List colouring hypergraphs and extremal results for acyclic graphs

by

Martin Pei

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2008

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

We study several extremal problems in graphs and hypergraphs. The first one is on list-colouring hypergraphs, which is a generalization of the ordinary colouring of hypergraphs. We discuss two methods for determining the list-chromatic number of hypergraphs. One method uses hypergraph polynomials, which invokes Alon's combinatorial nullstellensatz. This method usually requires computer power to complete the calculations needed for even a modest-sized hypergraph. The other method is elementary, and uses the idea of minimum improper colourings. We apply these methods to various classes of hypergraphs, including the projective planes.

We focus on solving the list-colouring problem for Steiner triple systems (STS). It is not hard using either method to determine that Steiner triple systems of orders 7, 9 and 13 are 3-list-chromatic. For systems of order 15, we show that they are 4-list-colourable, but they are also "almost" 3-list-colourable. For all Steiner triple systems, we prove a couple of simple upper bounds on their list-chromatic numbers. Also, unlike ordinary colouring where a 3-chromatic STS exists for each admissible order, we prove using probabilistic methods that for every s, every STS of high enough order is not s-list-colourable.

The second problem is on embedding nearly-spanning bounded-degree trees in sparse graphs. We determine sufficient conditions based on expansion properties for a sparse graph to embed every nearly-spanning tree of bounded degree. We then apply this to random graphs, addressing a question of Alon, Krivelevich and Sudakov, and determine a probability p where the random graph $G_{n,p}$ asymptotically almost surely contains every tree of bounded degree. This p is nearly optimal in terms of the maximum degree of the trees that we embed.

Finally, we solve a problem that arises from quantum computing, which can be formulated as an extremal question about maximizing the size of a type of acyclic directed graph.

Acknowledgements

First, a billion thanks to my supervisor Penny Haxell. Her mathematical agility together with her endless patience, encouragement and optimism made this thesis a reality. Also, thanks to my officemates and office neighbours (who are too numerous to name) who gave me many helps and diversions during my six years at the University of Waterloo. My family and my friends from KWCACMYF and mCCF (also too numerous to name) provided emotional support and prayers during my down times, and I appreciate it greatly. Finally, I thank God for moving me forward and keeping me alive throughout these tumultuous years.

Contents

List of Figures

Chapter 1

Introduction

We study three different extremal problems for graphs and hypergraphs in this thesis. The subject of extremal combinatorics contains many diverse topics, and many different methods from various areas of mathematics have been successfully applied to solve extremal problems. We will use several of these methods to tackle the three problems that we solve here.

Generally speaking, extremal questions ask for the maximum or minimum of certain parameters in a graph (or hypergraph) such that some properties hold, and what kind of structure do such "extremal" examples have. One basic example is the simplest form of Túran's theorem, which asks for the maximum number of edges that a graph on n vertices can have without having a triangle. The solution is the complete bipartite graph where the two parts of the partition are equal or nearly equal. We refer the readers to [12] for more information regarding extremal graph theory.

The first problem that we look at concerns the list-colouring of hypergraphs. In Chapters 2, 3 and 4, we use various methods such as algebraic, probabilistic, computational and extremal techniques to determine or bound the list chromatic number of many hypergraphs, focusing in particular on Steiner triple systems. We also investigate the relationship between list-colouring hypergraphs and its "predecessors" such as the list-colouring of graphs and the colouring of hypergraphs, where we find many interesting similarities and differences between these topics. For example, the greedy colouring result in graphs and the gap between ordinary and list colouring of graphs both extend to hypergraphs. However, the fact that the list chromatic number of graphs grows with their minimum degrees does not hold for hypergraphs in general (see Section 4.1). When we look at hypergraphs that come from combinatorial designs, we find that for designs of small size, their list colourability tends to be close to their ordinary colourability. However, for large designs, the two parameters can differ by a lot. So far, very little is known about this topic in the literature.

In Chapter 5, we study properties of sparse graphs that embed all nearly-spanning bounded-degree trees. We use probabilistic and graph theoretic approaches as main methods for solving this problem. In particular, we are interested in parameters of sparse random graphs that would satisfy these properties asymptotically. Our improvements over known results are near optimal in terms of the maximum degree of the trees.

Finally, in Chapter 6, we solve an extremal problem arising in quantum computing by considering acyclic directed graphs.

1.1 Background

We begin by giving some basic definitions and background information that we will use in this thesis.

Graphs and hypergraphs

A hypergraph $H = (V, E)$ consists of a set of vertices V and a set of edges E where each edge is a subset of V of size at least 2. For a positive integer r, H is r-uniform if each edge is an r-subset of V. A 2-uniform hypergraph is called a graph. The degree of a vertex v, denoted $d_H(v)$, is the number of edges in H that contain v. We will give an illustration of the definition in Section 2.1. For more background in graph theory, we refer the readers to [25].

Colouring and list-colouring

For a positive integer k, a hypergraph H is k-colourable if given a fixed set of k colours, there exists an assignment to each vertex of one of the k colours such that no edge is monochromatic. Such a colouring is called a proper colouring. We say that H is k-chromatic if k is the smallest integer such that H is k-colourable. This k is also called the chromatic number of H, denoted $\chi(H)$.

List-colouring is a generalization of colouring. A k-list-assignment L of H is a function that maps each vertex of H to a set (called "list") of k colours. An

L-colouring is a colouring where each vertex receives a colour from its list in L. Such a colouring is proper if there are no monochromatic edges. We say that H is k-list-colourable (or k-choosable) if for every k-list-assignment L, there is a proper L-colouring. The smallest k such that H is k-list-colourable is called the list chromatic number (or choice number) of H, denoted $\chi_l(H)$.

Some block designs and hypergraphs

Most of the hypergraphs that we consider in this thesis come from combinatorial design theory. In the most general form, a design is a pair $(X, \mathcal{A})$ where X is a set of elements called points, and $\mathcal{A}$ is a collection of subsets of X called blocks. Essentially we can consider the points as vertices and the blocks as edges of a hypergraph. We will use the terms vertices and points interchangeably, and also with the terms edges and blocks. We refer the readers to Stinson [64] for the basic information about designs that is presented here.

A broad class of designs is called Balanced Incomplete Block Design (BIBD). For positive integers v, k, λ where $v > k \geq 2$, a (v, k, λ)-BIBD is a design $(X, \mathcal{A})$ where $|X| = v$, every block contains exactly k points, and every pair of points is in exactly λ blocks. Note that the number of blocks in such a BIBD is $\lambda\binom{v}{2}/\binom{k}{2}$.

A BIBD where the number of vertices equals the number of blocks is called a symmetric BIBD. We will state the following result regarding symmetric BIBDs:

Theorem 1.1 (Ryser [61])**.** *In a symmetric (v, k, λ)-BIBD, every pair of distinct blocks intersect at exactly λ points.*

For a positive integer n, a projective plane of order n is defined as an $(n^2 + n + 1, n + 1, 1)$-BIBD. Notice that the number of blocks in a projective plane of order n is $\binom{n^2+n+1}{2}/\binom{n+1}{2} = n^2 + n + 1$, so it is a symmetric BIBD. On the other hand, any symmetric BIBD with $\lambda = 1$ must be a projective plane, i.e. there exists an n that satisfies the parameters of a projective plane. By Theorem 1.1, we know that each pair of blocks must intersect at exactly one point. So far, only projective planes whose orders are prime powers are known to exist. There is only one projective plane of order 2, and it is called the Fano Plane. It has 7 points and 7 blocks, and can be represented as Figure 1.1.

For an integer n, a Steiner triple system of order n, denoted STS(n), is an $(n, 3, 1)$-BIBD. The number of blocks of an STS(n) is $\binom{n}{2}/3 = n(n-1)/6$. A necessary condition for the existence of an STS(n) is that $n \equiv 1, 3 \bmod 6$. Such

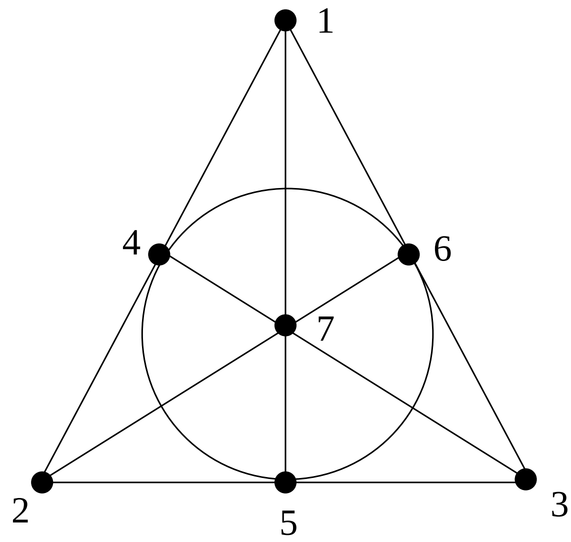

Figure 1.1: The Fano Plane.

an n is called admissible for STS. Bose [15] constructed Steiner triple systems of all orders $n \equiv 3 \bmod 6$, and Skolem [63] modified the construction for all orders $n \equiv 1 \bmod 6$. Hence an STS(n) exists if and only if n is admissible. The smallest non-trivial Steiner triple system is the Fano plane, which is a STS(7).

Random graphs

The random graph $\mathcal{G}(n, p)$ is a probability distribution on graphs with a fixed set of n vertices where each possible edge exists randomly and independently with probability p. In this model, each graph with m edges would occur with probability $p^m(1-p)^{n-m}$. We will talk about the random graph $G(n, c/n)$ for some constant c, so the probability that each edge exists is linear with respect to n^{-1}, and it is considered to be a sparse random graph. In this model, the expected number of edges is $c(n-1)/2$, and so the expected average degree of this random graph is about c. Let P be a property that a graph on n vertices may or may not have, e.g. the graph is k-colourable, or the graph is bipartite. Such a property P holds asymptotically almost surely (a.a.s.) for $G(n, p)$ if the probability that P is true approaches 1 as n approaches infinity. For an introduction to random graphs and probabilistic methods, see e.g. [42] and [7].

1.2 List-Colouring Hypergraphs

Colouring is perhaps the most studied problem in graph theory. It simply asks what is the minimum number of colours that is required to colour the vertices of a graph so that adjacent vertices have different colours. The most famous problem in this area is the four-colour conjecture that started in the 19th century. The conjecture asks if every planar graph can be coloured using four colours. This seemingly innocent problem was only solved in late 20th century, and the proof involves very heavy computations done by a computer (see [59]).

The subject of list-colouring of graphs was introduced independently by Vizing [67] and Erdős, Rubin and Taylor [30]. One important open problem in this area is the following conjecture regarding line graphs, which was raised by Vizing and others, and first appeared in print form in Bollobás and Harris [14]. (When G is a graph, $L(G)$ is the line graph of G, i.e. vertices in $L(G)$ represent edges of G, and vertices in $L(G)$ are adjacent if and only if their corresponding edges in G are adjacent.)

Conjecture 1.2 (The List Colouring Conjecture)**.** *For every graph G,*

$$\chi(L(G)) = \chi_l(L(G)).$$

This is a somewhat surprising conjecture since the gap between $\chi(G)$ and $\chi_l(G)$ can be arbitrarily large (Theorem 2.1 by Erdős, Rubin and Taylor [30]). So far, only special cases of this conjecture have been proven, the most famous one being from Galvin [36] who proved that the conjecture is true for all bipartite multigraphs. Another interesting result is from Kahn [44], who proved that for hypergraphs of bounded edge size, the list colouring conjecture is asymptotically true with respect to the maximum degree (note that the line graph of a hypergraph is a graph).

There have been works on the list colourability of planar graphs, perhaps inspired by the Four-Colour Theorem. The proofs for the list-colourability results are much simpler than the computationally-heavy proof of the Four-Colour Theorem. Alon and Tarsi [8] first proved that bipartite planar graphs are 3-list-colourable. Then Voigt [68] proved that not all planar graphs are 4-list-colourable, meaning that there can be a gap between the chromatic and list-chromatic number of planar graphs. But Thomassen [65] showed that this gap cannot be too large by proving that every planar graph is 5-list-colourable.

There are several surveys of fundamental works on list colouring, including Alon [2], Tuza [66] and Kratochvíl, Tuza and Voigt [47]. The authors of [47] noted that list coloring of hypergraphs is an area where very little is known. We list a couple of results in this topic, both of which take results for colourings of graphs and extend them to list colourings of hypergraphs: Benzaken, Gravier and Škrekovski [11] extended Hajós' Theorem for constructing non-k-colourable graphs into hypergraphs that are not k-list-colourable; and Kostochka, Stiebitz and Wirth [46] proved the hypergraph list-colouring version of Brooks' Theorem, which essentially says that if a hypergraph H has at least two edges and each edge has size at least 3, then $\chi_l(H)$ is at most the maximum degree Δ. Note that in Section 2.1.2, as background, we will give a weaker (but easier) result that $\chi_l(H) \leq \Delta + 1$, which is an extension of the greedy colouring scheme for graphs.

In terms of colouring of designs, Jensen and Toft [43] listed several problems involving the colouring of designs, including projective planes and some triple systems. For triple systems, a survey of results regarding the chromatic number can be found in Colbourn and Rosa [19]. Steiner triple systems seem to have gained special attention. Results include early ones like Erdős and Hajnal [28], who proved that for any $k \geq 2$, there exist k-chromatic partial Steiner triple systems (triple systems where each pair of elements appears at most once). In a more recent paper, de Brandes, Phelps and Rödl [24] used probabilistic methods to show that Steiner triple systems can have arbitrarily large chromatic number. Colbourn and Rosa [19] noted that the colourings of designs is one of the few subjects in triple systems where probabilistic methods have effective applications.

Colouring smaller Steiner triple systems also gained some attention, and we now know that the chromatic number of these systems can fall into a small range of numbers. For example, Mathon, Phelps and Rosa [50] showed that all Steiner triple systems of size between 7 and 15 are 3-chromatic by providing a list of all such systems. Using elementary methods and detailed case analysis on the structure of subsystems of Steiner triple systems, Horak [41] proved that all Steiner triple systems with at most 25 points are 4-colourable. Since all nontrivial Steiner triple systems have chromatic number at least 3, we know that the chromatic numbers of these small systems are either 3 or 4.

The goal of the first half of this thesis is to merge these two subjects and attempt to solve list-colouring problems for some of the most common designs. So far, very little is known about this topic in the literature, save for an example of the Fano plane in Ramamurthi and West [55]. In Chapter 2, we will present two methods for solving list-colouring problems. One is through hypergraph polynomials, which uses a deep result by Alon [3] known as combinatorial nullstellensatz. Using this method, we often resort to computers to do the heavy computations required to solve the problem. The other is through elementary methods by manipulating objects that we call minimum improper colourings (MICs). We solve the problem for all symmetric BIBDs (which include projective planes) as illustrations for both methods.

We then move the focus to solving the list-colouring problem for Steiner triple systems. We deal with small STS in Chapter 3. In particular, we show that Steiner triple systems of orders 9 and 13 have list-chromatic number 3, which matches their chromatic numbers. For systems of order 15, we first show that they are all 4-list-colourable, and then use both methods to show that they are "almost" 3-list-colourable. In Chapter 4, we prove several general bounds of the list-chromatic numbers for all STS. We will use probabilistic methods to show that, unlike the chromatic number, the list-chromatic number of STSs will grow as the order of the STS increases. We will prove an upper bound on the lowest possible list-chromatic number for all STSs of order n, which is around $\log n$. These results imply various bounds on other parameters regarding the list chromatic number of STSs.

1.3 Tree Embeddings

The most well-known conjecture from this area is the Erdős-Sós conjecture from 1963 [27].

Conjecture 1.3 (Erdős-Sós [27])**.** *Let T be a tree with d edges, and let G be a graph with average degree greater than $d-1$. Then G contains T as a subgraph.*

Ajtai, Komlós, Simonovits and Szemerédi have an unpublished proof that the conjecture is true for sufficiently large d using difficult methods. Special cases of this conjecture have been proven using elementary methods, e.g. McLennan [51] showed that the conjecture is true when the tree has diameter at most 4; Brandt and Dobson [16] proved the special case when the graph has girth 5, which Saclé and Wozniak [62] later improved by only requiring the graph to be C_4-free, and Haxell [39] improved it further by only requiring the graph to contain no $K_{2,r}$

where $r = \lfloor t/18 \rfloor$; Yin and Li [70] proved the case when the complement of the graph is C_4-free, and Dobson [26] proved the case when the complement of the graph does not contain $K_{2,4}$.

A fundamental result in the area of tree embeddings is from Friedman and Pippenger [34], who gave a sufficient condition based on expansion properties for a graph to contain all small trees (Theorem 5.5). Suppose that $N_G(X)$ denotes the set of neighbours of a subset of vertices X in G. Friedman and Pippenger proved that if $|N_G(X)| \geq (d+1)|X|$ whenever $|X| \leq 2n-2$, then the graph G contains any tree on at most n vertices and maximum degree at most d. The expansion factor can be improved using a theorem by Haxell [39] (Theorem 5.6), implying that the same statement holds if $|N_G(X)| \geq 3d|X|$ whenever $|X| \leq n/d+1$ (Corollary 5.7).

The problem of embedding "large" trees was mostly studied with embedding long paths in random graphs (maximum degree 2 for the problem). Erdős conjectured that $\mathcal{G}(n, c/n)$ contains a path of length at least $(1-\alpha(c))n$ a.a.s. where $0 < \alpha(c) < 1$ for all $c > 1$, and $\alpha(c)$ approaches 0 as c approaches infinity. This was proved by Ajtal, Komlós and Szemerédi [1] and Fernandez de la Vega [31]. Bollobás [13] improved this result by showing that $\alpha(c)$ decreases exponentially as c increases. Frieze [35] settled the question by proving that $\alpha(c) = (1+o(1))ce^{-c}$. This implies that the random graph $\mathcal{G}(n, c/n)$ contains a nearly-spanning path when $c = O(\log(1/\varepsilon))$.

In terms of embedding large trees other than paths, Fernandez de la Vega [32] showed that for a fixed tree T_n on n vertices with maximum degree $d+1$, there are constants C_1, C_2 with $N = C_1 n$ such that $G(N, C_2 d/N)$ a.a.s. contains T_n. In fact, this was proved with the constants $C_1 = C_2 = 8$, and the author noted that the proof works when C_2 is arbitrarily close to 1 (but not exactly one) as long as C_1 and d are sufficiently large. (To rephrase this, it means that $G(n, 8d/n)$ a.a.s. embeds a tree on $(1-7/8)n$ vertices with maximum degree $d+1$.) Note, however, that this result embeds only one fixed tree in the random graph. The results that we give in Chapter 5 embed all trees of maximum degree d.

One application of the tree embedding problem is in the study of fault tolerant linear arrays, which was raised by Rosenberg [60] and was studied by Alon and Chung [5]. The problem here is to find the minimum number of vertices and edges of a graph such that after removing all but ε portion of vertices or edges, the remaining graph still contains a path of length m. A natural extension of this is to replace the requirement of a path of length m by all trees of maximum degree d

and size m. This problem is also related to the size-Ramsey number (see e.g. [5], [40]), which asks for the least number of edges in a graph with the property that any two-colouring of the edges yields a monochromatic copy of a certain graph (in this case, a tree).

Our focus in this thesis is on a problem raised by Alon, Krivelevich and Sudakov [6]. They have proved that given a positive $\varepsilon < 1/2$ and a positive integer d, the random graph $\mathcal{G}(n, c/n)$ contains every tree on $(1-\varepsilon)n$ vertices with maximum degree d when

$$c = O\left(\frac{d^3 \log d \log^2(2/\varepsilon)}{\varepsilon}\right).$$

They asked the question of what is the best possible order for c, given that a result in [35] implies that the order for c cannot be smaller than $O(d\log(1/\varepsilon))$. In Chapter 5, we improve the results in [6] and prove that

$$c = O\left(\frac{d \log d \log^2(2/\varepsilon)}{\varepsilon}\right)$$

is sufficient. In terms of the parameter d, this is only $\log d$ away from being best possible. Improving the dependence on ε remains an interesting open problem.

1.4 Quantum Computing

In a quantum computer, the basic unit of information is stored as a "qubit," which is analogous to a "bit" in a classical computer. The bit and the qubit are quite different, however. A bit can store either a one or a zero, but a qubit occupies states describable as a linear combination of possible outcomes (perhaps a one or a zero). We may think of it as a "probability vector," which is a linear combination of mutually orthogonal vectors (one for each possible outcome), where the coefficient of each vector is complex. The state of a qubit cannot simply be read off like a bit, instead they need to be measured. The norm of the coefficients constitute the probability that such a state is the result of a measurement. One major problem with quantum computing is that any interaction with the outside world such as a measurement could turn a "coherent" state into a "decoherent" state, so sometimes error correction is needed. Another special property in quantum computing is in entanglement, where operations can be performed simultaneously on qubits that are separated at a physical distance. We refer the readers to [52] for more specific information related to quantum computing.

In Chapter 6, we will solve an extremal problem based on the one-way measurement model of quantum computing. Flow systems in this model may be described as graphs. We describe a transformation from these graphs into directed graphs, and those graphs that transform into acyclic directed graphs have "good" properties for the quantum problem. Determining the maximum size of "good" graphs would simplify the analysis of an algorithm for determining if a flow system is good. We do not assume any background on quantum computing for the readers (nor the author of this thesis), and the extremal problem itself can be seen as a strictly graph theoretical result.

Note: Sections 4.1 and 4.2 represent joint work with Penny Haxell. The content of Chapter 6 represents joint work with Niel de Beaudrap, and has appeared in [23].

Chapter 2

List-Colouring Hypergraphs

In this chapter, we will introduce a couple of techniques for determining the list-colourability of hypergraphs, and use them to determine the list chromatic number of several classes of hypergraphs. These are the tools that we will use extensively in Chapters 3 and 4 in solving the problem for Steiner triple systems. The structure of this chapter is as follows: First, in Section 2.1, we will give a couple of basic results that extend the standard results in list-colouring graphs. As symmetric BIBDs and projective planes are the key examples in this chapter, we will give known results about their colourability in Section 2.2. We will then consider two techniques in solving list-colouring problems. The first technique is in Section 2.3, where we will present a generalization of hypergraph polynomials that were introduced by Ramamurthi and West [55] that is computationally easier to work with (Theorem 2.6), and use it to determine the list chromatic number of some small hypergraphs through computations done by a computer. The second technique is introduced in Section 2.4, and it uses elementary methods to handle objects that we called Minimal Improper Colourings (MICs). We can use this method to determine exactly the list chromatic number of all symmetric BIBDs, and in particular all projective planes (Section 2.4.2), extending a result of Ramamurthi and West [55].

2.1 Background

We first give an example that illustrates the definition of list colouring and list chromatic number. Consider the Fano plane with the lists of size 2 as assigned in

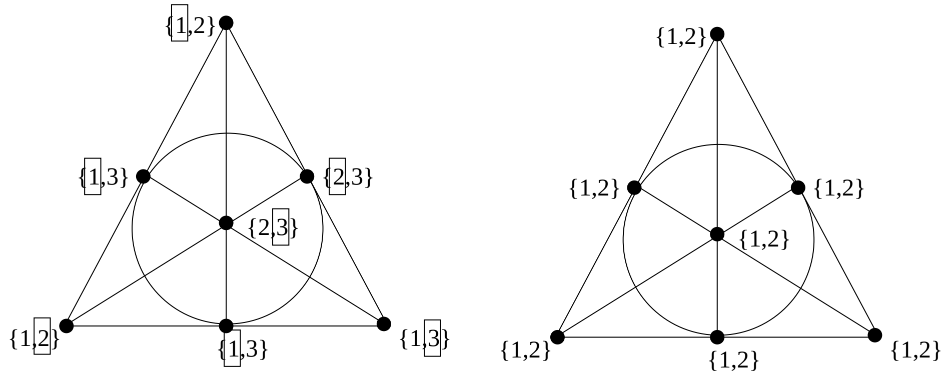

Figure 2.1: Using the Fano plane to illustrate the definition of list colouring.

the left part of Figure 2.1. The highlighted colours represent a proper colouring of the Fano plane from these lists. A different 2-list-assignment is given in the right part of Figure 2.1. It is not difficult to see that it is impossible to properly colour the Fano plane from these lists. Therefore, we conclude here that the Fano plane is not 2-list-colourable.

For the remainder of the section, we will first consider how much difference there can be between ordinary colouring and list colouring in the hypergraph case. Then we will give a simple greedy upper bound that "carries over" from the graph colouring case.

2.1.1 The Gap Between $\chi(H)$ and $\chi_l(H)$

Note that $\chi_l(H) \geq \chi(H)$ for all H: Let $k = \chi_l(H)$, and give a list-assignment L to H where each vertex receives the same list of k colours. Since H is k-list-colourable, there exists a proper L-colouring. Since L only assigns k different colours to the lists, this is also a proper k-colouring. Hence, $\chi(H) \leq k = \chi_l(H)$. The gap between $\chi_l(H)$ and $\chi(H)$ can be arbitrarily large, however, as we will show in this section.

The standard examples for graphs whose discrepancy between the chromatic number and list chromatic number is large were given by Erdős, Rubin and Taylor [30]:

Theorem 2.1 (Erdős, Rubin and Taylor [30])**.** *Let $k \geq 3$ be an integer. Then the complete bipartite graph $K_{m,m}$ where $m = \binom{2k-1}{k}$ is not k-list-colourable.*

Here, the list-assignment used to achieve this theorem is one where the m vertices in each part of the bipartition receive all possible k-subsets of a $(2k-1)$-set. We now prove a generalization of this result for r-uniform hypergraphs.

Theorem 2.2. *Let $r \geq 3$ and $k \geq 3$ be integers. Then there exists a 2-colourable r-uniform hypergraph that is not k-list-colourable.*

Proof. Let $m = \binom{rk-(r-1)}{k}$. We define a complete r-partite r-uniform hypergraph $H_{r,k}$ where the vertices of $H_{r,k}$ consist of r disjoint sets $V_1, \ldots, V_r$ of m vertices each, and the edges are all possible r-subsets $\{(v_1, \ldots, v_r) : v_i \in V_i, i = 1, \ldots, r\}$. Note that this is 2-colourable by colouring V_1 with one colour and the remaining vertices with another colour. To show that $H_{r,k}$ is not k-list-colourable, we give the list assignment L where for each V_i, all possible k-subsets of a fixed $(rk-(r-1))$-set (the "colours") appear as lists for the vertices. Let c be any L-colouring. Let C_i be the set of colours given to vertices in V_i by c, and let $D = \sum_{i=1}^{r} |C_i|$. Then for all i, $|C_i| \geq rk-(r-1)-(k-1)$, for otherwise a vertex in V_i with k of the remaining colours as its list would contribute one more colour to C_i. So

$$D \geq r(rk-(r-1)-(k-1)) = r^2k - r^2 + 2r - rk.$$

If every colour is in at most $r-1$ of the C_i's, then

$$D \leq (rk-(r-1))(r-1) = r^2k - r^2 + 2r - rk - 1,$$

which is a contradiction. Therefore, there exists at least one colour that appears in all C_i, and the edge containing a vertex with this colour from each V_i is monochromatic. Hence no L-colouring is proper, and $H_{r,k}$ is not k-list-colourable. □

2.1.2 Greedy Upper Bound

A graph is d-degenerate if every subgraph has a vertex of degree at most d. It is well-known that the chromatic number of a graph that is d-degenerate is at most $d+1$ (see e.g. [69]). The case for list-colouring hypergraphs is the same: A hypergraph is d-degenerate if every subhypergraph has a vertex of degree at most d; and $d+1$ is an upper bound for the list-chromatic number of a d-degenerate hypergraph. Note that if a hypergraph has maximum degree d, it is d-degenerate. For completeness, we will prove this result.

Lemma 2.3. *Let H be a d-degenerate hypergraph. Then $\chi_l(H) \leq d+1$.*

Proof. Suppose H has n vertices. Let L be any $(d+1)$-list-assignment for H. Order the vertices of H (backwards) as follows: Let v_n be a vertex of minimum degree in H (which must be no more than d). Suppose we have ordered $v_{i+1}, \ldots, v_n$. Then we pick v_i to be a vertex of minimum degree in the subhypergraph H_i induced by $V(H) \setminus \{v_{i+1}, \ldots, v_n\}$. Note that v_i has degree at most d in H_i. Now we attempt to L-colour H. Start by picking any colour for v_1 from $L(v_1)$. We wish to colour the vertices in order so that each edge that is fully coloured is not monochromatic. Suppose we have coloured $v_1, \ldots, v_{i-1}$. Consider the edges E_i in the subhypergraph H_i (which is the same as the subhypergraph induced by $v_1, \ldots, v_i$) containing v_i. Because of the way we ordered the vertices, $|E_i| \leq d$. For each of these edges, all vertices but v_i are coloured. Any edge in E_i that is not monochromatic among vertices other than v_i is already properly coloured regardless of which colour we give to v_i. So we only need to consider edges E'_i in E_i that are monochromatic without v_i. Since $|E'_i| \leq d$, there is at least one colour in $L(v_i)$ (which has size $d+1$) that is distinct from the colours of each edge in E'_i without v_i. We give that colour to v_i, and we have properly coloured all edges in E'_i. When we have coloured all the vertices, we would have a proper L-colouring. Hence $\chi_l(H) \leq d+1$. □

2.2 Colouring Symmetric BIBDs

We start with a result that is recorded in Jensen and Toft [43]:

Lemma 2.4. *A projective plane of order $n \geq 3$ is 2-colourable.*

We note that such a 2-colouring may be produced by finding three edges that do not share a vertex, then assign a colour to the set of vertices that appear in these three edges exactly once, and assign the remaining vertices with another colour.

Recall from Section 1.1 that the Fano plane is a projective plane of order 2. It is easy to see that the Fano is 3-colourable, but not 2-colourable. Later in Lemma 2.13, we will give a short proof which shows that symmetric BIBDs with $\lambda \geq 2$ have list chromatic number 2, which implies that they have chromatic number 2. Combining with Lemma 2.4, we see that every symmetric BIBD except the Fano plane has chromatic number 2.

In Section 2.4, we will show that for all symmetric BIBDs and projective planes, the chromatic number equals the list chromatic number.

2.3 Hypergraph Polynomials

We introduce the first of two techniques in approaching list-colouring hypergraphs in this section. We will begin with a description of graph polynomials, and see how Ramamurthi and West [55] extend them to hypergraph polynomials. We will discuss the advantages and disadvantages of their method, and derive our own way of creating hypergraph polynomials. These polynomials are potentially easier to deal with. We will see how these polynomials help with list colouring, and finally we utilize these tools and the computer to determine the list chromatic number of several hypergraphs.

2.3.1 Graph Polynomials and List Colouring

For a graph G with n vertices and m edges, let the vertices be $v_1, v_2, \ldots, v_n$. Each vertex v_i has a corresponding vertex variable x_i. The graph polynomial of G is $f_G(x_1, \ldots, x_n) = \prod\{(x_i - x_j) : i < j, v_iv_j \in E(G)\}$. Suppose that each vertex variable x_i is assigned a value a_i. If we consider each a_i as the "colour" of vertex v_i, then the colouring is proper if and only if $f_G(a_1, \ldots, a_n) \neq 0$. This is because for any edge $v_iv_j \in E(G)$, $x_i - x_j = 0$ if and only if $x_i = x_j$, i.e. v_i and v_j have been assigned the same colour. Notice that when this polynomial is expanded, each monomial has total degree m, the number of edges in the graph. Consider one such monomial $M = x_1^{d_1}x_2^{d_2}\cdots x_n^{d_n}$ (where $\sum_{i=1}^n d_i = m$), and suppose that each vertex v_i is given a list $L(v_i)$ of at least $d_i + 1$ colours. Through algebraic methods, Alon and Tarsi [8] proved that if the coefficient of M in $f_G(x_1, \ldots, x_n)$ is not zero, then there is a proper L-colouring of G. In particular, if $d = \max\{d_i : i = 1, \ldots, n\}$, then $\chi_l(G) \leq d + 1$.

Alon later generalized the algebraic result in [8] as an application of his theorems on "combinatorial nullstellensatz," which is based on Hilbert's nullstellensatz:

Theorem 2.5 (Alon [3]). *Let $\mathbb{F}$ be any field, and let f be a polynomial in $\mathbb{F}[x_1, \ldots, x_n]$ with total degree $\sum_{i=1}^n t_i$ where each t_i is a nonnegative integer. If the coefficient of* *the monomial* $\prod_{i=1}^{n} x_i^{t_i}$ *is nonzero and* $S_1, \ldots, S_n$ *are subsets of* $\mathbb{F}$ *with* $|S_i| > t_i$ *for all* i*, then there are elements* $s_1, \ldots, s_n$ *where* $s_i \in S_i$ *such that* $f(s_1, \ldots, s_n) \neq 0$.

For the remainder of this section, we will use the field of rational numbers $\mathbb{Q}$ whenever we apply this theorem. But for simplicity, the "colours" that we consider in sets like $S_1, \ldots, S_n$ in the theorem will be integers.

In addition to the list colouring application, Alon and Tarsi [8] found a combinatorial interpretation for the coefficients of monomials in f_G. Consider any orientation D of G. A subgraph of D is called Eulerian if for each vertex, the in-degree equals the out-degree (connectivity of the subgraph is not a factor). Such a subgraph is even or odd if it has an even or odd number of edges respectively. Consider the monomial M as above, and let D be an orientation of G where v_i is the source of an edge d_i times. Then the absolute value of the coefficient of M in $f_G(x_1, \ldots, x_n)$ equals the absolute difference between the number of even Eulerian subgraphs and the number of odd Eulerian subgraphs in D. In particular, this means that if the number of even Eulerian subgraphs is different from the number of odd Eulerian subgraphs in D, then the coefficient of M is not zero, hence G has a proper list colouring when each vertex v_i is given a list of $d_i + 1$ colours. Using this interpretation, Alon and Tarsi [8] proved that all planar bipartite graphs are 3-list-colourable. Fleischner and Stiebitz [33] used this to prove the cycle-plus-triangles problem, which is that a graph on $3n$ vertices that consists of a cycle of length $3n$ and n pairwise disjoint triangles is 3-list-colourable.

2.3.2 Extending to Hypergraph Polynomials

For some prime number r, given an r-uniform hypergraph H with n vertices $v_1, \ldots, v_n$, Ramamurthi and West [55] extended the idea of a graph polynomial to a hypergraph polynomial $f_H(x_1, \ldots, x_n)$ in the following way: Let θ be a primitive r-th root of unity, and let $v_1, \ldots, v_n$ be an ordering of the vertices. For each edge $e = \{v_{i_0}, \ldots, v_{i_{r-1}}\}$ where $i_0 < i_1 < \cdots < i_{r-1}$, create a polynomial $p_e = x_{i_0} + \theta x_{i_1} + \cdots + \theta^{r-1} x_{i_{r-1}}$ where $i_0 < i_1 < \cdots < i_{r-1}$. Then $f_H = \prod_{e \in E(H)} p_e$. As in the case for graphs, we can think of the values of $x_1, \ldots, x_n$, say they are $a_1, \ldots, a_n$ respectively, as the colours for $v_1, \ldots, v_n$. In order to use Theorem 2.5 and link this hypergraph polynomial to list colouring, we need to ensure that if $f_H(a_1, \ldots, a_n) \neq 0$, then the colouring c where $c(v_i) = a_i$ is a proper colouring of H:

If c is not a proper colouring, then there exists some edge e and some colour a such that $c(v) = a$ for all $v \in e$. But then $p_e(a_1, \ldots, a_n) = a \cdot (1 + \theta + \theta^2 + \cdots + \theta^{r-1}) = 0$ since a property of θ is that $1+\theta+\theta^2+\cdots+\theta^{r-1} = 0$. Therefore, $f_H(a_1, \ldots, a_n) = 0$.

In the discussion above, the only property of the roots of unity that we have used is the fact that $1+\theta+\theta^2+\cdots+\theta^{r-1} = 0$. We may generalize this as follows: For each edge e, suppose the factor that it contributes to the hypergraph polynomial f_H is $p_e = c_0 x_{i_0} + \cdots + c_{r-1} x_{i_{r-1}}$ for some constants $c_0, \ldots, c_{r-1}$ where $c_0 + \cdots + c_{r-1} = 0$ (in fact, these constants can change among different edges, as long as they add up to 0). Then the resulting hypergraph polynomial still satisfies the property that if $f_H(a_1, \ldots, a_n) \neq 0$, then the colouring c where $c(v_i) = a_i$ is a proper colouring of H. Combining this and Theorem 2.5, we get the following result.

Theorem 2.6. *Let $H = (V, E)$ be an r-uniform hypergraph where the vertices are $v_1, v_2, \ldots, v_n$. For each $e \in E$ where $e = \{v_{i_0}, v_{i_1}, \ldots, v_{i_{r-1}}\}$, define a polynomial $p_e = c_0 x_{i_0} + c_1 x_{i_1} + \cdots + c_{r-1} x_{i_{r-1}}$ for some numbers $c_0, c_1, \ldots, c_{r-1}$ (which are not necessarily the same for all edges), where $c_0 + c_1 + \cdots + c_{r-1} = 0$. Let $f_H(x_1, \ldots, x_n) = \prod_{e \in E} p_e$. Suppose that $d_1, \ldots, d_n$ are constants such that $\sum_{i=1}^n d_i = |E(H)|$, and $d = \max\{d_i : i = 1, \ldots, n\}$. If the coefficient of $\prod_{i=1}^n x_i^{d_i}$ in $f_H(x_1, \ldots, x_n)$ is not zero, then $\chi_l(H) \leq d + 1$.*

We now briefly describe how one can apply this theorem. Any hypergraph polynomial $f_H(x_1, \ldots, x_n)$ is homogeneous with degree $|E(H)|$. Each monomial can be obtained by taking one term (which represents a vertex) from each p_e, so it is natural to define an orientation of the hypergraph as setting one vertex from each edge as its source vertex. Suppose a monomial M is of the form $\prod_{i=1}^n x_i^{d_i}$. Then the powers d_i of x_i form a degree sequence $d_1, \ldots, d_n$. Each orientation where each vertex v_i is selected as source d_i times contributes a coefficient to M. So to compute the coefficients of M, we need to consider all possible orientations that have degree sequence $d_1, \ldots, d_n$ and sum up all the contributions.

One advantage of the hypergraph polynomial defined by Ramamurthi and West is that there is a combinatorial interpretation for the coefficients of the monomials called "balanced partitions" (which we will not describe here) that is a generalization of the interpretation for graph polynomials. So there is a possibility for combinatorial proofs of hypergraph list colouring results, potentially creating results similar to the applications of Alon and Tarsi. However, this interpretation is more complicated for hypergraphs, it requires that r be a prime number, and it is computationally more difficult to apply in general.

There are two applications using balanced partitions given in the paper by Ramamurthi and West [55]. One is for the Fano plane, which we will illustrate using Theorem 2.6 in Section 2.3.3. The other application is for a family of k-uniform hypergraphs with girth g and chromatic number i constructed by Kostochka and Nešetřil [45]. In this application, the authors chose a degree sequence where there is only one possible orientation that contributes to the monomial associated with that sequence. Therefore, in essence, any hypergraph polynomial that we define can be used to prove this result (as long as the coefficient of the source variable in the polynomial is nonzero in each edge).

Theorem 2.6 makes it possible to consider polynomials that are simpler to deal with computationally, even though they do not have any (obvious) combinatorial interpretations. For example, in the 3-uniform case, we may set $e_p = x_{i_0} + x_{i_1} - 2x_{i_2}$ for each edge $e = \{v_{i_0}, v_{i_1}, v_{i_2}\}$. Or in the 4-uniform case (which is not defined for the hypergraph polynomials of Ramamurthi and West), we may set $e_p = x_{i_0} - x_{i_1} + x_{i_2} - x_{i_3}$ for each edge $e = \{v_{i_0}, v_{i_1}, v_{i_2}, v_{i_3}\}$.

We note that there is a limitation as to what we can prove using any hypergraph polynomials in terms of list chromatic numbers. Since each monomial in a hypergraph polynomial has total degree $|E(H)|$, the average degree for a variable within any monomial is $|E(H)|/|V(H)|$. So within each monomial, there must be a variable of degree at least $\lceil |E(H)|/|V(H)| \rceil$. Therefore, using the polynomial method, we can only prove statements of the form $\chi_l(H) \leq k$ for $k \geq \lceil |E(H)|/|V(H)| \rceil + 1$. Anything lower than this will require us to use a different method of proof.

We will use the Fano plane as a small example of how to apply Theorem 2.6 in the next subsection, and present some computational results in the remaining subsections.

2.3.3 Example: Fano Plane

We now give an example showing that the Fano Plane is 3-list-colourable using the polynomial method. In [55], Ramamurthi and West used the Fano Plane as their example in illustrating the technique of balanced partitions. Here we give a polynomial argument for the same result. We label the 7 vertices as $\{1, 2, 3, 4, 5, 6, 7\}$, and the edges are

$$\{124, 235, 136, 157, 267, 347, 456\}.$$

See Figure 1.1 for reference. We define a hypergraph polynomial associated with the Fano Plane as follows:

$$\begin{aligned} f(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = &(x_1 + x_2 - 2x_4)(x_2 + x_3 - 2x_5)(x_1 + x_3 - 2x_6)\cdot \\ &(x_1 + x_5 - 2x_7)(x_2 + x_6 - 2x_7)(x_3 + x_4 - 2x_7)\cdot \\ &(x_4 + x_5 - 2x_6). \end{aligned}$$

Notice that the coefficients within each factor are $1, 1, -2$, which sum to 0. Consider the monomial $x_1^2 x_2^2 x_3^2 x_4$. We wish to determine whether the coefficient of this monomial in f is zero or not. To do this, we look at all orientations with the degree sequence $2, 2, 2, 1, 0, 0, 0$. For the edge 456, it must be the case that its source is 4, since vertices 5 and 6 have 0 in the degree sequence and cannot be chosen as sources. For the edges 157, 267 and 347, it must be the case that their sources are 1, 2 and 3 respectively. For the remaining three edges 124, 235 and 137, there are exactly two ways to orient them so that 1, 2 and 3 become the source exactly once: either with 1, 2 and 3 as sources respectively, or with 2, 3 and 1 as sources respectively. Notice that in all cases, the coefficient for the sources in f is always 1. Since there are two possible orientations with the degree sequence $2, 2, 2, 1, 0, 0, 0$ and both contribute a coefficient of 1 to the monomial $x_1^2 x_2^2 x_3^2 x_4$, we conclude that this monomial has coefficient 2 in f. Therefore, using Theorem 2.6, the Fano Plane is 3-list-colourable. Note that Lemma 3.1 implies that the Fano Plane is not 2-colourable, so it is not 2-list-colourable, hence its list chromatic number is in fact 3.

2.3.4 Computations with Hypergraph Polynomials

As the hypergraph grows, the hypergraph polynomial grows as well, and it becomes difficult to compute the coefficients by hand. So we turn to mathematical software such as Maple for help with the computations.

Before we get into the results for the computations, we first discuss some basic aspects of computational and complexity issues for colourings in general. Suppose we are given a hypergraph (or graph) H with n vertices and m edges. To prove that H is k-colourable, it is sufficient to present an actual colouring as a certificate to its k-colourability, and it is easy to check that such a colouring is proper (just go through each edge and make sure it is not monochromatic, and check that at most k colours are used). On the other hand, to prove that H is k-list-colourable, there is no obvious certificate for it. One could potentially use a brute force algorithm and create all possible k-list-assignments for vertices in H and check that each assignment has a proper colouring. However, the number of choices of list assignments is extremely large. For our hypergraph H, each assignment could use up to nk distinct colours, which we can map to a fixed set of nk colours. Then there are up to $\binom{nk}{k}^n$ configurations for the list assignments. In addition, given a list assignment, determining whether this assignment has a proper colouring or not is an NP-Complete problem in most cases. In fact, the problem is NP-Complete even for 3-uniform hypergraphs where all the lists are the same and have size two (see, e.g. [37]). Note that this is not true for graphs, since it is equivalent to determining if a graph is bipartite, and there are polynomial algorithms to solve this problem. If we use brute force to check if a list assignment has a proper colouring, this would take $k^n m$ steps (there are k^n possible colourings from each list, and for each colouring, we need to check that each of the m edges is not monochromatic). This gives a worst case running time of $\binom{nk}{k}^n k^n m$ to determine whether H is k-list-colourable or not.

This is where hypergraph polynomials are very helpful. By calculating the coefficient of a certain monomial in the polynomial and determining that it is nonzero, we have concrete evidence of the list-colourability of the hypergraph using Theorem 2.6. For a given hypergraph polynomial, one could at worst perform k^m multiplications to expand the polynomial and get the value of the desired coefficient, which is a lot better than the brute force algorithm described above. However, there are a couple of drawbacks. First, one needs to track the coefficients of an exponential number of monomials during the calculation. So in using the polynomial method, we could be very limited by the space constraint. Also, we gain no information if we find that the coefficient of a desired monomial is 0. We could use a different polynomial, but there are infinitely many polynomials to choose from. Even if the coefficient of the same monomial is 0 in every such polynomial, we still cannot conclude anything. In essence, the polynomial method cannot be used to prove that a hypergraph is not k-list-colourable. However, if we can use the method to produce a positive result (i.e. the coefficient of a desired monomial is nonzero), it would have been done in a more efficient way than using brute force. So there is an

i	1	4	7	1	2	3	1	2	3	3	2	1	10
j	2	5	8	5	6	4	4	5	6	5	4	6	11
k	3	6	9	9	7	8	7	8	9	7	9	8	12
l	10	10	10	11	11	11	12	12	12	13	13	13	13

Table 2.1: Projective plane of order 3 [18]

element of luck involved in choosing a "good" polynomial and monomial to work with.

For more discussions on the complexity of list-colouring problems and some variants, see [49] and [48]. For more on complexity theory in general, see [37] and [53].

Using a standard personal computer, we can only compute the coefficients for relatively small hypergraphs, since we have mentioned that the amount of memory needed is substantial. We will present some results on small projective planes obtained using Theorem 2.6 as computed by Maple. Note that these results can also be proved using the elementary techniques in the next major section, but we include them here as illustrations of how Theorem 2.6 can be applied.

2.3.5 Computational Results on Small Projective Planes

Projective planes have the same number of vertices as edges. Therefore, any chromatic polynomial associated with them has the property that the total degree of each term in the expansion of the polynomial is equal to the number of vertices. To prove that such a projective plane is 2-list-colourable, we just need to show that the coefficient of the term that is the product of all vertex variables is nonzero.

There are 13 vertices and edges in a projective plane of order 3. Only one projective plane of this order exists [18]. We list the edges as the columns in Table 2.1, and a representative diagram is shown in Figure 2.2. We define our hypergraph polynomial f to be the one where each edge (i, j, k, l) as listed in the table contributes a factor of $(x_i - 2x_j + 3x_k - 2x_l)$ to the polynomial. Note that the coefficients within each factor $1, -2, 3, -2$ add up to 0. Now Maple finds that the coefficient for the term $x_1 x_2 \cdots x_{13}$ in f is 124416, so using Theorem 2.6, we conclude that this projective plane is 2-list-colourable.

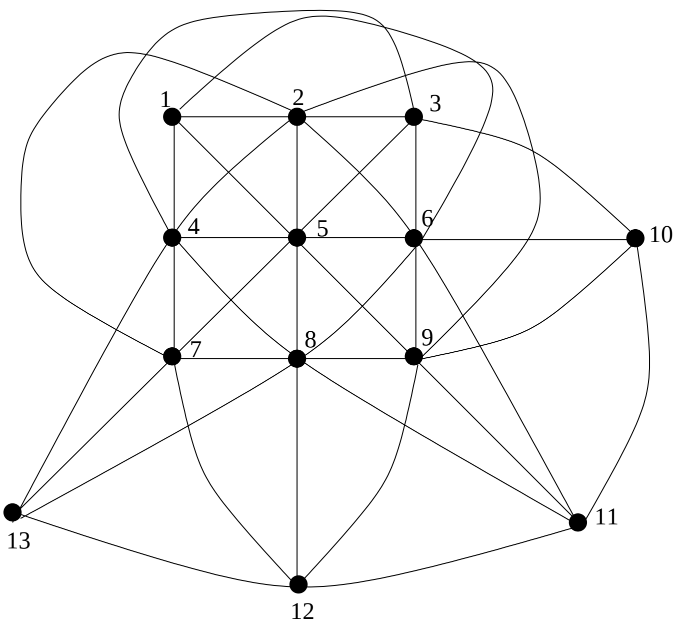

Figure 2.2: Diagram for the projective plane of order 3.

[Note: When we used the "natural" factor of $(x_i - x_j + x_k - x_l)$ instead of $(x_i - 2x_j + 3x_k - 2x_l)$, the coefficient for the term $x_1 x_2 \cdots x_{13}$ is 0. So this does not give us any new information.]

There are 21 vertices and edges in a projective plane of order 4, and there is also only one projective plane of this order [18]. We list the edges as the rows in Table 2.2. We define our hypergraph polynomial f to be the one where each edge (i, j, k, l, m) as listed in the table contributes a factor of $(x_i - 2x_j + 2x_k - 3x_l + 2x_m)$ to it. Note that the coefficients within each factor $1, -2, 2, -3, 2$ add up to 0. Now Maple finds that the coefficient of the monomial $x_1 x_2 \cdots x_{21}$ in f is -4894888400, so using Theorem 2.6, we conclude that this projective plane is 2-list-colourable as well. Later in Section 2.4.1, we will use elementary methods to show that all projective planes except for the Fano plane are 2-list-colourable.

2.4 Minimum Improper Colourings

We now turn to elementary methods in approaching the subject. To utilize the idea of minimum improper colourings, we need to set up each problem as follows: Let $H = (V, E)$ be a hypergraph. Suppose that H is not k-list-colourable. Then there

i	j	k	l	m
1	3	11	17	20
5	9	11	13	15
4	7	11	16	19
2	6	11	18	21
8	10	11	12	14
2	8	13	17	19
5	6	7	8	20
1	6	12	15	19
4	5	12	17	21
1	7	13	14	21
1	4	8	9	18
1	2	5	10	16
3	5	14	18	19
9	10	19	20	21
3	8	15	16	21
2	3	7	9	12
3	4	6	10	13
7	10	15	17	18
12	13	16	18	20
6	9	14	16	17
2	4	14	15	20

Table 2.2: Projective plane of order 4 [18]

exists a k-list-assignment L such that every L-colouring is improper, i.e. there exists at least one monochromatic edge for each possible L-colouring. An L-colouring c that contains the smallest number of monochromatic edges among all L-colourings is called a minimum improper colouring (MIC).

There are a few useful facts about MICs: Let $H, k \geq 2, L$ be as above, and let c be a MIC. Suppose that v is a vertex in a monochromatic edge e, $c(v) = 1$, and $x \in L(v) \setminus \{1\}$. Then,

Lemma 2.7. *Changing the colour of v to x will create at least one monochromatic edge f that is coloured x and $e \cap f = \{v\}$.*

Proof. Changing the colour of v to x destroys at least one monochromatic edge (i.e. edge e), so by the minimality of c, at least one monochromatic edge f must be created. This edge f is coloured x in the new colouring, it must contain v (since this is the only vertex whose colour has changed), and cannot contain other vertices of e (which still have colour 1 in the new colouring). □

Define $switch(c, v, x)$ to be the colouring obtained from c by changing the colour of v to x (always assuming that $x \in L(v)$). Define $E_c(v)$ to be the set of monochromatic edges that share v as a common vertex in the colouring c. Define $E_c(v, x)$ to be the set of edges that become monochromatic in $switch(c, v, x)$. For convenience, we will define $e_c(v, x)$ to be any edge in $E_c(v, x)$. Note that for any $e \in E_c(v)$ and any $f \in E_c(v, x)$, $e \cap f = \{v\}$. Then we have the following:

Lemma 2.8. *For any $x \in L(v) \setminus \{1\}$, $1 \leq |E_c(v)| \leq |E_c(v, x)|$.*

Proof. When we change the colour of v to x, $|E_c(v)|$ monochromatic edges were destroyed. So by minimality of c, at least $|E_c(v)|$ monochromatic edges must be created. □

Corollary 2.9. *For a MIC c, the number of monochromatic edges that share v as a common vertex is at most $\lfloor d_H(v)/k \rfloor$.*

Proof. Let m be the number of monochromatic edges that share v as a common vertex. For each $x \in L(v) \setminus \{c(v)\}$, $|E_c(v, x)| \geq m$. Since each list has size k, we must have

$$d_H(v) \geq m + \sum_{x \in L(v) \setminus \{c(v)\}} |E_c(v, x)| \geq m + (k-1)m = km,$$

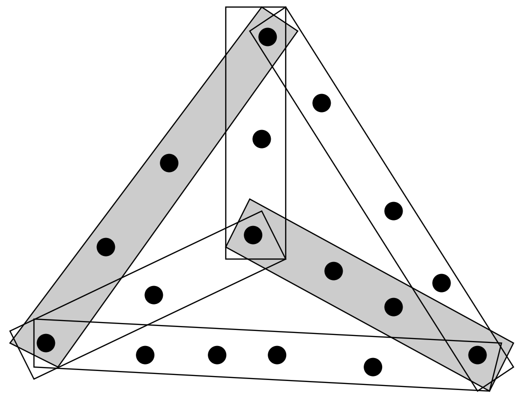

Figure 2.3: An almost-intersecting hypergraph.

hence $m \leq \lfloor d_H(v)/k \rfloor$. □

Finally, we note the following simple lemma.

Lemma 2.10. *If $|E_c(v)| = |E_c(v,x)|$, then switch(c,v,x) is also a MIC.*

2.4.1 Almost-Intersecting Hypergraphs

We call a hypergraph $H = (V, E)$ *almost-intersecting* if for every edge e, there is at most one edge f in E that is disjoint from e (i.e. $e \cap f = \emptyset$). Some small examples include a 3-uniform 6-cycle on 6 vertices: $V = \mathbb{Z}_6$, $E = \{0,1,2\} + \mathbb{Z}_6$ (which can be extended to larger cycles), or any K_4-like hypergraphs (e.g. Figure 2.3). Also, the class of projective planes is almost-intersecting (actually, they are "always intersecting").

The main theorem for this section is the following:

Theorem 2.11. *If the edges of an almost-intersecting hypergraph H all have cardinality at least 3 and H is 2-colourable, then H is 2-list-colourable.*

Proof. Suppose not. Then there exists a 2-list-assignment L such that all possible L-colourings contain monochromatic edges. Let c be a MIC with respect to L. Let e be a monochromatic edge which is, wlog, coloured 1. Let f be the edge in H that is disjoint from e, if it exists. (If f doesn't exist, then we may ignore case 1 below, and ignore mentions of f in case 2.) Note that all edges other than e and

f must intersect both e and f. Let $v \in e$ and suppose, without loss of generality, that $L(v) = \{1, 2\}$. We break into two cases:

Case 1: If f is monochromatic, then f must be coloured 2 in c, since $e_c(v, 2)$ intersects f. For any $v' \in e$, suppose $L(v') = \{1, x\}$. Then $e_c(v', x)$ intersects f, so $x = 2$. For any $v'' \in f$, suppose $L(v'') = \{2, y\}$. Then $e_c(v'', y)$ intersects e, so $y = 1$. Therefore, all vertices in e and f have the list $\{1, 2\}$. We can now obtain a proper L-colouring c' as follows: For vertices w not in $e \cup f$, if $L(w)$ contains a colour x that is neither 1 nor 2, then $c'(w) = x$. Since edges containing at least one of these vertices intersect e (all of whose vertices have lists $\{1, 2\}$), they must be properly coloured regardless of how e is coloured in c'. The remaining vertices all have lists $\{1, 2\}$ and they form a subhypergraph of H, which is 2-colourable by assumption. So assign c' to such a 2-colouring using the colours 1 and 2. We now have a proper L-colouring of H.

Case 2: Suppose f is properly coloured. Let $v_1, \ldots, v_k$ be the vertices in e, and let $L(v_i) = \{1, x_i\}$ and $e_i = e_c(v_i, x_i)$ for each $i = 1, \ldots, k$. Note that $k \geq 3$ since each edge contains at least 3 vertices by assumption. We want to show that $x_1 = x_2 = \cdots = x_k$. Among the edges $e_2, \ldots, e_k$, at most one is disjoint from e_1, say it is e_k. Then $e_2, \ldots, e_{k-1}$ intersect e_1 outside of e (since $e_i \cap e = \{v_i\}$), so $x_1 = x_2 = \cdots = x_{k-1}$. Since $k \geq 3$ and e_1 is disjoint from e_k, e_k must intersect e_{k-1}, hence $x_k = x_{k-1}$. So all vertices in e have the same list, say they are all $\{1, 2\}$. Now we can obtain a proper L-colouring c' in the same way as in case 1. Note that in this case, we do not change the colouring of f, so it is still properly coloured. The remaining edges intersect e, so they are properly coloured by the same arguments as in case 1. $\square$

Note that the two small examples we mentioned at the beginning of this section are both 2-colourable, so using this theorem, we can conclude that they are both 2-list-colourable.

2.4.2 Projective Planes and Symmetric BIBDs

Projective planes are almost-intersecting hypergraphs, so we can combine Lemma 2.4 and Theorem 2.11 to conclude the following:

Corollary 2.12. *A projective plane of order $n \geq 3$ is 2-list-colourable.*

All symmetric BIBDs are almost-intersecting hypergraphs as well, and for $\lambda \geq 2$, it is easy to show that they are 2-colourable. But there is a simpler proof of their 2-list-colourability.

Lemma 2.13. *A symmetric (v, k, λ)-BIBD with $\lambda \geq 2$ is 2-list-colourable.*

Proof. Let H be such a symmetric BIBD, and suppose that it is not 2-list-colourable. Then there exists a 2-list-assignment L with no proper L-colourings. Let c be a MIC, let e be a monochromatic edge coloured 1, and let $v \in e$ where $L(v) = \{1, 2\}$. Consider $f \in E_c(v, 2)$ and $c' := \mathit{switch}(c, v, 2)$. Now f must be monochromatic with colour 2 in c'. However, $|f \cap e| = \lambda \geq 2$, so there exists another vertex $w \in f \cap e$, which is coloured 1 in both c and c', contradicting the fact that f is monochromatic with colour 2 in c'. $\square$

In summary, we can conclude the following.

Theorem 2.14. *Every symmetric BIBD except for the Fano plane is 2-list-chromatic.*

Chapter 3

List-Colouring Small Steiner Triple Systems

For the next two chapters, we will focus on solving the list-colouring problem for Steiner triple systems. In this chapter, we will apply the techniques introduced in Chapter 2 to STSs of orders 9, 13 and 15. We will begin by presenting some known results in the (ordinary) colouring of Steiner triple systems in Section 3.1. This provides the groundwork for our investigation into list-colouring Steiner triple systems. In Section 3.2, we will use computations on hypergraph polynomials to solve the list-colouring problem for STS(9) and STS(13). The remainder of the chapter focuses on STS(15). Using minimum improper colourings, we will first prove that each STS(15) is 4-list-colourable in Section 3.3. We then use both techniques from Chapter 2 to show that STS(15) is "almost" 3-list-colourable: first using computations on hypergraph polynomials in Section 3.4, then using MICs in Section 3.5.

3.1 Colouring Steiner Triple Systems

For an admissible n, the chromatic spectrum, denoted Spec(n), is defined to be the set of values k such that there exists at least one STS(n) that is k-chromatic (this notation is used in [41]). The minimum and the maximum values in Spec(n) are denoted $\mathrm{Spec}_m(n)$ and $\mathrm{Spec}_M(n)$, respectively. Determining Spec(n) is still largely unsolved for most n. We can similarly define ListSpec(n) to be the set of values

k such that there exists at least one STS(n) that is k-list-chromatic, and define $\mathrm{ListSpec}_m(n)$ and $\mathrm{ListSpec}_M(n)$ in the same way.

We now present (or recall) some of the known results in colouring Steiner triple systems that we will use in the following two chapters. We begin with the following result (see Colbourn and Rosa [19]).

Lemma 3.1. *If H is a STS(n) where $n \geq 7$, then $\chi(H) \geq 3$.*

In particular, this implies that every non-trivial Steiner triple system must have list chromatic number at least 3. For any admissible n, the constructions by Bose and Skolem mentioned in Section 1.1 yield 3-chromatic triple systems (see e.g. [19]). We list this as a theorem.

Theorem 3.2. *For every admissible $n \geq 7$, there exists a 3-chromatic STS(n).*

Writing in terms of Spec(n) notation, this means that $\mathrm{Spec}_m(n) = 3$ for every non-trivial admissible n. Also, a result from de Brandes, Phelps and Rödl [24] mentioned in Section 1.2 implies that $\mathrm{Spec}_M(n) \to \infty$ as $n \to \infty$. In particular, this also implies that $\mathrm{ListSpec}_M(n) \to \infty$ as $n \to \infty$. Moreover, Phelps and Rödl [54] proved the following upper bound on $\mathrm{Spec}_M(n)$.

Theorem 3.3 (Phelps and Rödl [54])**.** *For each admissible n, $\mathrm{Spec}_M(n) \leq C\sqrt{n/\log n}$ for some constant C.*

For small Steiner triple systems, recall from Section 1.2 that Mathon, Phelps and Rosa [50] showed that all STS(n) where $7 \leq n \leq 15$ are 3-chromatic, i.e. $\mathrm{Spec}(n) = \{3\}$ for these values of n. For the remainder of this chapter, we will show that the corresponding list chromatic statement is true for $n \leq 13$, and "almost" true for $n = 15$. In particular, we will show that $\mathrm{ListSpec}(15) \subseteq \{3, 4\}$.

3.2 Computations on Small STSs

In this section, we will record the results of the computations on Steiner triple systems of orders 9 and 13.

There is only one Steiner triple system of order 9 [18], and its 12 blocks are listed as columns of Table 3.1. We define our hypergraph polynomial f to be the one where

i	1	4	7	1	2	3	1	2	3	1	2	3
j	2	5	8	4	5	6	5	6	4	6	4	5
k	3	6	9	7	8	9	9	7	8	8	9	7

Table 3.1: STS(9) [18]

each edge (i, j, k) as listed in the table contributes a factor of $(x_i + x_j - 2x_k)$. Now Maple finds that the coefficient of the monomial $x_1^2 x_2 x_4 x_5^2 x_6 x_7^2 x_8^2 x_9$ in f is 144, so by Theorem 2.6, this Steiner triple system is 3-list-colourable.

In fact, this polynomial result says something stronger than just STS(9) is 3-list-colourable. This result implies that if we give an arbitrary list L where vertex 3 has a list of size 1 (so the colour for this vertex is fixed), vertices 2, 4, 6 and 9 have lists of size 2 and the remaining vertices have lists of size 3, then there is always a proper L-colouring.

There are two Steiner triple systems of order 13 [18], and each of their 26 blocks are listed as rows of Table 3.2. Notice that the ratio between the number of blocks and the number of points is exactly 2. So in order to show that such systems are 3-list-colourable, we are forced to consider the monomial $M = x_1^2 x_2^2 \cdots x_{13}^2$. However, if we generate the hypergraph polynomial in the same manner as we did for the STS(9), the computation for the coefficient of the monomial takes too many resources and cannot be completed on our computer. So we need an approach that reduces the size of the polynomial.

The idea is that for a block (i, j, k), the factor that this block contributes to our hypergraph polynomial is of the form $ax_i + bx_j + cx_k$ for some coefficients a, b, c where $a + b + c = 0$. This 3-term factor could be reduced to a 2-term factor if we set the three coefficients to be a permutation of $1, -1, 0$. This means that the polynomial we get represents the graph polynomial for a graph obtained by dropping one vertex from each edge. If this graph is 3-list-colourable, then the original hypergraph is 3-list-colourable as well. Therefore, if we find that the coefficient of M is nonzero in this smaller polynomial, then we can conclude that our Steiner triple system is 3-list-colourable.

For each block (i, j, k), we ask Maple to randomly produce one of six possible factors: $x_i - x_j$, $x_j - x_i$, $x_i - x_k$, $x_k - x_i$, $x_j - x_k$ and $x_k - x_j$. Once all the factors are generated, it is relatively quick for Maple to compute the coefficient of M, so it

#1				#2		
0	1	2		0	1	2
0	3	4		0	3	4
0	5	6		0	5	6
0	7	8		0	7	8
0	9	10		0	9	10
0	11	12		0	11	12
1	3	5		1	3	5
1	4	7		1	4	7
1	6	8		1	6	8
1	9	11		1	9	11
1	10	12		1	10	12
2	3	9		2	3	9
2	4	5		2	4	5
2	6	10		2	6	10
2	7	12		2	7	11
2	8	11		2	8	12
3	6	11		3	6	11
3	7	10		3	7	12
3	8	12		3	8	10
4	6	12		4	6	12
4	8	9		4	8	9
4	10	11		4	10	11
5	7	11		5	7	10
5	8	10		5	8	11
5	9	12		5	9	12
6	7	9		6	7	9

Table 3.2: The two STS(13)s. [18]

can be repeated several times until we find a polynomial with a nonzero coefficient of M in a reasonable amount of time.

For system #1 in the table, Maple finds the following polynomial:

$$\begin{aligned} f =&(x_0 - x_2)(x_0 - x_4)(x_6 - x_0)(x_7 - x_0)(x_9 - x_0)(x_{12} - x_0)\cdot \\ &(x_3 - x_5)(x_7 - x_1)(x_1 - x_8)(x_9 - x_{11})(x_1 - x_{10})(x_2 - x_3)\cdot \\ &(x_5 - x_2)(x_{10} - x_2)(x_7 - x_{12})(x_{11} - x_8)(x_6 - x_{11})(x_{10} - x_7)\cdot \\ &(x_3 - x_{12})(x_6 - x_4)(x_9 - x_8)(x_{11} - x_4)(x_7 - x_5)(x_8 - x_5)\cdot \\ &(x_9 - x_5)(x_7 - x_9). \end{aligned}$$

The coefficient of M in f is -3, so this STS(13) is 3-list-colourable.

For system #2 in the table, Maple finds the following polynomial:

$$\begin{aligned} f =&(x_2 - x_1)(x_4 - x_0)(x_0 - x_5)(x_7 - x_8)(x_9 - x_{10})(x_{12} - x_{11})\cdot \\ &(x_5 - x_1)(x_4 - x_1)(x_6 - x_8)(x_1 - x_9)(x_{10} - x_1)(x_3 - x_9)\cdot \\ &(x_4 - x_2)(x_2 - x_{10})(x_{11} - x_2)(x_8 - x_{12})(x_6 - x_3)(x_3 - x_7)\cdot \\ &(x_{10} - x_3)(x_6 - x_{12})(x_8 - x_4)(x_{10} - x_{11})(x_7 - x_{10})(x_{11} - x_5)\cdot \\ &(x_5 - x_{12})(x_6 - x_9). \end{aligned}$$

The coefficient of M in f is also -3, so this STS(13) is also 3-list-colourable.

In summary, we have the following:

Theorem 3.4. *All Steiner triple systems of order n where $n \leq 13$ are 3-list-colourable.*

3.3 The 4-List-Colourability of STS(15)

There are 80 Steiner triple systems of order 15 [18]. We know from Section 3.1 that each of them is 3-colourable. In this section, we will show the following:

Theorem 3.5. *Each STS(15) is 4-list-colourable.*

Proof. Let H be an arbitrary STS(15), and suppose that it is not 4-list-colourable. So there exists a 4-list-assignment L such that there is no proper L-colouring. Let c be a MIC with respect to L. Since each vertex has degree 7, by Corollary 2.9, we know that each vertex is in at most one monochromatic edge of c. So all monochromatic edges in c are disjoint. Let e be one such edge, say it is coloured 1. There are 12 vertices outside of e, and we will call them spare vertices.

Let $e = \{v_1, v_2, v_3\}$, and let $L_i = L(v_i) \setminus \{1\}$ for $i = 1, 2, 3$. Note that each L_i has 3 colours, since $1 \in L(v_i)$ for each i. For each $x \in L_i$, $E_c(v_i, x)$ uses at least a pair of spare vertices of colour x. Since there are 12 spare vertices, there can be at most 6 distinct colours in $L_1 \cup L_2 \cup L_3$. We make the following two straightforward claims.

Claim 3.6. *If a colour x appears in at least 2 of L_1, L_2, L_3, then at least 3 spare vertices are coloured x in c.*

Proof. Say $x \in L_1 \cap L_2$. Then $e_c(v_1, x)$ and $e_c(v_2, x)$ each contains two spare vertices, both coloured x in c. However, they cannot contain the same pair of vertices since each pair can only be in one edge in a Steiner triple system, so at least 3 spare vertices are coloured x. □

Claim 3.7. *If a colour $x \in L_i$ is such that switch(c, v_i, x) is not a MIC, then at least 4 spare vertices are coloured x in c.*

Proof. Since *switch*(c, v_i, x) is not a MIC, $|E_c(v_i, x)| \geq 2$. Let $f, g \in E_c(V_i, x)$. We know that $f \cap g = \{v_i\}$ and all vertices in $(f \cup g) \setminus \{v_i\}$ are coloured x in c. So there must be at least 4 spare vertices coloured x in c. □

For a monochromatic edge e in a MIC c, we define

$$\mathcal{E}(c, e) = \{e\} \cup \bigcup_{v \in e, x \in L(v) \setminus \{c(v)\}} E_c(v, x).$$

For a colour x, we say that x is saturated in c by a set of edges E if each pair of vertices coloured x in c can be found in an edge of E. For example, suppose that the colour x appears in all of L_1, L_2, L_3 and there are exactly 3 spare vertices coloured x in c. Then each of $E_c(v_1, x)$, $E_c(v_2, x)$ and $E_c(v_3, x)$ contains at least one pair of spare vertices coloured x. But there are only 3 pairs of spare vertices coloured x, hence x is saturated in c by $\mathcal{E}(c, e)$.

Based on the remarks before Claim 3.6, we divide the rest of the proof according to the number of distinct colours in the set $L_1 \cup L_2 \cup L_3$.

Case 1: If there are exactly 6 colours, then exactly two spare vertices are assigned to each colour. At least one colour, say 2, must appear in at least two lists. By Claim 3.6, at least three spare vertices are coloured 2, which is a contradiction.

Case 2: Suppose that there are 5 colours in $L_1 \cup L_2 \cup L_3$. Then at least two colours, say 2 and 3, must appear in at least two lists. Each of the two colours uses up at least 3 spare vertices. The three remaining colours, say 4, 5 and 6, each uses at least 2 spare vertices. But there are only 12 spare vertices, so these must be exact, i.e. exactly 3 spare vertices are coloured 2 and 3, and exactly 2 for each colour 4, 5 and 6. This means that colours 4, 5 and 6 appear in only one of L_1, L_2, L_3 (Claim 3.6). Therefore, colours 2 and 3 are in all three lists. So we may assume that the lists are $L_1 = \{2, 3, 4\}$, $L_2 = \{2, 3, 5\}$ and $L_3 = \{2, 3, 6\}$. Notice that all colours from 1 to 6 are saturated in c by $\mathcal{E}(c, e)$. Consider the colouring $c' := switch(c, v_1, 4)$ and the edge $e' = e_c(v_1, 4)$. Since there are only two spare vertices coloured 4, c' must be a MIC. Let $w \in e' \setminus \{v_1\}$, and let $x \in L(w) \setminus \{1, 4\}$ for some colour x. Since c' is a MIC, $e'' = e_{c'}(w, x)$ exists, so some vertices are coloured x in c'. Note that $e'' \notin \mathcal{E}(c, e)$ since it does not contain any vertex coloured 1 in c. The possibilities for x are 2, 3, 5 and 6. However, all of these colours are saturated by $\mathcal{E}(c, e)$, i.e. there does not exist another pair of vertices in these colours that has not been used. Therefore, we cannot have e'', which is a contradiction.

Case 3: Suppose that there are 4 colours in $L_1 \cup L_2 \cup L_3$. Then at least one colour, say 2, must appear in all three lists. For the remaining three colours, there are two possibilities, and we divide this case into these two subcases.

Subcase 3a: The lists are, without loss of generality,

$$L_1 = \{2, 3, 4\}, L_2 = \{2, 3, 5\}, L_3 = \{2, 4, 5\}.$$

Since each of the four colours appears in at least two lists, each requires at least three spare vertices. But there are 12 spare vertices, so each colour receives exactly three spare vertices. Consider the colouring $c' = switch(c, v_1, 2)$ and the edge $e' = e_c(v_1, 2)$. Suppose that $e' = \{v_1, w_1, w_2\}$ where w_1, w_2 are coloured 2 in both c and c'. Note that c' is a MIC as there are only three spare vertices coloured 2. Consider the lists $L(w_1)$ and $L(w_2)$. For each $i = 1, 2$, $e_{c'}(w_i, x)$ exists for any $x \in L(w_i) \setminus \{2\}$, so x must be a colour that is used in c'. The possibilities for x

are 1, 3, 4 and 5. However, since the colour 1 is saturated by e in both c and c', this is not a valid option. Therefore, $L(w_1) = L(w_2) = \{2, 3, 4, 5\}$. There are three vertices coloured 3 in both c and c', so three pairs of these vertices are available. However, each of these four edges $e_c(v_1, 3)$, $e_c(v_2, 3)$, $e_{c'}(w_1, 3)$ and $e_{c'}(w_2, 3)$ requires a distinct pair of vertices coloured 3, which is a contradiction.

Subcase 3b: The lists are, without loss of generality,

$$L_1 = \{2, 3, 4\}, L_2 = \{2, 3, 4\}, L_3 = \{2, 3, 5\}.$$

The colours 2, 3 and 4 appear in at least two lists, so each colour receives at least three spare vertices. The colour 5 appears in only one list, so at least two spare vertices are coloured 5. So far, the colours of 11 spare vertices are determined. Let u be the remaining spare vertex. At least one of the colours 2 and 3 have exactly three spare vertices, say it is the colour 2. Consider the colouring $c' = \mathit{switch}(c, v_1, 2)$ and the edge $e' = e_c(v_1, 2)$. Let $w \in e' \setminus \{v_1\}$ where w is coloured 2 in both c and c'. Note that c' is a MIC as there are only three spare vertices coloured 2. Now $e_{c'}(w, x)$ exists for any $x \in L(w) \setminus \{2\}$, so x must be a colour that is used in c'. The possibilities for x are 1, 3, 4 and 5. A different colour (say the colour of u) cannot be considered since at least two spare vertices need to use that colour, yet the colour of only one spare vertex is not determined. In c' not counting the vertex u, the colour 1 is saturated by e, the colour 3 is saturated by $e_c(v_i, 3)$ for $i = 1, 2, 3$, and the colour 5 is saturated by $e_c(v_3, 5)$. Therefore, if any one of these three colours, say x, is in $L(w)$, then u must be coloured x in order to provide a pair for $e_{c'}(w, x)$. However, at least two of the colours 1, 3 and 5 are in $L(w)$, which is a contradiction.

Case 4: Suppose that there are only 3 colours in $L_1 \cup L_2 \cup L_3$. Therefore, we may assume that

$$L_1 = L_2 = L_3 = \{2, 3, 4\}.$$

Each of the three colours 2, 3 and 4 receives at least three spare vertices. We first want to obtain a c so that at least one of the three colours receives exactly three spare vertices. The only way that this does not happen is when all three colours receive four spare vertices each. Consider a vertex w coloured 2 in c. At least one colour x in $L(w)$ is not 3 nor 4. We replace c by $\mathit{switch}(c, w, x)$. We claim that this new colouring is still a MIC, and this is true provided that we did not create a new monochromatic edge when we switched the colour. This is certainly true if x is not 1 since w would be the only vertex coloured x. The only vertices coloured 1 are in e, so if x is 1, any new monochromatic edge must contain w and two vertices in e, which cannot happen.

Now we may assume that exactly three spare vertices w_1, w_2, w_3 are coloured 2 in c. At this point, we can also assume that if any colouring c' is a MIC, then vertices in any monochromatic edge e' in c' all have the same lists, which we may assume to be $\{1, 2, 3, 4\}$. Otherwise, we can apply the previous three cases. Since there are only three spare vertices coloured 2, $c_i = switch(c, v_i, 2)$ is a MIC for each $i = 1, 2, 3$. Therefore, $L(w_1) = L(w_2) = L(w_3) = \{1, 2, 3, 4\}$. Suppose that

$$e_1 = e_c(v_1, 2) = \{v_1, w_1, w_2\},$$

$$e_2 = e_c(v_2, 2) = \{v_2, w_2, w_3\},$$

$$e_3 = e_c(v_3, 2) = \{v_3, w_1, w_3\}.$$

We see that $e_{c_1}(w_1, 1)$ forces at least one spare vertex to have colour 1. The following 6 edges force at least four spare vertices to be coloured 3:

$$e_c(v_1, 3), e_c(v_2, 3), e_c(v_3, 3), e_{c_1}(w_1, 3), e_{c_2}(w_2, 3), e_{c_3}(w_3, 3).$$

Notice that these are distinct edges since each contains at least one unique vertex. Similarly, the following 6 edges force at least four spare vertices to be coloured 4:

$$e_c(v_1, 4), e_c(v_2, 4), e_c(v_3, 4), e_{c_1}(w_1, 4), e_{c_2}(w_2, 4), e_{c_3}(w_3, 4).$$

Again, these are distinct edges. But now, the colours of all spare vertices have been determined. So there must be exactly one spare vertex of colour 1 (call it u), four of colour 3, and four of colour 4. Four vertices of the same colour provide exactly 6 pairs, and for colours 3 and 4, each has at least 6 edges that want to claim a pair. So all colourings of the form $switch(c, v_i, x)$, $switch(c_i, w_i, x)$ where $i = 1, 2, 3$ and $x = 3, 4$ are MICs. Therefore, all spare vertices coloured 3 and 4 have the list $\{1, 2, 3, 4\}$. Since all STS(15) are 3-colourable, we may use colours 1, 2 and 3 to properly colour all vertices except for u, where we can assign to it a colour from $L(u)$ that is neither 1, 2 nor 3. This gives a proper L-colouring, which is a contradiction. □

3.4 The 3-List-Colourability of STS(15) I

In the following two sections, we will work toward the 3-list-colourability of Steiner Triple Systems of order 15. We will give two results using two different methods which show that STS(15)s are “almost” 3-list-colourable.

In this section, we give a computational result using hypergraph polynomials. Recall that there are 15 vertices and 35 edges in an STS(15). Since the number of edges is more than three times the number of vertices, we cannot use the polynomial method to prove that an STS(15) is 3-list-colourable. However, we can try to find the coefficient of monomials where 5 vertex variables have power 3 and the remaining 10 vertex variables have power 2. (Note that we cannot have fewer than 5 vertex variables with power 3, for otherwise the total degree of the hypergraph polynomial would be less than 35, the total number of edges.) Indeed, after having computed through all 80 STS(15)s using Maple, we conclude the following:

Theorem 3.8. *For any STS*(15) *on the vertex set* V*, there exists a set of five vertices* $W \subset V$ *such that every list-assignment* L *where vertices in* W *are given lists of size 4 and vertices not in* W *are given lists of size 3 has a proper* L*-colouring.*

The approach used in these computations is the same as the one we used in the computations for STS(13) in Section 3.2. For each block of 3 vertices, we randomly pick two vertices v, w to form a factor $x_v - x_w$. We form a random hypergraph polynomial this way, and ask Maple to find the coefficient of a fixed monomial where 5 of the vertex variables have power 3 and the rest have power 2. If the coefficient turns out to be 0, we repeat this process. The full results are listed in Appendix A. Here we give a sample of the results.

For STS#5, the list of blocks can be tabulated as follows:

`00000001111112222223333444455556666`
`13579bd3478bc3478bc789a789a789a789a`
`2468ace569ade65a9edbceddecbcbdeedbc`

In this case, the random hypergraph polynomial that Maple finds is

$$\begin{aligned} f(x) =&(x_2-x_1)(x_3-x_0)(x_6-x_5)(x_0-x_7)(x_a-x_0)(x_b-x_0)(x_e-x_0)\\ &\cdot(x_3-x_5)(x_6-x_4)(x_1-x_9)(x_a-x_1)(x_1-x_d)(x_e-x_c)(x_3-x_2)\\ &\cdot(x_2-x_5)(x_a-x_7)(x_8-x_9)(x_b-x_e)(x_c-x_2)(x_7-x_3)(x_8-x_c)\\ &\cdot(x_9-x_3)(x_3-x_a)(x_4-x_7)(x_8-x_4)(x_9-x_c)(x_a-x_4)(x_c-x_5)\\ &\cdot(x_b-x_8)(x_9-x_d)(x_a-x_5)(x_e-x_6)(x_6-x_d)(x_9-x_b)(x_c-x_6). \end{aligned}$$

And the coefficient of

$$x_0^3x_1^3x_2^3x_3^3x_4^3x_5^2x_6^2x_7^2x_8^2x_9^2x_a^2x_b^2x_c^2x_d^2x_e^2$$

in $f(x)$ is -1. Hence we may conclude that Theorem 3.8 holds for STS #5.

3.5 The 3-List-Colourability of STS(15) II

We now give a second theorem which shows that an STS(15) is "almost" 3-list-colourable. This uses the elementary method involving MICs from Section 2.4. Recall once again that each STS(15) is 3-colourable, and each vertex has degree 7. We will prove the following:

Theorem 3.9. *For any STS(15) and any 3-list-assignment L to the vertices, there is an L-colouring that contains at most one monochromatic edge.*

We will prove this theorem in two steps. We begin by showing the following.

Lemma 3.10. *For any STS(15) and any 3-list-assignment L to the vertices, there does not exist any MIC with respect to L which contains a pair of monochromatic edges that intersect at a vertex.*

Proof. Let H be an arbitrary STS(15), and let L be any 3-list-assignment. We may assume that there is no proper L-colouring, and let c be a MIC with respect to L such that there are two monochromatic edges $E_1 = E_c(v) = \{e_1, e_2\}$ that intersect at a vertex v. We will show that there exists a proper L-colouring, which would contradict the fact that c is a MIC.

We may assume that both e_1 and e_2 are coloured 1, and $L(v) = \{1, 2, 3\}$. Consider $E_2 = E_c(v, 2)$ and $E_3 = E_c(v, 3)$. By Lemma 2.8, both $|E_2|, |E_3| \geq 2$.

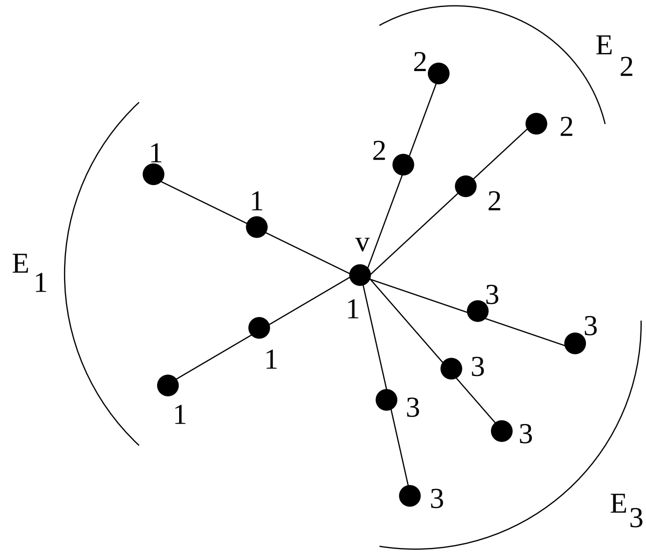

Figure 3.1: Diagram for Claim 3.11. The labels represent the colouring.

Note that v has degree 7 and edges in E_1, E_2, E_3 all contain v but are otherwise disjoint. Therefore, $|E_2|, |E_3| \leq 3$, and at most one of $|E_2|$ and $|E_3|$ is exactly 3. We make the following (generalized) claim:

Claim 3.11. *Let c be a MIC with two monochromatic edges joined at a vertex v where $L(v) = \{x, y, z\}$ and $c(v) = x$. Then $|E_c(v, y)| = |E_c(v, z)| = 2$.*

Proof. Let $E_2 = E_c(v, y)$, $E_3 = E_c(v, z)$, and assume that the colours x, y, z are 1, 2, 3 respective. Suppose, wlog, we have $|E_2| = 2$ and $|E_3| = 3$ (see Figure 3.1). Now the colour of every vertex in H is determined. We may assume that not all the lists are $\{1, 2, 3\}$, since any STS(15) is 3-colourable. So a colour 4 exists in the list of some vertex. Since c is a MIC, if this colour 4 is in say $v' \in e_1 \setminus \{v\}$, then $E_c(v', 4)$ cannot exist as no vertex is coloured 4 by c. Similarly, $c' := switch(c, v, 2)$ is a MIC, so no vertex in any edge in E_2 can have the colour 4 in its list. Therefore, some vertex w in some edge in E_3 must have colour 4. By recolouring w with 4 in c, we do not create a new monochromatic edge (since only one vertex has this colour), so it is still a MIC, and the new colouring satisfies $|E_2| = |E_3| = 2$. $\square$

Now there are two vertices in H that are not in any of the edges in E_1, E_2, E_3. We call them the spare vertices, and let them be s and t. (See Figure 3.2.) The colouring of all vertices except the spare vertices has been determined. Now we can claim the following:

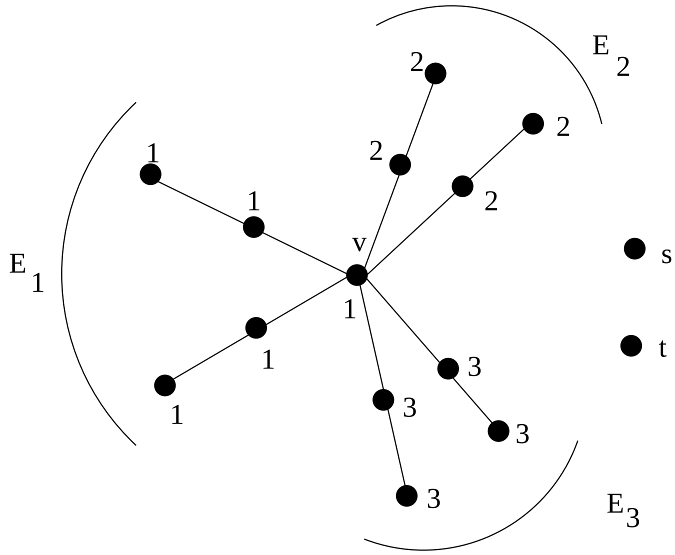

Figure 3.2: Diagram for Claim 3.12.

Claim 3.12. *The lists of all vertices in edges in E_1, E_2 and E_3 are $\{1, 2, 3\}$.*

Proof. We will first prove this for vertices in E_1. Then since both $switch(c, v, 2)$ and $switch(c, v, 3)$ are MICs, we may apply Claim 3.11 and the same argument in the rest of this proof to both of them. This gives the result for vertices in E_2 and E_3.

Suppose not all the vertices in E_1 have the list $\{1, 2, 3\}$. So the list for a vertex w in $V(E_1) \setminus \{v\}$ contains a colour other than 1, 2 or 3, say the colour 4. Since c is a MIC, $E_c(w, 4)$ must exist, and it must use at least two vertices of colour 4. But since all but the spare vertices have colours 1, 2 or 3, it must be the case that $|E_c(w, 4)| = 1$, and $e_c(w, 4) = \{w, s, t\}$. Therefore, $c(s) = c(t) = 4$. No other vertices in E_1 can have colour 4, since the colour 4 is saturated in c by $e_c(w, 4)$, and we may recolour such a vertex to 4 to get fewer monochromatic edges than c, which is a contradiction. Furthermore, for similar reasons, the lists of these vertices in E_1 cannot have any new colours. So it must be the case that all vertices in E_1 except w have the list $\{1, 2, 3\}$, and $L(w)$ contains 1, 4 and another colour, which must be either 2 or 3. Without loss of generality, $L(w) = \{1, 2, 4\}$. There are only four vertices coloured 2, namely vertices in E_2 except v. Now $E_2 \cup \{e_c(u, 2) : u \in V(E_1) \setminus \{v\}\}$ uses six pairs of vertices of colour 2, so the colour 2 is saturated in c by these edges.

Now $c' := switch(c, w, 4)$ is a MIC, so for any colour x in $L(s)$ or $L(t)$ that is not 4, $E_{c'}(s, x)$ or $E_{c'}(t, x)$ exists. So x must be a colour that is used by c, either 1, 2 or 3. But x cannot be 2, for the vertices coloured 2 in c (and therfore c') are saturated. Therefore, $L(s) = L(t) = \{1, 3, 4\}$. Now there are four vertices coloured 3, namely vertices in E_3 except v. But there are seven edges each requiring a pair of these vertices: the two edges in E_3, three edges $e_c(u, 3)$ where $u \in V(E_1) \setminus \{v, w\}$, $e_{c'}(s, 3)$ and $e_{c'}(t, 3)$. This is not possible. Therefore, all vertices in E_1 have the list $\{1, 2, 3\}$. $\square$

Let W be the set of vertices in H that have the list $\{1, 2, 3\}$. From the previous claim, we know that $|W| \geq 13$. Since any STS(15) is 3-colourable, there is a proper colouring for the partial STS(15) induced by W. If $|W| = 15$, then we are done. Otherwise, for the remaining one or two vertices, assign to each a colour in its list that is not 1, 2 or 3. This does not create a monochromatic edge since up to two such vertices have these new colours. Therefore, there is a proper L-colouring, contradicting the assumption that c is a MIC. $\square$

Proof of Theorem 3.9. Let H be an arbitrary STS(15), and let L be any 3-list-assignment. We may assume that there is no proper L-colouring, and let c be a MIC with respect to L. Using Lemma 3.10, we may assume that monochromatic edges in c are disjoint. To prove the theorem, it suffices to show that there cannot exist two disjoint monochromatic edges in c. We make a couple of observations first.

Claim 3.13. *Suppose that e is a monochromatic edge with colour 1, $v \in e$, $x \in L(v) \setminus \{1\}$ and $|E_c(v, x)| = 1$. Then $e_c(v, x)$ does not intersect any other monochromatic edge in c.*

Proof. Since $|E_c(v, x)| = 1$, $c' := switch(c, v, x)$ is a MIC. If $e_c(v, x)$ intersects any monochromatic edge f in c (which is also monochromatic in c'), then c' is a MIC that contains two monochromatic edges that intersect at a vertex. By Lemma 3.10, this is not possible. $\square$

Claim 3.14. *Suppose that e_1, e_2 are two monochromatic edges, $v \in e_1$, $w \in e_2$, $x \in L(v) \cap L(w)$ where x is not the colour of e_1 nor e_2. If $|E_c(v, x)| = |E_c(w, x)| = 1$, then $e_c(v, x)$ and $e_c(w, x)$ are disjoint.*

Proof. Otherwise, $c' := switch(switch(c, v, x), w, x)$ is a MIC with two monochromatic edges $e_c(v, x)$ and $e_c(w, x)$ intersecting at a vertex. Once again, by Lemma 3.10, this is not possible. $\square$

Suppose that there are at least two disjoint monochromatic edges, e_1 and e_2. We first consider the case where both e_1 and e_2 have the same colour, say 1. There are nine spare vertices in this case, so there can be at most four colours in the lists of vertices in e_1 and e_2 in addition to colour 1. Consider the twelve couples

$$\mathcal{C} = \{(v, x) : v \in e_1 \cup e_2, x \in L(v) \setminus \{1\}\}.$$

Since c is a MIC, for each $(v, x) \in \mathcal{C}$, $E_c(v, x)$ exists and uses at least a pair of spare vertices of colour x, since it is not possible for such an edge to intersect both e_1 and e_2. Any colour can appear in $\mathcal{C}$ at most six times. We wish to determine the minimum number of spare vertices of a colour that are needed based on how many times this colour appears in $\mathcal{C}$. This is recorded in the table below. For the cases where a colour appears in $\mathcal{C}$ at most five times, the way this is calculated is that if there are k spares of the same colour, then they can accommodate up to $\binom{k}{2}$ pairs, which is the maximum for the number of times that this colour can appear in $\mathcal{C}$. For the case when a colour (say x) appears in $\mathcal{C}$ six times, this colour appears in the lists of all six vertices in $e_1 \cup e_2$. Now four spare vertices of colour x can accommodate at most six pairs, so if there are exactly four spare vertices of colour x, then $|E_c(v, x)| = 1$ for each $v \in e_1 \cup e_2$. In particular, for a $v \in e_1$, there must exist a $w \in e_2$ such that $e_c(v, x)$ and $e_c(w, x)$ intersect at a spare vertex. By Claim 3.14, this is not possible. Therefore, at least five spare vertices are needed for a colour that appears six times in $\mathcal{C}$.

# times a colour appears in $\mathcal{C}$	# spares needed
1	2
2, 3	3
4, 5	4
6	5

Suppose that x_1, x_2, x_3, x_4 are the four possible colours in $\mathcal{C}$ (some may not exist). If $f(x_i)$ is the number of times x_i appears in $\mathcal{C}$ and $g(x_i)$ is the minimum number of spare vertices of x_i needed, then we must satisfy

$$f(x_1) + f(x_2) + f(x_3) + f(x_4) = 12, \tag{3.1}$$

and

$$g(x_1) + g(x_2) + g(x_3) + g(x_4) \leq 9. \tag{3.2}$$

But according to the table,

$$g(x_i) \geq f(x_i) - 1, \tag{3.3}$$

and equality holds only when $f(x_i)$ is five or six. But at least three of the four colours must satisfy $g(x_i) = f(x_i) - 1$, and the sum of their f-values must be at least 15. This would contradict (3.1). Therefore, we cannot have two disjoint monochromatic edges of the same colour.

Suppose now that e_1 is coloured 1 and e_2 is coloured 2. We define

$$\mathcal{C}_1 = \{(v, x) : v \in e_1, x \in L(v) \setminus \{1\}\},$$

$$\mathcal{C}_2 = \{(v, x) : v \in e_2, x \in L(v) \setminus \{2\}\},$$

and redefine $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$. If all the colours in $\mathcal{C}$ are not 1 nor 2, then the argument above follows and we are done. So we may assume that the colour 1 appears at least once in $\mathcal{C}_2$, or the colour 2 appears at least once in $\mathcal{C}_1$ (or both). Note that if $2 \in L(v)$ for some $v \in e_1$, then according to Claim 3.13, edges in $E_c(v, 2)$ can intersect e_2 provided that $|E_c(v, 2)| \geq 2$. In this case, each edge of $E_c(v, 2)$ uses a distinct spare vertex of colour 2. If $|E_c(v, 2)| = 1$, then $e_c(v, 2)$ uses two spare vertices of colour 2. So regardless of the size of $E_c(v, 2)$, at least two spare vertices are coloured 2. Similarly, if $1 \in L(w)$ for some $w \in e_2$, then at least two spare vertices are coloured 1.

Once again, using (3.1) and (3.2), we need at least three of the four possible colours in $\mathcal{C}$ to satisfy (3.3) with equality. In this case, we have additional possibilities of when this equality holds, and that is when the colour 2 is in $\mathcal{C}_1$ three times, or when the colour 1 is in $\mathcal{C}_2$ three times. If only one of the two occurs, say it is for the colour x_1, then at least two other colours x_2 and x_3 must satisfy (3.3) with equality where $f(x_2), f(x_3) \geq 5$. Then $f(x_1) + f(x_2) + f(x_3) \geq 13$, which is not possible. So it must be that both cases occur, which means that $f(1) = f(2) = 3$, and there is a colour x_3 where $f(x_3) \geq 5$ and the equality holds for (3.3). These three colours already satisfy (3.2) with equality, so a fourth colour cannot exist in $\mathcal{C}$. Therefore, $f(x_3) = 6$.

Now we have that the lists for all vertices in $e_1 \cup e_2$ are the same, say $\{1, 2, 3\}$. There are two spare vertices of colour 1, two spare vertices of colour 2, and five spare vertices of colour 3. Now we may assume that not all the lists of the spare vertices are $\{1, 2, 3\}$, since H is 3-colourable and we would be able to find a proper L-colouring. So there exists a colour 4 in one of the spare vertices that is currently coloured x. Switch the colour of this spare vertex to 4, which does not create any new monochromatic edge since no other vertices are coloured 4. But now there are not enough spare vertices of colour x to support the pairs required by $f(x)$, so this is a contradiction. We can now conclude that there exists an L-colouring that has at most one monochromatic edge. $\square$

Since any STS(15) is 3-colourable and Theorems 3.8 and 3.9 are good steps toward the 3-list-colourability of STS(15), we make the following conjecture.

Conjecture 3.15. *Each STS*(15) *is 3-list-chromatic.*

Chapter 4

Bounds on List-Colouring All Steiner Triple Systems

In this chapter, we will present three general bounds for list-colouring all Steiner triple systems, and discuss various implications on the list chromatic spectrum of STSs. We will use probabilistic methods to prove two different bounds: In Section 4.1, we will prove that large Steiner triple systems do not have constant list chromatic numbers, i.e. there is a growing lower bound on the minimum list chromatic number for large STS. In Section 4.2, we give an upper bound on the list chromatic number of an STS based on its chromatic number. These results will show that in general, the list chromatic numbers of Steiner triple systems behave differently from the chromatic numbers. Finally, in Section 4.3, we will give a simple upper bound on $\mathrm{ListSpec}_M(n)$, the highest possible list-chromatic number that an $\mathrm{STS}(n)$ can have, using a simple application of minimum improper colourings.

4.1 A Lower Bound on Large STS

In [4], Alon proved that the list chromatic number for a graph of minimum degree d is at least $(\frac{1}{2} - o(1)) \log_2 d$. So graphs with large minimum degree have large list chromatic number. This is not true for hypergraphs in general, however. Consider the 3-uniform hypergraph H_n obtained by adding a vertex v to a graph K_n and adding v to each edge in K_n. As $n \to \infty$, the minimum degree of H_n (which is $n-1$) also approaches ∞. Also, H_n is 2-list-colourable, since we can give v any colour in its list, and for each of the remaining vertices, we can give it a colour in its list other than the one given to v. So it is possible for a hypergraph to have an arbitrarily large minimum degree while maintaining constant list chromatic number.

This is not true for Steiner triple systems, however. In this section, we will use probabilistic methods to prove the following theorem.

Theorem 4.1. *For every integer s there exists an $n_0 = n_0(s)$ such that every STS(n) with $n \geq n_0$ has list chromatic number greater than s.*

The value of $n_0(s)$ that we use for this theorem has order around s^{6s}. Using this value, the theorem gives us the following lower bound on $\mathrm{ListSpec}_m$.

Corollary 4.2. *There exists a (small) constant $c > 0$ such that for any admissible n,*

$$ListSpec_m(n) \geq \frac{c \log n}{\log \log n}.$$

The rest of the section is organized as follows: We first give an outline for the proof of Theorem 4.1 in Section 4.1.1, and then prove this theorem in Section 4.1.2. We then prove Corollary 4.2 in Section 4.1.3. Finally, we discuss the implications of these results in Section 4.1.4.

4.1.1 Idea of the Proof for Theorem 4.1

The proof proceeds in three steps. Let H be a STS(n) where $n \geq n_0$. First we choose a small subset S of vertices such that each vertex v in H has about the same number of pairs x, y in its neighbourhood (i.e. pairs such that vxy is an edge of H) such that both x and y are in S. In the second step, we assign lists of size s to the vertices of S such that no vertex has too many pairs in its neighbourhood that are both in S and get the same colour in their lists. Then in the third step we assign lists to the rest of the vertices so that for each colouring c of S from its lists, there exists a vertex v of $V(H) \setminus S$ whose list is contained in the set of colours *forbidden* by c at v (where a colour i is called forbidden if both vertices of some pair in the neighbourhood of v are assigned colour i by c). Thus with such a list assignment, no colouring c of S could be extended to a proper colouring of the whole of H, implying that H has list chromatic number greater than s.

The way we ensure that suitable lists can be chosen for $V(H)\backslash S$ in the third step is as follows. For each colouring c of S, since there is only a constant (M) number of colours, many pairs of vertices in S are monochromatic. Since H is an STS, each pair is in the neighbourhood of some vertex v. Since by Step 2 no vertex can have too many monochromatic pairs of the same colour in its neighbourhood, this implies that many vertices each have many colours appearing on monochromatic pairs in their neighbourhoods. In other words, for many vertices, each of them has many colours forbidden by c. This implies that only a very small proportion of the total possible number of list assignments to the vertices $V(H) \setminus S$ avoid assigning to any vertex a list that is contained in its set of forbidden colours under c. This proportion is so small that even when we take the union over all possible colourings c of S (recalling S is a small set), the number of such list assignments that could extend some colouring c is smaller than the total number of list assignments possible for $V(H)\backslash S$. Therefore we can choose a suitable list assignment as described above.

4.1.2 The Proof of Theorem 4.1

We first state the Chernoff bounds, which we use extensively in the proof.

Theorem 4.3 (Chernoff bounds [17])**.** *Let X be the sum of n independent binary random variables $X_1, \ldots, X_n$, and let μ be the expected value of X (which is the sum of $E[X_i]$). Let $0 < \delta \leq 1$. Then*

1. $\mathbb{P}[X < (1-\delta)\mu] < e^{-\mu\delta^2/2}$;
2. $\mathbb{P}[X > (1+\delta)\mu] < e^{-\mu\delta^2/4}$; *and*
3. $\mathbb{P}[|X-\mu| > \delta\mu] < 2e^{-\mu\delta^2/4}$.

Let s be given, let $M = 36s^3$, and set

$$n_0 = 12^9 (Ms\log s)^3 \binom{M}{s}^3. \tag{4.1}$$

(Throughout this section, the notation log indicates the natural logarithm). We will show that there exists a list assignment L of the vertices of H, where each vertex receives s colours taken from the set $\{1, \ldots, M\}$, such that there is no proper colouring of H from the lists L. We mention that here we do not attempt to optimize the constants in the definition of n_0, but note that the order of magnitude of n_0 is about s^{6s}.

Corresponding to Step 1 in the previous section, we begin by choosing a special subset S of vertices of H. We want S to have the following two properties.

Property 1. *The size of S is between $\frac{4M}{s}\sqrt{n\log n}$ and $\frac{12M}{s}\sqrt{n\log n}$.*

Property 2. *Every vertex in H has at least $\frac{16M^2\log n}{s^2}$ and no more than $\frac{48M^2\log n}{s^2}$ pairs of vertices in its neighbourhood in S.*

We now proceed to prove that there exists an S with these two properties. We put each vertex of H into S randomly and independently with probability $p = \frac{8M}{s}\sqrt{\frac{\log n}{n}}$. Then the expected size of S is $\frac{8M}{s}\sqrt{n\log n}$.

Claim 4.4. *The probability that S satisfies property 1 is greater than 1/2.*

Proof. By part 3 of the Chernoff bound (Theorem 4.3), using $\delta = 1/2$, we find that

$$\begin{aligned}\mathbb{P}\left[\frac{4M}{s}\sqrt{n\log n} \le |S| \le \frac{12M}{s}\sqrt{n\log n}\right] &> 1 - 2\exp\left(-\frac{8M\sqrt{n\log n}}{16s}\right)\\ &= 1 - 2\exp\left(-36s^2\sqrt{n\log n}/2\right)\\ &> \frac{1}{2},\end{aligned}$$

where the last inequality is true since $n \ge n_0 \ge 2$ implies that $\exp(-36s^2\sqrt{n\log n}/2) < 1/4$. $\square$

Claim 4.5. *The probability that S satisfies property 2 is more than 1/2.*

Proof. Let v be a vertex of H. Since H is an STS, the neighbourhood of v consists of $\frac{n-1}{2}$ disjoint pairs of vertices. Let X_v denote the random variable that counts the number of pairs xy in the neighbourhood of v that are both in S. Since each vertex of H is in S with probability p, the expected value of X_v is

$$E[X_v] = \frac{n-1}{2}p^2 = \frac{32(n-1)M^2\log n}{s^2 n}.$$

For a fixed vertex v, we find that using part 1 of Chernoff bounds with $\delta = 1/2$ and the fact that $n \ge 3$, we get

$$\mathbb{P}\left[X_v < \frac{16M^2\log n}{s^2}\right] \le \mathbb{P}\left[X_v < \frac{32(1-\frac{1}{2\sqrt{2}})(n-1)M^2\log n}{s^2 n}\right] < e^{-E[X_v]/16}.$$

Using part 2 of the Chernoff bounds, we get

$$\mathbb{P}\left[X_v > \frac{48M^2 \log n}{s^2}\right] \leq \mathbb{P}\left[X_v > \frac{48(n-1)M^2 \log n}{s^2 n}\right] < e^{-E[X_v]/16}.$$

Combining the two we get

$$\begin{aligned}
\mathbb{P}\left[\frac{16M^2 \log n}{s^2} \leq X_v \leq \frac{48M^2 \log n}{s^2}\right] &> 1 - 2e^{-E[X_v]/16} \\
&= 1 - 2\exp\left(-\frac{2(n-1)M^2 \log n}{s^2 n}\right) \\
&> 1 - 2\exp\left(-\frac{M^2 \log n}{s^2}\right).
\end{aligned}$$

Therefore the probability that every vertex v has at least $\frac{16M^2 \log n}{s^2}$ pairs in its neighbourhood that are in S, and at most $\frac{48M^2 \log n}{s^2}$ such pairs, is bounded below by

$$\begin{aligned}
1 - 2ne^{-\frac{M^2 \log n}{s^2}} &= 1 - 2n(n^{-\frac{M^2}{s^2}}) \\
&= 1 - 2n(n^{-36^2 s^4}) \\
&> \frac{1}{2},
\end{aligned}$$

since $n \geq n_0 \geq 2$. □

Combining Claims 4.4 and 4.5, we get that

Claim 4.6. *There exists an $S \subseteq V(H)$ such that Properties 1 and 2 hold.*

We now fix an S that satisfies both Properties 1 and 2. For a vertex v we denote by $d_S(v)$ the number of pairs in the neighbourhood of v that are both in S. Then by Property 2 we have

$$\frac{16M^2 \log n}{s^2} < d_S(v) < \frac{48M^2 \log n}{s^2}.$$

Now for Step 2, we choose list assignments for the vertices in S. We claim the following.

Claim 4.7. *There exists a choice of lists for vertices in S such that for each vertex v in H and each colour i, the number of pairs of vertices of S in the neighbourhood of v that have colour i in both of their lists is at most $2d_S(v)\frac{s^2}{M^2}$.*

Proof. We will also do this randomly. We give each vertex of S one of the s-subsets of $\{1,\ldots,M\}$ uniformly at random. Thus the probability that a vertex of S receives colour i in its list is $\frac{\binom{M-1}{s-1}}{\binom{M}{s}} = \frac{s}{M}$. Let v be a vertex and i a colour. Let the random variable Y_v^i count the number of pairs xy in the neighbourhood of v such that both x and y are in S and receive the colour i in their lists. Then the expected value of Y_v^i is $d_S(v)\frac{s^2}{M^2}$. Thus by part 2 of Chernoff and property 2 of S, we find that the probability that there are more than $2d_S(v)\frac{s^2}{M^2}$ pairs incident to v that get colour i in their lists is at most

$$e^{-d_S(v)\frac{s^2}{4M^2}} < e^{-\frac{16\log n}{4}}.$$

Thus the probability that some vertex v has more than $2d_S(v)\frac{s^2}{M^2}$ neighbour pairs getting colour i for some i is bounded above by

$$Mne^{-\frac{16\log n}{4}} = Mn^{-3} < 1,$$

where the last inequality holds since $n \geq n_0 > 36s^3 = M$. Therefore, at least one list assignment would satisfy the conclusion of this claim. $\square$

We fix a choice of lists L for the vertices in S that satisfies the conditions in Claim 4.7. Let $d^i(v)$ denote the number of pairs in the neighbourhood of v that have colour i in both of their lists. From the claim, we know that $d^i(v) \leq 2d_S(v)\frac{s^2}{M^2}$ for any choice of vertex v and colour i.

Now our task for Step 3 of the argument is to choose lists for the remaining $n-|S|$ vertices so that there is no proper list colouring in H with these lists. Let c be a partial colouring that assigns to each vertex of S a colour from its list. Let us say that a colour i is forbidden at v by c if both vertices of some pair in the neighbourhood of v are coloured i by c. We denote by $F_c(v)$ the number of colours forbidden at v by c. We claim the following.

Claim 4.8. *For each choice of c, there are at least $\frac{n}{72s^2}$ vertices v such that $F_c(v) \geq s$.*

Proof. We first fix a partial colouring c. Then the total number of pairs of vertices in S that are monochromatic pairs under c is $B = \sum_{i=1}^{M}\binom{C(i)}{2}$, where $C(i)$ denotes the number of vertices in S that receive colour i under c. We will bound B in two ways.

First, we give a lower bound for B. Since $\sum_{i=1}^{M} C(i) = |S|$, the value of B is minimized when all $C(i)$ are as close in size as possible. Therefore, using property 1 for S, we have

$$\begin{aligned} B &\geq M\binom{\lfloor\frac{|S|}{M}\rfloor}{2} \geq \frac{M}{2}\left(\frac{|S|}{M}-1\right)^2 - \frac{|S|}{2} \\ &\geq \frac{|S|^2}{2M} - \frac{3|S|}{2} > \frac{|S|^2}{4M} > \frac{4Mn\log n}{s^2}, \end{aligned} \tag{4.2}$$

where the second last inequality uses the fact that $n \geq n_0 \geq s^4$, which implies

$$|S| \geq 4M\sqrt{n\log n}/s > 6M.$$

Moreover, note that since H is a Steiner triple system, each of these monochromatic pairs lies in the neighbourhood of (exactly) one vertex v.

Now we give an upper bound for B. We define the real number α such that the number of vertices v for which $F_c(v) < s$ is $(1-\alpha)n$. Then note that each of the remaining αn vertices trivially has $F_c(v) \leq M$. Also, if $F_c(v) < s$ then certainly v has at most $d^i(v)$ monochromatic pairs in colour i for each $i \in F_c(v)$, so v has in total at most $2sd_S(v)\frac{s^2}{M^2}$ monochromatic pairs in its neighbourhood altogether under c. Similarly, if $F_c(v) < M$, then v has at most $2Md_S(v)\frac{s^2}{M^2}$ monochromatic pairs in its neighbourhood under c.

Therefore, using property 2 of S, we have

$$\begin{aligned} B &\leq (1-\alpha)n\left(2sd_S(v)\frac{s^2}{M^2}\right) + \alpha n\left(2Md_S(v)\frac{s^2}{M^2}\right) \\ &= 2nd_S(v)\frac{s^2}{M^2}((1-\alpha)s+\alpha M) \\ &\leq 2n(\frac{48M^2\log n}{s^2})\frac{s^2}{M^2}((1-\alpha)s+\alpha M) \\ &= 96n\log n((1-\alpha)s+\alpha M). \end{aligned} \tag{4.3}$$

Combining the two inequalities (4.2) and (4.3), we get

$$\frac{4Mn\log n}{s^2} < 96n\log n(s-\alpha s+\alpha M).$$

This implies

$$\frac{M}{s^2} < 24(s-\alpha s+\alpha M).$$

Recalling that $M = 36s^3$, we get

$$36s < 24(s + \alpha(M - s))$$

which implies

$$\alpha > \frac{12s}{24(M-s)} = \frac{s}{72s^3 - 2s} > \frac{1}{72s^2}.$$

Therefore for each colouring c of S, there are at least $\frac{n}{72s^2}$ vertices v such that $F_c(v) \geq s$. $\square$

The number of colourings c of S is $s^{|S|}$. By Claim 4.8, each one results in at least $\frac{n}{72s^2} - |S| \geq \frac{n}{144s^2}$ vertices v of $V(H) \setminus S$ each with at least s forbidden colours (the inequality is true since $n \geq n_0 \geq 144^3 \cdot M^4$ implies that $|S| \leq 12\frac{M}{s}\sqrt{n \log n} < n/144s^2$). For each colouring c, there are therefore at most $(\binom{M}{s} - 1)^{\frac{n}{144s^2}} \binom{M}{s}^{n-|S|-\frac{n}{144s^2}}$ possible list assignments to the vertices of $V(H) \setminus S$ that could have a proper colouring that extends c, since if a vertex v had a list consisting of s forbidden colours then no proper colouring could exist. Thus the number of list assignments to $V(H) \setminus S$ for which some colouring c of S could be extended to a proper colouring of H is at most

$$s^{|S|}\binom{M}{s}^{n-|S|}\left(1 - \frac{1}{\binom{M}{s}}\right)^{\frac{n}{144s^2}} < \binom{M}{s}^{n-|S|} s^{|S|} \exp\left(-\frac{n}{144s^2\binom{M}{s}}\right), \tag{4.4}$$

which is less than the total number $\binom{M}{s}^{n-|S|}$ of list assignments for $V(H) \setminus S$ provided that

$$s^{|S|} \exp\left(-\frac{n}{144s^2\binom{M}{s}}\right) < 1.$$

This is true if and only if

$$|S| \log s - \frac{n}{144s^2\binom{M}{s}} < 0.$$

Using property 1 of S, this is true when

$$\frac{12M}{s}\sqrt{n \log n} \log s < \frac{n}{144s^2\binom{M}{s}}.$$

Squaring both sides and rearranging the inequality, we see that this holds if and only if

$$\left(12^3 M s \log s \binom{M}{s}\right)^2 < \frac{n}{\log n}.$$

Taking $K_0 = 12^3 M s \log s \binom{M}{s}$, we simplify this to the inequality

$$K_0^2 < \frac{n}{\log n}.$$

Since $n/\log n$ is an increasing function, if this inequality is true for the lower bound of n, it is true for all n. But since $n \geq n_0 = K_0^3$ and $3 \log K_0 < K_0$, indeed we have $K_0^2 < n/\log n$. Therefore, (4.4) is true, and there exists an assignment of lists to $V(H) \setminus S$ such that no colouring c of S can be extended to all of H. Therefore the list chromatic number of H is greater than s. $\square$

4.1.3 Proof of Corollary 4.2

Using (4.1), Theorem 4.1 shows that when

$$n \geq 12^9 (36 s^4 \log s)^3 \binom{36 s^3}{s}^3,$$

any STS(n) is not s-list-colourable. We may crudely estimate this as

$$n \geq c_0 s^{24s} \tag{4.5}$$

for some large constant c_0. This is equivalent to

$$\begin{aligned}
\log n &\geq 24 s \log c_0 s \\
&= c_1 s + 24 s \log s \\
&= s \log s (c_1/\log s + 24),
\end{aligned}$$

where $c_1 = 24 \log c_0$. If we fix n, then for $c = 1/(c_1 + 24)$ and substituting $s = c \log n/\log\log n$, we have

$$\begin{aligned}
s \log s (c_1/\log s + 24) &\leq s \log s (c_1 + 24) \\
&= \frac{\log n}{\log\log n} \log\left(\frac{c \log n}{\log\log n}\right) \\
&= \frac{\log n}{\log\log n} (\log\log n + \log c - \log\log\log n) \\
&\leq \log n,
\end{aligned}$$

since $c < 1$ implies that $\log c < 0$. Therefore, when $s = c \log n/\log\log n$, (4.5) holds. Hence, $\mathrm{ListSpec}_m(n) \geq c \log n/\log\log n$. $\square$

4.1.4 Comparing χ and χ_l for STS

Theorem 4.1 raises an interesting question. Recall that $\mathrm{Spec}_m(n) = 3$ for every admissible $n \geq 7$. However, Corollary 4.2 tells us that $\mathrm{ListSpec}_m(n) \to \infty$ as $n \to \infty$. So this provides the first confirmed instance that we have seen where the chromatic number of an STS differs from its list chromatic number. So we can ask, when do the chromatic number and the list-chromatic number differ for an STS? In particular, what is the smallest admissible n such that there exists an STS(n) whose list-chromatic number is strictly greater than its chromatic number? Let N be this number.

Based on the results in Sections 2.3.3 and 3.1, we see that $N \geq 15$. If Conjecture 3.15 about Steiner triple systems of order 15 is true, then we would have $N \geq 19$, the next admissible order. From the proof of Theorem 4.1, we see that when we plug in $s = 3$, this gives

$$N < 5.865692 \cdot 10^{19}.$$

However, we believe that the true value for N should be closer to 19 rather than this astronomical number.

4.2 An Upper Bound on χ_l in Terms of χ

Here we give a bound that relates the list chromatic number of a Steiner triple system to its chromatic number, showing that they cannot differ wildly, i.e. by at most a factor of $\log n$. This result will produce various bounds on $\mathrm{ListSpec}_m$ and $\mathrm{ListSpec}_M$ in general. In particular, we will show that our bound on $\mathrm{ListSpec}_m$ in Corollary 4.2 is not too far from the truth. The idea for the proof of this lemma comes from Exercise 2.7.9 in [7], about proving that a bipartite graph with n vertices has list-chromatic number at most $\log_2 n$.

Lemma 4.9. *Let n be given, and suppose H is a STS(n) with chromatic number k. Then H is $\lceil k(\log n + 1)\rceil$-list-colourable.*

Proof. Fix a vertex colouring c of H with k colours. Let lists of length $\lceil k(\log n+1)\rceil$ be assigned to each vertex of H. Let U denote the union of all the lists, then certainly $|U| \geq \lceil k(\log n + 1)\rceil$. We partition U randomly into k subsets $U_1, \ldots, U_k$ by putting each $u \in U$ into U_i randomly and independently with probability $\frac{1}{k}$.

We estimate the probability that a vertex v that is coloured i by c has a colour from U_i in its list. Since each element in its list is in U_i with probability $\frac{1}{k}$, the probability that no colour in v's list is in U_i is

$$\left(1 - \frac{1}{k}\right)^{\lceil k(\log n+1)\rceil} < e^{-\log n - 1}.$$

Thus the probability that some vertex v fails to have an element of U_i in its list, where i is its colour under c, is at most

$$ne^{-\log n-1} < \frac{1}{e} < 1.$$

Thus there exists a partition of U such that each vertex v has a colour in its list from U_i where $c(v) = i$. Then we can give H a colouring by giving each v a colour in its list from U_i where $c(v) = i$. We claim that no edge is monochromatic under this colouring. For suppose an edge is monochromatic in colour j, and let i be such that $j \in U_i$. Then since the only vertices that get any colour at all in U_i, hence in particular j, are all coloured i by c, it must be true that this edge is monochromatic in colour i under c. But this contradicts the fact that c is a colouring of H. Thus this colouring is a valid list colouring of H, and the proof is complete. □

Together with Theorem 3.2 which states that for every admissible $n \geq 7$ there exists a 3-colourable STS(n), the above result implies that for every admissible $n \geq 7$ there exists an STS(n) whose list chromatic number is at most $\lceil 3(\log n+1)\rceil$. This implies the following.

Corollary 4.10. *For every admissible n, ListSpec$_m(n) \leq \lceil 3(\log n + 1)\rceil$.*

Together with Corollary 4.2, we have pinned down the order of ListSpec$_m$ to somewhere between $\log n / \log\log n$ and $\log n$.

Furthermore, recall that Theorem 3.3 by Phelps and Rödl [54] gives an upper bound Spec$_M(n) \leq C\sqrt{n/\log n}$ for some constant C, where C is large. When combined with Lemma 4.9, this gives the following.

Corollary 4.11. *There exists a constant C such that for every admissible n, ListSpec$_M(n) \leq C\sqrt{n \log n}$.*

Note that for relatively "small" n, we can obtain a better upper bound on ListSpec$_M(n)$ using a simple argument based on minimum improper colourings. We will give this proof in the next section.

4.3 An Upper Bound on $\mathbf{ListSpec}_M(n)$

Here we give a simple proof for an upper bound on the list chromatic number of all Steiner Triple Systems that is better than the greedy upper bound (Lemma 2.3), and better than Corollary 4.11 when n is small. Recall that STS(n) exists only for $n \equiv 1, 3 \bmod 6$. For $t \geq 1$, if $n = 6t+1$, then each vertex has degree $3t$, so by the greedy upper bound, its list chromatic number is at most $3t+1$. If $n = 6t+3$, then each vertex has degree $3t+1$, so its list chromatic number is at most $3t+2$. We now give a proof that improves this upper bound by approximately two-thirds.

Lemma 4.12. *For each $t \geq 1$, if H is an STS(n) where $n = 6t+1$, then $\chi_l(H) \leq 2t+1$. If $n = 6t+3$, then $\chi_l(H) \leq 2t+2$.*

Proof. We will only prove the case for $n = 6t+1$, as the case for $n = 6t+3$ is similar. Suppose that H is not $(2t+1)$-list-colourable. Then there exists a $(2t+1)$-list-assignment L such that H is not properly L-colourable. Let c be a MIC with respect to L, and let $e = \{v_1, v_2, v_3\}$ be a monochromatic edge with colour 1. There are $6t-2$ vertices remaining, which we call spare vertices. There are $2t$ colours in $L_1 = L(v_1) \setminus \{1\}$, and each such colour x must take up at least 2 spare vertices in order to accomodate $e_c(v_1, x)$. So at least $4t$ spare vertices are used up already. Now there are $2t$ colours in $L_2 = L(v_2) \setminus \{1\}$. If a colour x in L_2 is not in L_1, then x requires an additional two spare vertices. If $x \in L_1 \cap L_2$, then x requires one additional spare vertex to cover both $e_c(v_1, x)$ and $e_c(v_2, x)$. Therefore, at least $2t$ additional spare vertices are needed. However, this means a total of $6t$ spare vertices are needed while there are only $6t-2$ available, which is a contradiction. Hence H must be $(2t+1)$-list-colourable. □

This implies the following.

Corollary 4.13. *For each admissible n, $ListSpec_M(n) \leq \frac{1}{3}n + 1$.*

Chapter 5

Embedding Nearly-Spanning Bounded-Degree Trees

In this chapter, we will see a set of sufficient conditions for a graph to contain (as subgraphs) all nearly-spanning trees of a fixed maximum degree, in terms of the graph's expansion properties. We will also apply this to random graphs. In particular, we answer a question about embedding large trees in random graphs asked by Alon, Krivelevich and Sudakov [6].

There are two main results that we will prove in this chapter, both of which improve upon results of [6]. One is a general result regarding embedding nearly-spanning trees in expanding graphs (Theorem 5.3), and the other is a result about embedding in random graphs (Corollary 5.4). We will give the statements of the results of [6] and our improved results in Section 5.1. In Section 5.2, we will prove Theorem 5.3, and finally in Section 5.3, we will prove Corollary 5.4.

5.1 Embedding nearly-spanning trees

Given a graph G on n vertices, a small constant $0 < \varepsilon < 1/2$, and an integer $d \geq 2$, we wish to find conditions on G such that it contains every nearly-spanning tree with maximum degree at most d. A nearly-spanning tree is one that has $(1-\varepsilon)n$ vertices. Our proof will follow the general ideas of [6].

First, we need to define the notion of expansion. Let G be a graph, and let $X \subset V$. Define $N_G(X)$ to be the set of vertices that are adjacent to at least one vertex in X (the "neighbours"). Note that $N_G(X)$ may contain some vertices of X. Let c and $\alpha < 1$ be two positive numbers. A graph $G = (V, E)$ is called an (α, c)-*expander* if for all $X \subset V$ with $|X| \leq \alpha|V|$,

$$|N_G(X)| \geq c|X|.$$

The main result that Alon, Krivelevich and Sudakov have proved is the following:

Theorem 5.1 (Alon, Krivelevich and Sudakov [6]). *Let $d \geq 2$, $0 < \varepsilon < 1/2$. Let $G = (V, E)$ be a graph on n vertices of minimum degree δ and maximum degree Δ. Let n, δ, Δ satisfy the following conditions:*

1. *(the orer of the graph is sufficiently large)*

$$n \geq \frac{480 d^3 \log(2/\varepsilon)}{\varepsilon},$$

2. *(the maximum degree is not too large compared to the minimum degree)*

$$\Delta^2 \leq \frac{1}{K} e^{(\delta/8K)-1} \text{ where } K = \frac{20 d^2 \log(2/\varepsilon)}{\varepsilon},$$

3. *(local expansion) every subgraph G_0 of G with minimum degree at least $\frac{\varepsilon\delta}{40 d^2 \log(2/\varepsilon)}$ is a $(\frac{1}{2d+2}, d+1)$-expander.*

Then G contains a copy of every tree T on at most $(1-\varepsilon)n$ vertices with maximum degree at most d.

A consequence of this theorem is the following result regarding embedding nearly-spanning trees in random graphs:

Corollary 5.2 (Alon, Krivelevich and Sudakov [6]). *Let $d \geq 2$, $0 < \varepsilon < 1/2$, and*

$$c \geq \frac{10^6 d^3 \log d \log^2(2/\varepsilon)}{\varepsilon}.$$

Then the random graph $\mathcal{G}(n, c/n)$ asymptotically almost surely (a.a.s.) contains every tree T on at most $(1-\varepsilon)n$ vertices with maximum degree at most d.

In the same paper, the authors speculate that this lower bound for c may not be necessary, perhaps $c = O(d\log(1/\varepsilon))$ is sufficient to embed all nearly spanning trees of maximum degree at most d a.a.s. This is best possible: d is needed because a random graph needs to have vertices of degree at least d in order to embed trees of maximum degree d; and as mentioned in the introduction, Frieze [35] showed that $c = O(\log(1/\varepsilon))$ is needed to embed a path of length $(1-\varepsilon)n$, so this is a lower bound for embedding trees. We are close to achieving the speculated bound on c with respect to the parameter d (with only an extra factor of $\log d$), improving the bound of c by a factor of d^2. We first make a refinement on the main theorem:

Theorem 5.3. *Let $d \geq 2$, $0 < \varepsilon < 1/2$. Let $G = (V, E)$ be a graph on n vertices of minimum degree δ and maximum degree Δ. Let n, δ, Δ satisfy the following conditions:*

1. *(the order of the graph is sufficiently large)*
$$n \geq 60d/\varepsilon,$$
2. *(the maximum degree is not too large compared to the minimum degree)*
$$\Delta^2 \leq \frac{1}{K} e^{(\delta/8K)-1} \text{ where } K = \frac{40\log(2/\varepsilon)}{\varepsilon},$$
3. *(local expansion) every subgraph G_0 of G with minimum degree at least $\frac{\varepsilon\delta}{80\log(2/\varepsilon)}$ is a $(\frac{1}{4d+1}, 3d)$-expander.*

Then G contains a copy of every tree T on at most $(1-\varepsilon)n$ vertices with maximum degree at most d.

We note that condition 2 ensures that the minimum degree δ has order at least $\Omega(K)$, which is $\Omega(\log(1/\varepsilon)/\varepsilon)$. Since $n > \delta$, n must be at least $\Omega(\log(1/\varepsilon)/\varepsilon)$ as well, and this is a lower bound for n in terms of ε. In some cases, this may already be larger than the lower bound given in condition 1, which gives a lower bound for n in terms of d and ε.

Using this theorem, we can prove the following improvement on Corollary 5.2:

Corollary 5.4. *Let $d \geq 2$, $0 < \varepsilon < 1/2$, and*
$$c \geq \frac{10^7 d\log d \log^2(2/\varepsilon)}{\varepsilon}.$$
Then the random graph $\mathcal{G}(n, c/n)$ asymptotically almost surely contains every tree T on at most $(1-\varepsilon)n$ vertices with maximum degree at most d.

5.2 Embedding trees in expanding graphs

We will prove Theorem 5.3 in this section. First, we will give an overall framework for this proof in Section 5.2.1. In three subsequent sections, we will provide the tools that are necessary in the proof: Section 5.2.2 describes results that we use to embed small trees, Section 5.2.3 shows how we can split the tree that we want to embed into small trees, and Section 5.2.4 splits the graph into pieces that have special properties. Finally in Section 5.2.5, we give the proof of Theorem 5.3.

5.2.1 Approach to proving Theorem 5.3

We will give an outline for how the proof of Theorem 5.3 works. We emphasize again that the general approach comes from Alon, Krivelevich and Sudakov [6]. The gist of it is that we will cut the tree into pieces that are "small" relative to the graph, find large subgraphs of G that have high expansion property, and use Corollary 5.7 (which shows that one can embed small trees into expanding graphs) to sequentially embed such subtrees into these subgraphs. Note that [6] used Theorem 5.5 to embed small trees, and by using Corollary 5.7 instead, we obtain a saving of a factor of d in the main result of Corollary 5.4.

First, in Section 5.2.2, we will divide the tree that we would like to embed into several pieces. The way we divide the tree is different from [6], and this leads to a saving of another factor of d in Corollary 5.4. Given the tree T, we cut it down into a constant number s (dependent only on ε) of subforests $T_1, T_2, \ldots, T_s$, with the exception that T_1 must always be a tree. The number of vertices in each T_i is about a fraction of the size of T. Each of the subforests T_i where $i > 1$ has the property that there exists a vertex v_{i-1} (called its "root") in T_{i-1} such that v_{i-1} is adjacent to a vertex in each of the subtrees of T_i. In the embedding process, we will embed the subforests one at a time. Whenever we try to embed T_i where $i > 1$, its root has already been embedded in T_{i-1}, so we may attempt to embed T_i together with its root (which would form a tree) using Corollary 5.7.

In Section 5.2.4, we will mention a result from [6] where, using the Lovász Local Lemma, condition 2 of our theorem regarding the relationship between the maximum degree and the minimum degree of the graph guarantees the existence of a partition of $V(G)$ into a constant number of pieces $S_1, S_2, \ldots, S_K$ such that each vertex in the graph has many neighbours (about a fraction of the minimum degree) in each piece. We pick the s smallest ones $S_1, \ldots, S_s$ which will only occupy at most $\varepsilon n/2$ vertices, a small fraction of the graph. These pieces will eventually provide us with subgraphs of G that have large enough minimum degree so that condition 3 in our theorem would imply the expansion property necessary to apply Corollary 5.7.

The proof of the main theorem is in Section 5.2.5. We embed the subforests in sequential order $T_1, T_2, \ldots, T_s$. We pick an arbitrary vertex to be the root of T_1. When we embed T_i, we consider the subgraph U_i which includes S_i, the embedded root of T_i, and vertices outside of any S_j's which have not been used in embedding $T_1, \ldots, T_{i-1}$. Since this subgraph U_i contains S_i, it has high minimum degree and hence high expansion factor. Also, since $S_1, \ldots, S_s$ are small and n is sufficiently large by condition 1 of the theorem, the size of U_i is large enough compared to T_i so that we can apply Corollary 5.7 to embed T_i into U_i. Note that each S_i is only used once in the entire process, namely when we embed T_i. Unused vertices that are not in any S_j's are "recycled" after each embedding.

5.2.2 Embedding small trees

The proof of the main theorem essentially depends on the ability to embed small trees into graphs of high expansion factor. Alon, Krivelevich and Sudakov relied on the following result from Friedman and Pippenger:

Theorem 5.5 (Friedman and Pippenger [34])**.** *Let T be a tree on t vertices of maximum degree d rooted at r. Let $H = (V, E)$ be a non-empty graph such that for each $X \subset V$ with $|X| \leq 2t - 2$,*

$$|N_H(X)| \geq (d+1)|X|.$$

Let v be an arbitrary vertex in H. Then H contains a copy of T as a subgraph, rooted at v.

To improve the result of Alon, Krivelevich and Sudakov, we need the following refinement on the theorem of Friedman and Pippenger, proved by Haxell:

Theorem 5.6 (Haxell [39])**.** *Let T be a tree on t vertices of maximum degree d rooted at r. Let $\emptyset = T_0 \subset T_1 \subset T_2 \subset \cdots \subset T_l \subset T$ be a sequence of subtrees of T such that T can be obtained from T_l by adding leaves to T_l. Let $d = d_1 \geq \cdots \geq d_l \geq 1$*

be a sequence of integers such that for each i with $1 \leq i \leq l$ and each $v \in V(T)$, $deg_T(v) - deg_{T_{i-1}}(v) \leq d_i$ (where $deg_G(u)$ is the degree of u in the graph G). Let $t_i = |E(T_i)|$. Suppose there is an integer $k \geq 1$ and a graph H satisfying the following $l+2$ conditions:

(0) $|N(X)| \geq d|X| + 1$ *for all* $X \subset V(H)$, $1 \leq |X| \leq 2k$,

(i) $|N(X)| \geq d_i|X| + t_i + 1$ *for all* $X \subset V(H)$, $k < |X| \leq 2k$ *(for $1 \leq i \leq l$),*

($l+1$) $|N(X)| \geq t + 1$ *for all* $X \subset V(H)$, $|X| = 2k + 1$.

Let v be an arbitrary vertex of H. Then H contains a copy of T as a subgraph, rooted at v.

And here is an application that improves Corollary 5.2 by a factor of d:

Corollary 5.7. *Let T be a tree on t vertices of maximum degree d rooted at r. Let $H = (V, E)$ be a non-empty graph such that for each $X \subset V$ with $|X| \leq t/d + 1$,*

$$|N_H(X)| \geq 3d|X|.$$

Let v be an arbitrary vertex in H. Then H contains a copy of T as a subgraph, rooted at v.

Proof. Let T_1 be the tree obtained from T by removing a leaf. We use Theorem 5.6 with $l = 1$ and $k = t/2d$ to prove this result. Suppose that $|N_H(X)| \geq 3d|X|$ for all $X \subset V$ with $|X| \leq t/d + 1$. We need to show that conditions (0), (1) and (2) in Theorem 5.6 hold.

(0) For $1 \leq |X| \leq t/d$, $|N_H(X)| \geq 3d|X|$ clearly implies $|N_H(X)| \geq d|X| + 1$, so this condition holds.

(1) For $t/2d < |X| \leq t/d$, we first note that $d_1 \leq d$, and $t_1 = t - 1 \leq t < 2d|X|$, which implies that $t_1 + 1 \leq 2d|X|$. So

$$|N_H(X)| \geq 3d|X| = d|X| + 2d|X| \geq d_1|X| + t_1 + 1,$$

hence this condition holds.

(2) For $|X| = t/d + 1$, $t = d(|X| - 1)$. Therefore, $|N_H(X)| \geq 3d|X| \geq t + 1$, and this condition holds.

Since all three conditions are satisfied, Theorem 5.6 implies that there exists a copy of T in H, rooted at v. □

5.2.3 Splitting the tree

We need the following result to help us in splitting the tree into subforests. This is a modification of the tree splitting method in [6]. Our method reduces the number of pieces by a factor of d^2, and this is another key in improving the results in [6].

We first define a *pseudo-rooted subforest* T^* of a rooted tree T as follows: Let T be rooted at r. Let $r_1, \ldots, r_l$ be vertices in T that have a common parent v in T. Let $T_1, \ldots, T_l$ be subtrees of T where each T_i contains r_i and all descendants of r_i in T. Then T^* consists of $T_1 \cup \cdots \cup T_l$. We call the vertex v the root of T^*. Note that if T^* is a pseudo-rooted subforest of T, then $T - T^*$ is a tree.

Proposition 5.8. *Let k be a positive integer. Let T be a tree on at least $k+1$ vertices. Root T at any vertex. Then there exists a pseudo-rooted subforest T' of T such that the number of vertices in T' is between k and $2k-2$.*

Proof. Let r be the root of T. Let L_i be the set of vertices of distance i from r, $i \geq 1$. For each vertex v, define $t(v)$ to be the number of vertices in the subtree of T rooted at v. Define i_0 to be the largest i such that at least one vertex v in L_i has $t(v) \geq k$. Since there is only one vertex v in L_1 and $t(v) \geq |V(T)| - 1 \geq k$, we see that $i_0 \geq 1$. Let u be any vertex in L_{i_0} satisfying $t(u) \geq k$. If $t(u) = k$, then the subtree of T rooted at u satisfies the conclusion of this proposition, and we are done. Otherwise, let $u_1, \ldots, u_l$ be the children of u, and let T_i be the subtree of T rooted at u_i, $1 \leq i \leq l$. Order the indices so that $|T_1| \leq |T_2| \leq \cdots \leq |T_l|$. Let j be the smallest index such that $\sum_{i=1}^{j} |T_i| \geq k$. But $\sum_{i=1}^{j-1} |T_i| < k$ and $|T_j| < k$, so $\sum_{i=1}^{j} |T_i| \leq 2k-2$. Therefore, $T' = \cup_{i=1}^{j} T_i$ satisfies the conclusion of this proposition, and we are done. $\square$

We can now split a tree into one subtree and a constant number of pseudo-rooted forests as follows.

Corollary 5.9. *Suppose $0 < \varepsilon < 1/2$, and T is an arbitrary tree on $(1-\varepsilon)n$ vertices. Then we can cut T into a subtree T_1 and $s-1$ disjoint pseudo-rooted subforests $T_2, \ldots, T_s$ such that for each $i > 1$, the root v_i of T_i is in $\cup_{j<i} T_j$, and*

$$\frac{\varepsilon n/2 + \sum_{j>i} |V(T_j)|}{16} \leq |V(T_i)| \leq \frac{\varepsilon n/2 + \sum_{j>i} |V(T_j)|}{8}.$$

For T_1, the upper bound holds, but not necessarily the lower bound. (The union of the vertices of the subforests is $V(T)$.) Also, $s \leq 20 \log(2/\varepsilon)$.

Proof. We will choose the subforests one by one in reverse order. So our choices for the pseudo-rooted subforests $T'_1, T'_2, \ldots, T'_s$ will satisfy

$$\frac{\varepsilon n/2 + \sum_{j<i} |V(T'_j)|}{16} \leq |V(T'_i)| \leq \frac{\varepsilon n/2 + \sum_{j<i} |V(T'_j)|}{8},$$

except for T'_s; and the root v'_i of each T'_i is in $T - \cup_{j \leq i} T'_j$. At the end, we will set $T_i = T'_{s-i+1}$ and $v_i = v'_{s-i+1}$.

For T'_1, we use Proposition 5.8 to obtain a subforest such that all components are connected to the same vertex v'_1 of $T - T'_1$, and the number of vertices is between $\frac{\varepsilon n}{32}$ and $\frac{\varepsilon n}{16}$. We now remove T'_1 from T, and note that $T - T'_1$ is still a tree.

Suppose we have obtained $T'_1, \ldots, T'_{i-1}$ that satisfy the conditions stated above. Let $T' = T - \cup_{j<i} T'_j$. If T' contains fewer than $\frac{\varepsilon n/2 + \sum_{j<i} |V(T'_j)|}{8}$ vertices, then set $T_i = T'$, $i = s$, and we are done. Otherwise, use Proposition 5.8 to obtain a pseudo-rooted subforest T'_i whose number of vertices is between $\frac{\varepsilon n/2 + \sum_{j<i} |T'_j|}{16}$ and $\frac{\varepsilon n/2 + \sum_{j<i} |T'_j|}{8}$ with root v'_i in $T - \cup_{j \leq i} T'_j$, and continue with this process.

It remains to bound the number of subforests s that we have created. Let $a_i = \varepsilon n/2 + \sum_{j<i} |V(T'_j)|$. Then $a_0 = \varepsilon n/2$ and $a_i \leq \varepsilon n/2 + |V(T)| \leq n - \varepsilon n/2 \leq n$. Also, $a_{i+1} = a_i + |V(T'_{i+1})| \geq (1 + \frac{1}{16}) a_i$. Therefore,

$$\frac{2}{\varepsilon} \geq \frac{a_s}{a_0} \geq \left(1 + \frac{1}{16}\right)^s.$$

Solving the inequality, we get $s \leq 20 \log(2/\varepsilon)$. □

5.2.4 Splitting vertex degrees

The following result shows that when the minimum degree is not far from the maximum degree, it is possible to partition the vertices of the graph into pieces so that every vertex has a large number of neighbours in each piece. This is essential in finding subgraphs that have high minimum degree.

Lemma 5.10 ([6]). *Let numbers K, δ, Δ satisfy*

$$K\Delta^2 e^{(-\delta/8K)+1} < 1.$$

Let $H = (V, E)$ be a graph with minimum degree δ and maximum degree Δ. Then H contains K pairwise disjoint sets of vertices $S_1, \ldots, S_K$ such that every vertex of H is adjacent to at least $\delta/2K$ vertices in each S_i.

This is the same theorem as in [6]. The proof is an application of Lovász's Local Lemma.

5.2.5 Proof of Theorem 5.3

Suppose $G = (V, E)$ is a graph on n vertices that satisfies conditions 1, 2, and 3 of the theorem. Let T be a tree on at most $(1-\varepsilon)n$ vertices of maximum degree at most d. Use Corollary 5.9 to split the tree into s pseudo-rooted subforests or subtrees $T_1, \ldots, T_s$ with their respective roots $v_1, \ldots, v_s$, where $s \leq 20\log(2/\varepsilon)$. Denote $t = |V(T)|$, and $t_i = |V(T_i)|$ for $1 \leq i \leq s$. Put $K' = 2s/\varepsilon \leq \frac{40\log(2/\varepsilon)}{\varepsilon}$. (Note that in condition 2 of this theorem, the right hand side of the inequality decreases as K grows. Since substituting $K = \frac{40\log(2/\varepsilon)}{\varepsilon}$ satisfies the inequality by assumption, the K' we use here also satisfies the inequality.) Then using condition 2 of this theorem and Lemma 5.10, there exist K' mutually disjoint sets of vertices $S_1, \ldots, S_{K'}$ such that every vertex of G is adjacent to at least $\delta/2K' \geq \frac{\varepsilon\delta}{80\log(2/\varepsilon)}$ vertices in each S_i. Pick the s smallest sets among $S_1, \ldots, S_{K'}$, and renumber them as $S_1, \ldots, S_s$. The total size of these s sets is at most $\frac{sn}{K'} = \frac{\varepsilon n}{2}$.

Let x_1 be an arbitrary vertex that is not in any S_i's. This will be the root for the embedded T_1 (recall that T_1 is a tree). Let $U_1 = V - \cup_{j\neq 1} S_j$, and let $G_1 = G[U_1]$. Since U_1 contains S_1, the minimum degree of G_1 is at least $\frac{\varepsilon\delta}{80\log(2/\varepsilon)}$. By condition 3, G_1 is a $(\frac{1}{4d+1}, 3d)$-expander, hence for all $X \subset U_1$ with $|X| \leq |U_1|/(4d+1)$, we have $|N_{G_1}(X)| \geq 3d|X|$. In order to apply Corollary 5.7 and conclude that T_1 can be embedded into G_1 with root x_1, we need to show that $\frac{|U_1|}{4d+1} \geq \frac{t_1}{d} + 1$. We know that

$$|U_1| \geq n - \sum_{j=1}^{s} |S_i| \geq n - \varepsilon n/2.$$

From Corollary 5.9, we see that

$$t_1 \leq \frac{\varepsilon n/2 + \sum_{j>1} t_j}{8} \leq \frac{\varepsilon n/2 + t}{8} \leq \frac{\varepsilon n/2 + (1-\varepsilon)n}{8} = \frac{n - \varepsilon n/2}{8} \leq \frac{|U_1|}{8}.$$

Therefore,

$$\frac{t_1}{d} + 1 \leq \frac{|U_1|}{8d} + 1 \leq \frac{|U_1|}{4d+1}.$$

The last inequality is true provided that

$$|U_1| \geq 13d \geq \frac{32d^2 + 8d}{4d - 1},$$

which is confirmed by

$$|U_1| \geq n - \varepsilon n/2 \geq 3n/4 \geq 45d$$

using condition 1 of the theorem. So we can indeed embed T_1 into G_1 with root x_1 using Corollary 5.7.

Suppose that we have already embedded $T_1, \ldots, T_{i-1}$, such that the root of each subforest T_j is embedded to x_j, and the embedded T_j does not use any vertices in $\cup_{l \neq j} S_l$. We now wish to embed the subforest T_i by embedding the tree $T_i^* = T[V(T_i) \cup \{v_i\}]$. The root v_i of T_i^* has already been embedded to some vertex x_i in the previous steps. Let U_i be the set of vertices of $G - \cup_{j \neq i} S_j$ that have not been used in the embedding of $T_1, \ldots, T_{i-1}$, except for x_i. Let $G_i = G[U_i]$. Since U_i contains S_i and no edge in G_i is used in embedding $T_1, \ldots, T_{i-1}$, the minimum degree of G_i is at least $\frac{\varepsilon\delta}{80 \log(2/\varepsilon)}$. By condition 3, G_i is a $(\frac{1}{4d+1}, 3d)$-expander, hence for any $X \subset U_i$ with $|X| \leq |U_i|/(4d+1)$, we have $|N_{G_i}(X)| \geq 3d|X|$. In order to apply Corollary 5.7 and conclude that T_i^* can be embedded into G_i with root x_i, we need to show that $\frac{|U_i|}{4d+1} \geq \frac{t_i+1}{d} + 1$. For U_i, we have

$$|U_i| \geq n - \sum_{j=1}^{s} S_j - \sum_{j<i} t_j \geq n - \varepsilon n/2 - \sum_{j<i} t_j.$$

From Corollary 5.9, we see that

$$\begin{aligned} t_i &\leq \frac{\varepsilon n/2 + \sum_{j>i} t_j}{8} = \frac{\varepsilon n/2 + t - \sum_{j \leq i} t_j}{8} \\ &\leq \frac{n - \varepsilon n/2 - \sum_{j<i} t_j}{8} \leq \frac{|U_i|}{8}. \end{aligned}$$

Therefore,

$$\frac{t_i + 1}{d} + 1 \leq \frac{|U_i|}{8d} + 2 \leq \frac{|U_i|}{4d+1}.$$

The last inequality is true provided that

$$|U_i| \geq 30d \geq \frac{64d^2 + 16d}{4d - 1},$$

which is confirmed by

$$|U_i| \geq n - \varepsilon n/2 - (1 - \varepsilon)n \geq n\varepsilon/2 \geq 30d$$

using condition 1 of the theorem. So we can indeed embed T_i^* into G_i with root x_i using Corollary 5.7.

We may continue this process until T_s, at which point T would be entirely embedded into G. Hence the theorem holds. $\square$

5.3 Embedding trees in random graphs

We aim to prove Corollary 5.4, which states that for $0 < \varepsilon < 1/2$ and $d \geq 2$, if

$$c \geq \frac{10^7 d \log d \log^2(1/\varepsilon)}{\varepsilon},$$

then the random graph $\mathcal{G}(n, c/n)$ contains every tree of maximum degree at most d with at most $(1-\varepsilon)n$ vertices, asymptotically almost surely. We first show that there are only a small number of vertices with very high and very low degrees, and we may remove them without reducing the size of the graph too much. This way, conditions 1 and 2 of Theorem 5.3 can be satisfied. The hard work is then to show that such a sparse random graph has good expansion properties (i.e. each subgraph of certain minimum degree is an expander). Again, we mimic the proof in [6] here. The key in this proof is the following lemma:

Lemma 5.11 ([6] essentially)**.** *For every integer $d \geq 2$, reals $0 < \theta < 1/2$ and $D \geq 50\theta^{-1}$, the random graph $\mathcal{G}(n, \frac{4D}{n})$ a.a.s. contains a subgraph G^* with the following properties:*

1. $|V(G^*)| \geq (1-\theta)n$*;*
2. $D \leq d_{G^*}(v) \leq 10D$ *for all* $v \in V(G^*)$*; and*
3. *every induced subgraph G_0 of G^* of minimum degree at least $D_0 = 200d \log D$ is a $(\frac{1}{4d+1}, 3d)$-expander.*

To prove this lemma, we need the following properties about random graphs in general:

Proposition 5.12 ([6])**.** *Let $\mathcal{G}(n,p)$ be a random graph with $np > 20$. Then the following two items occur a.a.s.:*

(i) The number of edges between any two disjoint sets $A, B \subseteq V$ with $|A| = a$ and $|B| = b$ such that $abp \geq 32n$ is at least $abp/2$ and at most $3abp/2$.

(ii) For every subset of vertices S with size a where $a \leq n/4$, $G[S]$ contains less than $anp/2$ edges.

The proof of this proposition consists of simple applications of Chernoff bounds. This statement is not changed from [6].

Proof of Lemma 5.11. In the first part of the proof, we will remove vertices of low degree and high degree so that at least $(1-\theta)n$ vertices remain, in order to satisfy conditions 1 and 2. Let $G = \mathcal{G}(n,p)$ be a random graph with $p = \frac{4D}{n}$, and let X be the set of $\theta n/2$ vertices with the largest degrees in the graph. Since $np = 4D \geq 200\theta^{-1} \geq 400$ and $\theta n/2 \leq n/4$, we may apply part (ii) of Proposition 5.12 to see that a.a.s. there are fewer than $|X|np/2 = 2D|X|$ edges in $G[X]$. Also, since $|X|(n-|X|)\frac{4D}{n} \geq \frac{4D}{n}\frac{\theta n}{2}(1-\frac{\theta}{2})n \geq 2D\theta(n/2) = D\theta n \geq 50n$, we may apply part (i) of Proposition 5.12 to see that a.a.s. there are at most $3|X|(n-|X|)p/2 \leq 3|X|n\frac{4D}{n}/2 = 6D|X|$ edges between X and $V(G)-X$. Therefore, the sum of vertex degrees in X is at most $10D|X|$, which means there is at least one vertex in X of degree at most $10D$. By definition of X, we see that there are no more than $\theta n/2$ vertices with degree larger than $10D$ in G. Remove these vertices from G to obtain G'.

We now want to remove vertices of low degree. In G', if there is a vertex of degree less than D, then we remove it from G'. Repeat this deletion until each vertex in the remaining graph has degree at least D. Suppose we have deleted more than $\theta n/2$ vertices by the end of this process. We claim that this does not happen a.a.s., and we wish to use part (i) of Proposition 5.12 to show that. Let Y be the first $\theta n/2$ vertices that we deleted. Then $|V(G')-Y| > (1-\frac{\theta}{2})n \geq n/2$. So $p|Y||V(G')-Y| > \frac{4D}{n}\frac{\theta n}{2}\frac{n}{2} = D\theta n \geq 50n$, and the assumption of part (i) of the proposition is satisfied. Therefore, the number of edges between Y and $V(G')-Y$ is a.a.s. at least $p|Y||V(G')-Y|/2 > D\theta n/2$. However, the choice of Y implies that the number of edges between Y and $V(G')-Y$ is at most $|Y|D \leq D\theta n/2$, which a.a.s. cannot occur. Therefore, this process of deletion ends with no more than $\theta n/2$ vertices deleted a.a.s. We denote that remaining graph by G^*. Note that G^* has at least $(1-\theta)n$ vertices, and the degree of each vertex in G^* is between D and $10D$, so it satisfies the first two conditions of this lemma.

It remains to show that the third condition holds. Suppose that it does not hold, and there exists a subset of vertices U such that $G_0 = G^*[U]$ has minimum degree at least D_0, but it is not a $(\frac{1}{4d+1}, 3d)$-expander. So there exists a set $X \subset U$ such that $|X| = t \leq |U|/(4d+1)$ and $C = N_{G_0}(X)$ satisfies $|C| \leq 3d|X|$. Also, there are at least $D_0|X|/2 = 100dt\log D$ edges between X and C. If $t < \frac{\log D}{D}n$, then the probability that G contains such sets X and C is at most

$$\begin{aligned}
\mathbb{P}_t &\le \binom{n}{t}\binom{n}{3dt}\binom{3dt^2}{100dt\log D}p^{100dt\log D} \\
&\le \left[\left(\frac{en}{t}\right)\left(\frac{en}{3dt}\right)^{3d}\left(\frac{3etp}{100\log D}\right)^{100d\log D}\right]^t \\
&\le \left[e\left(\frac{e}{3d}\right)^{3d}\left(\frac{n}{t}\right)^{3d+1}\left(\frac{12etD}{100n\log D}\right)^{100d\log D}\right]^t \\
&\le \left[\left(\frac{n}{t}\right)^{4d}\left(\frac{e}{8}\cdot\frac{tD}{n\log D}\right)^{100d\log D}\right]^t \\
&= \left[\left(\frac{e}{8}\right)^{100d\log D}\left(\frac{D}{\log D}\right)^{4d}\left(\frac{tD}{n\log D}\right)^{100d\log D-4d}\right]^t \\
&\le \left[(e^{-1})^{100d\log D}\cdot e^{4d\log(D/\log D)}\left(\frac{tD}{n\log D}\right)^{96d\log D}\right]^t \\
&< \left[e^{-96d\log D}\left(\frac{t}{n\log D/D}\right)^{96d\log D}\right]^t \\
&= \left[D^{-96d}\left(\frac{t}{n\log D/D}\right)^{96d\log D}\right]^t
\end{aligned}$$

We wish to conclude that $\mathbb{P}_t = o(n^{-1})$. We split into two subcases to do this. When $t < \log n$,

$$\mathbb{P}_t \le \left(\frac{D\log n}{n}\right)^{96d\log D}.$$

Since D is a constant, $D\log n/n^{.9} < 1$ for sufficiently large n, i.e. $D\log n/n < n^{-.1}$. Therefore, for sufficiently large n,

$$\mathbb{P}_t \le (n^{-.1})^{96d\log D} = o(n^{-1}).$$

Now when $\log n \le t < \frac{\log D}{D}n$,

$$\mathbb{P}_t \le D^{-96d\log n}.$$

Note that

$$n^{-1} = D^{-\frac{\log n}{\log D}}.$$

Since $-96d < -10/\log D$, we can conclude that

$$\mathbb{P}_t \le D^{-96d\log n} < D^{-\frac{10\log n}{\log D}} = n^{-10} = o(n^{-1}).$$

We now deal with the case of $t \geq \frac{\log D}{D} n$. Note that there are no edges in G between X and $Y = U - (X \cup N_{G_0}(X))$. Using $t = |X| \leq |U|/(4d+1)$ and $|N_{G_0}(X)| \leq 3dt$, we get

$$\begin{aligned}
|Y| &\geq |U| - |X| - |N_{G_0}(X)| \\
&\geq |U| - \frac{|U|}{4d+1} - 3dt \\
&\geq |U| - \frac{|U|}{4d+1} - 3d\frac{|U|}{4d+1} \\
&= (4d+1-1-3d)|U|/(4d+1) \\
&= d|U|/(4d+1) \\
&\geq dt.
\end{aligned}$$

So the probability of G having these sets is at most

$$\begin{aligned}
\mathbb{P}_t &\leq \binom{n}{t}\binom{n}{dt}(1-p)^{dt^2} \leq \left[\frac{en}{t}\left(\frac{en}{dt}\right)^d e^{-pdt}\right]^t \\
&\leq \left[\left(\frac{en}{t}\right)^{2d} e^{-pdt}\right]^t = \left[\left(\frac{en}{t}\right)^2 e^{-pt}\right]^{dt} \\
&\leq \left[\left(\frac{en}{n\log D/D}\right)^2 e^{-\frac{4D}{n}\cdot\frac{\log D}{D}n}\right]^{dt} \\
&\leq (D^2 D^{-4})^{dt} = o(n^{-1}).
\end{aligned}$$

(The first line uses the fact that $1 - x \leq e^{-x}$.) So the probability that G^* does not satisfy the third condition is at most $\sum_{t=1}^{n} \mathbb{P}_t = o(1)$. Hence, the third condition is satisfied a.a.s. $\square$

Proof of Corollary 5.4. Let $\theta = 0.01\varepsilon$, $D = c/4$, and $\varepsilon_1 = \frac{\varepsilon-\theta}{1-\theta} \geq 0.99\varepsilon$. Since $D \geq 50\theta^{-1} = 5000/\varepsilon$, Lemma 5.11 implies that $\mathcal{G}(n, c/n)$ a.a.s. contains a subgraph G^* with $n_1 \geq (1-\theta)n$ vertices such that the minimum degree is at least D, the maximum degree is at most $10D$, and every induced subgraph with minimum degree at least $200d \log D$ is an $(\frac{1}{4d+1}, 3d)$-expander. Condition 1 of Theorem 5.3 is satisfied since we are dealing with an asymptotic result where $n \to \infty$. To check condition 2, we need to verify that since $\Delta \leq 10\delta$,

$$\Delta^2 \leq 100D^2 \leq \frac{1}{K}e^{D/8K-1},$$

which we can rearrange as

$$100K \leq \frac{e^{D/8K-1}}{D^2}.$$

Since the right hand side is an increasing function in D, we may simply replace D by its lower bound $c/4 \geq 10^7 d \log d \log^2(2/\varepsilon)/4\varepsilon$. This gives

$$\frac{4000\log(2/\varepsilon)}{\varepsilon} \leq \frac{\varepsilon^2 e^{10^4 d\log d \log(2/\varepsilon)-1}}{10^{12}d^2\log^2 d\log^4(2/\varepsilon)},$$

which simplifies to

$$\frac{10^{16}d^2\log^2 d\log^5(2/\varepsilon)}{\varepsilon^3} \leq \left(\frac{2}{\varepsilon}\right)^{10^4 d\log d},$$

which holds.

To show that condition 3 holds, it suffices to show that

$$200d\log D \leq \frac{\varepsilon_1 D}{80\log(2/\varepsilon_1)},$$

which we can rearrange as

$$\frac{16000d\log(2/\varepsilon_1)}{\varepsilon_1} \leq \frac{D}{\log D}.$$

Since $x/\log x$ is an increasing function for $x > 3$, we may replace the right hand side by the lower bound for D to get

$$\begin{aligned}
\frac{D}{\log D} &\geq \frac{10^7 d\log d\log^2(2/\varepsilon)}{4\varepsilon}\cdot\frac{1}{\log(10^7/4)+\log d+\log\log d+2\log\log(2/\varepsilon)-\log\varepsilon}\\
&\geq \frac{10^7 d\log(2/\varepsilon)}{4\varepsilon}\cdot\frac{\log d\log(2/\varepsilon)}{18+\log d+\log\log d+2\log\log(2/\varepsilon)}\\
&\geq \frac{10^7 d\log(2/\varepsilon)}{4\varepsilon\cdot 22} \geq \frac{10^5 d\log(2/\varepsilon)}{(88/100)\varepsilon}\\
&\geq \frac{10^5 d\log(2/\varepsilon)}{.99\varepsilon} \geq \frac{10^5 d\log(2/\varepsilon)}{\varepsilon_1}\\
&\geq \frac{5\cdot 10^4 d(2\log(2/\varepsilon))}{\varepsilon_1} \geq \frac{16000d\log(2/\varepsilon_1)}{\varepsilon_1},
\end{aligned}$$

where the third inequality is due to the fact that $\log d, \log\log d, \log\log(2/\varepsilon) < \log d\log(2/\varepsilon)$, and where the last inequality can be justified by

$$\begin{aligned}
\log(2/\varepsilon_1) &\leq \log(2/.99\varepsilon)\\
&\leq \log(2/\varepsilon) - \log .99 = \log(2/\varepsilon)(1-\log(.99)/\log(2/\varepsilon))\\
&\leq \log(2/\varepsilon)(1-\log .99/\log 4) \leq 2\log(2/\varepsilon).
\end{aligned}$$

So by Theorem 5.3, G^* contains every tree on $(1-\varepsilon_1)n_1 \geq (1-\varepsilon_1)(1-\theta)n = (1-\varepsilon)n$ vertices with maximum degree at most d. $\square$

Chapter 6

An Extremal Result for Quantum Computing

We consider an extremal graph problem that arises from the one-way measurement model of quantum computing. We will roughly describe the one-way measurement model in Section 6.1, then formulate and solve the corresponding extremal problem in Section 6.2.

6.1 One-way Measurement Model and Flows

The one-way measurement model of quantum computation [56, 57, 58, 21] is a scheme that consists entirely of one-qubit measurements on a particular class of entangled states called the cluster states (groups of qubits that are entangled). We can imagine a program or algorithm that runs on the one-way measurement model as a system of qubits that are linked together in the way that is dictated by the program. Some qubits are set apart as input, and some as output. Once the input is set (through two-qubit entanglements), only single-qubit operations are performed for the rest of the program until the output is formed.

These programs or algorithms can be described in part by a graph G where each vertex represents a qubit, and each edge represents entanglement operations performed on the qubits at the two ends of the edge. Two (not necessarily disjoint) subsets of vertices, I and O represent the input and output of the algorithm respectively. The triple (G, I, O) is called a geometry. A flow $(f, \preccurlyeq)$ in a geometry

(G, I, O) is defined as a function $f : (V(G) \setminus O) \to (V(G) \setminus I)$ and a partial order $\preccurlyeq$ on $V(G)$ such that

- x is adjacent to $f(x)$;
- $x \preccurlyeq f(x)$; and
- if y is adjacent to $f(x)$, then $x \preccurlyeq y$.

A flow can be thought of as a partial order describing when a qubit can be measured. If $x \preccurlyeq y$, then y cannot be measured before x is measured.

The concept of a flow was introduced by Danos and Kashefi [20]. Given a geometry, the existence of a flow is a sufficient condition for the geometry to underlie a "unitary embedding," independent of the measurements to be performed on each qubit. This is a good property for a geometry to have, as such geometries have a more deterministic behaviour (in the probabilistic environment) and are considered to be stable (not easily destroyed by entanglement operations). The main result of this chapter is the following: In a geometry (G, I, O) where $|V(G)| = n$ and $|O| = k$, if it has a flow, then the maximum number of edges that G may have is $kn - \binom{k+1}{2}$, and this bound is tight. We use a counting argument to prove the upper bound, and that leads naturally to a construction that achieves the bound. An algorithm in [22] efficiently determines whether a geometry has a flow. As a consequence of the main result here, one can initially check that G does not have too many edges, before proceeding with the algorithm. This would improve the running time for the algorithm in [22] from $O(km)$ to $O(k^2n)$ (where m is the number of edges).

6.2 Corresponding Graph Problem

The problem of bounding the number of edges in a flow can be reduced to the following extremal problem:

Problem. Let n, k be integers where $n \geq k$. Let G be a graph on n vertices which includes k mutually disjoint directed paths $P_1, P_2, \ldots, P_k$ that cover $V(G)$. Let $D(G, P_1, \ldots, P_k)$ be a directed graph derived from $V(G)$ as follows: for each edge xy in G that is not in any path P_i, say $x \in P_i$ and $y \in P_j$, replace xy with a directed edge from the predecessor of x in P_i to y, and a directed edge from the predecessor of y in P_j to x (when these predecessors are well-defined). What is the maximum number of edges $\Gamma(n,k)$ that G may have, under the constraint that $D(G, P_1, \dots, P_k)$ is acyclic?

We claim the following:

Theorem 6.1. $\Gamma(n,k) = kn - \binom{k}{2}$ *for all integers* $n \geq k \geq 1$.

In this section, we will prove this theorem by bounding the number of edges between any two paths P_i and P_j, and then give a construction which saturates this bound.

6.2.1 Upper bound

To provide an upper bound on $\Gamma(n,k)$, we make the following observations. Let G and $P_1, \dots, P_k$ be as described in the problem above, and let $D = D(G, P_1, \dots, P_k)$. We will use the notation $v \to w$ in a digraph to represent a directed edge from v to w.

Observation 1. Consider any one of the paths $P_i = v_1 \to v_2 \to \cdots \to v_{n_i}$. If D is acyclic, then $v_a v_b \notin E(G)$ for any pair of vertices where $a < b - 1$. Otherwise, D would contain the cycle $v_a \to v_{a+1} \to \cdots \to v_{b-1} \to v_a$.

Observation 2. Consider any two distinct paths $P_i = v_1 \to v_2 \to \cdots \to v_{n_i}$ and $P_j = w_1 \to w_2 \to \cdots \to w_{n_j}$. If D is acyclic, then there cannot be two edges $v_a w_b$, $v_c w_d \in E(G)$ where $a < c$ and $b > d$. Otherwise, $D(G, P_1, \dots, P_k)$ would contain the cycle $v_a \to \cdots \to v_{c-1} \to w_d \to \cdots \to w_{b-1} \to v_a$.

The first observation implies that other than the edges contained in the paths P_i themselves, the only edges G can have are between pairs of paths, which we will call connecting edges. The second observation imposes a constraint on the connecting edges that may exist between any two paths. We use these observations to prove the following.

Lemma 6.2. $\Gamma(n,k) \leq kn - \binom{k+1}{2}$ *for all integers* $n \geq k \geq 1$.

Proof. Consider a graph G and dipaths $P_1, \ldots, P_k$ as above, where each path P_i has n_i vertices such that $D(G, P_1, \ldots, P_k)$ is acyclic. We first bound the number of connecting edges in G that may exist between each pair of paths P_i and P_j.

Define a function λ from the connecting edges of G to the integers as follows: For any connecting edge $v_a w_b$ where v_a is the a-th vertex of P_i and w_b is the b-th vertex of P_j, let $\lambda(v_a w_b) = a + b$. Consider two distinct connecting edges $v_a w_b, v_c w_d \in E(G)$ between the same two paths P_i and P_j, and we may assume that $a \leq c$. By Observation 2, if $a < c$, then $b \leq d$. Also, if $a = c$, then $b \neq d$. Therefore, $\lambda(v_a w_b) = a + b \neq c + d = \lambda(v_c w_d)$. This means that any two connecting edges between P_i and P_j have different images in the function λ. Since $2 \leq \lambda(e) \leq n_i + n_j$, there are at most $n_i + n_j - 1$ connecting edges between P_i and P_j.

Applying this to all pairs of paths P_i and P_j, the number of connecting edges in G is then bounded above by

$$\begin{aligned}
\sum_{1\leq i<j\leq k} (n_i + n_j - 1) &= \tfrac{1}{2}\left[\sum_{i=1}^{k}\sum_{j=1}^{k}(n_i + n_j - 1) - \sum_{i=1}^{k}(2n_i - 1)\right] \\
&= \tfrac{1}{2}\left[\sum_{i=1}^{k}(kn_i + n - k) - \sum_{i=1}^{k}(2n_i - 1)\right] \\
&= \tfrac{1}{2}[(kn + kn - k^2) - (2n - k)] \\
&= kn - n - \tfrac{1}{2}(k^2 - k).
\end{aligned}$$

Since the number of edges in the paths P_i themselves is $n - k$, the total number of edges G may have is at most $kn - k - \frac{1}{2}(k^2 - k) = kn - \binom{k+1}{2}$. □

6.2.2 Lower bound

Consider the following construction for any n and k. Let $n_1, n_2, \ldots, n_k$ be an integer partition of n such that $n_1 \leq n_2 \leq \cdots \leq n_k$. For each $1 \leq i \leq k$, let $P_i = v_{i,1}\, v_{i,2} \cdots v_{i,n_i}$. We then define $G(n_1, \ldots, n_k)$ to be the graph containing these paths, as well as the following edges for each $1 \leq i < j \leq k$:

(*i*) If $n_i > 1$, then for each $1 \leq r < n_i$, add the edge $v_{i,r}v_{j,r}$;

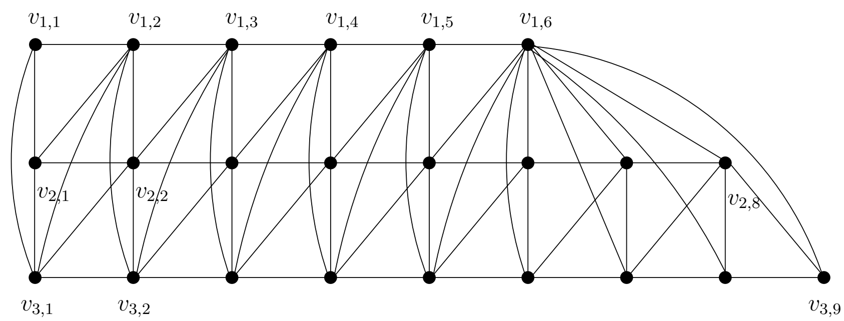

Figure 6.1: The graph $G(n_1, n_2, n_3)$ for $n_1 = 6$, $n_2 = 8$, $n_3 = 9$.

(ii) If $n_j > 1$, then for each $1 \leq r < n_i$, add the edge $v_{i,r+1}v_{j,r}$;

(iii) For each $n_i \leq r \leq n_j$, add the edge $v_{i,n_i}v_{j,r}$.

An example of this construction with $k = 3$ and $n_1 = 6$, $n_2 = 8$, $n_3 = 9$ is illustrated in Figure 6.1.

In constructing the digraph associated with $G = G(n_1, \ldots, n_k)$, the edge-rules (i) – (iii) for G yield the following arc-rules for $D(G, P_1, \ldots, P_k)$ for each $1 \leq i < j \leq k$:

$$(i) \begin{cases} \text{(a)} \ \ v_{i,r-1} \rightarrow v_{j,r} & \text{for } 1 < r \leq n_i \text{ (if } n_i > 1\text{), and} \\ \text{(b)} \ \ v_{j,r-1} \rightarrow v_{i,r} & \text{for } 1 < r \leq n_i \text{ (if } n_j > 1\text{);} \end{cases}$$

$$(ii) \begin{cases} \text{(c)} \ \ v_{i,r} \rightarrow v_{j,r} & \text{for } 1 \leq r < n_i - 1 \text{ (if } n_i > 1\text{), and} \\ \text{(d)} \ \ v_{j,r-1} \rightarrow v_{i,r+1} & \text{for } 1 < r \leq n_i - 1 \text{ (if } n_i > 1\text{);} \end{cases}$$

$$(iii) \begin{cases} \text{(e)} \ \ v_{i,n_i-1} \rightarrow v_{j,r} & \text{for } n_i \leq r \leq n_j\text{, and} \\ \text{(f)} \ \ v_{j,r-1} \rightarrow v_{i,n_i} & \text{for } \max\{n_i, 2\} \leq r \leq n_j \text{ (if } n_j > 1\text{).} \end{cases}$$

We can then prove:

Lemma 6.3. *The digraph $D(G, P_1, \ldots, P_k)$ described above is acyclic.*

Proof. Any arc produced by one of the rules (a) – (e) is of the form $v_{a,s} \rightarrow v_{b,r}$ with $s < r$ and no constraints on a and b, or $v_{a,r} \rightarrow v_{b,r}$ with $a < b$. In either case, we have $(s, a) < (r, b)$ in the lexicographic ordering on ordered pairs of integers. Then, if there are arcs in $D(G, P_1, \ldots, P_k)$ for $v_{b,r} \to v_{a,s}$ where $(r, b) > (s, a)$, they must arise from the rule (f), in which case $s = n_a$.

Note that none of the rules (a) – (f) produce arcs which leaves vertices v_{i,n_i} for any $1 \leq i \leq k$; then, there are no non-trivial walks which leave such a vertex. Then, it is easy to show by induction that if there is a directed walk between distinct vertices $v_{a,s}$ and $v_{b,r}$, either $(s, a) < (r, b)$ or $r = n_b$.

Let $v_{a,s}$ and $v_{b,r}$ be two vertices, with a directed walk W from $v_{a,s}$ to $v_{b,r}$. Because of the existence of W, we know that $s \neq n_a$; then, there is a directed walk from $v_{b,r}$ to $v_{a,s}$ only if $(r, b) < (s, a)$. We would then have $r = n_b$, in which case there are no directed walks from $v_{b,r}$ to *any* other vertices in $D(G, P_1, \ldots, P_k)$. So, for any two distinct vertices $v_{a,s}$ and $v_{b,r}$, there cannot be a directed walk from $v_{a,s}$ to $v_{b,r}$ and also from $v_{b,r}$ to $v_{a,s}$, in which case $D(G, P_1, \ldots, P_k)$ is acyclic. $\square$

As well as giving rise to an acyclic digraph $D(G, P_1, \ldots, P_k)$, we also have:

Lemma 6.4. $\left|E\big(G(n_1, \ldots, n_k)\big)\right| = kn - \binom{k+1}{2}$, *for any* $n \geq k \geq 1$ *and integer partition* $n_1 \leq \cdots \leq n_k$ *of* n.

Proof. Between any pair of paths P_i and P_j in $G(n_1, \ldots, n_k)$, there are $n_i - 1$ connecting edges of type (i), $n_i - 1$ connecting edges of type (ii), and connecting edges of type $n_j - n_i$. There are then $n_i + n_j - 1$ connecting edges between P_i and P_j, which saturates the upper bound for connecting edges between pairs of paths in Lemma 6.2. Summed over all pairs of paths and including the edges in the paths P_i themselves, the total number of edges in $G(n_1, \ldots, n_k)$ is $kn - \binom{k+1}{2}$. $\square$

Chapter 7

Future Work

There are several interesting possibilities for future research work. We begin with list colouring small Steiner triple systems. In Chapter 3, we have essentially solved the problem for STS of order at most 13. For STS(15), we have shown that such systems are almost 3-list-colourable. An obvious question is whether or not they are indeed 3-list-chromatic. We cannot do better with the hypergraph polynomial method, so it is very likely that a new technique using elementary methods is required to solve this problem. Beyond STS(15), the next order is 19. Now there are more than 11 million STS(19) (see [18]), so to compute hypergraph polynomials of every STS(19) is not feasible. However, the size of one STS(19) is still small enough that computing the coefficient of one hypergraph polynomial is still possible.

We have presented results on list colouring large Steiner triple systems in Chapter 4. In particular, we proved that the order of $\mathrm{ListSpec}_m(n)$ is between $\log n/\log\log n$ and $\log n$. It would be interesting to further narrow down this range. Also, our upper bound of $O(\sqrt{n\log n})$ for the list chromatic spectrum seems far from the lower bound of $\log n/\log\log n$. Our bound is driven by the upper bound on the chromatic spectrum, so any improvements in $\mathrm{Spec}_M(n)$ would improve $\mathrm{ListSpec}_M(n)$. Furthermore, the proof of Theorem 4.1 has the potential to extend to other designs, for example, Steiner quadruple systems, or triple systems where each pair of vertices is in k edges for some constant k.

We can ask the more general question of which parameters of a hypergraph dictate the behaviour of the list chromatic number. In graphs, the minimum degree is an important factor in the list chromatic number, as higher minimum degree implies higher list chromatic number. In hypergraphs, the minimum degree alone is not enough to determine a similar behaviour. It seems that the co-degree of a hypergraph plays a role as well, where the co-degree of a pair of vertices is the number of edges that contain both vertices. Steiner triple systems have co-degree 1 for each pair of vertices, and we have shown that in this case, the list chromatic number increases along with the minimum degree. However, we also gave examples with constant list chromatic number where the co-degree is at least $O(n)$. So one question is what role does the co-degree play in the behaviour of list chromatic number of hypergraphs?

Finally, in Chapter 5, we have proved that random graphs $G(n, c/n)$ where

$$c \geq \frac{10^7 d \log d \log^2(2/\varepsilon)}{\varepsilon}$$

a.a.s. contains all nearly-spanning trees of maximum degree d. We also mentioned that the order of c cannot be lower than $O(\log(1/\varepsilon))$. So the question is, can we improve c to the best possible bound? Also, we can ask whether or not this result extends to other types of graphs, for example, random bipartite graphs.

Appendix A

Computation Results for STS(15)

In this appendix, we record the computational results used in proving Theorem 3.8 in Section 3.4. We first briefly describe the code used in Maple for the computations, and then present the results for all 80 $STS(15)$s.

We first give the code used for generating a random hypergraph polynomial given a Steiner triple system. The procedure `randSTS15poly` requires `incmatrix` as an input matrix listing the vertices in each block. Then for each block in the STS, say with vertices $\{1, 2, 3\}$, the procedure randomly picks a number `k` between 0 and 5 to determine which factor this block contributes to the overall hypergraph polynomial. For example, if `k` is 0, then the factor it produces is $x_3 - x_1$; if `k` is 1, then the factor it produces is $x_3 - x_2$, etc.

```
randSTS15poly := proc (incmatrix)
temppolylong := 1;
for i from 1 to 35 do
  k := rand() mod 6;
  if (k = 0) then temppolylong := temppolylong *
    (x[incmatrix[i,3]] - x[incmatrix[i,1]]);
  elif (k = 1) then temppolylong := temppolylong *
    (x[incmatrix[i,3]] - x[incmatrix[i,2]]);
  elif (k = 2) then temppolylong := temppolylong *
    (x[incmatrix[i,1]] - x[incmatrix[i,2]]);
  elif (k = 3) then temppolylong := temppolylong *
    (x[incmatrix[i,1]] - x[incmatrix[i,3]]);
  elif (k = 4) then temppolylong := temppolylong *
    (x[incmatrix[i,2]] - x[incmatrix[i,3]]);
  else temppolylong := temppolylong *
```

```
    (x[incmatrix[i,2]] - x[incmatrix[i,1]]);
  end if;
end do;
RETURN(temppolylong)
end proc;
```

The following code asks Maple to first generate a random hypergraph polynomial `p` using `randSTS15poly`, and then find the coefficient for the monomial

$$M = x_0^3 x_1^3 x_2^3 x_3^3 x_4^3 x_5^2 x_6^2 x_7^2 x_8^2 x_9^2 x_a^2 x_b^2 x_c^2 x_d^2 x_e^2$$

in that polynomial. To find this coefficient, we utilize a Maple function called `coeftayl`, which finds the coefficient of an input term in a Taylor expansion of the input polynomial. We first find the "coefficient" of x_0^3 in `p` and call this `p[0]`. Now `p[0]` is actually a polynomial in the variables $x_1, \ldots, x_{14}$, and we then find the coefficient of x_1^3 in `p[0]` and call it `p[1]`. This process is repeated until x_{15}^2, in which case the result is the coefficient of M in `p`.

```
p := randSTS15poly(A);
p[0] := coeftayl(p, x[0]=0, [3]);
p[1] := coeftayl(p[0], x[1]=0, [3]);
p[2] := coeftayl(p[1], x[2]=0, [3]);
p[3] := coeftayl(p[2], x[3]=0, [3]);
p[4] := coeftayl(p[3], x[4]=0, [3]);
p[5] := coeftayl(p[4], x[5]=0, [2]);
p[6] := coeftayl(p[5], x[6]=0, [2]);
p[7] := coeftayl(p[6], x[7]=0, [2]);
p[8] := coeftayl(p[7], x[8]=0, [2]);
p[9] := coeftayl(p[8], x[9]=0, [2]);
p[10] := coeftayl(p[9], x[10]=0, [2]);
p[11] := coeftayl(p[10], x[11]=0, [2]);
p[12] := coeftayl(p[11], x[12]=0, [2]);
p[13] := coeftayl(p[12], x[13]=0, [2]);
p[14] := coeftayl(p[13], x[14]=0, [2]);
```

Note that technically speaking, we could use a `for` loop to accomplish this task, but mysteriously it was not working for us in Maple.

We now present a table which lists the results found by Maple. Note the following when interpreting this table:

- For each Steiner triple system, the incidence table of its block structure is presented first. The 15 vertices are labelled from 0 to 9 and a to e. Each column contains the three vertices of a block. The 80 systems and their blocks are listed in the same order as in [18].

- Right below the block structure is an encoding for the desired random polynomial found by Maple. For each block, say it is $[p, q, r]^T$ in the first table, and $[\alpha, \beta, \gamma]^T$, then it contributes the factor $\alpha x_p + \beta x_q + \gamma x_r$ to the polynomial. For example, in STS #1, the first block gives the factor $x_1 - x_0$, and the second block gives the factor $x_4 - x_3$.

- The "Coeff" listed for each system is the coefficient of the monomial M in the polynomial generated from above, as calculated by Maple.

We use STS #1 as an example. The random polynomial that Maple generated is

$$\begin{aligned} f(x) = &(x_1 - x_0)(x_4 - x_3)(x_0 - x_5)(x_7 - x_8)(x_0 - x_9)(x_c - x_0)(x_d - x_e) \\ &\cdot (x_3 - x_1)(x_4 - x_6)(x_1 - x_7)(x_1 - x_8)(x_b - x_1)(x_e - x_c)(x_2 - x_3) \\ &\cdot (x_2 - x_5)(x_2 - x_7)(x_9 - x_2)(x_2 - x_b)(x_d - x_c)(x_b - x_3)(x_c - x_8) \\ &\cdot (x_d - x_3)(x_e - x_a)(x_7 - x_c)(x_4 - x_8)(x_4 - x_9)(x_d - x_4)(x_7 - x_5) \\ &\cdot (x_8 - x_5)(x_5 - x_9)(x_a - x_c)(x_e - x_6)(x_8 - x_d)(x_6 - x_c)(x_6 - x_b). \end{aligned}$$

The coefficient of M in $f(x)$ is -3.

STS #1 Coeff = -3
`00000001111112222223333444455556666`
`13579bd3478bc3478bc789a789a789a789a`
`2468ace569ade65a9edbcdecbeddebcedcb`
`-0+0+-0-0++-0+++-+0-0-00++---+0-0++`
`+--+-0+++--+--0-0--0-0-+--0++-+0+00`
`0+0-0+-0-000+0-0+0+++++-00+000-+---`

STS #2 Coeff = -7
`00000001111112222223333444455556666`
`13579bd3478bc3478bc789a789a789a789a`
`2468ace569ade65a9edbcdecbededcbdebc`
`-0+0-00-00---++0--+0--00-0---+0+-0+`
`0-0+0+-+++00+-0-000-+0+-0+00+0-0+-0`
`++--+-+0--++00-+++-+0+-++-++0-+-0+-`

STS #3 Coeff = -1
`00000001111112222223333444455556666`
`13579bd3478bc3478bc789a789a789a789a`
`2468ace569ade65a9edbcdedebcedcbcbed`
`-00+-+00+0+-0+00+-0-++-++00--+++---`
`++-00--+0-0++--+0+-00-+-0++++-00+++`
`0-+-+0+--+-0-0+--0++-000---000--000`

STS #4 Coeff = -7
`00000001111112222223333444455556666`
`13579bd3478bc3478bc789a789a789a789a`
`2468ace569ade65a9edbcdedbecedcbcebd`
`-000+-0+-0-++---+-----+++00+-000+-0`
`0+++-0+-+-+00+++-0000+00--+-++-+00-`
`+---0+-00+0--0000++++0--0+-00-+--++`

STS #5 Coeff = −1

```
00000001111112222223333444455556666
13579bd3478bc3478bc789a789a789a789a
2468ace569ade65a9edbceddecbcbdeedbc
-----------------------------------
0-0+---00+-+0-+000--0-++-0--00--+0-
-+--0+0+-000-+0-++++++--+++0-++00+0
+0+0+0+-+-+-+0-+--00-0000-0++-0+--+
```

STS #6 Coeff = 1

```
00000001111112222223333444455556666
13579bd3478bc3478bc789a789a789a789a
2468ace569ade65a9edbceddecbcdbeebdc
-----------------------------------
++-+-00-0+0--0-+-0+0-+-+0-0+-0--0++
-00-+--0-0+00-+00+-+000-+0-0+++0+00
0-+00++++--+++0-+-0-+-+0-++-0-0+---
```

STS #7 Coeff = 20

```
00000001111112222223333444455556666
13579bd3478bc3478bc789a789a789a789a
2468ace569ade65a9edbdececbdcedbdbce
-----------------------------------
-+00+++++-0+0--+-----0---+--0-+0--0+
+-+-----0+---++-000+++00-00+0--+++-
00-+0000-0+0+000+++0-0++0++-+0+00-0
```

STS #8 Coeff = −1

```
00000001111112222223333444455556666
13579bd3478bc34789c78ab789a789a789a
2468ace569ade65abedc9deedcbdebcbcde
-----------------------------------
++-0-++-0+0-0+-+0+---++-00-0---00--
--+-00-+-0-0+00-+-0++--+-+++0++--00
000++-00+-++--+0-0+00000+-0-+00++++
```

STS #9 Coeff = 2

```
00000001111112222223333444455556666
13579bd3478bc34789c78ab789a789a789a
2468ace569ade65abedc9deedbcdecbbcde
-----------------------------------
0000-++-+0------+00++-0+-++00-0-++-
+++-0-0+--00000+-+---+--0--+-0++--0
---++0-00++++++00-+000+0+00-++-000+
```

STS #10 Coeff = −6

```
00000001111112222223333444455556666
13579bd3478bc34789c78ab789a789a789a
2468ace569ade65abedc9deedcbdcbebedc
-----------------------------------
0-0-+-00+-0-0---+-00--0++00+-00+0++
-0+0-0+-0+-0-++0-0--0++-0-+-+++-+00
++-+0+-+-0+++00+0++++0-0-+-00--0---
```

STS #11 Coeff = −13

```
00000001111112222223333444455556666
13579bd3478bc34789c789a789a78ab789a
2468ace569ade65abedcdbeecdbd9cebecd
-----------------------------------
+000+0-00-+-0+0-+-0+++00+00+-+-0+00
----0+++++-++0-+-++-00+-0++0+-++0+-
0+++--0--000--+000-0---+----000---+
```

STS #12 Coeff = −2

```
00000001111112222223333444455556666
13579bd3478bc34789c789a789a789a78ab
2468ace569ade65abedcdbeecdbbecdd9ce
-----------------------------------
00++--0--000--++-+-+00--++00+0+--0+
-+--0+-0++-+0+00+0+--+++0-+-0+0++-0
+-00+0++0-+-+0--0-00+-00-0-+---00+-
```

STS #13 Coeff = 9

```
00000001111112222223333444455556666
13579bd3478bc34789c789a789a78ab789a
2468ace569ade65abedcebdedcbd9cebcde
-----------------------------------
+0++0+0--+---+-+++00+0------++-00++
0--0--++0-+00-00-0-+0-++00++000-+0-
-+0-+0-0+00++0+-0-+--+00++00--++--0
```

STS #14 Coeff = 15

```
00000001111112222223333444455556666
13579bd3478bc34789c789a78ab789a789a
2468ace569ade65abedcebdd9ceedcbbcde
-----------------------------------
0+-0-+++0-++0++++0-+000+++++-+-0-00
--++00---0-0-0-00+00++--000-+0+++-+
+00-+-00++0-+-0---+---+0---00-0-0+-
```

STS #15 Coeff = -8

```
0000000111111222222333344445555666
13579bd3478bc34789a78ac789a789b789a
2468ace569ade65bcdee9bddbecadcecebd
```

```
0-0--0+-+++--0-+00++00-----+--+---+
-+-+0+00-00++++---00--0++0+0000+0+0
+0+0+--+0--00-00++--+++00+0-++-0+0-
```

STS #16 Coeff = -6

```
0000000111111222222333344445555666
13579bd3478bc34789a789a78bc789a789a
2468ace569ade65bcdeedcba9eddebccbed
```

```
0+-0--0++00000+++-0++-00-+0-00-++0-
+-0-+0-0--+++---00---++++0-0--00-++
-0++0++-0+---+00-++000--0-+++++-0-0
```

STS #17 Coeff = 7

```
0000000111111222222333344445555666
13579bd3478bc34789a789a789c78ab789a
2468ace569ade65bcdeedcbabedd9cecebd
```

```
---0+00+000--0++-0-0++0+0+-+00-00-0
00+---+0++-+++000-0+--+-+00-++++-+-
++0+0+----+00---+++-00-0--+0--0-+0+
```

STS #18 Coeff = 5

```
0000000111111222222333344445555666
13579bd3478bc34789a78ac789b789a789a
2468ace569ade65bcede9bdadcedebccbde
```

```
0-0--0++00-+0++-+000+++-++++0--++00
-++0+-0-+++-+-0+-++-0-0+-0-0-+0-0-+
+0-+0+-0--00-0-00--+-0-00-0-+0+0-+-
```

STS #19 Coeff = -3

```
0000000111111222222333344445555666
13579bd3478bc34789a789a78ab789c789a
2468ace569ade65bdceebdcc9deaebddceb
```

```
0+0-++-0+0+0+--++00++++00+++0--0+0-
+0++--0-0+--00+---+0000+---0-+0-0-+
---000++--0+-+000+------+00-+0++-+0
```

STS #20 Coeff = -31

```
0000000111111222222333344445555666
13579bd3478bc34789a78ac789b789a789a
2468ace569ade65bdece9bdacdedbcecebd
```

```
--000+-00+--+++0-0+-+++0-++0-++0--+
0++-+0+++-0+--0+0--+0-0-+0--000-++0
+0-+--0--0+000--++00-0-+0-0++--+00-
```

STS #21 Coeff = 12

```
0000000111111222222333344445555666
13579bd3478ac34789b789a789a78ab789c
2468ace569bde65acedbdcecedbd9ceeabd
```

```
--000+-00+--+++0-0+-+++0-++0-++0--+
0++-+0+++-00--0+0-0+000-+00-000-++0
+0-+--0--00000-00+0000000--+0--000-
```

STS #22 Coeff = 3

```
0000000111111222222333344445555666
13579bd3478ac34789b789a789c789a78ab
2468ace569bde65acedbdceeabdcedbd9ce
```

```
+-0--+-+0++00-+-+++++0+0+++++0+--++
00-0+-+----++0-+-0-0--0+-0-00--0+-0
-+++0000+00--+000-0-0+--0-0--+0+00-
```

STS #23 Coeff = -2

```
0000000111111222222333344445556667
13579bd3478bc34589c789a58ab789789aa
2468ace569ade67abedcdbed9cebececdbd
```

```
+-+0000+++---++-0000+++--0--0+000-+
00-+++-000000--+--++-00+0+00--++-00
-+0---+---+++000++--0--0+-+++0--++-
```

STS #24 Coeff = -4

```
0000000111111222222333344445556667
13579bd3478bc34589c789a589a78978aba
2468ace569ade67abedcdbebecdecdd9ceb
```

```
++0---0-0-0-+++0-++++00-++0-++0+000
0--0++-++++0-0--+---0++0-0-+0-+----
-0++00+0-0-+0-0+0000---+0-+0-0-0+++
```

STS #25 Coeff = 15

```
00000001111112222223333444455566667
13579bd3478bc34589c789a589a78978aba
2468ace569ade67abedcedbecbdbdcd9cee
+-0++0-0+0--0-0-+-00+++0---+00+0+-+
-0---+++-+++-0-+-+----0-+0+0-----0-
0++00-0-0-00+++000++00-+0+0-++0+0+0
```

STS #26 Coeff = −6

```
00000001111112222223333444455566667
13579bd3478bc3458ac789a589a789789ab
2468ace569ade67b9edcbededbcacddecbe
---+-0-+0-++00+---+++-000+00000-+++
00000++-++---+0++0-00+--+--+++-0---
+++-+-00-000+--00+0--0++-0+---++000
```

STS #27 Coeff = 12

```
00000001111112222223333444455566667
13579bd3478bc34589a78ab589c789789aa
2468ace569ade67bcded9ceaebdcdeebcdb
0+0+----+-000+++--00-+-++0+0--++0+-
--+-00++0+-+-0--+0+-00000+0-0+-0+-0
+0-0++00-0+-+-000+-++-+----++00--0+
```

STS #28 Coeff = 5

```
00000001111112222223333444455566667
13579bd3478bc34589a78ab589c789789aa
2468ace569ade67bcedd9ceaebdedccbdeb
+-00-+00--00++00++++0---0-0-++++-+0
00-++0++0+-+00---00-++00-+-0--0000+
-++-0---+0+---++0--0-0+++0++00--+--
```

STS #29 Coeff = 10

```
00000001111112222223333444455566667
13579bd3478bc34589a789a58bc789789aa
2468ace569ade67bcedebdca9eddeccdbeb
++-+++00-++-0-+++00+--0-+0-++00-0-0
-0+-00+++--0-+---++-00+00+00---++++
0-00----000++0000--0++-+--+-0++0-0-
```

STS #30 Coeff = −10

```
00000001111112222223333444455566667
13579bd3478bc34589a78ab589c789789aa
2468ace569ade67bdced9ceabedcedecbdb
+0--0-0+0+--0+0-0-+000-++-++--+++0-
0++++0----00-0+0++-++-+-0+--0+000++
--00-++0+0+++--+-00--+00-000+0----0
```

STS #31 Coeff = 8

```
00000001111112222223333444455566667
13579bd3478bc34589a789a58bc789789aa
2468ace569ade67bdcedbeca9edcedecbdb
0++++0+-0000+++-++++0-0---+0+-0+-+0
+0000-0++-++--0+0--0+0++++0+-0--+0+
-----+-0-+--00-0-00--+-000--0++00--
```

STS #32 Coeff = 7

```
00000001111112222223333444455566667
13579bd3478bc34589a78ab589c789789aa
2468ace569ade67bdcec9deaebddceebdcb
++0++-+-+++-00-0--0-00+++-0--+-+-++
0--00+00000+++0+0++0----0+++0000+--
-0+--0-+---0--+-+0-+++00-0-0+-+-000
```

STS #33 Coeff = −2

```
00000001111112222223333444455566667
13579bd3478bc34589a78ab589c789789aa
2468ace569ade67bdecc9deaebdecddbceb
+++--+0++++--++00-+++0+0++0000-+0+-
--0++--0-0-0+0---+0-0+--0-++-++-+0+
00-000+-0-0+0-0++0-0--0+-0--+-00--0
```

STS #34 Coeff = 1

```
00000001111112222223333444455566667
13579bd3478bc34589a789a589c78978baa
2468ace569ade67bdecdbceaebdecdc9edb
0+0-+++++++0+-++0+--+++0-0--0-+-0-0
--+000-0--0+0+0--00+-0-+++0+-0-0-0+
+0-+--0-00---0-0+-+00-0-0-+0++0+++-
```

STS #35   Coeff = 1
00000001111112222223333444455566678
13579bd3478bc34569a689a579a78a789bc
2468ace569ade78bcdedbecedcbc9daebed
+--++-+---00+0-------00-0-+++0+-0-+
-0+--+-0+++-0-000++0++-+-0000--++00
0+00000+00-+-++++00+0-+0++---+00-+-

STS #36   Coeff = 12
00000001111112222223333444455566678
13579bd3478bc34569a689a579a78a789bc
2468ace569ade78bceddbceecdbd9caebed
-0-0-0++0+-00---000+-000+0-++00----
+-0-0--0--0++++0+-+00+-+0+000--++0+
0+++++0-+0+--00+-+--+-+---+--++00+0

STS #37   Coeff = −7
00000001111112222223333444455566678
13579bd3478bc34569a68ab579c78978aa9
2468ace569ade78bced9dceaebdceddbebc
-0-0--+0++0+0000++-0---+000-+0-00+0
+++-0+0+00--++++0-0-00+0-++0--++-0+
0-0++0----+0-----0++++0-+--+0+0-+--

STS #38   Coeff = 1
00000001111112222223333444455566678
13579bd3478bc34569a68ab579a78978ac9
2468ace569ade78bdce9cdecedbadebecdb
+-+-00+++00-++++0+-0+-+0+00+00-+0+0
-+00--00-+++00--+0+-000--+----0--0+
00-+++--0--0--00--0+-+-+0-+0+++0+--

STS #39   Coeff = 1
00000001111112222223333444455566678
13579bd3478bc34569a68ab579c78978aa9
2468ace569ade78bdce9dceabedecdcebdb
0+00-+000-+-00++-00000+0+-+--+-+0-+
-0--0-++-+-+-+--+-+--+-+-0-+0-+--00
+-+++0--+000+-000+-++-0-0+00+000++-

STS #40   Coeff = −6
00000001111112222223333444455566678
13579bd3478bc34569a689a579a78a789bc
2468ace569ade78bdcecbedecdbd9caebed
+0-000+++-+0-00-0-+0++0-+--++0++-0+
0+0+++--0+0-+++0-00+0-++-++-0+-00--
--+---00-0-+0--+++---0-00000--0-++0

STS #41   Coeff = 7
00000001111112222223333444455566678
13579bd3478bc34569a689c57ab78a789a9
2468ace569ade78bdceabed9dceecdcebbd
++++-0+00-0-++0+-+0-0-+00+-+0--+00+
0---+-0++++00-+-00-0++0---00-0+--+0
-0000+---0-+-0-0+-++-0-++0+-++00+--

STS #42   Coeff = 2
00000001111112222223333444455566678
13579bd3478bc34569a68ab579c78978aa9
2468ace569ade78bdec9cdeaebddeccbebd
00-0-+0000+0-0-0+++-++-0+--0+0+++-0
++0-0-+---0-+-0+-0-0-0+-000---0--++
--+++0-+++-+0++-0-0+0-0+-+++0+-000-

STS #43   Coeff = 2
00000001111112222223333444455566678
13579bd3478bc34569a68ac579b78978aa9
2468ace569ade78bdec9ebdacdeedcbcedb
0+--00-0--00-0-00--++-0-0+-+00++--+
-000-+0-+0++0++--++0-+-+--0----00+-
+-+++-++0+--+-0++00-00+0+0+0++0-+00

STS #44   Coeff = 4
00000001111112222223333444455566678
13579bd3478bc34569a689a579a78a789bc
2468ace569ade78bdeccbdeedcbc9daebed
-0---+0-0++0-++-+++0+00-0----+-0-0-
+-++00+0--0-0--+0--+-++0+0+++-0+0+0
0+00+--++0-++000-00-0--+-+0000+-+-+

STS #45 Coeff = 1

00000001111112222223333444455566678
13579bd3478bc34569a68ac579a78978ab9
2468ace569ade78becd9bededbcacdcdbee
-00-0+0+00-0++--0-0+0---++--+-+0+-0
0+-0+-+0+++---00-++0++++00+00+0--+-
+-++-0----0+00+++0---000--0+-0-+00+

STS #46 Coeff = -7

00000001111112222223333444455566678
13579bd3478bc34569a68ab579c78978aa9
2468ace569ade78becd9dceabedceddcbeb
-+-+---+-00-+0000---0+++-0-00-00++-
00+00+00+-+0-+--+++++000+-0--+--0-+
+-0-+0+-0+-+0-++-000----0++++0++-00

STS #47 Coeff = 2

00000001111112222223333444455566678
13579bd3478bc34569a68ab579c78978aa9
2468ace569ade78bedc9cdeabeddeccdbeb
+0++0---0-0-+-+-+0-+-0+-0-000--++00
0----++++0++0000--00+-0+-0-++00-0+-
-+00+000-+-0-+-+0++-0+-0+++--++0--+

STS #48 Coeff = -7

00000001111112222223333444455566678
13579bd3478bc34568a69ab578c78979aa9
2468ace569ade79becd8dceabedcdedcbeb
+0---0-+--0-++--000-++0-----+00+0+0
--++0-000+-+0-0+-+-+---+++00---0+-+
0+00+++-+0+0-0+0+-+000+000++0++--0-

STS #49 Coeff = 6

00000001111112222223333444455566679
13579bd3478bc34568a689a578a78978abc
2468ace569ade79becddebccdbeadec9bed
-++00++-+---0+0+00+0-++--00++-0---0
00-----00++0---0-+0-0--0+++0-0++++-
+-0++00+-00++0+-+--++00+0---0+-000+

STS #50 Coeff = -4

00000001111112222223333444455566678
13579bd3478bc34568a69ac578a78979ab9
2468ace569ade79becd8bededbcadccdbee
-0+-00+-++-00++00-0++++-+0+-+0-+0+-
+-00---+00+---0++0+-00-00+-+0-+--0+
0+-+++00--0++0---+-0--0+--00-+00+-0

STS #51 Coeff = -3

00000001111112222223333444455566678
13579bd3478bc34568968ab579c79a78aa9
2468ace569ade7abecd9dce8ebddcecbdbe
-0+++0+-0-+0-+-0++-+++00+000+-0+--+
0-----0+++0-0-++--+---+---++0+--0+0
++000+-0-0-++00-000000-+0+---0+0+0-

STS #52 Coeff = 1

00000001111112222223333444455566678
13579bd3478bc345689689a579c79a78aab
2468ace569ade7abecdcdeb8ebddceb9dce
000+++0+-0+0---+-0--++0+0+-+++0+0--
-++--0+-0+--00000-0+----+-0-0---+0+
+--00--0+-0++++-+++000+0-0+0-0+0-+0

STS #53 Coeff = 3

00000001111112222223333444455566678
13579bd3478bc345689689a579c79a78aab
2468ace569ade7abecdbdec8ebddcec9dbe
-0++0000-+0-0-000-++--+-0+--+--+--0
0+-----++-++-0---0-000-0+-++0+00+0-
+-00+++-00-0++++++0-++0+-000-0+-0++

STS #54 Coeff = -1

00000001111112222223333444455566679
13579bd3478bc345689689a578c78a78aab
2468ace569ade7abecdbecd9ebdcded9cbe
0+++-0-0+--0--+0+0++-+000-+-0+00-0+
+-0-0+0+-+++0+-+0-0-+---++00----0-0
-0-0+-+-000-+00--+-000++-0-++0++++-

STS #55 Coeff = 5

```
00000001111112222223333444455566678
13579bd3478bc34568969ab578c78a79aa9
2468ace569ade7abecd8cde9ebdcdedbcbe
+---0+000+0----+0--++-0+-----+++000
-++0--+++-+0+0+--00--0+00+000000++-
000++0---0-+0+00+++00+--+0+++-----+
```

STS #56 Coeff = 1

```
00000001111112222223333444455566678
13579bd3478bc345689689a579c79a78aab
2468ace569ade7abedcdceb8ebdcdeb9cde
-0+000--+---0-0+0-000-+++-0-0++-+--
0-0++++00++0++--+++-++0-00-++--0-+0
++----0+-00+-0+0-0-+-0-0-++0-00+00+
```

STS #57 Coeff = 5

```
00000001111112222223333444455566678
13579bd3478bc34568968ab579c79a78aa9
2468ace569ade7abedc9cde8ebdcdedbcbe
++-+-00-+-++-0+00+-+00-0--00-000-++
-0+-+-+0-+0-++--+00--++++++++++0--
0-000+-+00-00-0+--+0+-0-00--0---+00
```

STS #58 Coeff = 13

```
00000001111112222223333444455566678
13579bd3478bc345689689a579c79a78aab
2468ace569ade7abedcbced8ebdcded9cbe
+0+-0++-0++--+00-+00+0-000----00+++
--0++00+-0-000--+--+-+0+-++00++-0-0
0+-0---0+-0++-++00+-0-+-+-0++0-+-0-
```

STS #59 Coeff = 1

```
0000000111111222222333344444555666 7a
13579bd3478bc345689689a578a789789cb
2468ace569ade7ebacdcbeddb9caecedbde
-----+-0++++0++-----+-++0---+--+-00
0++000+-00-----+0+000+-0-+++-++0++-
+00++-0+--00+000+0++-00-+000000-0-+
```

STS #60 Coeff = 2

```
0000000111111222222333344444555666 7a
13579bd3478bc34568968a9578a789789cb
2468ace569ade7ebacddecbcb9dadeebcde
--0+-+000-0--0--++00-+++0-0++-0+--+
0++-+0-+++-+++00-0+++000+++0-+-000-
+0-00-+--0+00-++0---0----0--00+-++0
```

STS #61 Coeff = −10

```
00000001111112222223333444455556666
13579bd3478ac34789b789c789a78ab789a
2468ace569bde65aecdbaedecdbd9cecdbe
-00+0+0-+--+0-+0-00-0+0+0+-0-+--++0
0----0+000+--+0-0++++-+0+0+-00++00+
+++0+--+-+00+0-++--0-0----0++-00---
```

STS #62 Coeff = −4

```
00000001111112222223333444455566667
13579bd3478ac34589b78ab589a789789ca
2468ace569bde67adcec9edbedcdceeabdb
+-+0-000+++---0--+0--+--00+++000---
000-0-+-0--0+0++0-++0-+0++-00---++0
-+-+++-+-00+0+-0+0-0+00+--0--+++00+
```

STS #63 Coeff = −3

```
00000001111112222223333444455566667
13579bd3478ac34589b78ab589a789789ca
2468ace569bde67aecdd9cebcdecdeeabdb
00-00-++-0-+--0+--+--0--0----00--0-
--++++-0+++-00+0+00+0+00+00+0++00-+
++0--00-0-00++--0+-0+-++-++0+--+++0
```

STS #64 Coeff = −2

```
00000001111112222223333444455566667
13579bd3478ac34589a789a589b78b789ca
2468ace569bde67cdbeecdbaecdd9ebaedc
+0-++-++0+0-+-+-000-00----0++0-+00-
--+00000+-++0+-++++++-++00-00+0-++0
0+0--+---0-0-000---0-+00+++---+0--+
```

STS #65 Coeff = -21
00000001111112222223333444455566678
13579bd3478ac34569b68ac579a789789ba
2468ace569bde78dacee9bdbedcacecdbde
---0-+0++0+00--+0-+++-00++++-+-+0-0
++++00--0+0+-++0-0-0-0++0-00+-00++-
000-+-+0----+00-++0-0+---0--00+--0+

STS #66 Coeff = -3
00000001111112222223333444455566678
13579bd3478ac34568968ac579a79b789ba
2468ace569bde7bdacee9bd8edcacecdbde
+00+--+++++-0-+++0-0+--+-+0+0+-++--
-+--++000000++---+++00000-+0+-+000+
0-+000-----+-0000-0--++-+0---00--+0

STS #67 Coeff = -52
00000001111112222223333444455566679
13579bd3478ac34568a68ac578b789789ab
2468ace569bde79cbdee9bdaecdbeddacce
-00-++0++-+-0-0+-+0--0-+++-+-+00---
++-+----00-0+++-0-++++00-00-+-++0+0
0-+000+0-+0+-0-0+0-00-+-0-+000--+0+

STS #68 Coeff = -5
00000001111112222223333444455566678
13579bd3478ac34569b68ac579a789789ba
2468ace569bde78adcee9bdbedccdeacbde
-0-++0+0-0--++++-0000-++000-0-++0-0
0-+-0+--+-++0---0-+--+---+-+-0--++-
++00--0+0+00-000++-++000+-+0++00-0+

STS #69 Coeff = -5
00000001111112222223333444455566679
13579bd3478ac3456a868ac578b789789ab
2468ace569bde79bcede9bdaecddecbadce
+-+-0+00+-00+---+-0++00-+-+0+-+--+-
00-0--++-+++000+0+-00--+00-+-000+-0
-+0++0--00---++0-0+--++0-+0-0+-+00+

STS #70 Coeff = 9
0000000111111222222333344445556667a
13579bd3478ac34568968ab5789789789cb
2468ace569bde7dbaece9cdceabadebcdde
+--00++---+00--000-+00-+0-0---0000+
-+0-+0-+++0+-00+--0-+++--0-0+0++--0
00++--0000--+++-+++0--00++++0+--++-

STS #71 Coeff = -3
0000000111111222222333344445556667a
13579bd3478ac34568968ab5789789789cb
2468ace569bde7cabdee9cddeabbecacdde
-00-+++0--00-0+000++0+--+++++0-0-++
0+-0---+0+-+0--++-----00000---0++-0
+-++000-+0+-++0--+00+0++---00++-00-

STS #72 Coeff = 2
0000000111111222222333344445556667a
13579bd3478ac34568968ab5789789789cb
2468ace569bde7dcbaee9cdaecbbedadcde
++-+-0+-+00+-0--0-0++0++0+---+-00+-
--0-0-0+----+-00-+--0+-0+0+++-++-0+
00+0++-00++00++++0+0--0---00000-+-0

STS #73 Coeff = -7
00000001111112222223333444455566679
13579bd3478ac34568a689a578b78c789ab
2468ace569bde7c9bdeecdbae9dbeddacce
--00--0-+++00-+++-++-+0-0+-0+-+0-++
00+-+0-000-+++0--+-00--+--0+-0-+0--
++-+0+++--0--0-0000-+0+0+0+-0+0-+00

STS #74 Coeff = 4
0000000111111222222333344445556667a
13579bd3478ac345698689a578b789789cb
2468ace569bde7abdecedbcce9daedbacde
00-00--+-++-000+0+0--+--+00+0--+++0
-+++-+00000++---+--++00+0+---++---+
+-0-+0+-+--0-++0-0+00-+0--+0+00000-

STS #75 Coeff $= 1$

```
000000011111122222233334444555666 7a
```

```
0000000111111222222333344445556667a
13579bd3478ac345689689a578b789789cb
2468ace569bde7cdbaeedbcae9dbecacdde
```

```
-0-0++-+000+++-----0+-++-+0++++0+-+
++0-00+0+-----0++0+-00--0--0-0---0-
0-++--0--++000+00+0+-+00+0+-0-0+0+0
```

STS #76 Coeff $= -3$

```
00000001111112222223333444455566678
13579bd3478ac34569b68ab578978979aac
2468ace569bde7da8cee9cdcbaedebcdbed
```

```
-0-+-00-0+00+0------0+-+-+0-00--0+-
+-+-0-+0---+-+++++0++-0-+-+0-+00--0
0+00++-++0+-0-0000+0-0+000-++-+++0+
```

STS #77 Coeff $= 2$

```
00000001111112222223333444455566678
13579bd3478ac34569a68ab578979b789ac
2468ace569bde7d8cebe9cdacebdcebaded
```

```
0+0+-0-+-0-+00--+-0-+++--+00+---+0+
----0++0+-0-+-+0-0-0--0000+-000+0-0
+0+0+-0-0++0-+0+0+++00-++--+-++0-+-
```

STS #78 Coeff $= 2$

```
00000001111112222223333444455566678
13579bd3478ac34569b689c578b78a79aa9
2468ace569bde7ac8edeabd9cdedebbdcec
```

```
0+++-0+00++00+00++0-0+++-0++0-0+++-
--000+0---0-+0-+0--+--000+-0++----0
+0--+--++0-+--+--0+0+0--+-0--0+000+
```

STS #79 Coeff $= -7$

```
00000001111112222223333444455566678
13579bd3478ac34568b689c57ab79a789a9
2468ace569bde79ecadaebd8dcecdbbdeec
```

```
00--0+---+-0--0-++-0--0-+-0-0--0-++
+-00--+0+0+-00++000-++++00+0-++-+00
-++++00+0-0+++-0--++00-0-+-++00+0--
```

STS #80 Coeff $= -7$

```
00000001111112222223333444455566678
13579bd3469ac34578b678a58ab78979c9a
2468ace578bde96aecdbcded9cebecaeddb
```

```
-+--0-0-0+0--0+0-0-0+-----+-00-00++
+000++++---+0+--0-0+0000++-0++0+-00
0-++-0-0+0+0+-0++++--+++000+--+-+--
```

Bibliography

[1] Ajtai, M., Komlós, J. and Szemerédi, E., The longest path in a random graph, *Combinatorica* 1(1981), 1-12.

[2] Alon, N., Restricted colorings of graphs, *Surveys in Combinatorics 1993 (Keele), London Math. Soc. Lect. Notes* 187 (Cambridge Univ. Press, 1993), 1-33.

[3] Alon, N., Combinatorial Nullstellensatz, *Combin. Probab. Comput.* 8(1999), 7-29.

[4] Alon, N., Degrees and choice numbers, *Random Structures Algorithms* 16(2000), 364-368.

[5] Alon, N. and Chung, F.R.K., Explicit construction of linear sized tolerant networks, *Discrete Math.* 72(1988), 15-19.

[6] Alon, N., Krivelevich, M. and Sudakov, B., Embedding nearly-spanning bounded degree trees, *Combinatorica* 27(2007), 629-644.

[7] Alon, N. and Spencer, J.H., *The Probabilistic Method*, Wiley-Interscience, 2000.

[8] Alon, N. and Tarsi, M., Colorings and orientations of graphs, *Combinatorica* 12 (1992), 125-134.

[9] Beck, J., On size Ramsey number of paths, trees, and circuits I, *J. Graph Theory* 7(1983), 115-129.

[10] Beck, J., On size Ramsey number of paths, trees and circuits II, in Mathematics of Ramsey Theory (Nešetřil, J. and Rödl, V., eds), *Algorithms and Combinatorics* 5(1990), Springer-Verlag, 34-45.

[11] Benzaken, C., Gravier, S. and Škrekovski, R., Hajós' theorem for list colorings of hypergraphs, *Discuss. Math. Graph Theory* 23(2003), 207-213.

[12] Bollobás, B., *Extremal Graph Theory*, New York: Academic Press, 1978.

[13] Bollobás, B., Long paths in sparse random graphs, *Combinatorica* 2(1982), 223-228.

[14] Bollobás B. and Harris, A.J., List colorings of graphs, *Graphs and Combinatorics* 1(1985), 115-127.

[15] Bose, R.C., On the construction of balanced incomplete block designs, *Ann. Eugenics* 9(1939), 353-399.

[16] Brandt, S. and Dobson, E., The Erdős-Sós conjecture for graphs of girth 5, *Discrete Math.* 150(1996), 411-414.

[17] Chernoff, H., A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations, *Annals of Mathematical Statistics* 23(1952), 493-50.

[18] Colbourn, C.J. and Dinitz, J.H., editors, *Handbook of Combinatorial Designs, Second Edition*, Chapman & Hall 2007.

[19] Colbourn, C.J. and Rosa, A., *Triple Systems*, Oxford 1999.

[20] Danos, V. and Kashefi, E., Determinism in the one-way model, *Phys. Rev. A* 74(2006), 052310.

[21] Danos, V., Kashefi, E. and Panangaden, P., The Measurement Calculus, *J. ACM* 54(2007).

[22] de Beaudrap, N., A complete algorithm to find flows in the one-way measurement model, arXiv:quant-ph/0603072 (2006).

[23] de Beaudrap, N. and Pei, M., An extremal result for geometries in the one-way measurement model, *Quantum Information and Computation* 8(2008), 430-437.

[24] de Brandes, M., Phelps, K.T. and Rödl, V., Coloring Steiner triple systems, *SIAM J. Algebraic Discrete Methods* 3(1982), 241-249.

[25] Diestel, R., *Graph Theory*, Springer-Verlag 1997.

[26] Dobson, E., Constructing trees in graphs whose complement has no $K_{2,s}$, *Combin. Probab. Comput.* 11(2002), 343-347.

[27] Erdős, P., Extremal problems in graph theory, Theory of graphs and its applications, Fiedler, M. (Ed.), Academic Press, New York, 1965, pp. 29-36.

[28] Erdős, P. and Hajnal, On chromatic number of graphs and set systems, *Acta Math. Acad. Sci. Hung.* 17(1966), 61-99.

[29] Erdős, P. and Lovász, L., Problems and results on 3-chromatic hypergraphs and some related questions, in *Infinite and Finite Sets* (Hajnal, A., Rado, R. and Sós, V.T., eds.), Colloq. Math. Soc. J. Bolyai 11, North Holland, Amsterdam, (1975)609-627

[30] Erdős, P., Rubin, A.L. and Taylor, H., Choosability in graphs, *West Coast Conference on Combinatorics, Graph Theory and Computing* (1979), 125-157.

[31] Fernandez de la Vega, W., Long paths in random graphs, *Studia Sci. Math. Hungar.* 14(1979), 335-340.

[32] Fernandez de la Vega, W., Trees in sparse random graphs, *J. Combin. Theory Ser. B* 45(1988) 77-85.

[33] Fleischner, H. and Stiebitz, M., A solution to a coloring theorem of P. Erdős, *Discrete Math.* 101(1992), 39-48.

[34] Friedman, J. and Pippenger, N., Expanding graphs contain all small trees, *Combinatorica* 7(1987), 71-76.

[35] Frieze, A., On large matchings and cycles in sparse random graphs, *Discrete Math.* 59(1986), 243-256.

[36] Galvin, F., The list chromatic index of bipartite multigraphs, *J. Combin. Theory Ser. B* 63(1995), 153-158.

[37] Garey, M.R. and Johnson, D.S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman 1979.

[38] Haddad, L., On the chromatic numbers of Steiner triple systems, *J. Combin. Des.* 7(1999), 1-10.

[39] Haxell, P.E., Tree embeddings, *J. Graph Theory* 36(2001), 121-130.

[40] Haxell, P.E. and Kohayakawa, Y., The size-Ramsey number of trees, *Israel Journal of Mathematics* 89(1995), 261-274.

[41] Horak, P., On the chromatic number of Steiner triple systems of order 25, *Discrete Math.* 299(2005), 120-128.

[42] Janson, S., Luczak, T. and Ruciński, A., *Random Graphs*, Wiley-Interscience, 2000.

[43] Jensen, T.R. and Toft, B., *Graph Coloring Problems*, Wiley 1995.

[44] Kahn, J., Asymptotically good list-colorings, *J. Combin. Theory Ser. A* 73(1996), 1-59.

[45] Kostochka, A.V. and Nešetřil, J., Properties of Descartes' construction of triangle-free graphs with high chromatic number, *Combin. Prob. Comp.* 8(1999), 467-472.

[46] Kostochka, A.V., Stiebitz, M. and Wirth, B., The colour theorems of Brooks and Gallai extended, *Discrete Math.* 162(1996), 299-303.

[47] Kratochvíl, J., Tuza, Zs. and Voigt, M., New trends in the theory of graph colorings: choosability and list coloring, *Contemporary Trends in Discrete Mathematics (Štiřín Castle, 1997)* (1999), 183-197.

[48] Kubale, M., Some results concerning the complexity of restricted colorings of graphs, *Discrete Applied Math.* 36(1992), 35-46.

[49] Mahadev, N.V.R., Roberts, F.S. and Santhanakrishnan, P., 3-choosable complete bipartite graphs, *DIMACS Technical Report* 91-62 (1991).

[50] Mathon, R.A., Phelps, K.T. and Rosa, A., Small Steiner triple systems and their properties, *Ars Combin.* 15(1983), 3-110.

[51] McLennan, A., The Erdős-Sós conjecture for trees of diameter four, *J. Graph Theory* 49(2005), 291-301.

[52] Nielsen, M.A. and Chuang, I.L., *Quantum Computation and Quantum Information*, Cambridge University Press 2000.

[53] Papadimitriou, C.H., *Computational Complexity*, Addison Wesley Longman 1994.

[54] Phelps, K.T. and Rödl, V., Steiner triple systems with minimum independence number, *Ars Combin.* 21(1986), 167-172.

[55] Ramamurthi, R. and West, D.B., Hypergraph extension of the Alon-Tarsi list coloring theorem, *Combinatorica* 25(2005), 355-366.

[56] Raussendorf, R. and Briegel, H.J., A one-way quantum computer, *Phys. Rev. Letters* 86(2001), 5188-5191.

[57] Raussendorf, R. and Briegel, H.J., Computational model underlying the one-way quantum computer, *Quantum Information and Computation* 2(2002), 443-486.

[58] Raussendorf, R., Browne, D.E. and Briegel, H.J., Measurement-based quantum computation on cluster states, *Phys. Rev. A* 68(2003) 022312.

[59] Robertson, N., Sanders, D.P., Seymour, P.D. and Thomas, R., The four colour theorem, *J. Combin. Theory Ser. B* 70(1997), 2-44.

[60] Rosenberg, A.L., Fault-tolerant interconnection networks, a graph-theoretic approach, *Workshop on Graph-Theoretic Concepts in Computer Science* (1983) (Trauner Verlag, Linz) 286-297.

[61] Ryser, H.J., A note on a combinatorial problem, *Proceedings of the American Mathematical Society* 1(1950), 422-424.

[62] Saclé, J-F. and Wozniak, M., The Erdős-Sós conjecture for graphs without C_4, *J. Combin. Theory Ser. B* 70(1997), 367-372.

[63] Skolem, Th., Some remarks on the triple systems of Steiner, *Mathematica Scandinavia* 6(1958), 273-280.

[64] Stinson, D.R., *Combinatorial Designs: Constructions and Analysis*, Springer 2004.

[65] Thomassen, C., Every planar graph is 5-choosable, *J. Combin. Theory Ser. B* 62(1994), 180-181.

[66] Tuza, Zs., Graph colorings with local constraints - a survey, *Discuss. Math. Graph Theory* 17(1997), 161-228.

[67] Vizing, V.G., Coloring the vertices of a graph in prescribed colors (in Russian), *Diskret. Analiz.* 29(1976), 3-10.

[68] Voigt, M., List colorings of planar graphs, *Discrete Math.* 120(1993), 215-219.

[69] West, D.B., *Introduction to Graph Theory*, Prentice-Hall 1996.

[70] Yin, J. and Li, J., The Erdős-Sós conjecture for graphs whose complements contain no C_4, *Acta Math. Appl. Sin. Engl. Ser.* 20(2004), 397-400.