x,y) = 1 - (1-x)^3 - (1-y)^3 - xy(3-x-y).$$

Given $\epsilon > 0$ there exists a positive integer n (such that xn and yn are integers) and almost cross-intersecting families on the n-cycle $\mathcal{A}, \mathcal{B}$, that consist of xn-sets and yn-sets respectively, such that

$$|\mathcal{A}| \geq (1-\epsilon)\binom{n}{xn} \qquad \text{and} \qquad |\mathcal{B}| \geq (1-\epsilon)\binom{n}{yn}.$$

Theorem 3.4.1 tells us that $r(x,y) > 0$ implies there exist almost cross-intersecting families that are almost everything for large n. Figure 3.4 shows a graph of $r(x,y) = 0$. The proof of Theorem 3.4.1 is similar to that of Theorem 3.2.1 and so unsurprisingly we find that $r(\frac{1}{4}, \frac{1}{4}) = 0$, which directly links the two results. As a side note we find that $r(x,y)$ is interestingly a factor of the much simpler expression $x(1-x)^3 - y(1-y)^3$,

$$x(1-x)^3 - y(1-y)^3 = (y-x)r(x,y).$$

Though what role, if any, the quantity $x(1-x)^3 - y(1-y)^3$ plays is unclear.

Before we begin proving Theorem 3.4.1 we will need a few technical lemmas. The lemmas will only consider $x, y \in (0,1)$ as it will be easier to deal with the remaining trivial cases separately. We start by rigorously proving some features of $r(x,y)$ which are apparent in Figure 3.4.

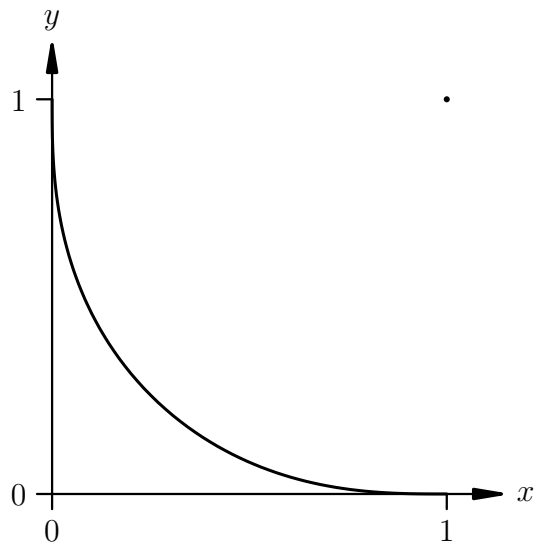

Figure 3.4: A graph of $r(x, y) = 0$. The points which satisfy $r(x, y) = 0$ consist of the curve from $(0, 1)$ to $(1, 0)$, and the isolated point at $(1, 1)$. Points lying above the curve satisfy $r(x, y) > 0$ (with the exception of the point at $(1, 1)$).

Lemma 3.4.2. *Given a fixed $0 < x < 1$, there is exactly one real value of y which satisfies $r(x, y) = 0$, which we will call y_0. Furthermore $0 < y_0 < 1$ holds true, and*

$$\begin{aligned} r(x, y) &< 0 \quad \textit{when } y < y_0, \\ r(x, y) &> 0 \quad \textit{when } y > y_0. \end{aligned}$$

Proof. $r(x, y)$ is a cubic in y, so has at most two turning points (a maxima and a minima). If we can show that $r(x, y) > 0$ at the minima then we can conclude that $r(x, y)$ has only one root.

$$\frac{dr}{dy} = 3y^2 + (2x - 6)y + (x^2 - 3x + 3)$$

hence $\frac{dr}{dy} = 0$ when $y = \frac{1}{3}(-x + 3 \pm \omega)$ where $\omega = \sqrt{-2x^2 + 3x}$. It is easy to check that $-2x^2 + 3x > 0$ for $0 < x < 1$, so two distinct turning points occur, and since the coefficient of y^3 in $r(x, y)$ is positive the minima must occur at

$y = \frac{1}{3}(-x + 3 + \omega)$. Therefore to show only one root occurs we need to show

$$r\left(x, \tfrac{1}{3}(-x + 3 + \omega)\right) = \frac{x}{27}\left(20x^2 - 45x + 27 + (4x - 6)\omega\right)$$

is greater than 0. This is true if and only if $20x^2 - 45x + 27 > (-4x + 6)\omega$. It is straightforward to check that both sides of this inequality are positive when $0 < x < 1$, so it is equivalent to prove $(20x^2 - 45x + 27)^2 - (-4x + 6)^2\omega^2 > 0$. Since

$$(20x^2 - 45x + 27)^2 - (-4x + 6)^2\omega^2 = 27(x - 1)^2((4x - 5)^2 + 2)$$

we are done.

Because $r(x, 0) = -(1 - x)^3 < 0$ and $r(x, 1) = x(1 - x)^2 > 0$, we can conclude that the only root, y_0, lies between 0 and 1. Since $r(x, y)$ is a cubic in y with precisely one root y_0, it is trivial to show $r(x, y) > 0$ when $y > y_0$, and $r(x, y) < 0$ when $y < y_0$. □

The proof of Theorem 3.4.1 involves the use of dominant runs just as in the proof of Lemma 3.2.8. However, the precise definition of a dominant run will now depend on x. Rather than define a dominant run directly using x it will be more convenient to define it using α, a variable representing the proportion of the run which is part of the underlying set. The next lemma describes the relationship between α and the variables a_0, a_1, and a_2 which will be used in specifying the number of $0, 1$, and 2 spaces in the definition of a dominant run. The lemma following it defines the relationship between x and α.

Lemma 3.4.3. *For α a real value let us define*

$$\begin{aligned}
\beta &= \sqrt{-8\alpha^2 + 12\alpha - 3}, \\
a_0 &= \tfrac{1}{6}(10\alpha - 3 - \beta), \\
a_1 &= \tfrac{1}{3}(-\alpha + \beta), \\
a_2 &= \tfrac{1}{6}(-2\alpha + 3 - \beta).
\end{aligned}$$

When $\frac{1}{3} < \alpha < 1$ holds, a_0, a_1, and a_2 are real numbers that are strictly greater

than 0. *Furthermore, they satisfy the following properties*

$$\begin{aligned} a_0 + a_1 + a_2 &= \alpha, \\ a_1 + 2a_2 &= 1 - \alpha, \\ a_0 a_2 &= a_1^2. \end{aligned}$$

It is easy to check that the claims asserted in the lemma are true so no proof will be given.

Lemma 3.4.4. *For* $0 < x < 1$ *there exists a unique* $\frac{1}{3} < \alpha < 1$ *such that*

$$x = \frac{a_0^2}{\alpha a_1 + a_0^2}$$

where a_0 *and* a_1 *are functions of* α *defined in Lemma 3.4.3.*

Proof. Define the function $s(\alpha)$ as

$$s(\alpha) = \frac{a_0^2}{\alpha a_1 + a_0^2}$$

By looking at some of the properties of $s(\alpha)$ we will show that it maps the interval $(\frac{1}{3}, 1]$ to $(0, 1]$ and that an inverse exists, which is sufficient to prove the lemma.

It is easy to check that $s(1) = 1$. When $\alpha = \frac{1}{3}$ both a_0 and a_1 are 0, so $s(\frac{1}{3})$ is ill-defined. Instead let us calculate $s(\frac{1}{3} + \epsilon)$ for a small $\epsilon > 0$. Evaluating β at $\alpha = \frac{1}{3} + \epsilon$ gives

$$\begin{aligned} \beta &= \sqrt{-8\left(\frac{1}{3} + \epsilon\right)^2 + 12\left(\frac{1}{3} + \epsilon\right) - 3} \\ &= \frac{1}{3}\sqrt{1 + 60\epsilon + O(\epsilon^2)}, \end{aligned}$$

and by applying the Taylor series expansion of $\sqrt{1+t}$ we get

$$\beta = \frac{1}{3} + 10\epsilon + O(\epsilon^2).$$

Consequently it is straightforward to show that $a_0 = O(\epsilon^2)$, and $\alpha a_1 = \epsilon + O(\epsilon^2)$.

Therefore

$$s(\tfrac{1}{3}+\epsilon) = \frac{O(\epsilon^4)}{\epsilon + O(\epsilon^2)} = \frac{O(\epsilon^3)}{1+O(\epsilon)}.$$

So we can make $s(\alpha)$ as close to 0 as we like by taking α very close to $\frac{1}{3}$. Furthermore $s(\frac{1}{3}+\epsilon) > 0$ holds by Lemma 3.4.3. Therefore for $\alpha \in (\frac{1}{3}, 1]$ the range of $s(\alpha)$ contains the interval $(0, 1]$ (by the intermediate value theorem).

It is straightforward to calculate that

$$\frac{ds}{d\alpha} = \frac{a_0^2}{\beta(\alpha a_1 + a_0^2)^2}$$

which is strictly greater than 0 for $\alpha \in (\frac{1}{3}, 1]$. This completes the proof as it shows $s(\alpha)$ is a strictly increasing function, and therefore provides a bijection between $(\frac{1}{3}, 1]$ and $(0, 1]$. □

The expression $\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha}$ will occur when we calculate the expected number of dominant runs in a xn-set. The following lemma will prove useful when dealing with it.

Lemma 3.4.5. *Given $0 < x < 1$, let $\alpha \in (\frac{1}{3}, 1)$ given by Lemma 3.4.4, satisfy*

$$x = \frac{a_0^2}{\alpha a_1 + a_0^2}.$$

If y satisfies $r(x, y) > 0$, then

$$1 - y < \frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha} < 1.$$

Proof. First let us simplify the expression

$$\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha}$$

by substituting $x = \frac{a_0^2}{\alpha a_1 + a_0^2}$ and eliminating a_2 by using the property $a_0 a_2 = a_1^2$, given by Lemma 3.4.3. Hence

$$\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha} = a_0^{2\alpha - a_0 + a_2} a_1^{1-\alpha-a_1-2a_2} \frac{\alpha}{\alpha a_1 + a_0^2}$$

which reduces further to

$$\frac{\alpha a_0}{\alpha a_1 + a_0^2}$$

using the fact that $2\alpha - a_0 + a_2 = 1$ and $1 - \alpha - a_1 - 2a_2 = 0$, see Lemma 3.4.3.

Consequently the inequalities we are required to prove can be simplified to

$$0 < y_0 < y,$$

where

$$y_0 = 1 - \frac{\alpha a_0}{\alpha a_1 + a_0^2},$$

to ease notation. To prove this we consider

$$r(x, y_0) = \frac{\alpha^2 a_0^2}{(\alpha a_1 + a_0^2)^3} \left(a_1^2 - a_0(\alpha - a_0 - a_1)\right).$$

Using Lemma 3.4.3 we see that $a_1^2 - a_0(\alpha - a_0 - a_1)$ simplifies to $a_1^2 - a_0 a_2$ which in turn reduces to 0. Hence $r(x, y_0) = 0$ and by Lemma 3.4.2 we know that $0 < y_0 < 1$. Furthermore, since y satisfies $r(x, y) > 0$, we have $y_0 < y$. □

We are now ready to begin proving Theorem 3.4.1.

Proof of Theorem 3.4.1. First let us deal with the trivial cases of when x is 0 or 1. If $x = 0$ then $r(x, y) > 0$ becomes $-(1 - y)^3 > 0$ which is never true, for $y \in [0, 1]$. If $x = 1$ then $r(x, y) = y(1 - y)^2 > 0$ which implies $y \neq 0$. Clearly we can find a positive n such that yn is an integer, since y is rational. It should be equally obvious that the n-set intersects with every yn-set provided $y \neq 0$. Hence taking $\mathcal{A}$ to be the family consisting of the n-set, and $\mathcal{B}$ to be the family of all yn-sets, is enough to satisfy the conditions of the theorem for $x = 1$.

We have dealt with the case when $x = 0$ or 1 and by symmetry this also covers the case when $y = 0$ or 1. We may therefore assume throughout the rest of the proof that $0 < x < 1$ and $0 < y < 1$. We will also be fixing $\frac{1}{3} < \alpha < 1$ such that

$$x = \frac{a_0^2}{\alpha a_1 + a_0^2}$$

which we are free to do by Lemma 3.4.4. We take the definitions of a_0, a_1, and a_2 as those given in Lemma 3.4.3.

We will prove the theorem in the standard way using runs and gaps. It should be clear that there exists arbitrarily large n such that xn and yn are integers. As such n get larger, we will show that the probability a random xn-set has at least one run of size $t+3$ tends to 1 as does the probability a random yn-set has no gaps of size t or greater, where

$$t = \lceil h(x,y) \log n \rceil .$$

The function $h(x,y)$ is not critical but in order for the proof to work we require that $h(x,y) > 0$ (so that $t > 0$) and that it satisfies

$$\log(1-y) < -\frac{1}{h(x,y)} < \log\left(\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha}\right). \tag{3.1}$$

Where these conditions come from will become apparent later. All these conditions can be met due to Lemma 3.4.5. We choose an $h(x,y)$ that satisfies these conditions by taking $-1/h(x,y)$ to be the average of its two bounds. Hence

$$h(x,y) = \frac{-2}{\log\left(\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha}\right) + \log(1-y)}.$$

Now that we have defined $h(x,y)$ we will show that a typical yn-set does not contain a gap of size t or greater.

Any set that contains a gap of size t or greater will have t consecutive points that are not in the set. An overestimate of the number of such sets is $n\binom{n-t}{yn}$, which as a proportion of all yn-sets is

$$\frac{n\binom{n-t}{yn}}{\binom{n}{yn}} \sim n(1-y)^t$$

for large n by Lemma 3.2.9. We want to show that $n(1-y)^t \to 0$ as $n \to \infty$. Observe that $n(1-y)^t \to 0$ if and only if $\log n + t\log(1-y) \to -\infty$. We can substitute in

$$\begin{aligned} t &= \lceil h(x,y) \log n \rceil \\ &= h(x,y) \log n + \delta \end{aligned}$$

where δ is some real number between 0 and 1 (depending on n). Hence

$$\log n + t\log(1-y) = (1 + h(x,y)\log(1-y))\log n + \delta\log(1-y),$$

which clearly tends to $-\infty$ if $1 + h(x,y)\log(1-y) < 0$. This holds true since

$$\log(1-y) < -\frac{1}{h(x,y)}$$

by the way we chose $h(x,y)$, see (3.1). Therefore the proportion of yn-sets that have a gap of size t or greater tends to 0.

To complete the proof we will show that the proportion of xn-sets that contain a $(t+3)$-run tends to 1, as $n \to \infty$. As in the proof of Lemma 3.2.8 we define a special type of $(t+3)$-run which we will refer to as a dominant run. However, our definition of a dominant run now depends on α which in turn is chosen so that

$$x = \frac{a_0^2}{\alpha a_1 + a_0^2},$$

see Lemma 3.4.4. A dominant run contains exactly $\lceil \alpha t \rceil + 1$ points from the set. This means that there are exactly $\lceil \alpha t \rceil$ spaces between the points from the set. The size of the spaces can be $0, 1$, or 2. Define b_0, b_1, and b_2 to be the number of spaces in a dominant run, of size $0, 1$, and 2 respectively. The size of b_0, b_1, and b_2 are given by

$$\begin{aligned} b_0 &= \lceil \alpha t \rceil - \lfloor (1-\alpha)t \rfloor + \lceil a_2 t \rceil, \\ b_1 &= \lfloor (1-\alpha)t \rfloor - 2\lceil a_2 t \rceil, \\ b_2 &= \lceil a_2 t \rceil. \end{aligned}$$

Unfortunately from the way we have defined b_0, b_1, and b_2 they may turn out to be negative. However,

$$\begin{aligned} b_0 &\sim a_0 t, \\ b_1 &\sim a_1 t, \\ b_2 &\sim a_2 t, \end{aligned}$$

and since a_0, a_1, and a_2 are all positive, for sufficiently large n the values of b_0, b_1, and b_2 will also be positive. Note that b_0, b_1, and b_2 have been carefully

chosen so that the number of spaces $b_0 + b_1 + b_2 = \lceil \alpha t \rceil$ as required. The total number of points that lie in the spaces of a dominant run is

$$0 \times b_0 + 1 \times b_1 + 2 \times b_2 = \lfloor (1-\alpha)t \rfloor$$

So the total number of points covered by the spaces in between the points from the set, the points from the set, and the points at the boundary is

$$\lfloor (1-\alpha)t \rfloor + (\lceil \alpha t \rceil + 1) + 2 = t + 3.$$

So a dominant run is a special type of $(t+3)$-run, as intended. We will prove that a typical set contains a $(t+3)$-run by showing that it contains a dominant run. The proof will be similar to that of Lemma 3.2.8.

Let M be the number of ways of arranging the spaces within a dominant run.

$$M = \frac{\lceil \alpha t \rceil !}{b_0! b_1! b_2!}$$

Stirling's formula tells us that for large n

$$\begin{aligned} M &\sim \frac{1}{2\pi} \sqrt{\frac{\lceil \alpha t \rceil}{b_0 b_1 b_2}} \left(\frac{\lceil \alpha t \rceil^{\lceil \alpha t \rceil}}{b_0^{b_0} b_1^{b_1} b_2^{b_2}} \right) \\ &\sim \frac{1}{2\pi t} \sqrt{\frac{\alpha}{a_0 a_1 a_2}} \left(\frac{\lceil \alpha t \rceil^{\lceil \alpha t \rceil}}{b_0^{b_0} b_1^{b_1} b_2^{b_2}} \right). \end{aligned}$$

Note that we can write

$$\begin{aligned} b_0 &= a_0 t + 3\delta_0, \\ b_1 &= a_1 t - 3\delta_1, \\ b_2 &= a_2 t + \delta_2, \\ \lceil \alpha t \rceil &= \alpha t + \delta_3, \end{aligned}$$

where $\delta_0, \delta_1, \delta_2$, and δ_3 are real numbers between 0 and 1 (dependent on n).

Thus

$$\begin{aligned}\lceil \alpha t \rceil^{\lceil \alpha t \rceil} =& (\alpha t + \delta_3)^{\alpha t + \delta_3} \\ =& (\alpha t)^{\alpha t + \delta_3} \left(1 + \frac{\delta_3}{\alpha t}\right)^{\alpha t + \delta_3} \\ \sim& (\alpha t)^{\alpha t + \delta_3} e^{\delta_3}\end{aligned}$$

We can do something similar for $b_0^{b_0}, b_1^{b_1}$, and $b_2^{b_2}$. Consequently

$$M \sim c_0 t^{\delta_4} \left(\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}}\right)^t$$

where

$$c_0 = \frac{e^{-3\delta_0 + 3\delta_1 - \delta_2 + \delta_3} \alpha^{\delta_3}}{2\pi a_0^{3\delta_0} a_1^{-3\delta_1} a_2^{\delta_2}} \sqrt{\frac{\alpha}{a_0 a_1 a_2}} \qquad \text{and} \qquad \delta_4 = -1 - 3\delta_0 + 3\delta_1 - \delta_2 + \delta_3.$$

Note that both c_0 and δ_4 depend on n but are bounded, in fact $\delta_4 \in [-5, 3]$.

Let X be the number of dominant runs in an xn-set. By considering the probability a dominant run occurs at a specific position, and the linearity of expectation, it is easy to show that

$$\mathbf{E}[X] = n \frac{M\binom{n-(t+7)}{xn-(\lceil \alpha t \rceil + 1)}}{\binom{n}{xn}}$$

Lemma 3.2.9 allows us to simplify this expression,

$$\mathbf{E}[X] \sim nMx^{\lceil \alpha t \rceil + 1}(1-x)^{t - \lceil \alpha t \rceil + 6}.$$

Substituting $\lceil \alpha t \rceil = \alpha t + \delta_3$ and our approximation of M gives

$$\mathbf{E}[X] \sim c_1 n t^{\delta_4} \left(\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha}\right)^t$$

where $c_1 = c_0 x^{1+\delta_3}(1-x)^{6-\delta_3}$. As in the proof of Lemma 3.2.8 we require that $\mathbf{E}[X] \to \infty$ as $n \to \infty$. Instead we will show the equivalent statement that

$$\log c_1 + \log n + \delta_4 \log t + t \log \left(\frac{\alpha^\alpha}{a_0^{a_0} a_1^{a_1} a_2^{a_2}} x^\alpha (1-x)^{1-\alpha}\right) \tag{3.2}$$

tends to infinity. Substituting $t = h(x,y)\log n + \delta_5$ (where $\delta_5 \in [0,1]$ depending on n) into (3.2) gives

$$\left(1 + h(x,y)\log\left(\frac{\alpha^\alpha}{a_0^{a_0}a_1^{a_1}a_2^{a_2}}x^\alpha(1-x)^{1-\alpha}\right)\right)\log n + \delta_4 \log t + \log c_1 + \delta_5 \log\left(\frac{\alpha^\alpha}{a_0^{a_0}a_1^{a_1}a_2^{a_2}}x^\alpha(1-x)^{1-\alpha}\right).$$

If we can show that the coefficient of $\log n$ is positive then we will be done, and this is true because

$$-\frac{1}{h(x,y)} < \log\left(\frac{\alpha^\alpha}{a_0^{a_0}a_1^{a_1}a_2^{a_2}}x^\alpha(1-x)^{1-\alpha}\right)$$

holds by (3.1). This proves that $\mathbf{E}[X] \to \infty$.

The remainder of the proof is almost identical to that of Lemma 3.2.8. By following a similar argument we can show that

$$\mathbf{Var}[X] \leq (2t+13)\mathbf{E}[X],$$

and an application of Chebyshev's inequality tells us that

$$\mathbf{P}\left[|X - \mathbf{E}[X]| \geq \mathbf{E}[X]^{\frac{2}{3}}\right] \to 0.$$

Hence the proportion of the xn-sets that do not contain a $(t+3)$-run tends to 0. □

3.5 Open problems

The main motivation for this chapter was Conjecture 3.1.2 made by Bohman, Frieze, Ruszinkó, and Thoma [1]. It still remains open and is the most intriguing question in the area. We have shown that $\alpha(n,k) = (1-o(1))\binom{n}{k}$ when $k/n \geq 1/4 + \epsilon$, which leads us to make the following conjecture.

Conjecture 3.5.1. *Conjecture 3.1.2 holds true with threshold value $\gamma = 1/4$.*

Although the proof given in Section 3.2 is not overly complicated, the end result of $1/4$ is such a simple value that it suggests there is perhaps a simpler and more illuminating argument. This hypothesis appears to be supported by

the fact we have an equally simple bound of $1/(2p+2)$ for the higher powered problem, and the relation $r(x,y)(y-x) = x(1-x)^3 - y(1-y)^3$ in the cross-intersection case. Searching for a more direct argument may be a worthwhile approach to answering whether $1/4$ is the threshold.

Chapter 4

Independent Sets in Trees

4.1 Introduction

An *independent set* of a graph $G = (V(G), E(G))$ is a subset of $V(G)$ in which no two members are neighbours in G. We call an independent set A an *independent r-set* if $|A| = r$.

For a given graph G a natural question to ask is what is the largest intersecting family of independent r-sets. The Erdős–Ko–Rado theorem [12] suggests that taking all independent r-sets that contain a fixed vertex v may be a good candidate. We define $\mathcal{I}(v, G, r)$ to be the number of independent r-sets in G that contain v. An interesting subproblem is to find the vertex v which yields the largest $\mathcal{I}(v, G, r)$.

Hurlbert and Kamat [20] make the following conjecture when G is a tree.

Conjecture 4.1.1. *For any tree T, and $r \in \mathbb{N}$, there exists a leaf x such that for all $v \in V(T)$,*

$$\mathcal{I}(x, T, r) \geq \mathcal{I}(v, T, r).$$

They show the conjecture holds for $r \leq 4$, although their proof is a bit unclear. We will give a slightly modified version of their proof and then show that the conjecture is false for $r > 4$.

Theorem 4.1.2. *For any tree T, and $r \leq 4$, there exists a leaf x such that for all $v \in V(T)$, $\mathcal{I}(x, T, r) \geq \mathcal{I}(v, T, r)$.*

To disprove the conjecture for $r > 4$ we consider the following family of trees T_n, see Figure 4.1. Formally T_n is a tree with $4n + 3$ vertices, $V(T_n) =$

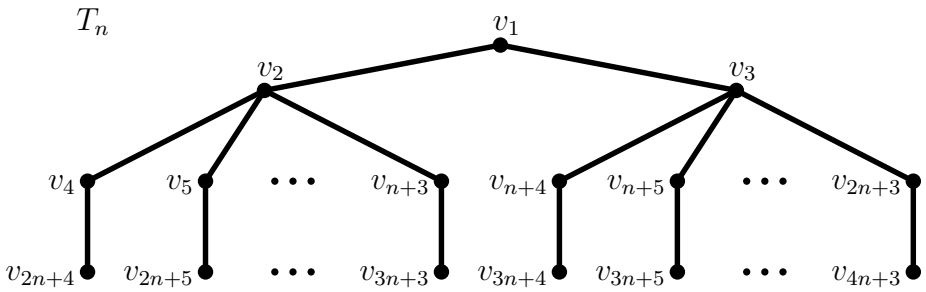

Figure 4.1: The tree T_n, with vertices labelled $v_1, \ldots, v_{4n+3}$.

$\{v_1, \ldots, v_{4n+3}\}$, and

$$E(T_n) = \{v_1v_2, v_1v_3\} \cup \{v_2v_{i+3} : 1 \leq i \leq n\} \cup \{v_3v_{i+n+3} : 1 \leq i \leq n\} \cup \{v_{i+3}v_{i+2n+3} : 1 \leq i \leq 2n\}.$$

Theorem 4.1.3. *Given $r > 4$ and $n \geq r^2$, for all leaves x of T_n,*

$$\mathcal{I}(x, T_n, r) < \mathcal{I}(v_1, T_n, r).$$

4.2 Proof of Theorem 4.1.2

The conjecture trivially holds for $r \leq 2$ as for all graphs G and vertices $v \in V(G)$, we have $\mathcal{I}(v, G, 1) = 1$, and $\mathcal{I}(v, G, 2) = |V(G)| - 1 - d(v)$, where $d(v)$ is the degree of the vertex v. If $|V(G)| < r$ then it is again trivial to prove. Hence to prove Theorem 4.1.2 we need only look at independent sets of size $r = 3, 4$, on trees with at least r vertices.

Let T be a tree and $\mathcal{C}$ be a family of independent r-sets that contain an internal vertex c. We will prove Theorem 4.1.2, by showing that there exists a leaf l for which we can construct an injective function f which maps the sets in $\mathcal{C}$ to independent r-sets on T that contain l.

It is easy to show that in any tree there exists a unique simple path between any two vertices. Let us define the *distance* between two vertices in a tree as the number of edges in the unique simple path that joins them. If there exists a leaf which is at a distance of at most two from vertex c then we choose it to

be our leaf l and take f to be

$$f(C) = \begin{cases} C, & \text{if } l \in C, \\ C \setminus \{c\} \cup \{l\}, & \text{if } l \notin C, \end{cases}$$

for $C \in \mathcal{C}$. It is simple to check f is injective and maps the sets in $\mathcal{C}$ to independent r-sets containing l. Thus $\mathcal{I}(c, T, r) \leq \mathcal{I}(l, T, r)$ in this case.

If all leaves are a distance of at least three away from c, we choose any leaf to be l. To define an appropriate function f we need to choose any two distinct neighbours of c, which we will call v_1, v_2 (this is always possible as c is an internal vertex, hence $d(c) \geq 2$). Also let us define vertex u to be the unique neighbour of l. Note that because the distance between c and l is at least three, the vertices c, v_1, v_2, u, and l are all distinct.

If $r = 3$ we define f as

$$f(C) = \begin{cases} C, & \text{if } l \in C, \\ C \setminus \{c\} \cup \{l\}, & \text{if } l \notin C, u \notin C, \\ \{l, v_1, x\}, & \text{if } l \notin C, u \in C, xv_1 \notin E(T) \text{ where } x \in C \setminus \{c, u\}, \\ \{l, v_2, x\}, & \text{if } l \notin C, u \in C, xv_1 \in E(T) \text{ where } x \in C \setminus \{c, u\}. \end{cases}$$

It is easy to check that for all $C \in \mathcal{C}$, f maps C to a 3-set containing l. It is also not hard to see that $f(C)$ is an independent set for the cases $l \in C$ and $l, u \notin C$. If $l \notin C$ and $u \in C$ then $C = \{c, u, x\}$ for some vertex $x \neq c, v_1, v_2, u, l$. If $xv_1 \notin E(T)$ then it is simple to show that $\{l, v_1, x\}$ is an independent set. If $xv_1 \in E(T)$ then $xv_2 \notin E(T)$ must hold, otherwise x, v_1, c, v_2 would form a 4-cycle. Hence similarly $\{l, v_2, x\}$ will be an independent set. To prove f is injective we consider the function $g : \text{im}(f) \to \mathcal{C}$ defined as

$$g(A) = \begin{cases} A, & \text{if } c \in A, \\ A \setminus \{l\} \cup \{c\}, & \text{if } c \notin A, v_1, v_2 \notin A, \\ A \setminus \{l, v_1\} \cup \{c, u\}, & \text{if } c \notin A, v_1 \in A, \\ A \setminus \{l, v_2\} \cup \{c, u\}, & \text{if } c \notin A, v_1 \notin A, v_2 \in A. \end{cases}$$

It is straightforward to check that $g(f(C)) = C$ for all $C \in \mathcal{C}$ which implies f is injective, and so completes the proof for $r = 3$.

If $r = 4$ then we define f for $C \in \mathcal{C}$ as follows:

- If $l \in C$ then $f(C) = C$.
- If $l, u \notin C$ then $f(C) = C \setminus \{c\} \cup \{l\}$.
- If $l \notin C$, $u \in C$, then $C = \{c, u, w_1, w_2\}$ for some vertices $w_1, w_2 \neq c, v_1, v_2, u, l$. We split this case into three subcases.
  - If $v_2w_1, v_2w_2 \notin E(T)$ then $f(C) = \{l, v_2, w_1, w_2\}$.
  - If at least one of v_2w_1, v_2w_2 is in $E(T)$, and $v_1w_1, v_1w_2 \notin E(T)$ then $f(C) = \{l, v_1, w_1, w_2\}$.
  - If at least one of v_2w_1, v_2w_2 is in $E(T)$, and at least one of v_1w_1, v_1w_2 is also in $E(T)$, then to avoid the 4-cycles $cv_1w_1v_2c$ and $cv_1w_2v_2c$, we must have that either $v_1w_1, v_2w_2 \in E(T), v_1w_2, v_2w_1 \notin E(T)$ or $v_1w_2, v_2w_1 \in E(T), v_1w_1, v_2w_2 \notin E(T)$. Since T is a tree with all its leaves at a distance of at least three from c, both w_1, w_2 are not leaves. Hence we can find neighbours x_1, x_2 of w_1, w_2 respectively such that $x_1, x_2 \neq v_1, v_2$. In fact x_1, x_2 are distinct from $c, v_1, v_2, u, l, w_1, w_2$, and importantly $v_1x_1, v_1x_2 \notin E(T)$. Consequently we define $f(C) = \{l, v_1, x_1, x_2\}$ which must be an independent set.

To prove that f is injective we again construct a function g such that $g(f(C)) = C$ for all $C \in \mathcal{C}$. For $A \in \mathrm{im}(f)$, g is defined as:

- If $c \in A$ then $g(A) = A$.
- If $c, v_1, v_2 \notin A$ then $g(A) = A \setminus \{l\} \cup \{c\}$.
- If $c \notin A$, $v_2 \in A$ then $g(A) = A \setminus \{l, v_2\} \cup \{c, u\}$.
- If $c, v_2 \notin A$, $v_1 \in A$ then A is of the form $\{l, v_1, a_1, a_2\}$ with $a_1, a_2 \neq v_2$.
  - If a_1v_2 or a_2v_2 is in $E(T)$ then $g(A) = \{c, u, a_1, a_2\}$.
  - If $a_1v_2, a_2v_2 \notin E(T)$, then $g(A) = \{c, u, b_1, b_2\}$, where b_1, b_2 are the neighbours of a_1, a_2 respectively on the unique simple paths joining a_1, a_2 to c.

This completes the proof for $r = 4$, and hence the proof of Theorem 4.1.2.

4.3 Proof of Theorem 4.1.3

Due to the symmetric nature of T_n, see Figure 4.1, it is easy to see that $\mathcal{I}(x, T_n, r)$ is the same for all choices of leaves x. Hence to prove Theorem 4.1.3 it is sufficient to show $\mathcal{I}(v_{4n+3}, T_n, r) < \mathcal{I}(v_1, T_n, r)$ for $r > 4$ and $n \geq r^2$.

First we count the number of independent r-sets containing v_1. If v_1 is in our set then v_2, v_3 cannot be in, as our set is independent. Consequently we have to choose the $r-1$ remaining elements of our set from the $2n$ disjoint edges $\{v_{i+3}v_{2n+3} : 1 \leq i \leq 2n\}$. Hence

$$\mathcal{I}(v_1, T_n, r) = \binom{2n}{r-1} 2^{r-1}.$$

Next we count the number of independent r-sets containing v_{4n+3} by splitting it into five cases.

(i) *v_1 is part of the set.*
Because the set must be independent v_2, v_3, v_{2n+3} cannot be in the set. The remaining $r-2$ vertices must be taken from the $2n-1$ disjoint edges $\{v_{i+3}v_{i+2n+3} : 1 \leq i \leq 2n-1\}$. Hence there are $\binom{2n-1}{r-2}2^{r-2}$ such independent r-sets.

(ii) *v_1 is not part of the set, and v_2, v_3 are.*
The remaining $r-3$ vertices must be taken from the set of $2n-1$ vertices $\{v_{2n+4}, \ldots, v_{4n+2}\}$. There are $\binom{2n-1}{r-3}$ such sets.

(iii) *v_1, v_3 are not part of the set, and v_2 is.*
The remaining $r-2$ vertices, are taken from the disjoint set of $n-1$ edges $\{v_{i+n+3}v_{i+3n+3} : 1 \leq i \leq n-1\}$ and the set of n vertices $\{v_{2n+4}, \ldots, v_{3n+3}\}$. The number of sets in which k of the vertices come from the $n-1$ edges is $\binom{n-1}{k}\binom{n}{r-2-k}2^k$. Hence overall there are

$$\sum_{k=0}^{r-2} \binom{n-1}{k}\binom{n}{r-2-k}2^k$$

independent r-sets containing v_2, v_{4n+3} but not v_1, v_3.

(iv) *v_1, v_2 are not part of the set, and v_3 is.*
The remaining $r-2$ vertices, are taken from the disjoint set of n edges

$\{v_{i+3}v_{i+2n+3} : 1 \leq i \leq n\}$ and the set of $n-1$ vertices $\{v_{3n+4}, \ldots, v_{4n+2}\}$. The number of such independent sets is

$$\sum_{k=0}^{r-2} \binom{n-1}{k} \binom{n}{r-2-k} 2^{r-2-k}.$$

(v) *v_1, v_2, v_3 are not part of the set.*
The remaining $r-1$ vertices, are taken from the disjoint set of $2n-1$ edges $\{v_{i+3}v_{i+2n+3} : 1 \leq i \leq 2n-1\}$. There are $\binom{2n-1}{r-1}2^{r-1}$ such sets.

Putting all this together gives

$$\mathcal{I}(v_{4n+3}, T_n, r) = \binom{2n-1}{r-3} + \binom{2n-1}{r-2}2^{r-2} + \binom{2n-1}{r-1}2^{r-1} + \sum_{k=0}^{r-2} \binom{n-1}{k} \binom{n}{r-2-k} \left(2^k + 2^{r-2-k}\right).$$

Applying the fact that $\binom{2n}{r-1} = \binom{2n-1}{r-1} + \binom{2n-1}{r-2}$ to $\mathcal{I}(v_1, T_n, r)$ we see that $\mathcal{I}(v_1, T_n, r) - \mathcal{I}(v_{4n+3}, T_n, r)$ equals

$$\binom{2n-1}{r-2}2^{r-2} - \binom{2n-1}{r-3} - \sum_{k=0}^{r-2} \binom{n-1}{k} \binom{n}{r-2-k} \left(2^k + 2^{r-2-k}\right).$$

Using the identity

$$\binom{2n-1}{r-2} = \sum_{k=0}^{r-2} \binom{n-1}{k} \binom{n}{r-2-k}$$

we get $\mathcal{I}(v_1, T_n, r) - \mathcal{I}(v_{4n+3}, T_n, r)$ is

$$-\binom{2n-1}{r-3} + \sum_{k=0}^{r-2} \binom{n-1}{k} \binom{n}{r-2-k} \left(2^{r-2} - 2^k - 2^{r-2-k}\right).$$

Since

$$\begin{aligned}\binom{2n-1}{r-3} &= \sum_{k=1}^{r-2}\binom{n-1}{k-1}\binom{n}{r-3-(k-1)} \\ &= \sum_{k=0}^{r-2}\frac{k}{n-k}\binom{n-1}{k}\binom{n}{r-2-k},\end{aligned}$$

we have $\mathcal{I}(v_1, T_n, r) - \mathcal{I}(v_{4n+3}, T_n, r)$ is

$$\sum_{k=0}^{r-2}\binom{n-1}{k}\binom{n}{r-2-k}\left(2^{r-2} - 2^k - 2^{r-2-k} - \frac{k}{n-k}\right). \tag{4.1}$$

If $n \geq r^2$ and $2 \leq k \leq r-4$ then

$$\begin{aligned}2^{r-2} - 2^k - 2^{r-2-k} - \frac{k}{n-k} &> 2^{r-2} - 2^k - 2^{r-2-k} - 1 \\ &= \left(2^k - 1\right)\left(2^{r-2-k} - 1\right) - 2 \\ &> 0.\end{aligned}$$

Therefore to prove $\mathcal{I}(v_1, T_n, r) > \mathcal{I}(v_{4n+3}, T_n, r)$ it will be enough to show that the terms corresponding to $k = 0, 1, r-3, r-2$ in (4.1) have a sum that is strictly greater than 0. These four terms sum to

$$\begin{aligned}&-\binom{n}{r-2} + (n-1)\binom{n}{r-3}\left(2^{r-3} - 2 - \frac{1}{n-1}\right) + \\ &\qquad \binom{n-1}{r-3}n\left(2^{r-3} - 2 - \frac{r-3}{n-r+3}\right) + \binom{n-1}{r-2}\left(-1 - \frac{r-2}{n-r+2}\right)\end{aligned}$$

which simplifies to

$$\binom{n}{r-3}\left((2n-r+2)\left(2^{r-3}-1\right) - \frac{2nr-2n-2r+6}{r-2}\right). \tag{4.2}$$

Since $n \geq r^2$ and $r > 4$, it is easy to check that (4.2) is larger than

$$(2n-r+2)\binom{n}{r-3}\left(2^{r-3} - 1 - \frac{r}{r-2}\right).$$

Note that for $r > 4$,

$$2^{r-3} - 1 - \frac{r}{r-2} > 2^{r-3} - 4 \geq 0,$$

hence (4.2) is greater than 0 and therefore so is (4.1). This implies that $\mathcal{I}(v_1, T_n, r) > \mathcal{I}(v_{4n+3}, T_n, r)$ for $r > 4$ and $n \geq r^2$ which completes the proof of Theorem 4.1.3.

Appendix A

GraphFinder

The following is a C++ implementation of the algorithm that produces a list of possible extremal vertex minimal tripartite graphs, as described in Chapter 1. The graphs it outputs are given in Figure 1.11. The code can also be found on the accompanying CD-ROM in the `GraphFinder` directory.

```
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;

//If bSaveToTxtFile is true it will save the output of the program to the file "Output.txt".
bool bSaveToTxtFile = false;

//CGraph stores data about a tripartite graph (with at most 3 vertices per vertex class, see
//Lemma 1.4.11).
//
//Member variables
//----------------
//
//M[][]       is the adjacency matrix of the graph.
//            An entry of 1 means the edge is present, 0 indicates it is missing.
//
//hasVertex[] tells us which vertices are part of the graph.
//            This allows us to represent smaller class sizes.
//            An entry of 1 means the vertex is part of the graph, 0 indicates it is missing.
//
//idEdge      is a 27 bit integer representation of M[][].
//            We use EdgeMatrix[][] to convert between M[][] and idEdge.
//
//idVertex    is a 9 bit integer representation of hasVertex[].
//            The least significant bit should correspond to hasVertex[0].
//
//classSize[] keeps track of the size of the vertex classes (for speed).
//            E.g. classSize[0] = hasVertex[0]+hasVertex[1]+hasVertex[2].
//
```

```
//Member functions
//----------------
//
//void AssignMatrixFromID(long id)   id should be a 27 bit integer. The function sets idEdge
//                                   to id, then uses EdgeMatrix[][] to construct the
//                                   corresponding adjacency matrix M[][].
//
//void AssignVerticesFromID(long id) id should be a 9 bit integer. The function sets idVertex
//                                   to id and then constructs the corresponding hasVertex[]
//                                   array. It also fills in classSize[].
//
//Notes
//-----
//
//The vertex classes are:
//Class 0 - vertices 0, 1, 2,
//Class 1 - vertices 3, 4, 5,
//Class 2 - vertices 6, 7, 8.
//
//Observe that vertex v lies in vertex class v/3 (where '/' is the integer division
//operator). This provides an efficient way of testing if two vertices lie in the same vertex
//class, which we will need to do often.
//
//If hasVertex[v] is zero it means vertex v should not be considered part of the tripartite
//graph. This is to allow CGraph to be flexible enough to represent tripartite graphs of
//smaller order. The adjacency matrix M[][] is of fixed size but since vertex v is not part
//of the graph, the entries in row v and column v are of no importance. Some functions assume
//(for speed purposes) that if the vertex is missing then its corresponding column and row
//entries should all be zero.
//
//To speed things up, functions that take CGraph objects as input do little (if any) checking
//that the objects are well-formed, e.g. entries of M[][] are all 0 or 1, M[][] is symmetric,
//that idEdge correctly represents M[][], or vertices that are not part of the graph have no
//neighbours. It is the responsibility of the process calling such functions to ensure that
//the CGraph objects are well-formed.
//
//We keep a record of the variables idEdge and idVertex as they provide a natural ordering of
//CGraph objects. This helps make the process of testing if two tripartite graphs are
//strongly-isomorphic to each other more efficient.
class CGraph
{
public:
    //Construct M[][] using EdgeMatrix[][] and id (a 27 bit integer), idEdge is set to id.
    void AssignMatrixFromID(long id);

    //Sets idVertex to id (a 9 bit integer), then constructs hasVertex[] and classSize[].
    void AssignVerticesFromID(long id);

    //The adjacency matrix of the graph.
    //0 = Edge missing.
    //1 = Edge present.
    long M[9][9];
```

```
    //Used to indicate which vertices are part of the graph.
    //0 = Vertex has been removed.
    //1 = Vertex is part of the graph.
    long hasVertex[9];

    //A 27 bit integer representing M[][].
    long idEdge;

    //A 9 bit integer representing hasVertex[].
    long idVertex;

    //The size of the vertex classes.
    long classSize[3];
};

//EdgeMatrix[][] is used to bijectively map the 27 possible edges of the tripartite graphs to
//27 bit integers. The entries indicate which bit (0 to 26) in CGraph::idEdge is the
//corresponding entry in the adjacency matrix CGraph::M[][]. Entries of -1 indicate that such
//an edge lies within a vertex class, so should always be zero in CGraph::M[][], and
//consequently does not get mapped to any of the 27 bits in CGraph::idEdge.
long EdgeMatrix[9][9] = {
    {-1, -1, -1, 18, 19, 20,  0,  1,  2},
    {-1, -1, -1, 21, 22, 23,  3,  4,  5},
    {-1, -1, -1, 24, 25, 26,  6,  7,  8},
    {18, 21, 24, -1, -1, -1,  9, 10, 11},
    {19, 22, 25, -1, -1, -1, 12, 13, 14},
    {20, 23, 26, -1, -1, -1, 15, 16, 17},
    { 0,  3,  6,  9, 12, 15, -1, -1, -1},
    { 1,  4,  7, 10, 13, 16, -1, -1, -1},
    { 2,  5,  8, 11, 14, 17, -1, -1, -1}};

//The 1296 permutations of the the 9 vertices that leaves the tripartite graph
//strongly-isomorphic to its original form.
long StrongIsoPermute[1296][9];

//Store the possible extremal vertex minimal graphs in Store[][0]. Also keep a copy of all
//graphs strongly-isomorphic to Store[i][0] in Store[i][1 to 1295]. Use Found to track how
//much of Store[][] has been filled in.
CGraph Store[200][1296];
long   Found;

//Graphs that are strongly-isomorphic to F7 and F9 (see Figure 1.8).
CGraph F7, F9;

//Displays the adjacency matrix of graph g. To help distinguish the vertex classes it uses
//the symbol '.' to represent edges that lie within a class.
void DisplayGraph(CGraph& g, ostream& os = cout);

//Display all the possible extremal vertex minimal graphs in Store[][0].
//Ignores graphs with idEdge = -1.
void DisplayStore(ostream& os = cout);

//Declared in CGraph. Construct M[][] using EdgeMatrix[][] and id (a 27 bit integer), idEdge
```

```
//is set to id.
//void CGraph::AssignMatrixFromID(long id);

//Declared in CGraph. Sets idVertex to id (a 9 bit integer), then constructs hasVertex[]
//and classSize[].
//void CGraph::AssignVerticesFromID(long id);

//Initializes the StrongIsoPermute[][] array and constructs the graphs F7 and F9.
void SetUp();

//Returns true if it can show the graph is not extremal or not vertex minimal (using Lemmas
//1.4.10, 1.4.13, and 1.4.14).
bool bBadClassSize(CGraph& g);

//Returns true if it can show the graph is not extremal or not vertex minimal (using
//Corollary 1.4.5).
bool bCanMoveEdgeWeights(CGraph& g);

//Returns true if g contains no triangles. We are not interested in such a graph by Theorem
//1.2.1.
bool bHasNoTriangles(CGraph& g);

//Returns true if it can show the graph is not extremal or not vertex minimal (using Lemma
//1.4.15). This function assumes Store[][] is filled with a complete list of possible
//extremal vertex minimal tripartite graphs.
bool bCanReplaceBy8(CGraph& g);

//Creates a graph strongly-isomorphic to gIn by using the vertex mapping given in
//StrongIsoPermute[p][]. The result is stored in gOut (which should be a different object
//to gIn).
void StrongIsoGraph(CGraph& gIn, long p, CGraph& gOut);

//Displays the adjacency matrix of graph g. To help distinguish the vertex classes it uses
//the symbol '.' to represent edges that lie within a class.
void DisplayGraph(CGraph& g, ostream& os)
{
    long i,j;

    for(i=0;i<9;i++)
    {
        if(g.hasVertex[i]==0)
            continue; //Ignore vertices that are not part of the graph.

        for(j=0;j<9;j++)
        {
            if(g.hasVertex[j]==0)
                continue; //Ignore vertices that are not part of the graph.

            //Display whether ij is an edge.
            if(i/3==j/3)
                os << " ."; //The edge lies within a vertex class.
            else
            {
```

```
                if(g.M[i][j]==0)
                    os << " 0"; //The edge is missing.
                else
                    os << " 1"; //The edge is present.
            }
        }

        os << endl;
    }

    os << endl;

    return;
}

//Display all the possible extremal vertex minimal graphs in Store[][0].
//Ignores graphs with idEdge = -1.
void DisplayStore(ostream& os)
{
    long i,order;

    os << "Adjacency matrices of possible extremal vertex minimal tripartite" << endl;
    os << "graphs (edges which would lie within a vertex class are indicated" << endl;
    os << "by the entry \".\"):" << endl;
    os << endl;

    for(order=0;order<=9;order++) //Display the smallest order graphs first.
    for(i=0;i<Found;i++)
    {
        if(Store[i][0].idEdge==-1)
            continue; //The graph was removed.

        if(Store[i][0].classSize[0]+Store[i][0].classSize[1]+Store[i][0].classSize[2]!=order)
            continue; //The graph does not have the right number of vertices.

        DisplayGraph(Store[i][0],os);

        os << endl;
    }

    os << endl;
}

//Construct M[][] using EdgeMatrix[][] and id (a 27 bit integer), idEdge is set to id.
void CGraph::AssignMatrixFromID(long id)
{
    long i,j;

    idEdge = id;

    //Fill in the adjacency matrix.
    for(i=0;i<9;i++)
    for(j=0;j<9;j++)
```

```
        if(i/3==j/3)
            M[i][j] = 0; //Edge lies within a class.
        else
            M[i][j] = (id>>EdgeMatrix[i][j])&1; //Assign the the correct bit of id.

    return;
}

//Sets idVertex to id (a 9 bit integer), then constructs hasVertex[] and classSize[].
void CGraph::AssignVerticesFromID(long id)
{
    long i;

    idVertex = id;

    for(i=0;i<9;i++)
        hasVertex[i] = (id>>i)&1; //Assign the "i"th bit of id.

    for(i=0;i<3;i++)
        classSize[i] = hasVertex[3*i]+hasVertex[3*i+1]+hasVertex[3*i+2];

    return;
}

//Initializes the StrongIsoPermute[][] array and constructs the graphs F7 and F9.
void SetUp()
{
    long i,n;
    long idEdge,idVertex;
    long P[4];

    //Fill in the StrongIsoPermute[][] array.
    {
        //Permute[][] stores the 6 perumtations of {0,1,2}.
        long Permute[6][3] = {{0,1,2}, {0,2,1}, {1,0,2}, {1,2,0}, {2,0,1}, {2,1,0}};

        //Permute class 0 by Permute[P[0]][].
        //Permute class 1 by Permute[P[1]][].
        //Permute class 2 by Permute[P[2]][].
        //Permute classes by Permute[P[3]][].
        n = 0;
        for(P[0]=0;P[0]<6;P[0]++)
        for(P[1]=0;P[1]<6;P[1]++)
        for(P[2]=0;P[2]<6;P[2]++)
        for(P[3]=0;P[3]<6;P[3]++)
        {
            //Calculate the new label for vertex i.
            //i is the "i%3" vertex in class "i/3", i.e. i = 3*(i/3)+(i%3).
            //Class "i/3" gets mapped to class Permute[P[3]][i/3].
            //The vertices in class "i/3" are permuted by Permute[P[i/3]][].
            for(i=0;i<9;i++)
                StrongIsoPermute[n][i] = 3*Permute[P[3]][i/3]+Permute[P[i/3]][i%3];
```

```
            n++;
        }
    }

    //Construct the graph F7 (see Figure 1.8).
    {
        //Vertices 7,6,5,4,3,1,0 are part of F7.
        //Vertices 8 and 2 are not part of F7.
        idVertex = 251; //Binary 011111011.

        //An array of the 9 edges in F7.
        long Edge[9][2] = {{0,4}, {0,5}, {0,7}, {1,3}, {1,4}, {1,6}, {3,7}, {4,6}, {5,6}};

        //Construct idEdge for F7 using Edge[][] and EdgeMatrix[][].
        idEdge = 0;
        for(i=0;i<9;i++)
            idEdge |= 1<<EdgeMatrix[Edge[i][0]][Edge[i][1]];

        F7.AssignVerticesFromID(idVertex);
        F7.AssignMatrixFromID(idEdge);
    }

    //Construct the graph F9 (see Figure 1.8).
    {
        //Vertices 0 to 8 are part of F9.
        idVertex = 511; //Binary 111111111.

        //An array of the 15 edges in F9.
        long Edge[15][2] = {
            {0,4}, {0,5}, {0,7}, {1,3}, {1,5},
            {1,6}, {1,8}, {2,4}, {2,6}, {2,7},
            {3,7}, {3,8}, {4,6}, {4,8}, {5,7}};

        //Construct idEdge for F9 using Edge[][] and EdgeMatrix[][].
        idEdge = 0;
        for(i=0;i<15;i++)
            idEdge |= 1<<EdgeMatrix[Edge[i][0]][Edge[i][1]];

        F9.AssignVerticesFromID(idVertex);
        F9.AssignMatrixFromID(idEdge);
    }

    return;
}

//Returns true if it can show the graph is not extremal or not vertex minimal (using Lemmas
//1.4.10, 1.4.13, and 1.4.14).
bool bBadClassSize(CGraph& g)
{
    //Our tests will involve testing the neighbourhoods of vertex 0 and 1.
    if(g.hasVertex[0]==0 || g.hasVertex[1]==0)
        return false;
```

```
    //Check whether vertices 0 and 1 have the same neighbours in class 1.
    if(g.M[0][3]==g.M[1][3] && g.M[0][4]==g.M[1][4] && g.M[0][5]==g.M[1][5])
    {
        if(g.classSize[2]==3)
            return true; //g is not extremal or not vertex minimal by Lemma 1.4.13.

        if(g.classSize[0]==2)
            return true; //g is not extremal by Lemma 1.4.10.

        if(g.M[0][3]==g.M[2][3] && g.M[0][4]==g.M[2][4] && g.M[0][5]==g.M[2][5])
        {
            //Vertices 0, 1, and 2 have the same neighbours in class 1, hence by Lemma
            //1.4.10 g is not extremal.
            return true;
        }

        if(g.M[0][6]==g.M[1][6] && g.M[0][7]==g.M[1][7] && g.M[0][8]==g.M[1][8])
        {
            //Vertices 0 and 1 have the same neighbourhood, hence by Lemma 1.4.14 g is not
            //vertex minimal.
            return true;
        }
    }

    return false;
}

//Returns true if it can show the graph is not extremal or not vertex minimal (using
//Corollary 1.4.5).
bool bCanMoveEdgeWeights(CGraph& g)
{
    long i,j;
    bool bC0,bC1;
    bool bIsEqual, bIsSubset;

    //We will try to apply Corollary 1.4.5 to
    //(i)  the edge 0,3 and the missing edge 0,4
    //(ii) the edge 0,3 and the missing edge 1,4

    if(g.hasVertex[0]==0 || g.hasVertex[1]==0
    || g.hasVertex[3]==0 || g.hasVertex[4]==0)
        return false; //A vertex is missing so cannot apply the test.

    if(g.M[0][3]==0)
        return false; //Edge 0,3 is missing.

    for(j=0;j<=1;j++)
        if(g.M[j][4]==0)
        {
            //g contains edge 0,3 and j,4 is missing (where j = 0 or 1).

            //Check C_{j,4} is a proper subset of C_{0,3}
            bIsEqual  = true;
```

```
            bIsSubset = true;
            for(i=6;i<9;i++)
            {
                bC0 = (g.M[i][j]==1 && g.M[i][4]==1);
                bC1 = (g.M[i][0]==1 && g.M[i][3]==1);

                if(bC0==true && bC1==false)
                    bIsSubset = false;

                if(bC0!=bC1)
                    bIsEqual = false;
            }

            if(bIsSubset==true && bIsEqual==false)
                return true; //g is not extremal or not vertex minimal by Corollary 1.4.5.
        }

    return false;
}

//Returns true if g contains no triangles. We are not interested in such a graph by Theorem
//1.2.1.
bool bHasNoTriangles(CGraph& g)
{
    long i,j,k; //Vertices in classes 0,1,2 respectively.

    //Note that if a vertex is missing it has no neighbours so cannot be part of a triangle.
    //This saves us having to use the hasVertex[] array.
    for(i=0;i<3;i++)
    for(j=3;j<6;j++)
    for(k=6;k<9;k++)
        if(g.M[i][j]==1 && g.M[i][k]==1 && g.M[j][k]==1)
            return false; //ijk forms a triangle.

    return true; //Could not find a triangle.
}

//Returns true if it can show the graph is not extremal or not vertex minimal (using Lemma
//1.4.15). This function assumes Store[][] is filled with a complete list of possible
//extremal vertex minimal tripartite graphs.
bool bCanReplaceBy8(CGraph& g)
{
    long i,j,k,n;
    long a0,b0,a1,b1;
    long G1[10][10];
    long G2[10][10];
    bool bC0,bC1;
    bool bSame;

    //ReplaceEdgeID[] Holds the CGraph::idEdge values for the eight graphs described in Lemma
    //1.4.15.
    long ReplaceEdgeID[8];
```

```
//We will try to apply Lemma 1.4.15 to
//(i)   the edge 6,0 and the missing edge 6,1,
//(ii)  the edge 6,0 and the missing edge 7,0,
//(iii) the edge 6,0 and the missing edge 7,1.
//Where class A in Lemma 1.4.15 is vertex class 2.
long cases[3][2] = {{6,1},{7,0},{7,1}};

//a1 = 6,           b1 = 0,
//a0 = cases[n][0], b0 = cases[n][1].

if(g.classSize[2]!=3)
    return false; //The class in which vertices a0, a1 lie in must be of size 3.

if(g.hasVertex[0]==0 || g.hasVertex[6]==0
|| g.hasVertex[1]==0 || g.hasVertex[7]==0)
    return false; //A vertex is missing so cannot apply the test.

if(g.M[0][6]==0)
    return false; //Edge 6,0 is missing.

a1 = 6;
b1 = 0;
for(n=0;n<3;n++)
{
    //Find suitable values for a0,b0, a1,b1.
    {
        a0 = cases[n][0];
        b0 = cases[n][1];

        if(g.M[a0][b0]==1)
            continue;

        //Check C_{a0,b0} = C_{a1,b1}
        bSame = true;
        for(i=3;i<6;i++)
        {
            bC0 = (g.M[i][a0]==1 && g.M[i][b0]==1);
            bC1 = (g.M[i][a1]==1 && g.M[i][b1]==1);

            if(bC0!=bC1)
                bSame = false;
        }

        if(bSame==false)
            continue;
    }

    //Create graphs G1, and G2.
    {
        //Copy g into G1 and G2.
        for(i=0;i<9;i++)
        for(j=0;j<9;j++)
        {
```

```
        G1[i][j] = g.M[i][j];
        G2[i][j] = g.M[i][j];
    }

    //Remove edge a1,b1 from G1.
    G1[a1][b1] = 0;
    G1[b1][a1] = 0;

    //Add edge a0,b0 to G2.
    G2[a0][b0] = 1;
    G2[b0][a0] = 1;

    //Fill in the neighbours for the vertex a2 in column/row 9.
    for(i=0;i<9;i++)
    {
        //Copy vertex a0.
        G1[9][i] = G1[a0][i];
        G1[i][9] = G1[i][a0];

        //Copy vertex a1.
        G2[9][i] = G2[a1][i];
        G2[i][9] = G2[i][a1];
    }

    //Fill in the entry a2,a2.
    G1[9][9] = 0;
    G2[9][9] = 0;

    //Add edge a2,b0 to G1.
    G1[9][b0] = 1;
    G1[b0][9] = 1;

    //Remove edge a2,b1 from G2.
    G2[9][b1] = 0;
    G2[b1][9] = 0;
}

//Fill the array ReplaceEdgeID[] by calculating the idEdge of G1, G2 once a vertex
//has been removed from class 2.
{
    //map[][] is used to remove vertex k+6 in class 2 by mapping vertex i to
    //vertex map[k][i] in G1, G2;
    long map[4][9] = {
        {0,1,2,3,4,5,  7,8,9},
        {0,1,2,3,4,5,6,  8,9},
        {0,1,2,3,4,5,6,7,  9},
        {0,1,2,3,4,5,6,7,8  }};

    for(k=0;k<4;k++)
    {
        ReplaceEdgeID[k] = 0;
        for(i=0;i<9;i++)
        for(j=0;j<9;j++)
```

```
                if(i/3!=j/3)
                    ReplaceEdgeID[k] |= G1[map[k][i]][map[k][j]]<<EdgeMatrix[i][j];
            }

            for(k=0;k<4;k++)
            {
                ReplaceEdgeID[k+4] = 0;
                for(i=0;i<9;i++)
                for(j=0;j<9;j++)
                    if(i/3!=j/3)
                        ReplaceEdgeID[k+4] |= G2[map[k][i]][map[k][j]]<<EdgeMatrix[i][j];
            }
        }

        //Check if any of the possible extremal vertex minimal graphs have the same idEdge as
        //ReplaceEdgeID[] by searching through Store[][].
        {
            bSame = false;
            for(i=0;i<8;i++)
            for(j=0;j<Found;j++)
            {
                if(bSame==true)
                {
                    //A graph in Store[][] has an idEdge which matches a member of
                    //ReplaceEdgeID[].
                    break;
                }

                if(Store[j][0].idEdge==-1)
                {
                    //The graph Store[j][0] is not an extremal vertex minimal graph, and
                    //hence neither are any of the graphs that are strongly-isomorphic to it.
                    continue;
                }

                for(k=0;k<1296;k++)
                    if(Store[j][k].idEdge==ReplaceEdgeID[i])
                    {
                        //One of the eight subgraphs of G1, G2 is possibly extremal and
                        //vertex minimal. So g maybe extremal and vertex minimal.
                        bSame = true;
                        break;
                    }
            }

            if(bSame==false)
            {
                //The eight subgraphs of G1, G2 are not extremal or not vertex minimal, hence
                //by Lemma 1.4.15 the graph g is not extremal or not vertex minimal.
                return true;
            }
        }
    }
```

```
    return false;
}

//Creates a graph strongly-isomorphic to gIn by using the vertex mapping given in
//StrongIsoPermute[p][]. The result is stored in gOut (which should be a different object
//to gIn).
void StrongIsoGraph(CGraph& gIn, long p, CGraph& gOut)
{
    long i,j;

    //Map vertex i in gIn to vertex StrongIsoPermute[p][i] in gOut.

    //Fill in hasVertex[].
    for(i=0;i<9;i++)
        gOut.hasVertex[StrongIsoPermute[p][i]] = gIn.hasVertex[i];

    //Fill in M[][].
    for(i=0;i<9;i++)
    for(j=0;j<9;j++)
        gOut.M[StrongIsoPermute[p][i]][StrongIsoPermute[p][j]] = gIn.M[i][j];

    //Create the idVertex.
    gOut.idVertex = 0;
    for(i=0;i<9;i++)
        gOut.idVertex |= gOut.hasVertex[i]<<i;

    //Create the idEdge.
    gOut.idEdge = 0;
    for(i=0;i<9;i++)
    for(j=0;j<9;j++)
        if(i/3!=j/3)
            gOut.idEdge |= gOut.M[i][j]<<EdgeMatrix[i][j];

    //Fill in classSize[].
    for(i=0;i<3;i++)
        gOut.classSize[i] = gOut.hasVertex[3*i]+gOut.hasVertex[3*i+1]+gOut.hasVertex[3*i+2];

    return;
}

int main()
{
    long   i,j,n,p,v;
    long   idVertex[64];
    long   notIsolatedID;
    long   idEdge;
    bool   bStoreChanged;
    CGraph g;

    //Initialize StrongIsoPermute[][] and construct F7, F9.
    SetUp();
```

```
//Go through all possible CGraph::idVertex values and store only those which have class
//sizes that are two or three. We are not interested in graphs with empty vertex classes
//by Theorem 1.2.1, or vertex classes of size one by Lemma 1.4.8.
n = 0;
for(i=0;i<(1<<9);i++)
{
    g.AssignVerticesFromID(i);

    if(g.classSize[0]>1 && g.classSize[1]>1 && g.classSize[2]>1)
    {
        idVertex[n] = i;
        n++;
    }
}

//Initialize the number of graphs found that are possibly extremal and vertex minimal.
Found = 0;

//Generate all possible tripartite graphs by cycling through all possible 27 bit
//integers representing CGraph::idEdge, and all possible CGraph::idVertex values stored
//in idVertex[].
for(idEdge=0;idEdge<(1<<27);idEdge++)
{
    g.AssignMatrixFromID(idEdge);

    //Work out which vertices have neighbours, and store the result in notIsolatedID.
    //This will aid us in creating a well-formed CGraph object. Much like
    //CGraph::idVertex, if vertex i has a neighbour then bit i is 1 otherwise it is 0.
    //Bit 0 is the least significant bit.
    notIsolatedID = 0;
    for(i=0;i<9;i++)
    for(j=0;j<9;j++)
        notIsolatedID |= g.M[i][j]<<i;

    for(v=0;v<64;v++)
    {
        if((idVertex[v] & notIsolatedID)!=notIsolatedID)
        {
            //If we assign idVertex[v] to g then g will not be well-formed as there will
            //exist a non-isolated vertex according to M[][] which is not part of the
            //graph according to hasVertex[].
            continue;
        }

        g.AssignVerticesFromID(idVertex[v]);

        //Check if we have enough memory to store the graph in Store[][]. Using the
        //sizeof operator is a standard trick to get the size of an array.
        //sizeof(Store)/sizeof(Store[0]) should equal 200.
        if(Found>=sizeof(Store)/sizeof(Store[0]))
        {
            //We should never run out of memory.
            cout << endl << endl;
```

```
    cout << "Error: Too many graphs, increase size of Store[][]." << endl;
    cout << endl;
    return 0;
}

//Start filling in Store[Found][] with all the graphs strongly-isomorphic to g.
for(p=0;p<1296;p++)
{
    StrongIsoGraph(g,p,Store[Found][p]);

    //If for any reason we wish to discard g and all its isomorphisms, we will
    //break out of this loop and deal with removing the graphs from Store[][].

    //Note that if Store[x][0] and Store[y][0] are strongly-isomorphic to each
    //other. Then the arrays Store[x][] and Store[y][] are just permutations of
    //each other. To avoid such duplication we'll only keep Store[Found][] if the
    //idEdge and idVertex of Store[Found][0] is the smallest of all the
    //isomorphisms in Store[Found][].
    if( Store[Found][0].idEdge   > Store[Found][p].idEdge
    || (Store[Found][0].idEdge  == Store[Found][p].idEdge
    &&  Store[Found][0].idVertex > Store[Found][p].idVertex))
        break;

    //Get rid of graphs whose class size can be reduced, or is too small (i.e.
    //those satisfying Lemmas 1.4.10, 1.4.13, and 1.4.14).
    if(bBadClassSize(Store[Found][p])==true)
        break;

    //Get rid of graphs where we can reduce the density of triangles by moving
    //edge weights (see Corollary 1.4.5).
    if(bCanMoveEdgeWeights(Store[Found][p])==true)
        break;

    //Get rid of graphs without a triangle (see Theorem 1.2.1).
    if(bHasNoTriangles(Store[Found][p])==true)
        break;

    //Get rid of graphs strongly-isomorphic to F7 (see Lemma 1.4.16).
    if(Store[Found][p].idEdge  ==F7.idEdge
    && Store[Found][p].idVertex==F7.idVertex)
        break;

    //Get rid of graphs strongly-isomorphic to F9 (see Lemma 1.4.18).
    if(Store[Found][p].idEdge  ==F9.idEdge
    && Store[Found][p].idVertex==F9.idVertex)
        break;
}

//If p!=1296 we broke out of the loop earlier than expected because
//   (i)  one of graphs was not extremal or not vertex minimal,
//or (ii) Store[Found][0] was not the graph with the smallest IDs.
//In either case we wish to discard everything in Store[Found][] which we can
//easily do by not updating Found.
```

```
        if(p==1296)
        {
            //We didn't break out of the loop early. We wish to retain the graphs in
            //Store[Found][] which we do by increasing the value of Found.
            Found++;
        }
    }
}

//We have now tested every possible tripartite graph (with class sizes at most three).
//Store[][] holds a list of possible extremal vertex minimal graphs, which we will reduce
//further by repeated applications of Lemma 1.4.15, as implemented by the function
//bCanReplaceBy8(). If Store[i][j] satisfies bCanReplaceBy8() then Store[i][j] and all
//graphs strongly-isomorphic to it (i.e. the graphs in the Store[i][] array) can be
//removed. For speed and convenience we will indicate that they have been removed by
//setting Store[i][0].idEdge to -1.

do
{
    bStoreChanged = false;

    //Go through each possible extremal vertex minimal graph.
    for(i=0;i<Found;i++)
    {
        if(Store[i][0].idEdge==-1)
            continue; //The graph and its isomorphisms have been removed.

        for(j=0;j<1296;j++)
            if(bCanReplaceBy8(Store[i][j])==true)
            {
                Store[i][0].idEdge = -1;

                //bCanReplaceBy8() depends on Store[][]. Since Store[][] has changed
                //there may be graphs that returned false but now will return true. Hence
                //we need to re-check those graphs and repeat this process, which we
                //indicate by setting bStoreChanged to true.

                bStoreChanged = true;
                break;
            }
    }
}while(bStoreChanged==true);

//Display the final list of possible extremal vertex minimal graphs.
DisplayStore();

//Save results to "Output.txt".
if(bSaveToTxtFile==true)
{
    fstream OutputFile;
    OutputFile.open("Output.txt", fstream::out | fstream::trunc);
    if(OutputFile.fail())
```

```
        {
            cout << endl << endl;
            cout << "Failed to open Output.txt." << endl;
            cout << endl;
            return 0;
        }
        DisplayStore(OutputFile);
        OutputFile.close();
    }

    cout << "Finished." << endl << endl;
    return 0;
}
```

Appendix B

DensityBounder

DensityBounder is an implementation of Razborov's method for calculating upper bounds of Turán densities for 3-graphs (see Section 2.2).

The C++ source code can be found in the `DensityBounder` directory of the CD-ROM. The program has been tested on Windows XP using Visual C++ 2008, and on Ubuntu 9.10 (Linux) using gcc 4.4.1, both running on a Pentium 4 CPU with 1 GB of RAM. The program exits immediately after finishing, and therefore should be run from a command-line interface rather than through the graphical user interface.

It is by default setup to calculate an upper bound for $\pi(\mathcal{F}')$ (see Section 2.3). This can be changed to $\pi(K_4^-)$ by commenting out

```
#define _HYPERGRAPHS_DO_JUMP_
```

and uncommenting

```
#define _FORBIDDING_K4MINUS_.
```

The positive semidefinite matrices can be loaded from precomputed ".soln" files which can be found in the `DensityBounder` directory of the CD-ROM. Alternatively the matrices can be computed using DensityBounder and a semidefinite program solver.

To compute the positive semidefinite matrices, DensityBounder generates a ".dat-s" file (the specific name for each problem is stored in the `filenameSDP[]` array). The ".dat-s" file will be the input to the semidefinite program solver. Next we use the solver to output a solution into a ".out" file (given by the `filenameOutput[]` array). In particular to get a solution using CSDP's standalone solver we type

```
csdp HypergraphsDoJump.dat-s HypergraphsDoJump.out
```

at the command-line. DensityBounder then takes the ".out" file, removes rounding errors, and stores the result in a ".soln" file (given by `filenameSoln[]`).

Once the ".soln" file has been created or loaded, DensityBounder calculates an upper bound for the Turán density using only integer type variables. It avoids any use of floating point numbers at this stage so that no rounding errors can occur.

Appendix C

DensityChecker

DensityChecker is a program which calculates the upper bounds of Turán densities for 3-graphs using Razborov's method (see Section 2.2) from data given in a text file.

The program's main purpose is to take a series of types, flags, and matrices, which constitute a proof of an upper bound, and verify they do indeed form a proof. The program independently calculates $\mathcal{H}$, checks all matrices are positive semidefinite, and calculates the upper bound of the Turán density.

The data DensityChecker needs to make the verification should be provided in a text file passed to it via the command-line. We provide an example of such a text file at the end of this section. The C++ source code for DensityChecker can be found on the CD-ROM in the `DensityChecker` directory. This directory also contains the text files `S1.txt`, `S2.txt`, `S3.txt`, `S4.txt`, `S5.txt`, and `K5.txt` (see Section 2.4.1) which can be passed as input to DensityChecker. For completeness, we have also provided the files `HypergraphsDoJump.txt` and `K4-.txt` based on the ".soln" files created by DensityBounder (see Appendix B).

```
#File name: Example.txt

#Everything after a # symbol is a comment and is ignored by DensityChecker. The file format
#is fairly tolerant of whitespace characters (i.e. spaces, tabs, and newline characters) to
#help make the file more readable for humans.

#This file can be found in the DensityChecker folder of the CD-ROM. It describes the data
#given by Razborov in the paper "On 3-hypergraphs with forbidden 4-vertex configurations"
#which proves that the Turan density is 5/9 when we forbid induced four vertex subgraphs to
#have one or four edges. To use DensityChecker to verify the data does prove an upper bound
#of 5/9, type "DensityChecker Example.txt" at the command-line.

#The data file should begin by specifying what program it has been written for.
```

```
Program : DensityChecker

#Next we declare the order of the graphs in our family $\mathcal{H}$. This is denoted by $l$
#in the thesis. The current implementation of DensityChecker can only handle 3-graphs with at
#most nine vertices.
Order of subgraphs H : 6

#Description of the forbidden family $\mathcal{F}$. Note that a 3-graph is represented by
#a number (indicating its order, n say) folowed by a colon then its edge set (assuming its
#vertices are labelled 1,2,...,n).
Number of forbidden graphs : 2
Forbidden graphs           :
4 : {123, 124, 134, 234}
4 : {123}

#In the traditional definition of Turan density we forbid the forbidden graphs from appearing
#as subgraphs, and so assign "no" to the following setting. However, in this particular
#problem we wish to forbid the graphs from appearing as *induced* subgraphs so we instead
#set it to "yes".
Forbid only induced subgraphs : yes

Number of terms : 4 #The bound involves four postive semidefinite matrices.

#Each term consists of a positive semidefinite matrix, and the basis it is written in. Each
#member of the basis is a linear combination of flags, hence we also need to define the
#flags and their type.


--- Term 1 --- #The start of the term is indicated by a simple header line.

#This term is equivalent to (9/5)[(e-4/9)^2]_1 in Razborov's notation.

#We need only define the order of the type. The edge set of the type can be determined from
#the flags. See term 2 for an example.
Order of type : 1

#The common order of the flags, denoted by $m$ in the thesis.
#Note that "order of flags"<=("order of type" + "order of subgraphs H")/2 must hold.
Order of flags : 3

#The number of flags in the flag list below.
Number of flags : 2

#The dimension of the matrix.
Matrix dimension : 1

#The edge sets of the flags used in describing the basis.
#The order of the flags has already been defined. The labelled vertices of the flag (which
#describe the type) are the vertices 1,...,"order of type". In this case there is only one
#labelled vertex: the vertex 1.
Flags :
F1 : {}
F2 : {123}
```

```
#The basis the matrix is written in.
#Each element of the basis is a linear combination of flags in the form:
#<a rational coefficient>(<label of the flag in the flag list above>) + ...
Basis:
B1 : 5/9(F1) - 4/9(F2)

#The entries of the matrix.
#The program expects to read in a list of rational numbers separated by whitespaces, which
#represent the entries (1,1), (1,2), ..., (1,"matrix dimension"), (2,1), (2,2), ... etc.
Matrix :
9/5


--- Term 2 ---

Order of type    : 4
Order of flags   : 5
Number of flags  : 17
Matrix dimension : 4

#The order of the type is four. Hence the labelled vertices are 1,2,3,4. By looking at the
#flag "F1" we can determine the edge set of the type is {124, 134, 234}, this is denoted by
#$\tau_1$ in Razborov's paper.

Flags :
F1  : {124, 134, 234, 135, 235, 145, 245}
F2  : {124, 134, 234, 125, 235, 145, 345}
F3  : {124, 134, 234, 125, 135, 245, 345}
F4  : {124, 134, 234, 125, 135, 235, 245}
F5  : {124, 134, 234, 125, 135, 235, 345}
F6  : {124, 134, 234, 125, 135, 235, 145}
F7  : {124, 134, 234, 125, 135, 235}
F8  : {124, 134, 234, 145, 245, 345}
F9  : {124, 134, 234, 135, 235, 245}
F10 : {124, 134, 234, 125, 235, 345}
F11 : {124, 134, 234, 135, 235, 145}
F12 : {124, 134, 234, 125, 235, 145}
F13 : {124, 134, 234, 125, 135, 345}
F14 : {124, 134, 234, 125, 135, 245}
F15 : {124, 134, 234, 245, 345}
F16 : {124, 134, 234, 145, 345}
F17 : {124, 134, 234, 145, 245}

Basis :
B1 : (F1)+(F2)+(F3)-(F4)-(F5)-(F6)
B2 : (F1)+(F2)+(F3)+2(F7)-(F8)
B3 : (F9)+(F10)+(F11)+(F12)+(F13)+(F14)
B4 : (F15)+(F16)+(F17)

#The matrix M_1 in Razborov's paper (multiplied by 9/5).
Matrix :
 96/40 -12/40 -66/40  75/40
```

```
-12/40  48/40   3/40 -39/40
-66/40   3/40  51/40 -51/40
 75/40 -39/40 -51/40  81/40

#We have used whitespaces to arrange the entries as they would appear in a matrix. This is to
#make it easier to read for humans. DensityChecker treats this the same as:
#Matrix: 96/40 -12/40 -66/40 75/40 -12/40 48/40 3/40 -39/40 -66/40 3/40 51/40 -51/40 75/40
#-39/40 -51/40 81/40


--- Term 3 ---
#The type is {134, 234} denoted by $\tau_2$ in Razborov's paper.

Order of type    : 4
Order of flags   : 5
Number of flags  : 15
Matrix dimension : 5

Flags :
F1  : {134, 234, 125, 135, 235, 145, 245}
F2  : {134, 234, 135, 235, 145, 245}
F3  : {134, 234, 345}
F4  : {134, 234, 125, 135, 235, 245}
F5  : {134, 234, 125, 235, 145, 245}
F6  : {134, 234, 125, 135, 235, 145}
F7  : {134, 234, 125, 135, 145, 245}
F8  : {134, 234, 125, 235, 145, 345}
F9  : {134, 234, 125, 135, 245, 345}
F10 : {134, 234, 125, 235, 145}
F11 : {134, 234, 125, 135, 245}
F12 : {134, 234, 145, 245, 345}
F13 : {134, 234, 135, 235, 345}
F14 : {134, 234, 145, 245}
F15 : {134, 234, 135, 235}

Basis :
B1 : (F1)-(F2)
B2 : (F1)-(F3)
B3 : (F4)+(F5)+(F6)+(F7)
B4 : (F8)+(F9)
B5 : (F12)+(F13)

#The matrix M_2 in Razborov's paper (multiplied by 9/5).
Matrix :
 192/40  -60/40   18/40 -126/40 132/40
 -60/40  120/40   75/40 -123/40 -57/40
  18/40   75/40   84/40 -165/40  -3/40
-126/40 -123/40 -165/40  393/40 -54/40
 132/40  -57/40   -3/40  -54/40  99/40


--- Term 4 ---
#The type is {134, 234} denoted by $\tau_2$ in Razborov's paper.
```

```
Order of type    : 4
Order of flags   : 5
Number of flags  : 15
Matrix dimension : 1

Flags :
F1  : {134, 234, 125, 135, 235, 145, 245}
F2  : {134, 234, 135, 235, 145, 245}
F3  : {134, 234, 345}
F4  : {134, 234, 125, 135, 235, 245}
F5  : {134, 234, 125, 235, 145, 245}
F6  : {134, 234, 125, 135, 235, 145}
F7  : {134, 234, 125, 135, 145, 245}
F8  : {134, 234, 125, 235, 145, 345}
F9  : {134, 234, 125, 135, 245, 345}
F10 : {134, 234, 125, 235, 145}
F11 : {134, 234, 125, 135, 245}
F12 : {134, 234, 145, 245, 345}
F13 : {134, 234, 135, 235, 345}
F14 : {134, 234, 145, 245}
F15 : {134, 234, 135, 235}

#The element g_0 in Razborov's paper.
Basis :
B1 : -(F4)+(F5)+(F6)-(F7)+2(F8)-2(F9)+2(F10)-2(F11)

Matrix :
9/10 #9/5 times 1/2.
```

Appendix D

Checking the Hypercube Turán Densities

HypercubeEdgeDensityChecker and HypercubeVertexDensityChecker are programs which take text files as input and uses the data they contain to calculate upper bounds for $\pi_{ce}(\mathcal{F})$ and $\pi_{cv}(\mathcal{F})$ respectively (see Sections 2.5.2 and 2.5.3). They behave much in the same way as DensityChecker (see Appendix C).

The C++ source code for HypercubeEdgeDensityChecker can be found in the `HypercubeEdgeDensityChecker` folder on the CD-ROM. The folder also contains the files `B.txt` and `B1B2.txt` which can be passed as input to HypercubeEdgeDensityChecker. The files follow a similar format to those created for DensityChecker. The differences are highlighted and explained via the comments in `B.txt`.

The C++ source code for HypercubeVertexDensityChecker can be found in the `HypercubeVertexDensityChecker` folder on the CD-ROM. The folder also contains the input files `B3-.txt`, `B3.txt`, and `B4B5.txt`. They follow a similar format to those created for DensityChecker. The file `B3-.txt` has been commented to highlight the differences.

Bibliography

[1] T. Bohman, A. Frieze, M. Ruszinkó, and L. Thoma, *G-intersecting Families*, Combinatorics, Probability and Computing, **10** (5) 367–384, (2001).

[2] T. Bohman, and R. R. Martin, *A Note on G-intersecting Families*, Discrete Mathematics, **260** 183–188, (2003).

[3] B. Bollobás, *Relations between sets of complete subgraphs*, Proceedings of the Fifth British Combinatorial Conference, Utilitas Mathematica, Winnipeg, **15** 79–84, (1976).

[4] A. Bondy, J. Shen, S. Thomassé, and C. Thomassen, *Density conditions for triangles in multipartite graphs*, Combinatorica, **26** (2) 121–131, (2006).

[5] B. Borchers, *CSDP, a C library for semidefinite programming*, Optimization Methods and Software, **11** (1) 613–623, (1999).

[6] P. Brass, H. Harborth, and H. Nienborg, *On the maximum number of edges in a C_4-free subgraph of Q_n^**, Journal of Graph Theory, **19** (1) 17–23, (1995).

[7] D. de Caen, *Extension of a theorem of Moon and Moser on complete subgraphs*, Ars Combinatoria, **16** 5–10, (1983).

[8] F. R. K. Chung, *Subgraphs of a Hypercube Containing No Small Even Cycles*, Journal of Graph Theory, **16** (3) 273–286, (1992).

[9] P. Erdős, *On a theorem of Rademacher–Turán*, Illinois Journal of Mathematics, **6** 122–127, (1962).

[10] P. Erdős, *On some extremal problems on r-graphs*, Discrete Mathematics, **1** (1) 1–6, (1971).

[11] P. Erdős, *On some problems in graph theory, combinatorial analysis and combinatorial number theory*, Graph Theory and Combinatorics, Academic Press, London–New York, 1–17, (1984).

[12] P. Erdős, C. Ko, and R. Rado, *Intersection theorems for systems of finite sets*, Quarterly Journal of Mathematics, Oxford Series 2, **12** 313-320, (1961).

[13] P. Erdős, and M. Simonovits, *A limit theorem in graph theory*, Studia Scientiarum Mathematicarum Hungarica, **1** 51–57, (1966).

[14] P. Erdős, and A. H. Stone, *On the structure of linear graphs*, Bulletin of the American Mathematical Society, **52** (12) 1087–1091, (1946).

[15] D. C. Fisher, *Lower bounds on the number of triangles in a graph*, Journal of Graph Theory, **13** (4) 505–512, (1989).

[16] P. Frankl, and Z. Füredi, *An exact result for* 3*-graphs*, Discrete Mathematics, **50** 323–328, (1984).

[17] P. Frankl, Y. Peng, V. Rödl, and J. Talbot, *A note on the jumping constant conjecture of Erdős*, Journal of Combinatorial Theory, Series B, **97** (2) 204–216, (2007).

[18] P. Frankl, and V. Rödl, *Hypergraphs do not jump*, Combinatorica, **4** 149–159, (1984).

[19] J. Goldwasser, *Personal communication*, (2010).

[20] G. Hurlbert, and V. Kamat, *Erdős–Ko–Rado theorems for chordal and bipartite graphs*, arXiv, (2009).

[21] J. R. Johnson, and J. Talbot, *G-intersection theorems for matchings and other graphs*, Combinatorics, Probability and Computing, **17** (4) 559–575, (2008).

[22] J. R. Johnson, and J. Talbot, *Vertex Turán problems in the hypercube*, Journal of Combinatorial Theory, Series A, **117** (4) 454–465, (2010).

[23] K. A. Johnson, and R. Entringer, *Largest induced subgraphs of the n-cube that contain no* 4*-cycles*, Journal of Combinatorial Theory, Series B, **46** (3) 346–355, (1989).

[24] E. A. Kostochka, *Piercing the edges of the n-dimensional unit cube*, Diskret. Analiz Vyp. 28 Metody Diskretnogo Analiza v Teorii Grafov i Logiceskih Funkcii, 55–64, (1976) [in Russian].

[25] L. Lovász, and M. Simonovits, *On the number of complete subgraphs of a graph II*, Studies in pure mathematics, Birkhäuser, Basel, 459–495, (1983).

[26] W. Mantel, *Problem 28*, Wiskundige Opgaven, **10** 60–61, (1907).

[27] K. Markström, *Personal communication*, (2010).

[28] Z. L. Nagy, *A Multipartite Version of the Turán Problem - Density Conditions and Eigenvalues*, The Electronic Journal of Combinatorics, **18** (1) P46, (2011).

[29] A. A. Razborov, *Flag Algebras*, Journal of Symbolic Logic, **72** (4) 1239–1282, (2007).

[30] A. A. Razborov, *On the Minimal Density of Triangles in Graphs*, Combinatorics, Probability and Computing, **17** (4) 603–618, (2008).

[31] A. A. Razborov, *On* 3*-Hypergraphs with Forbidden* 4*-Vertex Configurations*, SIAM Journal on Discrete Mathematics, **24** (3) 946–963, (2010).

[32] A. Thomason, and P. Wagner, *Bounding the size of square-free subgraphs of the hypercube*, Discrete Mathematics, **309** (6) 1730–1735, (2009).