8-2023

A machine learning approach to constructing Ramsey graphs leads to the Trahtenbrot-Zykov problem.

Emily S. Hawboldt
University of Louisville

A MACHINE LEARNING APPROACH TO CONSTRUCTING RAMSEY GRAPHS LEADS TO THE TRAHTENBROT-ZYKOV PROBLEM

By

Emily S. Hawboldt
B.M., University of Louisville, 2017
M.A., University of Louisville, 2019

A Dissertation
Submitted to the Faculty of the
College of Arts and Sciences of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy
in
Applied and Industrial Mathematics

Department of Mathematics
University of Louisville
Louisville, Kentucky

August 2023

A MACHINE LEARNING APPROACH TO CONSTRUCTING RAMSEY GRAPHS LEADS TO THE TRAHTENBROT-ZYKOV PROBLEM

Submitted by

Emily S. Hawboldt

A Dissertation Approved on

June 22, 2023

by the Following Dissertation Committee:

Dr. André Kézdy,
Dissertation Director

Dr. Robert Powers

Dr. D. Jacob Wildstrom

Dr. Gerard Williger

DEDICATION

To my future students. I look forward to more mathematical adventures with them.

ACKNOWLEDGMENTS

The machine learning aspect of my project was supported in part by the University of Louisville Graduate Student Council. Thanks also to the Department of Mathematics for funding the rest of my deep learning workstation and to Joel for helping assemble it.

It was a joy to talk to my longtime friend Karissa Jackson (Freedom House; MA, Indiana University REEI) about mathematics as she helped with the Bulitko translation.

Thank you to the UofL Counseling Center, particularly Michelle and Gabrielle. Thank you to the Disability Resource Center, and to Dr. Alica Miller for encouraging me to go. These resources helped me unlock potential at every step of my studies.

Dr. Adam Jobson was my first mentor in grad school. One of the greatest things he taught me was how to program.

Many thanks to the members of my committee for their thoughtful comments on my work. In particular, I thank Dr. Powers for encouraging me to pursue graduate studies in mathematics when I met him during my senior year as a music education major. It's never too late to learn more about mathematics.

Special thanks also to Dr. Kézdy for being an excellent mentor. The COVID-19 pandemic presented a unique set of challenges and I'm grateful to have had Dr. Kézdy's support during such a strange time. I will miss our conversations and work when I move on to my next chapter.

I'm grateful to the many folks in my life who have shown tremendous support over the last several years. It would be hard to name them all, but I'm confident they know who they are.

The greatest supporter of them all is my husband Dan. I'm sure we'll continue occasionally discussing graph theory over dinner for many years to come.

ABSTRACT

A MACHINE LEARNING APPROACH TO CONSTRUCTING RAMSEY GRAPHS LEADS TO THE TRAHTENBROT-ZYKOV PROBLEM

Emily S. Hawboldt

June 22, 2023

Attempts at approaching the well-known and difficult problem of constructing Ramsey graphs via machine learning lead to another difficult problem posed by Zykov in 1963 (now commonly referred to as the Trahtenbrot-Zykov problem): For which graphs F does there exist some graph G such that the neighborhood of every vertex in G induces a subgraph isomorphic to F?

Chapter 1 provides a brief introduction to graph theory. Chapter 2 introduces Ramsey theory for graphs. Chapter 3 details a reinforcement learning implementation for Ramsey graph construction. The implementation is based on board game software, specifically the AlphaZero program and its success learning to play games from scratch. The chapter ends with a description of how computing challenges naturally shifted the project towards the Trahtenbrot-Zykov problem. Chapter 3 also includes recommendations for continuing the project and attempting to overcome these challenges.

Chapter 4 defines the Trahtenbrot-Zykov problem and outlines its history, including proofs of results omitted from their original papers. This chapter also contains a program for constructing graphs with all neighborhood-induced subgraphs isomorphic to a given graph F. The end of Chapter 4 presents constructions from the program when F is a Ramsey graph. Constructing such graphs is a non-trivial

task, as Bulitko proved in 1973 that the Trahtenbrot-Zykov problem is undecidable. Chapter 5 is a translation from Russian to English of this famous result, a proof not previously available in English.

Chapter 6 introduces Cayley graphs and their relationship to the Trahtenbrot-Zykov problem. The chapter ends with constructions of Cayley graphs Γ in which the neighborhood of every vertex of Γ induces a subgraph isomorphic to a given Ramsey graph, which leads to a conjecture regarding the unique extremal Ramsey(4, 4) graph.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

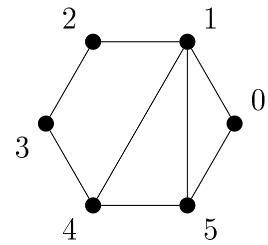
This dissertation connects two topics in graph theory: Ramsey graphs and the Trahtenbrot-Zykov (T-Z) problem. Broadly speaking, Ramsey theory deals with the inevitability of certain substructures as the size of a larger structure grows. The T-Z problem asks about global structures that admit uniform local structures. The problems are defined more precisely in their respective chapters – Chapter 2 for Ramsey graphs, and Chapter 4 for the T-Z problem.

This chapter covers basic definitions and notation. Chapter 2 introduces Ramsey theory as it relates to simple graphs and cliques. Chapter 3 outlines a method for generating Ramsey graphs by using machine learning, specifically reinforcement learning. Attempts at this machine learning implementation lead to a perspective on Ramsey graphs rooted in the T-Z problem.

Definition 1 (Graph)**.** *A* graph $G = (V, E)$ *consists of a vertex set* V *and a collection* E *of two-element subsets of these vertices, called edges. The vertex and edge set of* G *are denoted* $V(G)$ *and* $E(G)$*, respectively.*

Only simple, undirected graphs are considered in this work, i.e. graphs without loops or multiple edges. The *order* of a graph G is its number of vertices and is denoted $|G|$. The *size* of a graph G is its number of edges and is denoted $||G||$.

Example 1 (Graph)**.** Let G be the graph shown below:



- The vertex set is $V(G) = \{0, 1, 2, 3, 4, 5\}$.
- The edge set is $E(G) = \{\{0,1\}, \{0,5\}, \{1,2\}, \{1,4\}, \{1,5\}, \{2,3\}, \{3,4\}, \{4,5\}\}$.
- G is a graph of order 6 with size 8; that is, $|G| = 6$ and $||G|| = 8$.

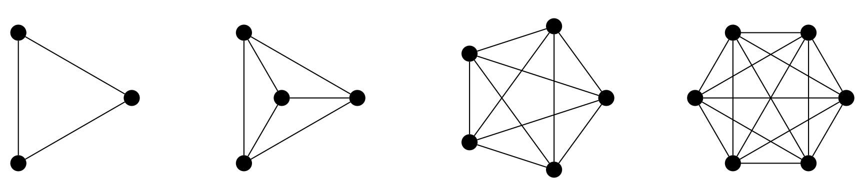
$\triangle$

Definition 2 (Adjacency)**.** *Let G be a graph with $u, v \in V(G)$. If $\{u, v\} \in E(G)$, then u and v are* adjacent *in G.*

The following class of graphs is important for both Ramsey theory and the T-Z problem.

Definition 3 (Complete graph)**.** *A* complete graph *is a graph in which all vertices are pairwise adjacent. Write K_n to denote the complete graph of order n.*

Example 2 (Complete graphs)**.** The complete graphs K_3, K_4, K_5, and K_6 are shown below:

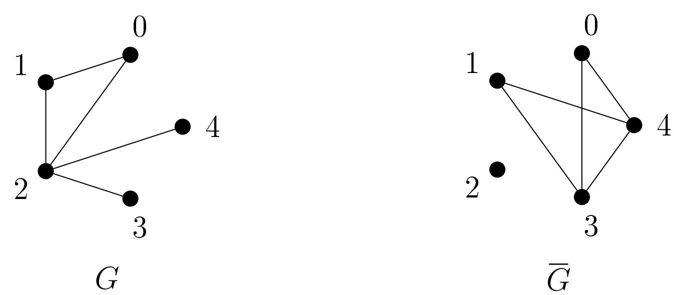


$\triangle$

Complete graphs include all possible edges. For graphs that aren't complete graphs, one might ask what kind of graph is formed by the edges not present in the graph. This coincides with the notion of graph *complement*.

Definition 4 (Graph complement). *Let G be a graph. The* complement *of G, denoted $\overline{G}$, is the graph with $V(\overline{G}) = V(G)$ and $\{u,v\} \in E(\overline{G})$ if and only if $\{u,v\} \notin E(G)$.*

Example 3 (Complement). A graph G and its complement are shown below:



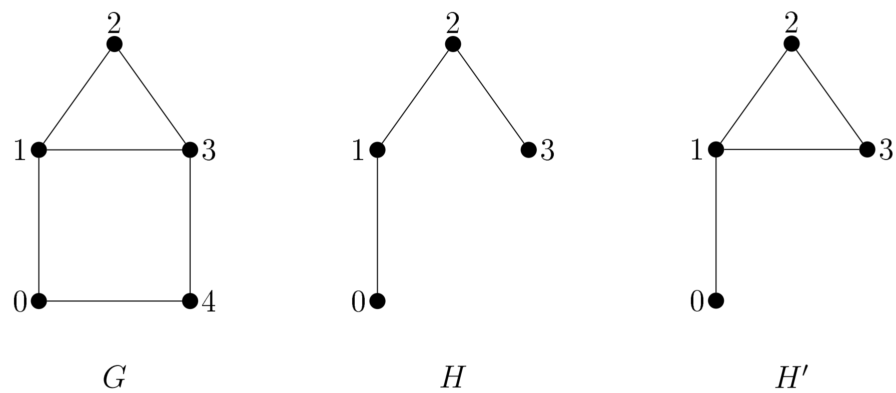
△

Ramsey theory and the T-Z problem both address graph substructures. There are two key types of substructures to consider: subgraphs, and induced subgraphs.

Definition 5 (Subgraph). *Let G and H be graphs. If $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$, then H is a* subgraph *of G.*

Definition 6 (Induced subgraph). *Let G and H be graphs. If $V(H) \subseteq V(G)$ and $\{u,v\} \in E(H)$ if and only if $\{u,v\} \in E(G)$ for all $u, v \in V(H)$, then H is an* induced subgraph *of G.*

Example 4 (Subgraph, induced subgraph)**.** Consider the following graphs:



Observe that H is a subgraph of G, but it is not an induced subgraph of G. On the other hand, H' is an induced subgraph of G. △

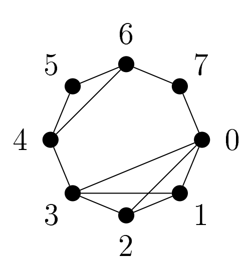
Certain subgraphs are of enough interest that they have special names; cliques and independent sets are two such subgraphs.

Definition 7 (Clique)**.** *A* clique *is a complete subgraph.*

Definition 8 (Independent set)**.** *An* independent set *(also called a* stable set *or* coclique*) is a set of vertices in a graph such that no two vertices in the set are adjacent to one another.*

In other words, a set of vertices in a graph forms an independent set if the vertices induce a clique in $\overline{G}$. Hence $\overline{K_m}$ is sometimes written to denote an independent set of order m.

Example 5. Consider the following graph:



Vertices $\{0, 1, 2, 3\}$ form a clique of order 4. The vertices $\{4, 5, 6\}$ form a clique of order 3. Each edge corresponds to a clique of order 2, and each vertex corresponds to a clique of order 1.

The vertices $\{1, 4, 7\}$ form an independent set of order 3. $\triangle$

Example 5 contains multiple cliques. It is often interesting to consider what the largest clique in a graph is.

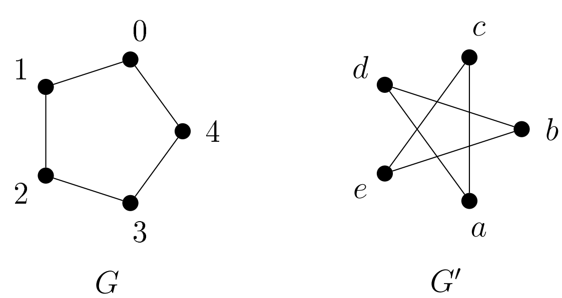
Definition 9 (Maximum clique; clique number)**.** *Let G be a graph. A* maximum clique *of G is a clique of largest order. The* clique number *of G, denoted $\omega(G)$, is the order of a maximum clique in G.*

Example 6 (Maximum clique; clique number)**.** The graph G in Example 5 has a maximum clique $\{0, 1, 2, 3\}$. It follows that $\omega(G) = 4$. $\triangle$

Sometimes, a graph (or subgraph) resembles a "copy" of some other graph. This is the notion of graph isomorphism, defined below.

Definition 10 (Graph isomorphism)**.** *Let G and G' be graphs. If there is some bijection $\phi : V(G) \to V(G')$ such that $\{u, v\} \in E(G)$ if and only if $\{\phi(u), \phi(v)\} \in E(G')$, then G and G' are* isomorphic*, denoted $G \cong G'$. That is, two graphs are isomorphic if there is an edge-preserving bijection between the vertex sets.*

Example 7 (Isomorphism)**.** Consider the following graphs:



Observe that G and G' are isomorphic; that is, $G \cong G'$. One satisfactory mapping $\phi : V(G) \to V(G')$ is given by $\phi(0) = a$, $\phi(1) = c$, $\phi(2) = e$, $\phi(3) = b$, $\phi(4) = d$. $\triangle$

Problems in Ramsey theory often focus on the presence of one copy of a given substructure; in contrast, the T-Z problem concerns several copies of a given substructure. More specifically, the T-Z problem asks about copies of subgraphs within vertex *neighborhoods*:

Definition 11 (Vertex neighborhood; G_v)**.** *Let G be a graph, and let $v \in V(G)$ be an arbitrary vertex. The (open)* neighborhood *of v in G is the set $N_G(v) := \{x \in V(G) : \{v, x\} \in E(G)\}$. Write G_v to denote the subgraph of G induced by $N_G(v)$.*

The T-Z problem asks for which graphs F there exists a graph G such that for each vertex $v \in V(G)$, the subgraph induced by the neighbors of v is isomorphic to F. Such a graph G is said to be *locally* F. More generally, G might simply be called a *local graph*. Chapter 4 describes the T-Z problem in more detail. It also outlines a program that constructs graphs that are locally F for given F. The program applies linear programming to the subgraph isomorphism problem to conduct a tree search for satisfactory graphs. Chapter 4 presents graphs constructed by the program that are locally Ramsey, i.e. locally F for some Ramsey graph F.

Constructing local graphs is a non-trivial task. The Russian mathematician V.K. Bulitko proved in 1973 that the T-Z problem is *undecidable*, i.e. that there is no general algorithm which, given any set of input graphs, always correctly determines whether local graphs exist for those graphs. Winkler also establishes the undecidability of the T-Z problem, independently of Bulitko [78]. Chapter 5 contains a translation of the first section of Bulitko's paper, which is not available in English.

Chapter 6 addresses (undirected) Cayley graphs (Definition 52). All Cayley graphs are local graphs. This work highlights Cayley graphs that are locally Ramsey.

Cayley graphs represent group structures. Groups and graphs are related through the notion of graph automorphisms.

Definition 12 (Automorphism). *Let G be a graph. An* automorphism *of G is an isomorphism from G to itself.*

Definition 13 (Automorphism group of a graph). *Let G be a graph. The set of all automorphisms of G forms a group under the operation of composition. This group is called the* automorphism group *of G, denoted $Aut(G)$.*

The automorphism group of a graph acts on the set of vertices of the graph. Certain classes of graphs are defined based on properties of this group action; the vertex-transitive graphs form such a class.

Definition 14 (Transitive action). *Let G be a group acting on a set X. The action is said to be* transitive *if for any $x, y \in X$ there exists some $g \in G$ such that $g \cdot x = y$.*

Definition 15 (Vertex-transitive graph). *Let G be a graph. If $Aut(G)$ acts transitively on $V(G)$, then G is* vertex-transitive. *That is, G is vertex transitive if for any $u, v \in V(G)$ there is some $\phi \in Aut(G)$ such that $\phi(u) = v$.*

Definition 16 (Circulant graph). *Let G be a graph. If $Aut(G)$ contains a cyclic subgroup that acts transitively on $V(G)$, then G is a* circulant *graph.*

Like Cayley graphs, the vertex-transitive graphs are local graphs. Chapter 6 draws connections between Cayley graphs and vertex-transitive graphs, namely the fact that every Cayley graph is also vertex-transitive. Chapter 6 also includes a definition of circulant graphs that is based on Cayley graphs.

The following basic graph classes are studied in both Ramsey theory and the T-Z problem.

Definition 17 (Path). *A* path *is a graph* $G = (V, E)$ *such that*

$$V(G) = \{v_0, v_1, \ldots, v_n\}$$

and

$$E(G) = \{\{v_0, v_1\}, \{v_1, v_2\}, \ldots, \{v_{n-1}, v_n\}\},$$

with all v_i *distinct. The vertices* v_0 *and* v_n *are the* endpoints *of the path. Write* P_n *to denote a path of order* n*.*

Example 8 (Path). The paths P_3, P_4, P_5, and P_6 are shown below:

$\triangle$

A graph is *connected* if there is a path between any two vertices of the graph.

Definition 18 (Cycle). *A* cycle *is a graph* $G = (V, E)$ *such that*

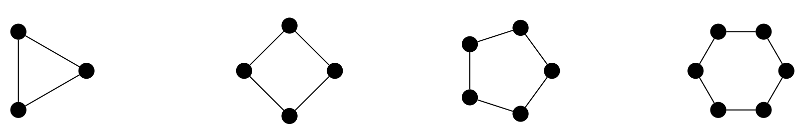
$$V(G) = \{v_0, v_1, \ldots, v_n\}$$

and

$$E(G) = \{\{v_0, v_1\}, \{v_1, v_2\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_0\}\},$$

with all v_i *distinct. Write* C_n *to denote a cycle of order* n*.*

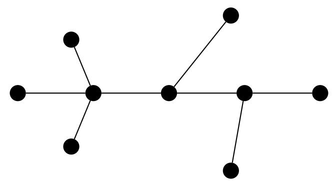
Example 9 (Cycle). The cycles C_3, C_4, C_5, and C_6 are shown below:



$\triangle$

Definition 19 (Tree)**.** *A* tree *is a connected cycle-free graph.*

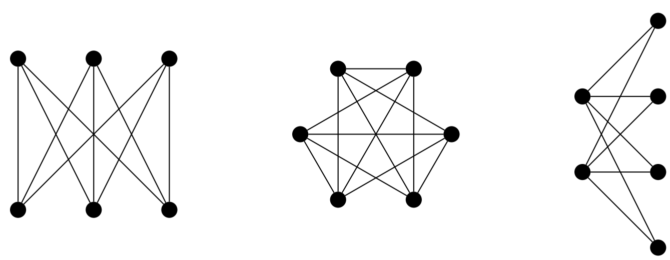
Example 10 (Tree)**.** A tree of order 9 is shown below:



△

Definition 20 (Complete multipartite graph)**.** *A* complete k-partite graph *is a graph with k independent sets in which there is an edge between every pair of vertices from different independent sets. A complete k-partite graph with independent sets of order $m_1, m_2, \ldots, m_k$ is denoted $K_{m_1,m_2,\ldots,m_k}$.*

Example 11 (Complete multipartite graph)**.** The complete multipartite graphs $K_{3,3}$, $K_{2,2,2}$ and $K_{2,4}$ are shown below:



△

Next are some basic definitions and notation necessary for arguments in later proofs.

Definition 21 (Distance between two vertices)**.** *Let G be a graph and let u and v be two arbitrary vertices of G. The* distance *between u and v, denoted $d_G(u,v)$, is the order of the shortest path connecting u and v. If no path connecting u and v exists, the distance between them is assumed to be infinite.*

Definition 22 (Degree of a vertex). *Let G be a graph. For a vertex $v \in V(G)$, the* degree *of v is denoted* $\deg(v, G)$ *and is defined as the number of vertices adjacent to v in G. That is,*

$$\deg(v, G) := |\{u \in V(G) : \{u, v\} \in E(G)\}|.$$

A graph in which all vertices have the same degree is a *regular* graph.

Chapter 2 includes proofs of elementary Ramsey theory results, in particular focusing on neighborhood arguments to foreshadow the T-Z problem. Chapter 4 presents proofs that were omitted from the papers in which they originally appeared. Chapter 5 is a translation from Russian into English of Bulitko's famous proof regarding the T-Z problem. Chapter 6 includes constructive proofs of T-Z results concerning Cayley graphs.

CHAPTER 2
RAMSEY THEORY

Ramsey theory is a famous and difficult branch of mathematics. This chapter provides a brief introduction to Ramsey theory with particular attention towards its role in graph theory. The recent plateau in progress identifying Ramsey numbers (Definition 25) makes it clear that new techniques for attacking the problem are needed. Chapter 3, describes how reinforcement learning is used to generate Ramsey graphs (Definition 26).

2.1 Introduction

In 1929, F.P. Ramsey [59] proved what would come to be known as Ramsey's Theorem. The theorem was presented as a result of formal logic:

Theorem 1 (Ramsey's theorem)**.** *Let Γ be an infinite class, and μ and r positive integers; and let all r-combinations of the members of Γ be divided in any manner into μ mutually exclusive classes C_i $(i = 1, 2, \ldots, \mu)$ so that every r-combination is a member of one and only one C_i; then, assuming the axiom of selections, Γ must contain an infinite subclass Δ such that all the r-combinations of the members of Δ belong to the same C_i.*

Ramsey's theorem led to a new field known as Ramsey theory. For readers interested in learning more about Ramsey theory, the book by Graham, Rothschild, and Spencer [26] gives an overview of the subject. While some Ramsey theoretic results precede the theorem, Ramsey's theorem gained popularity in the decades following his original paper. The result is particularly popular within the field of graph theory, where it is often stated in terms of edge colorings.

Definition 23 (Edge coloring). *Let G be a graph. An* edge coloring *of G is a map $\chi : E(G) \to C$, where C is a set of colors.*

Definition 24 (Monochromatic clique). *Let G be a graph and let χ be an edge coloring of G. A* monochromatic clique *of G is a clique such that all of its edges are colored with the same color under χ.*

Ramsey's theorem for graphs determines the inevitability of monochromatic cliques of a fixed order as the order of an edge-colored graph grows.

Theorem 2. *For any given number of colors, c, and any given positive integers $n_1, \ldots, n_c$, there is a number, denoted $R(n_1, \ldots, n_c)$ such that if the edges of a complete graph of order $R(n_1, \ldots, n_c)$ are colored with c different colors, then for some $i \in \{1, \ldots, c\}$, there is a monochromatic clique of order n_i colored i.*

When only two colors are considered, Ramsey's theorem might be stated in terms of graph complements as follows:

Theorem 3. *For any positive integers k and ℓ, there is a number n such that for any graph G of order $N \geq n$ either G contains a clique of order k or $\overline{G}$ contains a clique of order ℓ.*

The statement of Ramsey's theorem in Theorem 3 is the primary consideration of this work. Next is some vocabulary associated with Ramsey's theorem.

Definition 25 (Ramsey number). *The Ramsey number for (k, ℓ), denoted $r(k, \ell)$, is the smallest integer such that any graph of order $r(k, \ell)$ contains either a clique of order k or an independent set of order ℓ.*

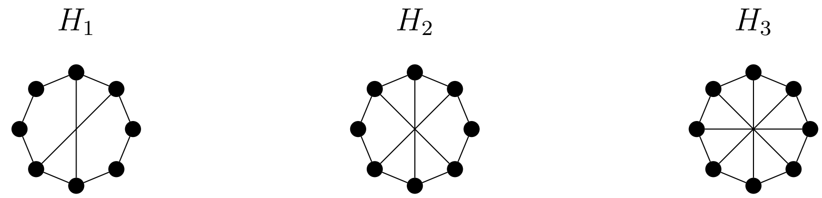


Figure 2.1: $R(3, 4; 8)$ graphs

Definition 26 (Ramsey graph)**.** *Let G be a graph that does not contain a K_k or $\overline{K}_\ell$. Such a graph G is a* Ramsey graph *for (k, ℓ). In general, $R(k, \ell; n)$ denotes a Ramsey graph of order n.*

Definition 27 (Critical Ramsey graph)**.** *Let $R(k, \ell; n)$ be a Ramsey graph. If $n = r(k, \ell) - 1$ (i.e. is of the largest order possible), then R is a* critical *Ramsey (k, ℓ) graph. Equivalently, R is Ramsey(k, ℓ)-critical.*

Example 12. Figure 2.1 shows Ramsey graphs for $(3, 4)$. It is shown later that these are in fact Ramsey$(3, 4)$-critical graphs. △

2.2 History of Ramsey theory

While some results similar to Ramsey's theorem appeared before Ramsey's paper in 1930, interest in Ramsey theory increased significantly in the years following his paper. Erdős showed particular interest with his 1935 paper [21] that offered a new proof of the theorem – one that improved on Ramsey number bounds originally given by Ramsey – as well as results regarding convex polygons formed from arbitrary sets of points in a plane. Erdős furthermore introduced some of the earliest techniques for addressing Ramsey numbers, perhaps most notably the probabilistic method used to improve lower bounds, introduced in 1947 [20]. The probabilistic method is a non-constructive argument for improving lower bounds, in contrast with the constructive approach of providing a counterexample. In 1975, Spencer made further improvements with the probabilistic method [71].

In 1955, Greenwood and Gleason proved results related to various upper bounds for Ramsey numbers, including cases involving three or more colors [29]. Among the several Ramsey numbers identified in this paper is the three-color Ramsey number $r(3,3,3) = 17$. Of particular interest in their paper is the following recurrence result:

Theorem 4. $r(k,m) \leq r(k-1,m) + r(k,m-1)$.

Proof. Let $p = r(k-1,m) + r(k,m-1)$. Let G be a graph of order p, and let $v \in V(G)$. Let $N_G(v)$ and $N_{\overline{G}}(v)$ be as indicated in Definition 11. Note that $|N_G(v)| + |N_{\overline{G}}(v)| + 1 = p$. It follows that either $|N_G(v)| \geq r(k-1,m)$ or $|N_G(v)| < r(k-1,m)$.

If $|N_G(v)| \geq r(k-1,m)$ then $N_G(v)$ either induces a clique of order $k-1$ or an independent set of order m. In the latter case, the proof is finished, so suppose the former. Since there is a clique of order $k-1$, this clique joined with the vertex v as a universal vertex results in a clique of order k in G, completing the proof.

Suppose instead that $|N_G(v)| < r(k-1,m)$. In this case, $|N_{\overline{G}}(v)| \geq r(k,m-1)$, and the argument is similar to the one presented above to produce the desired clique or independent set. □

Theorem 4 and its proof are widely known. The inclusion of the proof here serves to draw attention to the importance of *vertex neighborhoods* in the argument.

The following corollary has been useful for improving bounds of some Ramsey numbers.

Corollary 1. *If $k, m \in \mathbb{N}$ are such that $r(k,m-1)$ and $r(k-1,m)$ are both even, then $r(k,m) < r(k-1,m) + r(k,m-1)$, i.e.*

$$r(k,m) \leq r(k-1,m) + r(k,m-1) - 1.$$

Proof. Let $p = r(k, m-1) + r(k, m-1) - 1$ and let G be a graph of order p. Let $v \in V(G)$. Next,

$$|N_G(v)| + |N_{\overline{G}}(v)| = r(k-1, m) + r(k, m-1) - 2.$$

Consider three possibilities:

1. $|N_G(v)| > r(k-1, m) - 1$. In this case, $|N_G(v)| \geq r(k-1, m)$. Hence $N_G(v)$ induces either a clique of order $k-1$ or an independent set of order m. In either case, the proof is finished.

2. $|N_G(v)| < r(k-1, m) - 1$. In this case,

$$\begin{aligned} |N_{\overline{G}}(v)| &= r(k-1, m) + r(k, m-1) - 2 - |N_G(v)| \\ &> r(k-1, m) + r(k, m-1) - 2 - r(k-1, m) + 1 \\ &\geq r(k, m-1). \end{aligned}$$

   Hence $N_{\overline{G}(v)}$ induces either a clique of order k or an independent set of order $m-1$ in G. Either way, the proof is finished.

3. $|N_G(v)| = r(k-1, m) - 1$. Note that $r(k-1, m) - 1$ is odd since $r(k-1, m)$ is even. Hence there must be some vertex $u \in V(G)$ that falls under Case 1 or 2. Otherwise, this would imply that G (a graph of odd order) is regular of odd degree, which is not possible.

$\square$

While Corollary 1 may seem a simple result, it has yielded improvements in the search for $r(5, 5)$. First, the corollary led to improvements on the bounds (and eventual exact identification) of $r(4, 5)$. The identification of $r(4, 5)$ was then used to improve bounds on $r(5, 5)$ [54].

Paths, trees, forests, and cycles represent a few of the other popular graph classes studied in Ramsey theory. An extensive survey regarding Ramsey numbers is maintained at [58].

2.2.1 Known 2-color Ramsey numbers

Identifying Ramsey numbers is a notoriously difficult problem in mathematics. The process for establishing a Ramsey number $r(k, \ell)$ is twofold. To establish $r(k, \ell) \geq n$, one typically produces a counterexample of order $n - 1$. To establish $r(k, \ell) \leq n$, one must show that *every* graph of order n satisfies the property of containing a K_k or $\overline{K_\ell}$. Ramsey numbers of the form $r(k, k)$ are frequently called the *symmetric* Ramsey numbers. Currently, only two of the symmetric Ramsey numbers are known.

The proofs presented below are fairly simple and well-known results. They are included here because the neighborhood arguments used in the proofs foreshadow the Trahtenbrot-Zykov problem encountered in Chapter 4. These proofs might also help familiarize readers with the common early techniques in Ramsey theory.

Theorem 5. *$r(3, 3) = 6$.*

Proof. The cycle C_5 establishes $r(3, 3) \geq 6$. To establish $r(3, 3) \leq 6$, let G be a graph of order 6. Let $v \in V(G)$ be arbitrary. By the pigeonhole principle, v has either 3 neighbors or 3 non-neighbors in G. Without loss of generality, assume v has 3 neighbors. If any two of these neighbors are adjacent to one another, G contains a K_3. On the other hand, if the 3 neighbors are pairwise non-adjacent, they form an independent set of order 3 in G. □

The following lemma will be used together with Corollary 1 to establish $r(3, 4)$ later.

Lemma 1. *$r(2, k) = k$.*

Proof. Let G be a graph of order $k - 1$, and suppose $E(G) = \emptyset$. It follows that G does not contain a K_2 or an independent set of order k. Hence $r(2, k) > k - 1$.

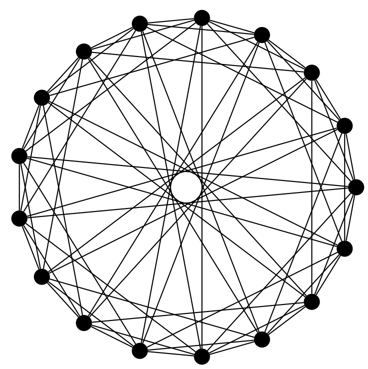


Figure 2.2: $R(4,4;17)$ graph

Next, let G' be a graph of order k. If G' contains any edges, then G' contains a K_2. On the other hand, if G' contains no edges, then G' contains an independent set of order k. Hence $r(2,k) \leq k$.

Thus $r(2,k) = k$. □

Theorem 6. $r(3,4) = 9$.

Proof. Figure 2.1 shows $r(3,4) > 8$, i.e. $r(3,4) \geq 9$. To show $r(3,4) \leq 9$, note that by Lemma 1, $r(2,4) = 4$ and by Theorem 5, $r(3,3) = 6$. Since both of these are even, by Corollary 1, $r(3,4) < r(2,4) + r(3,3)$, i.e. $r(3,4) < 10$. Hence $r(3,4) \leq 9$. It follows that $r(3,4) = 9$. □

Corollary 2. $r(4,4) = 18$.

Proof. Figure 2.2.1 shows $r(4,4) > 17$, i.e. $r(4,4) \geq 18$. Next, by Theorem 4 and Theorem 6, it follows that $r(4,4) \leq 9 + 9$, so $r(4,4) \leq 18$. Hence $r(4,4) = 18$. □

In 1995, McKay established $r(4,5) = 25$ [53]. McKay also made improvements on bounds for $r(5,5)$ as recently as 2018 [4], when he established $r(5,5) \leq 48$.

Theorem 7 ([22, 4])**.** $43 \leq r(5,5) \leq 48$.

In Theorem 7, the lower bound of 43 was established constructively in 1989 [22]. It is conjectured that $r(5,5)$ is precisely 43 due to the fact that despite the expenditure of extensive computer resources, attempts to construct $R(5,5;43)$ graphs have been unsuccessful [4].

Much of the progress on improving the upper bounds is due to Brendan McKay's work involving linear programming. Section 4.5 outlines how to apply linear programming in the subgraph isomorphism problem. Computers are a major tool in the search for Ramsey numbers, as even some of the early papers detail the extent to which computers were used [39]. The linear programming approach frequently employed by McKay involves a gluing procedure in which larger Ramsey graphs are constructed by gluing together graphs along some smaller Ramsey graph [52]. In 1992, McKay made an improvement of 1 on the upper bounds of each of $r(4,5), r(5,5)$, and $r(4,6)$, in particular establishing $r(5,5) \leq 53$. Improvements on the bounds of $r(4,5)$ later helped establish $r(5,5) \leq 50$. In 1995, $r(5,5)$ was improved from 50 to 49 using the gluing procedure [54]. This remained the best upper bound until 2018 when it was established, again by the linear programming gluing technique, that $r(5,5) \leq 48$.

McKay notes that a contributing factor to the 2018 progress is the ability to make computations that would have simply taken far too long in 1995, highlighting the importance of computing power in addressing Ramsey numbers. The next chapter details another computer-based approach to improving bounds on Ramsey numbers: the application of reinforcement learning to producing edge-colored graphs.

CHAPTER 3
REINFORCEMENT LEARNING AND RAMSEY GRAPHS

Section 3.1 of this chapter covers basic concepts regarding neural networks and reinforcement learning. Section 3.2 describes how we trained a reinforcement learning agent to generate Ramsey graphs (Definition 26). Section 3.3 contains results of simulations. Recommendations for continuing this project comprise Section 3.4, and Section 3.5 describes how the project led to the Trahtenbrot-Zykov problem defined in Chapter 4. Code for this chapter is publicly available on GitHub (Appendix I).

The application of reinforcement learning towards the problem of constructing Ramsey graphs is motivated by recent improvements with artificial intelligence and the game of Go. Go is a difficult game for a computer agent to master due to the large number of possible board positions and moves, as the game tree for Go has a significantly greater breadth and depth than chess – approximately 250^{150} possible move sequences as opposed to 35^{80} [65]. In 2015, Google's DeepMind, using their AlphaGo program [65], defeated a professional Go player without any in-game handicaps. The number of 2-colorings of the edges of a complete graph K_n is $2^{\binom{n}{2}}$, which surpasses the number of Go positions when $n = 50$:

$$
\begin{aligned}
2^{\binom{n}{2}} &\geq 250^{150} \\
n &\geq \frac{1 + \sqrt{1 + 4(300 \log_2 250)}}{2} \approx 49.4
\end{aligned}
$$

These tools, having conquered the game of Go, might reasonably be expected to tackle the task of 2-coloring edges of graphs of order less than 50, possibly helping to construct new Ramsey graphs. This is of particular interest in the case of $r(5,5)$, known to be between 43 and 48 and conjectured to in fact be precisely 43 [4], or for other Ramsey numbers for which the lower bounds are below 50 and might possibly be improved by the new tools of reinforcement learning. The application of reinforcement learning towards generating Ramsey graphs therefore seems plausible from a complexity standpoint. Reinforcement learning has been applied towards the task of constructing combinatorial counterexamples with success, as demonstrated by Wagner [75]. More specifically, the tools related to the game of Go have been applied towards problems in graph theory. In 2019, Huang et al. adapted AlphaGo Zero (Section 3.1.3) to color large graphs [37].

The following definition is central to this chapter.

Definition 28 (Ramsey game)**.** *Let n, k, ℓ be integers, with $k \leq n$ and $\ell \leq n$. Let G be a complete graph of order n with all edges colored black. The $r(k, \ell; n)$* game *is as follows: Two players take turns coloring black edges of G using their assigned color. Player 1 colors edges red while Player 2 colors edges blue. The game ends when either Player 1 colors a K_k red or Player 2 colors a K_ℓ blue, whichever happens first. The player who first completes a clique in their color loses the game. If both players avoid creating a monochromatic clique, the game ends in a draw.*

3.1 Reinforcement learning

Machine learning systems are trained rather than explicitly programmed [16]. Three main classes of machine learning are supervised learning, unsupervised learning, and reinforcement learning. This work primarily concerns reinforcement learning, which is a sort of "trial and error" approach to computers solving problems.

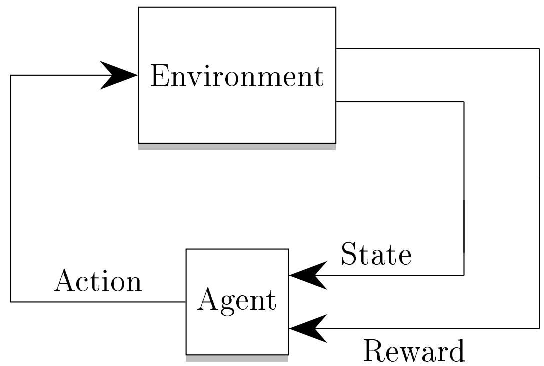


Figure 3.1: The reinforcement learning cycle

Definition 29 (Reinforcement learning)**.** Reinforcement learning *is a class of machine learning algorithms in which agents take* actions *within some prescribed* environment *in order to maximize* rewards *[16].*

The notions of *actions*, *environment*, and *rewards* in Definition 29 appear throughout this section. For an agent to receive information about its environment, the environment is converted to a computer representation. This computer representation of the environment is called the *state*. Note that the encoding of the environment as a state is a subjective process; choosing how to encode an environment is a step of machine learning known as *feature engineering*, in which the programmer's knowledge of the task and environment is used to extract useful representations of the environment to encode.

The objective in the Ramsey game is to color all edges of a complete graph red or blue in stages, and in such a way that avoids creating monochromatic cliques. Coloring edges represents an *action*. At each stage of the coloring process, the edge-colored graph is an *environment*. The *agent* is backed by a neural network. The *reward* is based on the result at the conclusion of the game – avoiding monochromatic cliques maximizes the reward, while coloring a monochromatic clique yields negative rewards. Section 3.2 contains more details of the implementation.

3.1.1 What is a neural network?

Neural networks are closely related to machine learning and specifically deep learning.

Definition 30. *Let S be a set of states and let D be a set of decisions. A neural network is a function $f : S \to D$ that takes some state $s \in S$ as input and yields a decision $d \in D$.*

More specifically, a neural network f might be defined by a function composition

$$f = (\phi_n \circ f_n) \circ \cdots \circ (\phi_0 \circ f_0),$$

where each $f_i, i \in \{0, \ldots, n\}$ corresponds to what is called a *layer* of the network, and each $\phi_i, i \in \{0, \ldots, n\}$ corresponds to an *activation function.* One of the simplest and most common layer types is the dense layer, which returns a linear transformation of the input data.

Definition 31 (Dense layer)**.** *Let $\mathbf{x}$ be a vector of input data for a neural network. Given a weight matrix $\mathbf{W}$ and a bias vector $\mathbf{b}$, a dense layer f returns a linear transformation:*

$$f(\mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{b}.$$

An *activation function* applies a nonlinearity to the output of a layer. Different activation functions are recommended for certain layer types; page 41 includes the recommendations followed for this project.

The definition of a dense layer includes a weight matrix. Weights are learned by the network during the training loop. Training consists of a forward pass of data through the network to get an output. The accuracy of this output is measured through a *loss function.* Information from this loss function is then used in a backward pass through the network, and weights are adjusted through a process known as *backpropagation*. For more information on loss functions and backpropagation, see [16]. While dense layers are used here as a simple example for the training process, the overall procedure for learning network weights to improve network predictions is similar for networks with other types of layers, such as convolutional layers.

The Conv2D layer

The major building block of our network is the Keras `Conv2D` layer, a convolutional layer. Broadly speaking, convolutional layers take snapshots of visual data in order to extract local patterns. Convolutional layers have two key hyperparameters: the number of *filters* and the *kernel* size. Informally, kernel size dictates the size of the snapshots taken of the input data, while the number of filters determines the depth of the layer's output. A kernel size of 3×3 is fairly standard. The number of filters is typically chosen empirically; see AlphaGo's data collected for versions of the network with different numbers of filters [65]. Another hyperparameter in the `Conv2D` layer is the convolutional kernel used for the convolution step; we use the Keras default, which is the Glorot uniform initializer.

Consider a `Conv2D` layer with kernel size 3×3 and d filters. As shown in Figure 3.2, the passage of data through a convolutional layer is as follows:

1. Pass the environment through the layer as an input state: view this as an $n \times k \times \ell$ array.

2. Collect as many $3 \times 3 \times \ell$ snapshots as possible; say $s = (n-2)(k-2)$ is the number of snapshots.

3. Take the dot product of each snapshot with the convolutional kernel to get s vectors with ℓ entries each.

4. The vectors get rearranged into a $n \times k \times d$ representation of the environment.

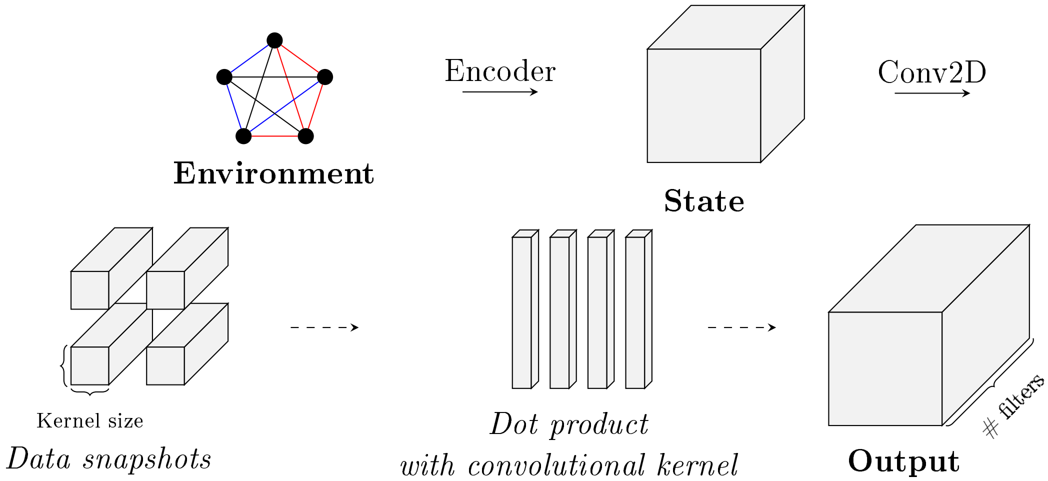


Figure 3.2: Keras Conv2D layer

A neural network that consists of mostly convolutional layers is called a Convolutional Neural Network (CNN). Convolutional layers are useful for detecting visual patterns in data, as they are specifically designed to process data that come in the form of multiple arrays [45].

Other important layers

The Keras `Dense` layer attempts to match relationships between any two input features [16]. This is in contrast with `Conv2D` layers, which look at local relationships. The Keras `Flatten` layer flattens input into one-dimensional data. Since convolutional layers are designed specifically for multidimensional data, `Flatten` layers are a way to pass convolutional layer outputs to other layer types, such as the `Dense` layer.

3.1.2 Tree search

This section outlines two types of tree search related to our neural network. The first tree search, Monte Carlo Tree Search, is a well-known search algorithm used in decision processes. The second tree search, AlphaZero Tree Search, is a modification of Monte Carlo Tree Search.

Monte Carlo Tree Search

Monte Carlo methods rely on repeated random sampling to estimate possible outcomes of uncertain events. Monte Carlo Tree Search (MCTS) is an algorithm for exploring game trees in search of optimal moves. Two key factors in MCTS are exploration and exploitation. Exploration tends to favor exploring many new game positions while exploitation favors looking deeper into moves known well. The following explanation of MCTS is based on the $r(k, \ell; n)$ game described in Definition 28, where Player 1 is assigned the color red and Player 2 is assigned the color blue.

Definition 32 (Game state). *A* game state *is an environment within a game.*

The attributes of a game state depend on the game being played. In the $r(k, \ell; n)$ game, the game state consists of a complete graph of order n under some (not necessarily proper) edge coloring using red, blue, and black. The current player is also an attribute of the game state.

Next are some definitions for the set-up of MCTS in the Ramsey game.

Definition 33 (Game tree node). *A* game tree node t_i *consists of an associated game state g_i, a number n_i corresponding to the number of visits to this game tree node, and numbers r_i, b_i, and d_i corresponding respectively to the number of red wins, blue wins, and draws resulting from simulations from t_i. Write $t_i = (g_i, n_i, r_i, b_i, d_i)$.*

Definition 34 (Root node). *A game tree node is the* root node *if it corresponds to the current (present) game state, i.e. if it corresponds to the game state for the beginning of the tree search being conducted.*

Definition 35 (Incomplete node). *A game tree node is* incomplete *if it is not a terminal node and if there are unexplored legal moves from its corresponding game state, i.e. if the node has potential children not yet added to the tree.*

Definition 35 notes the relationship between nodes and children. This notion is related to the game tree's structure as a directed graph in which legal moves between game states constitute edges.

Definition 36 (MCTS temperature)**.** *The* temperature *of Monte Carlo Tree Search is a nonnegative number c that determines how heavily the tree search favors an exploration based approach (exploring many new moves) as opposed to an exploitation based approach (repeatedly visiting moves it already knows well). A low value of c results in more exploitation while a high value of c results in more exploration.*

The temperature c is chosen empirically. It is fixed at the beginning of the search. There is not a set interval from which c should be chosen, as this varies with different games and implementations. Experimentation determines a value of c appropriate for the desired level of exploration or exploitation.

The temperature affects the *UCT score* for nodes, which dictates the tree search.

Definition 37 (Upper Confidence bound applied to Trees (UCT))**.** *Let $t_m = (g_m, n_m, r_m, b_m, d_m)$ be a game tree node that is a descendant of root node $t_0 = (g_0, n_0, r_0, b_0, d_0)$. Suppose P_0 is the current player in g_0 and w is the percentage of games P_0 has won starting from t_m. Suppose the parent of t_m has been visited N_m times. Let c be the temperature of the tree search. The* UCT score *for t_m is defined by*

$$U(t_m) := w + c\sqrt{\frac{\log N_m}{n_m}}$$

Definition 38 (Rollout)**.** *A* rollout *is a random simulation of gameplay from a given game state; that is, a rollout is a set of moves selected at random until a terminal game state is reached.*

The two hyperparameters for MCTS are the temperature and the number of rollouts to perform. Each rollout will correspond to a round of MCTS, and each round of MCTS consists of the following steps:

1. Selection: Starting from the root node, select child nodes until an incomplete node is reached. This node selection is based on the UCT score; the node with the highest UCT score is selected at each step of the tree traversal. Note that if the root node itself is incomplete, it will be selected.

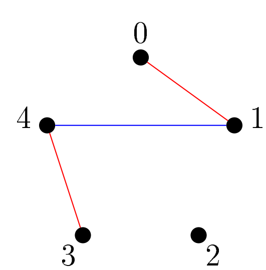
2. Expansion: Randomly choose any unexplored move to make from the incomplete node's game state and add the corresponding child node to the tree.

3. Simulation: Complete a rollout from the child node's game state. Record the result.

4. Backtrack: Travel back up the tree and update information for ancestors of the child node.

As more rollouts are carried out, tree node statistics regarding outcomes (wins, losses, draws) from a particular position become increasingly reliable. The number N of rollouts is typically chosen in a way that balances computational expense against the desire for reliable statistics, i.e. choosing the greatest number of rollouts one can afford given computational constraints.

Example 13 (Example of using MCTS to play Ramsey game)**.** This example shows what a round of MCTS looks like on the $r(3,3;5)$ game; see Definition 28 for details regarding the game. Suppose the current environment is as shown below:



It is currently Player 2's turn to choose some edge to color blue. Suppose 16 rounds of MCTS have already been completed, and the tree is as shown in Figure 3.3. Let $t_0 = (g_0, 16, 7, 9, 0)$, where $7 = r_0$ is the number of red wins from this state and $9 = b_0$ is the number of blue wins from this state.

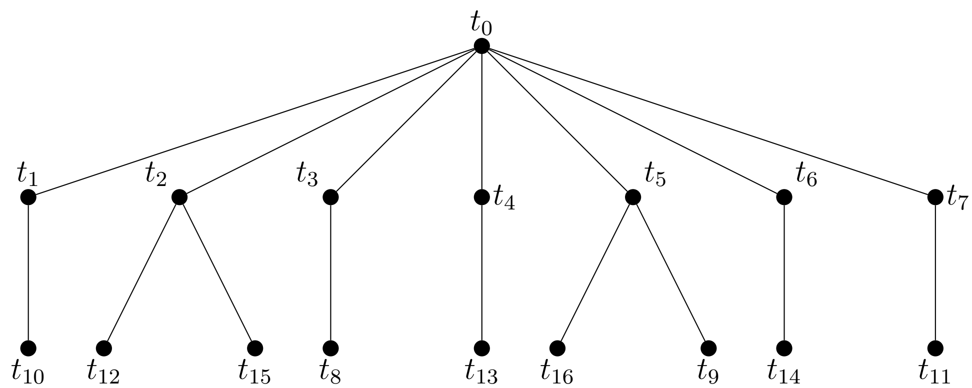
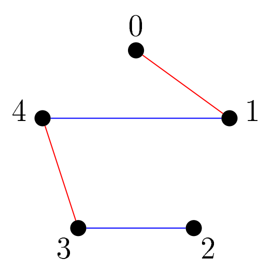


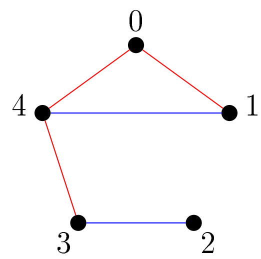
Figure 3.3: A partial MCTS tree

1. Selection: Suppose $t_6 = (g_6, 2, 1, 1, 0)$ is selected as having the highest UCT score among $\{t_1, t_2, \ldots, t_7\}$.

2. Expansion: The game state g_6 is shown below:



   To expand the tree, randomly choose any unexplored legal move that might be made next; suppose Player 1 colors $\{0, 4\}$ red. Add the node t_{17} with corresponding game state g_{17}, shown below:



3. Simulation: Complete a rollout from g_{17}. Suppose Player 1 (red) wins the game. This corresponds to node $t_{17} = (g_{17}, 1, 1, 0, 0)$.

4. Backtrack: Update statistics for nodes t_6 and t_0:

   - $t_6 = (g_6, 2+1, 1+1, 1, 0) = (g_6, 3, 2, 1, 0)$
   - $t_0 = (g+0, 16+1, 7+1, 9, 0) = (g_0, 17, 8, 9, 0)$

$\triangle$

The next type of tree search, AlphaZero Tree Search (AZTS), is closely related to MCTS. While MCTS relies heavily on randomness, AlphaZero receives some information from a neural network to guide the tree search.

AlphaZero Tree Search

The AlphaZero algorithm uses a modified form of MCTS. The AlphaZero Tree Search (AZTS) has hyperparameters for temperature and the number of rollouts, similarly to MCTS. The nodes in AZTS have different associated statistics based on the neural network backing the AlphaZero agent.

Definition 39 (Prior value of a move)**.** *Let s be a game state, and let f be the neural network for the AlphaZero agent. Let m be a legal move from s. Suppose $f(s) = (P, v)$, where P is a probability distribution over legal moves from s. The* prior value *of m is defined as $P(m)$.*

Definition 40 (Value of a game state)**.** *Let s be a game state, and let f be the neural network for the AlphaZero agent. Suppose $f(s) = (P, v)$. The* value *of s is defined as v.*

Definition 41 (AlphaZero game tree node)**.** *An* AlphaZero (AZ) game tree node *z_i consists of an associated game state g_i, a positive integer n_i corresponding to the number of visits to this game tree node, and an accumulated value v_i. The AZ game tree node is denoted $z_i = (g_i, n_i, v_i)$.*

Definition 42 (Expected value of AZ game tree node)**.** *Let $z_i = (g_i, n_i, v_i)$ be an AZ game tree node. The* expected value *Q of z_i is*

$$Q(z_i) := \begin{cases} \frac{v_i}{n_i} & \text{if } n_i \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Definition 43 (AlphaZero score)**.** *Let c be the temperature for AZTS. Let $z_t = (g_t, n_t, v_t)$ be an AZ game tree node with parent $z_s = (g_s, n_s, v_s)$. Let m_t be the move made from g_s to transition to g_t. The* AlphaZero (AZ) score *z_t is*

$$A(z_t) := Q(z_t) + \frac{c \cdot P(m_t) \cdot \sqrt{n_s}}{1 + n_t}$$

The overall steps for AZTS are similar to MCTS:

1. Selection: Starting from the root node, select child nodes until an incomplete node is reached. This node selection is based on the AZ score; the node with the highest AZ score is selected at each step of the tree traversal. Note that if the root node itself is incomplete, it will be selected.

2. Expansion: From the incomplete node, select the unexplored move with the highest prior value and add the corresponding child node to the tree.

3. Simulation: Determine and record the value of the child node game state.

4. Backtrack: Travel back up the tree and update information for ancestors of the child node. At each step of this process, the accumulated value of each node should be *subtracted* from its parent to account for the change in perspective due to players taking turns.

The move that is selected by the agent after completing all rollouts is simply the move corresponding to the child of the root node that has the most recorded visits. This is a reliable way to select a move because nodes with several recorded visits will have an expected value that is not only high but also trustworthy.

Example 14 (Example of using AZ tree search to play Ramsey game)**.**

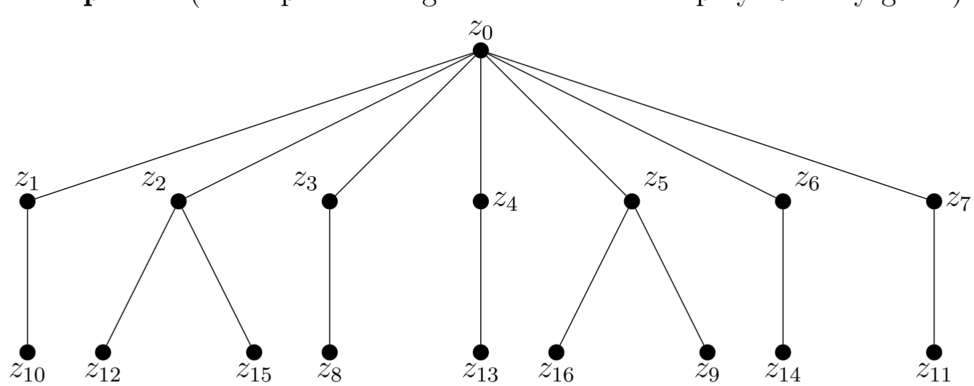
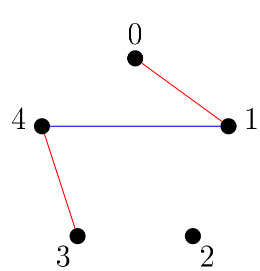


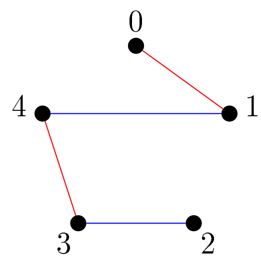
Figure 3.4: A partial AZTS tree

This example shows what a round of AZTS looks like on the $r(3, 3; 5)$ game; see Definition 28 for details regarding the game. Suppose the current environment is as shown below:



It is currently Player 2's turn to choose some edge to color blue. Suppose 16 rounds of AZTS have already been completed, and the tree is as shown in Figure 3.4. Let $z_0 = (g_0, 16, 0.25)$.

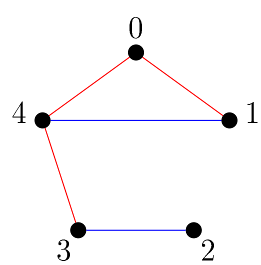
1. Suppose $z_6 = (g_6, 2, 0.7)$ is selected as having the highest AZ score among $\{z_1, z_2, \ldots, z_7\}$, where g_6 is shown below:



2. Expansion: Suppose the prior values for each valid move from g_6 are given below:

m	$\{0,2\}$	$\{0,3\}$	$\{0,4\}$	$\{1,2\}$	$\{1,3\}$	$\{2,4\}$
$P(m)$	0.18	0.12	0.2	0.35	0.08	0.07

   The move $\{1, 2\}$ is already represented by node z_{14} on the tree, so select the move $\{0, 4\}$ as the *unexplored* move with the highest prior value. Add the node z_{17} with corresponding game state g_{17} to the tree, where g_{17} is shown below:



3. Simulation: Let f be the neural network for the AlphaZero agent. Suppose $f(g_{17}) = (P, -0.23)$. Record the statistics for z_{17}, so $z_{17} = (g_{17}, 1, -0.23)$.

4. Backtrack: Update statistics for ancestors, so

$$z_6 = (g_6, 2 + 1, 0.7 + 0.23) = (g_6, 3, 0.93)$$
$$z_0 = (g_0, 16 + 1, 0.25 - 0.93) = (g_0, 17, -0.68)$$

$\triangle$

Tree search is a major component of several board game software implementations. The next section describes the development of a high-performance Go program.

3.1.3 Reinforcement learning and the game of Go

AlphaGo was first trained to play Go on a vast set of human-generated data. The data set included 30 million board positions and moves made by experts from those positions [65]. After this initial training, more improvements were made by using self-play. During self-play, moves were selected in accordance with a Monte Carlo Tree Search [65]. By completing thousands of high-level games in this way, more data were generated in order to further improve the network. AlphaGo's victory over Fan Hui in 2015 made it the first Go program to defeat a professional Go player without any handicaps; furthermore, AlphaGo went on to defeat the even higher-ranked Lee Sedol, widely considered one of the best Go players in the world, in 2016 [65].

AlphaGo consists of two networks, one each for policy and value. *Policy* corresponds to *actions* which might be taken in a particular game state, i.e. the policy network dictates which move the program should make at a particular point in time. The value network estimates the overall *value* of the game state for the player, i.e. whether it appears the player is on track to win (a high value) or lose (low value). Value thus corresponds to the concept of *reward* in the reinforcement learning cycle. Both the policy network and the value network for AlphaGo were trained on the same data set. Further improvements to the trained version of AlphaGo were made through self play with MCTS.

AlphaGo Zero is an algorithm based on AlphaGo with some key differences. The first such difference is that AlphaGo Zero consists of one network with two separate outputs, as opposed to two separate networks each with their own output. The two outputs from the AlphaGo Zero network are still policy and value as previously described.

Another key difference, perhaps the most significant, is in the training process. While AlphaGo relied on expert-level, human-generated data, AlphaGo Zero learned entirely from scratch by using self play to generate its own data set. Throughout self-play, moves were selected in accordance with AZTS. At first, the outputs for policy and random were seemingly random, but as more games were carried out (and the network weights adjusted accordingly), AlphaGo Zero saw tremendous improvement, surpassing all previous versions of AlphaGo in just 40 days [67]. This was a major improvement over the already impressive growth of AlphaGo, which took months to train [65].

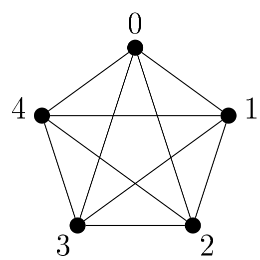
AlphaZero, a more general version of AlphaGo Zero, was introduced in late 2017 [66]. Overall, AlphaZero has a similar structure with a few key differences which allow the single algorithm to master several different games, including go, chess, and Shogi. The development of such a generalized approach to reinforcement learning with turn-based games made a similar approach to the construction of Ramsey counterexamples seem enticing.

3.2 Training a reinforcement learning agent to generate Ramsey graphs

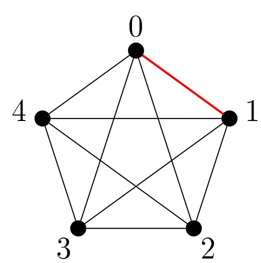
The objective for each player in the $r(k, \ell; n)$ game is to avoid creating a monochromatic clique in their color for as long as possible. Note that for certain values of n and k, it is in fact possible for the game to end in a draw – for example, when $n = 5$ and $k = \ell = 3$, both players are able to avoid creating a monochromatic triangle. This follows from the fact that $r(3,3) > 5$. Similarly, other values of n and k might guarantee that someone must lose the game. Considering $r(3,3) = 6$, it follows that for $n \geq 6$ and $k = \ell = 3$ one of the players will be forced to create a triangle in their color.

A hypothetical example of optimal play

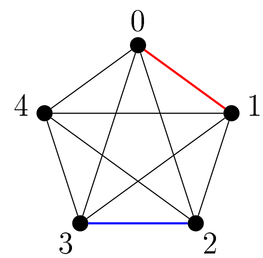
This section outlines an example of optimal play of the $r(3,3;5)$ game. Play is optimal when, if possible, it ends in a draw. If a draw is not possible, play is optimal when as many edges as possible are colored before a loss is declared. Let the vertices of the game graph be labeled using $\{0,1,2,3,4\}$:



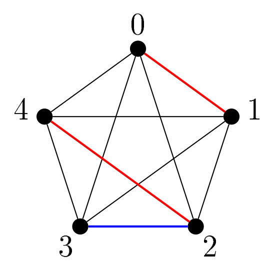
1. Player 1 chooses any edge. Without loss of generality, Player 1 colors $\{0,1\}$ red.



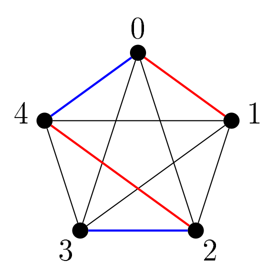
2. Player 2 avoids any edge incident to 0 or 1 since these edges present opportunities for Player 1 to color a triangle red. Without loss of generality, Player 2 chooses $\{2,3\}$ to color blue.



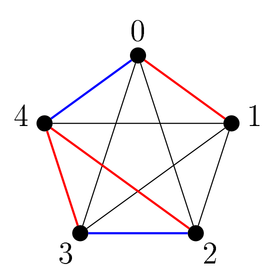
3. Player 1 avoids any edges incident to 0 or 1 since these are “high risk” edges that might lead to a red K_3. Without loss of generality, Player 1 chooses $\{2, 4\}$ to color red.



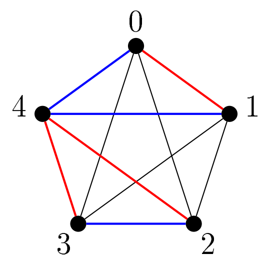
4. Player 2 avoids any edges incident to 2 or 3 since these are “high risk” edges that might lead to a blue K_3. Without loss of generality, Player 2 chooses $\{0, 4\}$ to color blue.



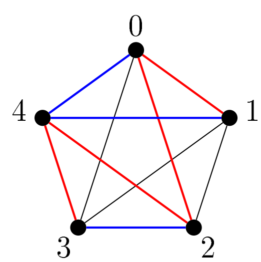
5. Player 1 colors $\{3, 4\}$ red because all of the other edges are risky.



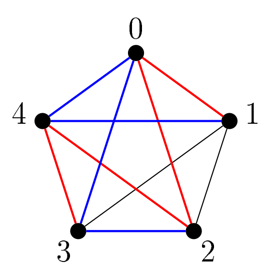
6. Player 2 colors $\{1, 4\}$ blue because all of the other edges are risky.



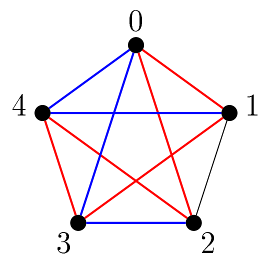
7. Player 1 can no longer avoid risky moves. $\{0, 2\}$ will make $\{1, 2\}$ dangerous later (and vice versa), similarly for $\{0, 3\}$ and $\{1, 3\}$. Without loss of generality, Player 1 chooses $\{0, 2\}$ to color red.



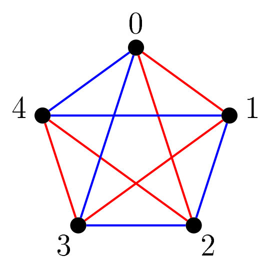
8. Player 2 avoids $\{1, 2\}$ because in this case, player 1 will likely choose $\{0, 3\}$ next, forcing blue to take $\{1, 3\}$ and lose. Player 2 also avoids $\{1, 3\}$ because in this case, Player 1 will likely choose $\{0, 3\}$ next, forcing Player 2 to take $\{1, 2\}$ and lose. Player 2 therefore chooses $\{0, 3\}$ to color blue.



9. Player 1 avoids $\{1, 2\}$ and chooses $\{1, 3\}$ to color red.



10. Player 2 colors $\{1, 2\}$ blue. The game ends in a draw.



3.2.1 Implementation

Environment and state

Definition 44 (Ramsey game graph)**.** *A* Ramsey game graph *is an environment at some turn of the $r(k, \ell; n)$ game. It is a complete graph of order n under some (not necessarily proper) edge coloring $\chi : E(K_n) \to \{red, blue, black\}$.*

Each Ramsey game graph must be encoded as a *state* to provide as input for a neural network. Recall from Section 3.1 that this encoding is a subjective process. It is important to capture features of the environment that are particularly relevant to the objective of the game. For the $r(k, \ell; n)$ game, it seems useful to encode information about monochromatic cliques within the environment. The following definitions lead to the encoding scheme in our implementation.

Definition 45 ($\omega_\chi(e)$)**.** *Let G be a Ramsey game graph, and let $e \in E(G)$. Define*

$$\omega_{red}(e) := \max\{p : e \textit{ is part of a red } K_p\},$$

and

$$\omega_{blue}(e) := \max\{p : e \textit{ is part of a blue } K_p\}.$$

Definition 46 (R_t, B_t)**.** *Let G be a graph and let $\chi : E(G) \to \{red, blue, black\}$ be an edge coloring of G. Define*

$$R_t := \{e \in E(G) : \omega_{red}(e) = t\},$$

and

$$B_t := \{e \in E(G) : \omega_{blue}(e) = t\}.$$

Next is the definition of our encoding scheme used for the Ramsey game.

Definition 47 (Encoding scheme for $r(k, k; n)$ game)**.** *The environment at each turn of the $r(k, k; n)$ game is encoded as a sequence $M^1, M^2, \ldots, M^{2k-1}$ of $n \times n$ matrices, where for $1 \leq t \leq k-1$ the matrices are specified as follows:*

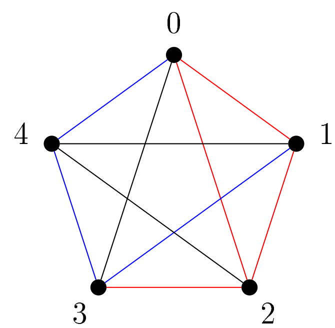
$$M^1_{i,j} = \begin{cases} 1 & \textit{if } \{i, j\} \textit{ is black} \\ 0 & \textit{otherwise} \end{cases}$$

$$M^{2t}_{i,j} = \begin{cases} 1 & \textit{if } \{i, j\} \in R_{t+1} \\ 0 & \textit{otherwise} \end{cases}$$

$$M^{2t+1}_{i,j} = \begin{cases} 1 & \textit{if } \{i, j\} \in B_{t+1} \\ 0 & \textit{otherwise} \end{cases}$$

A similar scheme might be used for the $r(k, \ell; n)$ game with $k \neq \ell$, but only the $r(k, k; n)$ scheme is considered here.

Example 15 (Example of encoding scheme)**.** A Ramsey game graph for the $r(3, 3; 5)$ game is shown below:



Its encoding $\mathbf{M} = M^1, M^2, M^3, M^4, M^5$ is as follows:

$$\mathbf{M} = \left[\begin{bmatrix} 0&0&0&1&0\\0&0&0&0&1\\0&0&0&0&1\\1&0&0&0&0\\0&1&1&0&0 \end{bmatrix} \begin{bmatrix} 0&0&0&0&0\\0&0&0&0&0\\0&0&0&1&0\\0&0&1&0&0\\0&0&0&0&0 \end{bmatrix} \begin{bmatrix} 0&0&0&0&1\\0&0&0&1&0\\0&0&0&0&0\\0&1&0&0&1\\1&0&0&1&0 \end{bmatrix} \begin{bmatrix} 0&1&1&0&0\\1&0&1&0&0\\1&1&0&0&0\\0&0&0&0&0\\0&0&0&0&0 \end{bmatrix} \begin{bmatrix} 0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0 \end{bmatrix}\right]$$

$\triangle$

The data encoded from gameplay could be augmented by including data corresponding to relabelings of graphs. Our implementation does not do this. AlphaGo exploits board symmetries [65], so this is an approach worth considering. AlphaZero does not make use of symmetry since the rules of chess are asymmetric [66], so its training set is not augmented.

Agent

In the reinforcement learning cycle, the agent receives information about its environment in order to take action. The agent in our implementation uses a neural network to select an action.

Network structure

The AlphaZero network is a Convolutional Neural Network (CNN). The network we attempt to implement is a heavily reduced version of the one implemented for AlphaZero in order to account for our relative hardware constraints (see Table 3.1), so our network is also a CNN. The main building block of our network is thus the Keras `Conv2D` layer. AlphaZero used a minimum of 40 convolutional layers leading up to the output, across what they describe as convolutional blocks and residual blocks in the network [66]. Our network consists of 8 convolutional layers, each with `ReLu` activation, which is the most popular activation function for convolutional layers. Each convolutional layer in our network has a kernel size of 3×3 and 32 filters. For comparison, AlphaZero uses 256 filters. The output from these convolutional layers is then passed through the layers corresponding to the policy and value outputs.

Policy has its own `Conv2D` layer, this time with only 2 filters. This output is passed to a `Flatten` layer and two `Dense` layers. The penultimate policy `Dense` layer uses a `ReLu` activation while the final `Dense` layer uses a `Softmax` activation in order to output a probability distribution over the legal moves from a game state. Similarly, value has its own `Conv2D` layer, this time with just 1 filter. It is then passed to a `Flatten` layer followed by two `Dense` layers. The last `Dense` layer uses a `tanh` activation, which is well-suited for binary classification problems [57] and outputs a value in the interval $(-1, 1)$. Figure 3.5 shows the architecture of our network. The overall structure of our network is derived largely from Max Pumperla and Kevin Ferguson's *Deep Learning and the Game of Go* [57]. Our code can be found on GitHub (Appendix I) and we encourage the reader to look there for any details regarding the network that might have been accidentally omitted here.

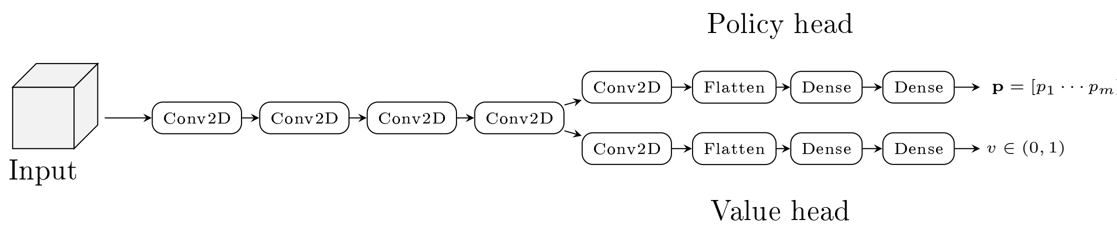


Figure 3.5: Sketch of network

Training cycle

The training cycle consists of several batches of self-play to generate data followed by adjustment of network weights using this data. For our hardware (see Table 3.1) playing the $r(3, 3; 5)$ game, a batch size of 2,000 games provides a balance between amount of data generated and time required to generate the data. Throughout self-play, moves are selected in accordance with AZTS with a temperature of $c = 0.4$ and 500 rollouts. AlphaGo Zero reported using 1,600 simulations to select each move, which required less than a half second for each search [67]. To select a move, our agent completes 500 rounds of AZTS and selects the root child node that has the most recorded visits.

Component	Our computer part	DeepMind AlphaGo [65] final version part
Processor	AMD FX$^{\text{TM}}$-8120 Eight-core Processor 3.10 GHz	48 CPUs total
RAM	Corsair Vengeance Pro 32 GB (4×8 GB) DDR3 1600 MHz	Not listed
GPU	NVIDIA GeForce RTX 2070 Super	8 GPUs total

Table 3.1: Computer hardware comparison

3.3 Simulations

We trained an agent to play the $r(3, 3; 5)$ game. Table 3.2 shows the development of the agent. The number of games refers to the number of self-play games

Agent name	# games	# samples	c	N	Time
`init`	N/A – initialized agent				
`v1`	5,000	45,000	0.4	500	1,500 min
`v2`	2,500	22,320	0.4	500	700 min

Table 3.2: Development of $r(3, 3; 5)$ agent.

Player 1	Player 2	Games played	P1 losses	Draws	Avg edges colored
`init`	`init`	100	1	5	8.47
`v1`	`v1`	100	20	17	9.8
`v2`	`v2`	100	25	58	9.69

Table 3.3: Performance improvements for the $r(3, 3; 5)$ agent

completed, and the number of samples corresponds to the number of game states generated by the agent across all games of self-play. Values of c and N correspond respectively to temperature and rollouts for AZTS, and the time recorded is the amount of time it took to complete self play. Note that `v1` is an improved version of `init`; that is, `init` completed 5,000 games of self-play, and this data was used to train the agent that would become `v1`. Similarly, `v2` is an improved version of `v1`.

Table 3.3 shows performance improvements for the agent playing the $r(3, 3; 5)$ game. Note that `init` is different from a purely random bot in that its moves are in fact still backed by the AZTS. The games played for evaluation are completed with the first move being chosen at random. After only 7,500 games of self-play (just over 36 hours of training time), our agent is able to achieve a draw over half of the time.

Our results for the $r(3, 3; 5)$ game are encouraging enough that we attempted to extend the experiment to playing the $r(4, 4; 17)$ game. With our implementation, one $r(4, 4; 17)$ game with $c = 0.4$ and $n = 250$ took over 5 hours, with only 127 of the 168 edges being colored before a loss. Our implementation thus does not scale well to playing on larger graphs. These constraints ultimately changed the direction of the project, as described in Section 3.5. The next section includes recommendations for overcoming these challenges.

3.4 Recommendations for continuing project

Training a reinforcement learning agent to construct larger Ramsey graphs seems to be a worthwhile task. For readers interested in continuing this project, some recommendations are offered here.

- Avoid Python's `deepcopy` function if possible. It is very computationally expensive and slow, as it constructs a new compound object and then recursively populates the new object with copies of the child objects found in the original object. Our implementation regularly calls `deepcopy` to create copies of our custom `GameState` objects, which involves copying a large amount of data. As seen in our code on GitHub, `deepcopy` is called primarily during the tree search to avoid modification of game states throughout that process. We are not sure how to avoid it ourselves, but it is likely possible. More experienced programmers might also consider manually defining a special `__deepcopy__()` method for the `GameState` class. There are likely many other opportunities for optimization in our code.

- Experiment with network architecture and hyperparameters.

- Consider a solitaire approach in which a solitaire agent chooses an edge and its color (as opposed to strictly alternating). It is unclear whether the algorithm might support such an approach or if changes might need to be made, so we disregard this approach in favor of a truly 2-player game. The solitaire approach might be helpful for training a bot to generate asymmetric Ramsey counterexamples since such graphs would require different numbers of red and blue edges.

- Experiment with reward structure. We chose ± 1 to mirror Go implementation as closely as possible. Another reward scheme worth considering (especially with a solitaire game) would be awarding the agent based on the total number of edges colored.

- Find other ways to speed up self-play, which is a CPU-intensive task. A GPU speeds up training of the network since it processes data very quickly, but generating the data for training is a major bottleneck of the project.

3.5 A change of direction

Due to the performance slowdown from the $r(3,3;5)$ game to the $r(4,4;17)$ game, it is tempting to abandon a "learn from scratch" approach and instead introduce some hints to the agent. In order for hints to apply to graphs of varying orders, hints should ideally be as general as possible. We therefore consider commonalities between the $R(3,3;5)$ and $R(4,4;17)$ graphs. The $R(4,4;17)$ graph is in Figure 2.2, and the $R(3,3;5)$ graph is isomorphic to C_5.

An interesting pattern emerges when one considers the subgraph induced by the neighbors of any particular vertex. In the $R(3,3;5)$ graph, this subgraph (for every vertex) is isomorphic to $\overline{K_2}$, which is itself a $R(2,3;2)$ graph. Perhaps more interesting is the case of $R(4,4;17)$, as the neighborhood of each vertex in this graph induces the same $R(3,4;8)$ graph. While this observation may not be completely surprising given Theorem 4, it does lead to another interesting problem in graph theory: The Trahtenbrot-Zykov problem, described in Chapter 4.

CHAPTER 4
THE TRAHTENBROT-ZYKOV PROBLEM

In 1963, Zykov [1] posed the following problems:

Question 1. *For which graphs F is there a graph G such that $G_v \cong F$ for every v in $V(G)$?*

Question 2. *For which graphs F are there only infinite G with G_v isomorphic to F for every $v \in V(G)$?*

Zykov notes that the first question was previously stated in a less general form by B.A. Trahtenbrot. The problem is thus frequently referred to as the Trahtenbrot-Zykov (T-Z) problem.

This chapter highlights seminal results related to the T-Z problem as well as examples of graphs related to those results. Section 4.1 briefly surveys some foundational papers related to the T-Z problem. Section 4.2 identifies some variants of the T-Z problem, including a conjecture of Szamkołowicz. Section 4.3 includes some existence results, while Section 4.4 focuses on non-existence results. Section 4.5 outlines a Python program that, given some graph F, attempts to construct a graph G such that $G_v \cong F$ for every $v \in V(G)$. This program is used in Section 4.6 to address graphs F of order 7 (related to a paper by Hall [31]) and in Section 4.7, which contains constructions from the program when F is a Ramsey graph.

Graphs in which all neighborhood induced subgraphs are isomorphic are sometimes called *local graphs*.

Definition 48 (Locally F; realizable; realization)**.** *A graph G in which G_v is isomorphic to F for every $v \in G$ is said to be* locally F*, and F is said to be* realizable*, with G being a* realization of F*.*

If G is *finite* and locally F, then F might be said to be *f-realizable*. In some contexts, if G is locally F, G might also be said to have *constant link F*, with F being a *link graph*.

The T-Z problem may also be considered in terms of families of graphs. Let $\mathscr{F}$ be a family of graphs. A graph G is *locally $\mathscr{F}$* if for every $v \in V(G)$, G_v is isomorphic to some $F \in \mathscr{F}$.

4.1 History

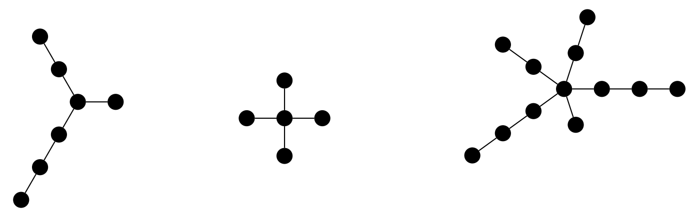
The complete graphs represent a class for which the Trahtenbrot-Zykov problem is trivial, as it is clear that K_n is locally K_{n-1} for all n. Complete symmetric multipartite graphs with all parts equal size are also trivially realizable; specifically, $K_{m,m,\ldots,m}$, with n parts, is realized by $K_{m,m,\ldots,m}$ with $n+1$ parts. The complete graphs and complete multipartite graphs are the only trivial cases of the T-Z problem.

Bulitko [12] proves that there is no general algorithm which, given any set of input graphs, will always correctly determine whether these graphs are realizable or not. This is a major result regarding the T-Z problem and is addressed in greater depth in Chapter 5.

Two papers by Brown and Connelly [9, 10] in 1973 and 1975, respectively, are frequently cited throughout the literature. The following definition is introduced to discuss their results.

Definition 49 (m-ad)**.** *An m-*ad *is a tree with m leaves and only one vertex of degree greater than two.*

Example 16 (m-ad)**.** Below is one example each of a 3-ad, 4-ad, and 5-ad, respectively.



△

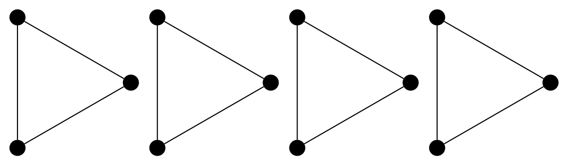
In [9] Brown and Connelly credit Zykov [1] for the T-Z problem in general. Brown and Connelly use a topological approach to obtain results about graphs which are locally a disjoint union of paths as well as graphs which are locally m-ad. Specifically, they state existence conditions for graphs that are locally F, where F is a finite disjoint union of paths, and graphs that are locally F for F a finite m-ad. The construction methods of Brown and Connelly are implemented in later papers both by themselves and others (e.g. Hall [31]).

In 1974, Chilton et al. showed that C_n is realizable for all $n, n \geq 3$ [15]. They establish conditions for graphs that are locally C_k for $k \in \{3, 4, 5\}$. They also give additional requirements when $k \geq 6$. The proofs are constructive in nature and use notions of graph automorphisms and some geometry. Cycles are an interesting family for the T-Z problem, as early published results regarding them were not correct; in fact, [15] was published to correct an erroneous result [3] regarding the realizability of some cycles.

Among realizability results for well-known classes of graphs are graphs that are locally Petersen [30], locally paths (of potentially varying orders) [55], and locally regular [85]. Some of the earliest papers on the T-Z problem [69, 68, 70], published as early as 1965, address locally Hamiltonian graphs.

Another local property studied is that of locally connected graphs, namely in the 1974 paper by Chartrand and Pippert [13]. Neither the property of being connected nor the property of being locally connected implies the other, as seen in Example 17. Chartrand and Pippert also explore the relationship between local connectivity and planarity. Sufficient conditions for locally connected graphs are stated in terms of degree sums and minimum degree.

Example 17. The graph mK_n consists of m disjoint copies of K_n. For $m > 1$, it is locally connected but not connected. The graph $4K_3$ is shown below as an example:



On the other hand, $C_n, n \geq 4$ is connected but not locally connected. $\triangle$

In [31], Hall resolves the T-Z problem for all graphs of up to order 6; that is, for all graphs G such that $|G| \leq 6$, it is determined whether or not G is realizable. These results are then used to determine all graphs of order up to 11 which are realizations of some graph, i.e. all graphs H with $|H| \leq 11$ such that H has constant link. While Hall states some general theorems regarding the existence or non-existence of graph realizations, some graphs are still left to ad-hoc methods for resolving the problem. Section 4.4 includes some proofs of results that were omitted from Hall's paper.

It is worth noting that every vertex-transitive graph (Definition 15) realizes some graph. However, vertex transitivity is not required for a graph to be a realization of some other graph; see [84] and [8] for some examples. Later in this chapter is another example, a non-vertex-transitive realization of a Ramsey graph.

4.2 Variants of the T-Z problem

There are several variants of the Trahtenbrot-Zykov (T-Z) problem. One of these variants considers subgraphs induced by more distant neighbors of each vertex.

Definition 50 (kth neighborhood)**.** *Let G be a graph, and let $u \in V(G)$ be arbitrary. Let $k \in \mathbb{N}$. The* kth neighborhood *of u in G, denoted $N_G^k(u)$, is defined as*

$$N_G^k(u) := \{v \in V(G) : d_G(u, v) = k\}.$$

G_u^k denotes the subgraph of G which is induced by $N_G^k(u)$.

Question 3. *Let $k \in \mathbb{N}$. For which graphs F does there exist a graph G such that $G_u^k \cong F$, for every $u \in V(G)$?*

The original T-Z problem corresponds to $k = 1$.

Szamkołowicz addresses Question 3 in [73] and [72]. In [73] he determines graphs for which the kth neighborhood of every vertex is edge-free. He extends these results to properties regarding the chromatic number of a graph, in particular drawing connections to Kőnig's Theorem. In [72] he offers conjectures and observations related to these conjectures; in particular, he conjectures the following:

Conjecture 1 (Szamkołowicz [72])**.** *Let G be a graph, and let $K(G) = \{k \in \mathbb{N} : V(G_u^k) \neq \emptyset\}$. Let $n \in \mathbb{N}$. If $\mathscr{G}(C_n) := \{G : \forall u \in V(G), \forall k \in K(G), G_u^k \cong C_n\}$, then $\mathscr{G}(C_3) = \{K_4\}$ and $\mathscr{G}(C_n) = \emptyset$ for $n \geq 4$.*

While the T-Z problem asks about graphs such that all neighborhoods are the same, others have researched graphs such that all of the neighborhoods are different.

Question 4. *Characterize graphs G for which G_u and G_v are not isomorphic for all $u, v \in V(G)$.*

Sedláček addresses Question 4 in [63]. The paper includes a minimal graph with all neighborhoods different from one another and furthermore proves that for all $n \geq 6$, there is a graph of order n in which all of the neighborhoods are non-isomorphic. In [64] Sedláček focuses on planar and outerplanar graphs in which all of the neighborhoods are non-isomorphic.

Another variant of the T-Z problem considers subgraphs induced by neighbors of edge endpoints, i.e. *edge neighborhoods.*

Question 5. *Let G be a graph with $e \in E(G)$. Let G_e denote the subgraph of G induced by the set of all vertices of G which are not endpoints of e and are adjacent to at least one endpoint of e. Characterize the graphs F with the property that there exists a graph G such that G_e is isomorphic to F for each edge e of G.*

Zelinka addressed Question 5 in 1986 [80]. Let $\mathscr{N}_e$ denote graphs F for which there is some G with every edge neighborhood isomorphic to F. Zelinka proves that $\mathscr{N}_e$ includes K_n for all n; $K_{m,n}$ for all m, n; C_k for $k \in \{3, 4, 6, 8\}$; and $\overline{C_n}, n \in \{3, 4\}$, among others. Sedláček also addresses Question 5 in [63] where he demonstrates the nonexistence of certain path edge-neighborhood graphs, i.e. that for $6 \neq d \geq 4$ there is no graph G that has P_d as the edge neighborhood of each $e \in E(G)$.

Zelinka also researches the T-Z problem in the context of digraphs. In [82] Zelinka addresses all digraphs of order at most 3 whose neighborhoods are all isomorphic.

Readers interested in more history regarding the T-Z problem might consider the survey paper by Hell [36] as well as the paper by Sedláček [62]. Some of their results are also included in Sections 4.3 and 4.4 to follow.

4.3 Existence results

As previously noted, the complete graphs and the complete multipartite graphs with equal parts represent classes of graphs that are trivially realizable. Other large classes of graphs are realizable, including (most) paths and cycles.

Theorem 8 (Paths and cycles [36])**.** *All paths and cycles, with the exception of P_3, are f-realizable.*

Hell [36] also includes results regarding realizability based on some graph operations, including disjoint unions, Cartesian products, conjuction, and composition of graphs. These methods of producing realizations of graphs based on other graph realizations lead to a more general result, which is that any graph can be made realizable.

Theorem 9 (Any graph can be made realizable [36])**.** *For every graph L, there exists a graph L' such that $L \cup L'$ is realizable.*

A natural corollary of Theorem 9 is that every connected graph is a component of some link graph.

As an example of a graph related to Theorem 9, consider $P_3 \cup K_2$. Hall [31] shows how one might construct finite or infinite realizations of $P_3 \cup K_2$. There are infinitely many realizations of $P_3 \cup K_2$ (finite and infinite) despite one of its components being isomorphic to P_3, which is non-realizable. The non-realizability of P_3 is established in the next section.

Edge subdivision is another simple way to produce link graphs, as described in the following theorem.

Theorem 10 ([36])**.** *Every graph G admits a realizable subdivision G'. Furthermore, G' can be chosen so that all of its subdivisions are also realizable.*

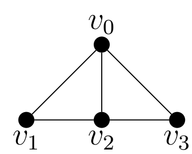
Hell notes that Theorem 10 was discovered independently by himself, Bulitko, and Brown and Connelly.

4.4 Non-existence results

Theorem 8 notes that P_3 is not f-realizable. More specifically, P_3 is not realizable at all.

Proposition 1. *Let P_3 be a path on 3 vertices. There is no graph G such that $G_v \cong P_3$ for every $v \in V(G)$.*

Proof. Let G be a graph and let $v_0 \in V(G)$. Suppose $G_{v_0} \cong P_3$ as shown below:



Consider v_1. It requires one more neighbor for a P_3 neighborhood. Furthermore, this neighbor must be adjacent to either v_0 or v_2. Since $G_{v_0} \cong P_3$ and $G_{v_2} \cong P_3$, it follows that the construction cannot be completed. Hence P_3 is not realizable. $\square$

Some new notation is required for the next results. Let L be a graph, and let $B \subseteq \mathbb{N}$ be non-empty. Let $D_L(B)$ denote the set of vertices of L which have degree b for some $b \in B$; that is,

$$D_L(B) = \{v \in V(L) : \deg(v, L) \in B\}.$$

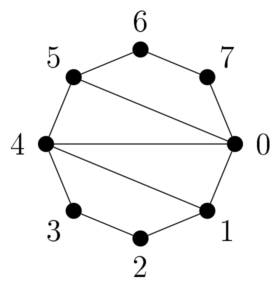
It is understood that $D_L(k) := D_L(\{k\})$, $k \in \mathbb{N}$. As before, $N_L(v)$ denotes the open neighborhood of v in L. In this open neighborhood, vertices of a certain degree might be considered, so define

$$N_L(v, B) := N_L(v) \cap D_L(B).$$

Next, let G be a graph and let $u \in V(G)$ be arbitrary. Let $D_u(B)$ denote the set of vertices $\{v \in V(G_u) : \deg(v, G_u) \in B\}$. Similarly, $N_u(v) := \{w \in V(G_u) : \{v, w\} \in E(G_u)\}$. Let G_{uv} denote the subgraph induced by $N_u(v)$.

The following example demonstrates the notation introduced so far in this section.

Example 18. Let G be the graph below:



Let $B = \{2, 3\}$. Observe the following:

$$
\begin{aligned}
D_G(B) &= \{v \in V(G) : \deg(v, G) \in B\} \\
&= \{1, 2, 3, 5, 6, 7\} \\
D_0(B) &= \{v \in G_0 : \deg(v, G_0) \in B\} \\
&= \{4\} \\
N_G(0, B) &= N_G(0) \cap D_G(B) \\
&= \{1, 4, 5, 7\} \cap \{1, 2, 3, 5, 6, 7\} \\
&= \{1, 5, 7\} \\
N_0(4) &= \{v \in G_0 : \{4, v\} \in E(G_0)\} \\
&= \{1, 5\}
\end{aligned}
$$

△

Lemma 2. *If G is a graph with $\{u, v\} \in E(G)$, then $G_{uv} = G_u \cap G_v$.*

Proof. Let $\{x_1, y_1\} \in E(G_{uv})$. Since x_1 is adjacent to both u and v, $x_1 \in G_u \cap G_v$. Similarly, $y \in G_u \cap G_v$. Hence $\{x_1, y_1\} \in E(G_u \cap G_v)$.

Next, let $\{x_2, y_2\} \in E(G_u \cap G_v)$. It follows that x_2 is adjacent to both u and v, so $x \in V(G_{uv})$. Similarly $y_2 \in V(G_{uv})$, so $\{x_2, y_2\} \in E(G_{uv})$. □

Corollary 3. *If G is a graph with $\{u, v\} \in E(G)$, then $G_{uv} = G_{vu}$.*

Proof. Observe that

$$\begin{aligned} G_{uv} &= G_u \cap G_v \\ &= G_v \cap G_u \\ &= G_{vu}. \end{aligned}$$

Hence $G_{uv} = G_{vu}$. □

The next theorem uses degree arguments and techniques that influence other frequently cited papers on the T-Z problem, namely those by Brown and Connelly [9] and Hall [31]. The proof presented here uses different notation but follows the same general ideas of the original proof.

Theorem 11 (Theorem 1 of Blass, Harary, Miller[8])**.** *Let $k \in \mathbb{N}_0$ and suppose L is a link graph with $D_L(k)$ non-empty. For $B \subseteq \mathbb{N}$, define*

$$f_L(k, B) := \min_{v \in D_L(k)} |N_L(v, B)|$$

and

$$F_L(k, B) := \max_{v \in D_L(k)} |N_L(v, B)|.$$

If B satisfies

$$k < f_L(k, B) + F_L(k, B)$$

then there is some edge $\{x, y\} \in E(L)$ such that $x, y \in D_L(B)$.

Proof. Suppose G has constant link L, and let $u \in V(G)$. Let $v \in G_u$ be such that

$$|N_u(v, B)| = \max_{x \in D_u(k)} |N_u(x, B)|,$$

i.e.

$$|N_u(v, B)| \geq |N_u(x, B)|$$

for every $x \in G_u$. Note that $x \in D_u(k)$ means x has degree k in G_u, where $G_u \cong L$ since G has constant link L. Observe that

$$\begin{aligned}\deg(u, G_v) &= |G_{vu}| \\ &= |G_{uv}| \\ &= k,\end{aligned}$$

by definition of v. Also, since G has constant link,

$$\min_{x \in D_u(k)} |N_u(x, B)| = \min_{x \in D_v(k)} |N_v(x, B)|.$$

By the assumption of the theorem,

$$\begin{aligned}k &< \min_{x \in D_u(k)} |N_u(x, B)| + |N_u(v, B)| \\ &= \min_{x \in D_v(k)} |N_v(x, B)| + |N_u(v, B)|.\end{aligned}$$

Now since

$$|N_v(u, B)| \geq \min_{x \in D_v(k)} |N_v(x, B)|,$$

it follows that

$$k < |N_v(u, B)| + |N_u(v, B)|.$$

Furthermore,

$$|N_v(u, B)| + |N_u(v, B)| - |N_v(u, B) \cap N_u(v, B)| = |N_v(u, B) \cup N_u(v, B)|,$$

where $N_v(u, B) \cup N_u(v, B) = V(G_{uv})$, and $|G_{uv}| = k$. Thus

$$|N_v(u, B)| + |N_u(v, B)| - |N_v(u, B) \cap N_u(v, B)| = k$$

i.e.

$$|N_v(u, B)| + |N_u(v, B)| = k + |N_v(u, B) \cap N_u(v, B)|.$$

Now,

$$k < |N_v(u, B)| + |N_u(v, B)|$$

$$k < k + |N_v(u, B) \cap N_u(v, B)|$$

$$0 < |N_v(u, B) \cap N_u(v, B)|,$$

so there must be some element $w \in N_v(u, B) \cap N_u(v, B)$. Note that this implies $\{u, v\} \in E(G_w)$. Furthermore,

$$\deg(u, G_w) = |G_{wu}| = |G_{uw}| = \deg(w, G_u)$$

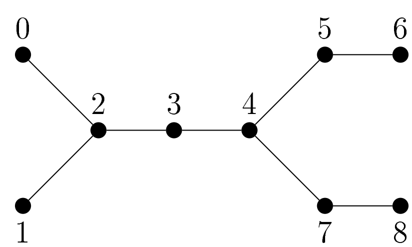
and

$$\deg(v, G_w) = |G_{wv}| = |G_{vw}| = \deg(w, G_v),$$

where $\deg(w, G_u), \deg(w, G_v) \in B$. Hence there is an edge $\{u, v\} \in E(G_w)$ with $u, v \in D_w(B)$. □

The paper by Blass, Harary, and Miller [8] addresses the realizability of trees. The following example is one included in their original paper, with further explanation included here.

Example 19 (Example from Blass, Harary, Miller [8])**.** Theorem 11 will be used in two different ways to show that this L is not realizable. The original paper [8] gives just one example of suitable k and B.



First, let $k = 2, B = \{3\}$ as suggested in the original paper. Relevant data are shown below.

$D_L(2) = \{3, 5, 7\}$

$v \in D_L(2)$	$N_u(v, B)$	$\lvert N_u(v, B)\rvert$
3	$\{2, 4\}$	2
5	$\{4\}$	1
7	$\{4\}$	1

It follows that $f_L(2, \{3\}) = 1$ and $F_L(2, \{3\}) = 2$. Since $2 < 1 + 2$, if L is a link graph, there should be an edge in L such that both endpoints have degree 3 in L. As this is not the case, L is not a link graph.

Next is another way to apply the theorem. Let $k = 3, B = \{2\}$. Relevant data is shown below.

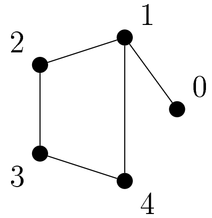
$D_u(3) = \{2, 4\}$

$v \in D_L(2)$	$N_u(v, B)$	$\lvert N_u(v, B)\rvert$
2	$\{3\}$	1
4	$\{3, 5, 7\}$	3

It follows that $f_L(3, \{2\}) = 1, F_L(3, \{2\}) = 3$ Since $3 < 1 + 3$, if L is a link graph, there should be an edge in L such that both endpoints have degree 2 in L. As this is not the case, L is not a link graph. $\triangle$

While Theorem 11 appears in a paper regarding the realizability of trees, it also determines non-realizability of some non-tree graphs, as seen in the following example.

Example 20. Let L be as shown below:



v	0	1	2	3	4
$N_L(v)$	$\{1\}$	$\{0, 1\}$	$\{1, 3\}$	$\{2, 4\}$	$\{1, 3\}$

Let $k = 1$, and let $B = \{3\}$. Next, $D_L(1) = \{0\}$, and

$$f_L(1, \{3\}) = F_L(1, \{3\}) = 1,$$

since the only neighbor of 0 of degree 3 in L is 1.

By Theorem 11, if L is a link graph, there should be an edge in L such that both endpoints have degree 3. As this is not the case, L is not a link graph. $\triangle$

Next is a theorem that is a slight generalization of Theorem 11. Let $\mathscr{B}$ be a family of graphs. Let G be a graph with $u, v \in V(G)$. Define

$$N_u(v, \mathscr{B}) := \{w \in G_u : \{v, w\} \in G_u \quad \text{and} \quad G_{uw} \in \mathscr{B}\}.$$

Theorem 12. *Let $\mathscr{B}$ be a family of graphs. Suppose L is a link graph and let $k \in \mathbb{N}_0$ be such that $D := D_L(k)$ is non-empty. Define*

$$f_L(k, \mathscr{B}) = \min_{v \in D} |N_L(v, \mathscr{B})|$$

and

$$F_L(k, \mathscr{B}) = \max_{v \in D} |N_L(v, \mathscr{B})|.$$

If $\mathscr{B}$ is a non-empty subset of $\mathscr{L} = \{L_x : x \in V(L)\}$ satisfying

$$k < f_L(k, \mathscr{B}) + F_L(k, \mathscr{B})$$

then there is an edge $\{x, y\} \in E(L)$ with $L_x, L_y \in \mathscr{B}$.

Proof. Suppose G has constant link L. Let $u \in V(G)$ be arbitrary and let $v \in D_u$ be such that

$$|N_u(v, \mathscr{B})| = \max_{x \in D_u} |N_u(x, \mathscr{B})|.$$

Note that because G has constant link,

$$\min_{x \in D_v} |N_v(x, \mathscr{B})| = \min_{x \in D_u} |N_u(x, \mathscr{B})|.$$

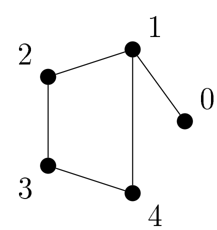
Thus, by the assumption,

$$k < |N_v(u, \mathscr{B})| + |N_u(v, \mathscr{B}|.$$

By similar reasoning as above, $N_v(u, \mathscr{B})$ and $N_u(v, \mathscr{B})$ share a common element w, i.e. $w \in N_v(u, \mathscr{B}) \cap N_u(v, \mathscr{B})$. Note that $w \in N_v(u, \mathscr{B}) \iff G_{vw} \in \mathscr{B}$ and $w \in N_u(v, \mathscr{B}) \iff G_{uw} \in \mathscr{B}$. As $G_{vw} = G_{wv}$ and $G_{uw} = G_{wu}$, it follows that $G_{wv}, G_{wu} \in \mathscr{B}$. Thus $\{u, v\} \in G_w$ is the desired edge with $G_{wu}, G_{wv} \in \mathscr{B}$. □

Next is a simple example of Theorem 12 applied to a potential link graph.

Example 21. Let L be as in Example 20:



The neighborhood subgraphs in L are outlined below:

v	0	1	2	3	4
L_v	K_1	$\overline{K_3}$	$\overline{K_2}$	$\overline{K_2}$	$\overline{K_2}$

Let $k = 1$, and let $\mathscr{B} = \{\overline{K_3}\}$. Next, $D_L(1) = \{0\}$, and

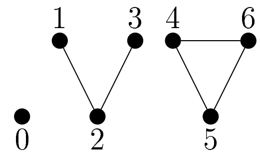
$$\min_{v \in D_L(1)} |N_L(v, \mathscr{B})| = \max_{v \in D_L(1)} |N_L(v, \mathscr{B})| = 1,$$

since the only neighbor of 0 with $\overline{K_3}$ as its neighborhood in L is 1.

By Theorem 12, if L is a link graph, there should be an edge in L such that both endpoints have $\overline{K_3}$ as a neighborhood. As this is not the case, L is not a link graph. △

The next example shows a graph which is non-realizable by Theorem 12 but is not ruled out as a link graph by Theorem 11.

Example 22. Consider the graph L shown below:



It is first verified that L cannot be ruled out as a link graph by Theorem 11. If $k = 1$, then $D_L(1) = \{1, 2\}$, where $N_L(1) = N_L(2) = \{3\}$. Next, observe the following:

B	{0}	{1}	{2}	{0, 1}	{0, 2}	{1, 2}	{0, 1, 2}
$N_L(1, B)$	$\emptyset$	$\emptyset$	{3}	$\emptyset$	{3}	{3}	{3}

It follows that $f_L(1, B) = F_L(1, B) = 1$ for all choices of B such that $2 \in B$. The triangle component of L will ensure that there is an edge with both endpoints in B.

Now let $k = 2$. Next,

$$D_L(2) = \{3, 4, 5, 6\}$$

$$N_L(3) = \{1, 2\}$$

$$N_L(4) = \{5, 6\}$$

The following table excludes data for vertices 5 and 6 since the data will be similar to vertex 4.

B	$N_L(3, B)$	$N_L(4, B)$
{0}	$\emptyset$	$\emptyset$
{1}	{1, 2 }	$\emptyset$
{2}	$\emptyset$	{5, 6 }
{0, 1}	{1, 2 }	$\emptyset$
{1, 2 }	{1, 2 }	{5, 6}
{0, 1, 2 }	{1, 2 }	{5, 6 }

The assumptions of the theorem are satisfied whenever both 1 and 2 are in B, i.e. for $B = \{1, 2\}$ and $B = \{0, 1, 2\}$. In both cases, $f_L(2, B) = F_L(2, B) = 2$. Any edge will satisfy the requirement that both endpoints have a degree in B.

Next, it is verified that L is determined by Theorem 12 to be non-realizable. Let $k = 1$ and $\mathscr{B} = \{\overline{K_2}\}$. Next, $D_L(1) = \{1, 2\}$. The neighborhood induced subgraphs of L are outlined below:

x	0	1	2	3	4	5	6
L_x	$\emptyset$	K_1	K_1	$\overline{K_2}$	K_2	K_2	K_2

It follows that $f(1, \{\overline{K_2}\}) = F(1, \{\overline{K_2}\}) = 1$ so the assumptions of Theorem 12 are satisfied. There is no edge in L such that the neighborhood of each endpoint induces $\overline{K_2}$, so L is not a link graph. $\triangle$

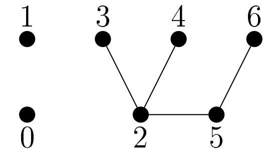
Hall [31] uses techniques similar to those of Blass, Harary, and Miller. The following theorem rules out realizability based on neighborhood-induced subgraphs within the potential link graph itself.

Theorem 13 (Theorem B of Hall [31])**.** *Let L be a link graph and let $\mathscr{B} \subseteq \{L_x : x \in V(L)\}$ be non-empty. Let $B = \{b \in V(L) : L_b \in \mathscr{B}\}$. Let $C = \bigcup_{b \in B} N_L(b)$ and let $\mathscr{C} = \{L_c : c \in C\}$. For each $Y \in \mathscr{C}$, there is an edge $\{a, b\} \in E(L)$ with $L_a \cong Y$ and $L_b \in \mathscr{C}$.*

Proof. Let G be a graph that is locally L. Let $a \in V(G)$ be arbitrary. Let $Y \in \mathscr{C}$ and let $c \in G_a$ be such that $G_{ac} \cong Y$. Note that $c \in C_a$. c has a neighbor in G_a, say b, such that $G_{ab} \in \mathscr{B}$, i.e. $b \in B_a$. Since $G_{ab} \in \mathscr{B}$, $G_{ba} \in \mathscr{B}$ so $a \in B_b$. Since $\{a, c\} \in E(G_b)$, it follows that $c \in C_b$, i.e. $G_{bc} \in \mathscr{C}$. Next, in G_c it follows that $G_{ca} = G_{ac} \cong Y$, and $G_{cb} = G_{bc} \in \mathscr{C}$. Hence in G_c, the edge $\{a, b\}$ yields the desired result. $\square$

The next example serves to illustrate the notation of Theorem 13. Theorem 13 does *not* eliminate the possibility of the graph in the example being a link graph.

Example 23. Let L be the graph shown below:



To show that L is not ruled out as a possible link graph by Theorem 13, it must be verified that a satisfactory edge $\{a, b\}$ exists for each choice of $\mathscr{B}$. First, observe the present neighborhood induced subgraphs:

x	0	1	2	3	4	5	6
L_x	$\emptyset$	$\emptyset$	$\overline{K_3}$	K_1	K_1	$\overline{K_2}$	K_1

Next is data regarding neighborhood subgraphs induced by the endpoints of each edge in L:

$\{x, y\}$	$\{2, 3\}$	$\{2, 4\}$	$\{2, 5\}$	$\{5, 6\}$
L_x	$\overline{K_3}$	$\overline{K_3}$	$\overline{K_3}$	$\overline{K_2}$
L_y	K_1	K_1	$\overline{K_2}$	K_1

The following table considers each possible choice of $\mathscr{B} \subseteq \mathscr{L} = \{L_x : x \in V(L)\}$ and identifies a satisfactory edge $\{a, b\}$.

$\mathscr{B}$	B	C	$\mathscr{C}$	Y	$\{a, b\}$
$\{\emptyset\}$	$\{0, 1\}$	$\{\emptyset\}$	$\{\emptyset\}$		
$\{\overline{K_3}\}$	$\{2\}$	$\{3, 4, 5\}$	$\{K_1, \overline{K_2}\}$	K_1	$\{5, 6\}$
				$\overline{K_2}$	$\{5, 6\}$
$\{K_1\}$	$\{3, 4, 6\}$	$\{2, 5\}$	$\{\overline{K_3}, \overline{K_2}\}$	$\overline{K_3}$	$\{2, 5\}$
				$\overline{K_2}$	$\{2, 5\}$
$\{\overline{K_2}\}$	$\{5\}$	$\{2, 6\}$	$\{\overline{K_3}, K_1\}$		$\{2, 4\}$
$\{\emptyset, \overline{K_3}\}$	$\{0, 1, 2\}$	$\{3, 4, 5\}$	already done –		
			any $\{\emptyset, L_x\}$ done.		

$\{\overline{K_3}, K_1\}$	{2, 3, 4, 6}	{2, 3, 4, 5}	$\{\overline{K_3}, K_1, \overline{K_2}\}$	$\overline{K_3}$	{2, 3}
				K_1	{5, 6}
				$\overline{K_2}$	{5, 6}
$\{\overline{K_3}, \overline{K_2}\}$	{2, 5}	{2, 3, 4, 5, 6}	$\{\overline{K_3}, K_1, \overline{K_2}\}$	done	
$\{K_1, \overline{K_2}$	{3, 4, 5, 6}	{2, 5, 6}	$\{\overline{K_3}, \overline{K_2}, K_1\}$	done	
$\{\emptyset, \overline{K_3}, K_1\}$	{0, 1, 2, 3, 4, 5, 6}	{2, 3, 4, 5}	done		
$\{\overline{K_3}, K_1, \overline{K_2}\}$	{2, 3, 4, 5, 6}	{2, 3, 4, 5, 6}	$\{\overline{K_3}, K_1, \overline{K_2}\}$	done	

It follows that Theorem 13 does not identify L as non-realizable. $\triangle$

Next is an example of a graph that is *not* realizable based on Theorem 13.

Example 24. Let L be the graph from Example 22. Theorem 13 rules out the possibility of L being a link graph. Let $\mathscr{B} = \{\overline{K_2}\}$. Next,

$$B = \{2\}$$

$$C = \{1, 3\}$$

$$\mathscr{C} = \{K_1\}$$

Let $Y \cong K_1$. By Theorem 13, if L is a link graph, it should contain an edge $\{a, b\}$ such that $L_a \cong K_1$ and $L_b \cong K_1$. Since no such edge exists, L is not realizable. $\triangle$

Hall expands on Theorem B in [31] by introducing Theorem BC, where "C" might suggest the notion of clique. Theorem BC rules out realizability based on vertices that induce cliques within the potential link graph. To state and prove Theorem BC, some additional notation is needed.

Let G be a graph. Define $\mathscr{K}_{G,k} := \{Q \subseteq V(G) : G[Q] \cong K_k\}$. That is, $\mathscr{K}_{G,k}$ consists of all vertex subsets in G that induce a clique of order k. For a vertex $v \in V(G)$, let $\mathscr{K}_{v,k}$ denote $\{Q \subseteq V(G_v) : G_v[Q] \cong K_k\}$.

For a subset of vertices $X \subseteq V(G)$, let G_X denote the subgraph of L induced by $\bigcap_{x \in X} N_G(x)$, and call G_X the *intersection neighborhood* of X in G.

Lemma 3. *If $y \in V(G)$ is arbitrary, and $X \subseteq V(G_y)$, then $G_{X \cup \{y\}} = (G_y)_X$.*

Proof. First, show that $V(G_{X \cup \{y\}}) \subseteq V((G_y)_X)$. Let $u \in V(G_{X \cup \{y\}})$. By definition of $G_{X \cup \{y\}}$, u is adjacent to each vertex of $X \cup \{y\}$. In particular, u is adjacent to y, so $u \in (G_y)$. Since $X \subseteq V(G_y)$ and u is adjacent to each vertex of X, it follows that u is adjacent to each vertex of X within G_y, so $u \in V((G_y)_X)$.

Next, show $V((G_y)_X) \subseteq V(G_{X \cup \{y\}})$. Let $u \in V((G_y)_X)$. It follows that u is adjacent to y. Since $X \subseteq V(G_y)$ and u is adjacent to each vertex of X within G_y, it follows that u is adjacent to each vertex of X. Hence u is adjacent to each vertex of $X \cup \{y\}$, i.e. $u \in V(G_{X \cup \{y\}})$. □

The proof of Theorem BC in [31] is omitted from the original paper. Our own proof is included below.

Theorem 14 (Theorem BC in [31])**.** *Let L be a link graph. Let $\mathscr{B}$ be a non-empty subset of neighborhood-induced subgraphs of L such that $B_{\mathscr{B}} \in \mathscr{K}_{L,k}$ for some k. Suppose that for all $Q \in \mathscr{K}_{L,k} - B$, L_B and L_Q are non-isomorphic. For every $Y \in \mathscr{C}_{\mathscr{B}}$, there is an edge $\{a, b\} \in E(L)$ with $L_a \cong Y$ and $L_b \in \mathscr{C}$. Furthermore, if $Y \in \mathscr{C} - \mathscr{B}$, then $L_b \in \mathscr{C} - \mathscr{B}$ also.*

Proof. Suppose G has constant link L. Let $\mathscr{B}$ be a non-empty set of neighborhood-induced subgraphs of L such that $B \in \mathscr{K}_{L,k}$ for some k. Let $Y \in \mathscr{C} - \mathscr{B}$, as the more general case of $Y \in \mathscr{C}$ was addressed in Theorem 13.

Let $a \in V(G)$ be arbitrary, and suppose $c \in C_a$ is such that $G_{ac} \cong Y$. It follows that c has a neighbor b in G_a such that $b \in B_a$. Note that $c \notin B_a$ by definition of Y.

Since $b \in B_a, G_{ab} = G_{ba} \in \mathscr{B}$, so $a \in B_b$. Furthermore, $\{a, c\} \in E(G_b)$. Since $a \in B_b$ and $\{a, c\} \in E(G_b)$ it follows that $c \in C_b$, i.e. $G_{bc} \in \mathscr{C}_{\mathscr{B}}$. If $G_{bc} \in \mathscr{C}_{\mathscr{B}} - \mathscr{B}$

then the proof is finished, as $\{a, b\}$ would be the desired edge in G_c with $G_{ca} \cong Y$ and $G_{cb} \in \mathscr{C} - \mathscr{B}$. Suppose to the contrary that $G_{bc} \in \mathscr{C}_{\mathscr{B}} \cap \mathscr{B}$.

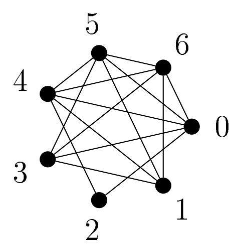
Since $G_{bc} \in \mathscr{B}$, it follows that $c \in B_b$. Let $X = B_b \cup \{b\}$. In G, X induces a clique of order $k+1$. By Lemma 3 G_X, the intersection neighborhood of X in G, is precisely $(G_b)_{B_b}$.

Now since $a \in B_b$, in G_a there is a clique $W \in \mathscr{K}_{a,k}$ such that $V(W) = X - a$. Note that $W \neq B_a$ since $c \in W$ but $c \notin B_a$. Furthermore, Lemma 3, $(G_a)_W = G_X = (G_b)_{B_b}$.

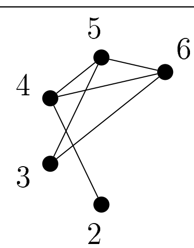
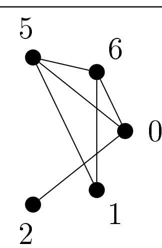
Hence $(G_a)_W = (G_b)_{B_b}$. Since $(G_b)_{B_b} \cong (G_a)_{B_a}$, it follows that $(G_a)_W \cong (G_a)_{B_a}$, a contradiction. $\square$

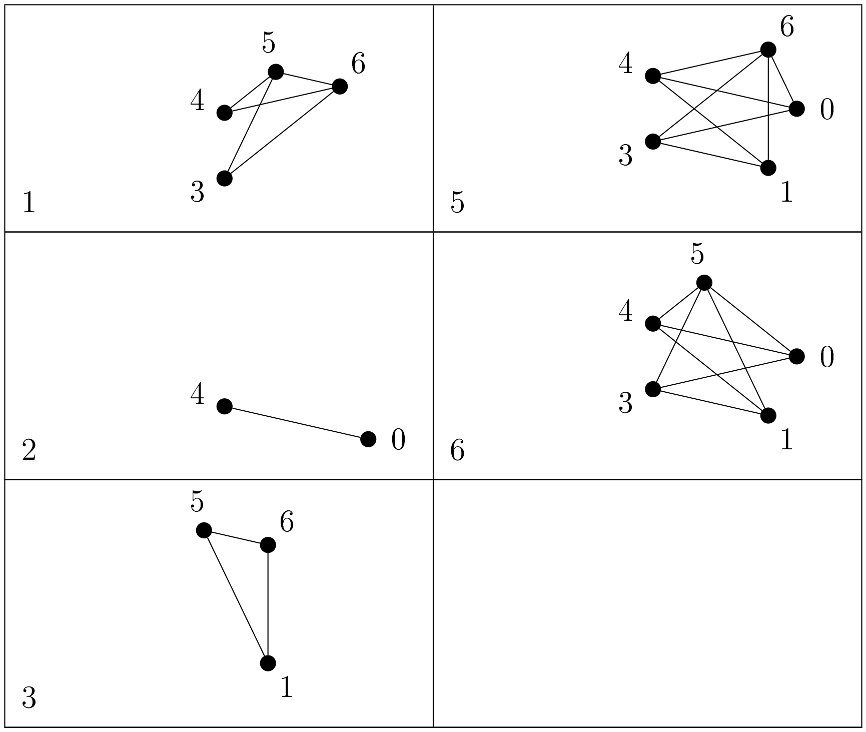
The next example shows a graph that is ruled out as a link graph by Theorem 14 but not by Theorem 13.

Example 25. Let L be the graph below:



The following table outlines the neighborhood subgraphs induced in L:

$v \in V(L)$ $\quad L_v$	$v \in V(L)$ $\quad L_v$
0 	4 



Note that $L_0 \cong L_4$. Let $X \cong L_0$, and let $\mathscr{B} = \{X\}$. It follows that $B = \{0, 4\}$, where $B \in \mathscr{K}_{L,2} = E(L)$. Observe the following:

$Q \in \mathscr{K}_{L,2}$	$V(L_Q)$	L_Q	$Q \in \mathscr{K}_{L,2}$	$V(L_Q)$	L_Q
$\{0, 2\}$	$\{4\}$	K_1	$\{1, 6\}$	$\{3, 4, 5\}$	P_3
$\{0, 3\}$	$\{5, 6\}$	K_2	$\{2, 4\}$	$\{0\}$	K_1
$\{0, 4\}$	$\{2, 5, 6\}$	$K_2 \cup K_1$	$\{3, 5\}$	$\{0, 1, 6\}$	P_3
$\{0, 5\}$	$\{3, 4, 6\}$	P_3	$\{3, 6\}$	$\{0, 1, 5\}$	P_3
$\{0, 6\}$	$\{3, 4, 5\}$	P_3	$\{4, 5\}$	$\{0, 1, 6\}$	P_3
$\{1, 3\}$	$\{5, 6\}$	K_2	$\{4, 6\}$	$\{0, 1, 5\}$	P_3
$\{1, 4\}$	$\{5, 6\}$	K_2	$\{5, 6\}$	$\{0, 1, 3, 4\}$	C_4
$\{1, 5\}$	$\{3, 4, 6\}$	P_3			

As seen in the above table, $L_Q \ncong L_B$ for all $Q \in \mathscr{K}_{L,2} - B$, so the assumptions of

Theorem 14 are satisfied. Next,

$$\begin{aligned}\mathscr{C}_{\mathscr{B}} &= \{L_x : x \in \bigcup_{b \in B} N_L(b)\} \\ &= \{L_x : x \in V(G)\}\end{aligned}$$

Let $Y \cong K_2$, and note that $Y \in \mathscr{C} - \mathscr{B}$. By Theorem 13, if L is a link graph, there is an edge $\{u, v\} \in E(L)$ such that $L_u \cong K_2$ and L_v is isomorphic to some graph in $\mathscr{C} - \mathscr{B}$. Observe that $\{u \in V(L) : L_u \cong K_2\} = \{2\}$. The only neighbors of 2 in L are 0 and 4, where $L_0 \in \mathscr{B}$ and $L_4 \in \mathscr{B}$. Hence L is not a link graph. $\triangle$

4.5 Tree search to construct graph realizations

This section details a realization construction program. The program is framed as a tree search and is implemented in Python. The code, included on GitHub (Appendix I), makes calls to Gurobi. Gurobi is powerful optimization software briefly described in Appendix I.1.2.

4.5.1 Linear programming formulation of subgraph isomorphism problem

Suppose, given two graphs H and G with $|H| \leq |G|$, it is asked whether G contains a subgraph that is isomorphic to H. This is known as the subgraph isomorphism problem. Algorithm 1 shows a linear programming formulation of the problem.

Algorithm 1 Subgraph Isomorphism

INPUT: Graphs H and G of order m and n, respectively

OUTPUT: True if H is isomorphic to a subgraph of G; false otherwise.

PROCEDURE:

Return true if the linear programming problem below returns m for objective. (x_{ij} is 1 if the ith vertex of H is mapped to jth vertex of G; zero otherwise)

$$\begin{aligned}
\text{maximize} \quad & \sum_{\substack{1 \le i \le m \\ 1 \le j \le n}} x_{i,j} \\
\text{subject to} \quad & \sum_{1\le i\le m} x_{i,j} \le 1 && \text{for all } j = 1, \ldots, n \quad \text{(injective)} \\
& \sum_{1\le j\le n} x_{i,j} = 1 && \text{for all } i = 1, \ldots, m \quad \text{(well-defined)} \\
& x_{i,r} + x_{j,s} \le 1 && \text{for all } \{u_i, u_j\} \in E(H) \\
& x_{i,s} + x_{j,r} \le 1 && \text{and all } \{v_r, v_s\} \notin E(G) \\
& && \quad \text{(edge-preserving)} \\
& x_{i,j} \text{ boolean.}
\end{aligned}$$

Algorithm 1 is used in our realization construction program, outlined below. Let L be a graph. The goal of the realization construction program is to construct a graph G such that G is locally L. While our implementation on GitHub (Appendix I) frames the problem as a tree search, the program is ultimately a recursion. Algorithm 2 shows the realization construction program. The subgraph isomorphism program in Algorithm 1 is used to check that $(*)$ in Algorithm 2 is satisfied.

4.6 Graphs of order 7

There are 1,044 graphs of order 7. These graphs are available on Brendan McKay's website [48]. Code that checks for realizability results based on Theorems 11, 12, 13, and 14 is available on GitHub (Appendix I). Our results are summarized in Table 4.3.

Algorithm 2 Search-L-Realizer

(*Note: To guarantee this algorithm halts, place a maximum bound on* $|V(G)|$.)

INPUT: Graphs L and G such that G satisfies

(∗) All vertices have neighborhood isomorphic to a subgraph of L.

OUTPUT: A supergraph of G realizing L, if such a graph exists; otherwise $\varnothing$

PROCEDURE:

```
if |V(G)| is too big then
    return ∅ // exceeded boundary of search
else
    if G_v ≅ L, for all v ∈ V(G) then
        return G
    else
        Let v ∈ V(G) such that G_v ≇ L and G_v is as close to L as possible
        if possible to add new vertices S and new edges to N_G(v) ∪ S so that v's
neighborhood completes to L and new graph satisfies (∗) then
            for All ways to complete the neighborhood of v to L do
                Let H be the next completion of v's neighborhood
                        (so G ⊂ H and H_v ≅ L and H satisfies (∗))
                set J = output of Search-L-Realizer on L, H
                if J ≠ ∅ then
                    return J
                end if
            end for
            return ∅ // loop exhausted all possible ways to complete N(v)
        else
            return ∅ // no way to complete N(v)
        end if
    end if
end if
```

After checking for realizability based on these theorems, 312 graphs of order 7 are left as potentially realizable. This includes some graphs which are definitely realizable, such as K_7.

Given 5 minutes to construct a realization of some L, the tree search constructs realizations of 18 of the 312 unresolved graphs. It runs out of time on 114 graphs. The tree search terminates for 180 graphs, which suggests those graphs may not have realizations of order 40 or smaller.

Theorem	# graphs of order 7 non-realizable by theorem
Theorem 11	355
Theorem 12	589
Theorem 12, NOT Theorem 11	234
Theorem 11, NOT Theorem 12	0
Theorem 13	486
Theorem 14	643
Theorem 13, NOT Theorem 14	15
Theorem 14, NOT Theorem 13	172

Table 4.3: Realizability of graphs of order 7

To continue resolving the realizability of graphs of order 7 (and larger graphs), it would be useful to code more known results, particularly existence theorems.

4.7 Realizations of certain Ramsey graphs

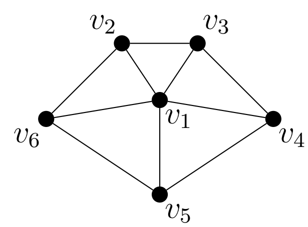
Section 3.5 describes how patterns detected in certain Ramsey graphs are related to the Trahtenbrot-Zykov problem. This section draws more connections between Ramsey graphs and the T-Z problem by exhibiting realizations of some critical Ramsey graphs.

4.7.1 Unique realization of $R(3,3;5)$

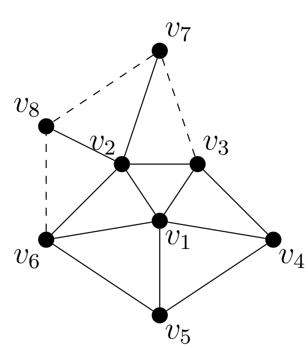
The unique $R(3,3;5)$ graph is isomorphic to C_5, which is uniquely realizable.

Proposition 2. *If G is a connected graph that is locally C_5, then G is isomorphic to the icosahedron graph.*

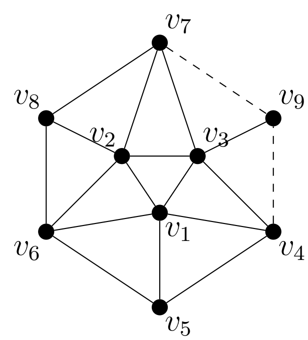
Proof. Suppose G is a connected graph which is locally C_5. Let $v_1 \in V(G)$ with $N_G(v_1) = \{v_2, v_3, v_4, v_5, v_6\}$. Let $\{v_2, v_3\}, \{v_2, v_6\}, \{v_3, v_4\}, \{v_4, v_5\}, \{v_5, v_6\}$ be the edges of the C_5 induced by these vertices. For any vertex $v \in V(G)$, the neighborhood of v is *established* if $G_v \cong C_5$. Hence, the neighborhood of v_1 is established.



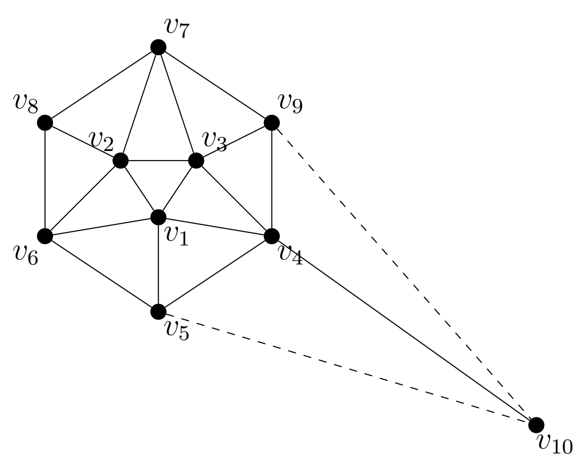
Next, consider v_2, whose currently known neighbors are $\{v_1, v_3, v_6\}$. Note that v_2 cannot be adjacent to any more of $\{v_2, v_3, v_4, v_5, v_6\}$ without disrupting the neighborhood of v_1. *Disrupting* the neighborhood means making changes that will result in an (established) C_5 neighborhood no longer being isomorphic to C_5. Let v_7 and v_8 be the remaining neighbors of v_2, so that $N_G(v_2) = \{v_1, v_3, v_6, v_7, v_8\}$. Without loss of generality, suppose the remaining edges of the C_5 are $\{v_3, v_7\}$, $\{v_6, v_8\}$, and $\{v_7, v_8\}$. The neighborhoods of v_1 and v_2 are now established.



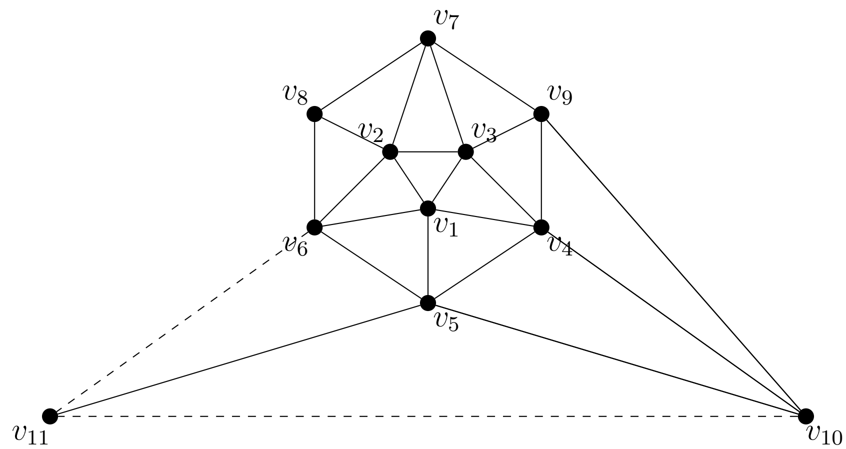
Now consider v_3, whose currently known neighbors are $\{v_1, v_2, v_4, v_7\}$. Again, v_3 cannot be adjacent to any more of the previously identified vertices, so let v_9 be its last remaining neighbor. Now $N_G(v_3) = \{v_1, v_2, v_4, v_7, v_9\}$. Let $\{v_7, v_9\}$ and $\{v_4, v_9\}$ be the remaining edges of the C_5. Thus the neighborhoods of v_1, v_2, and v_3 are established.



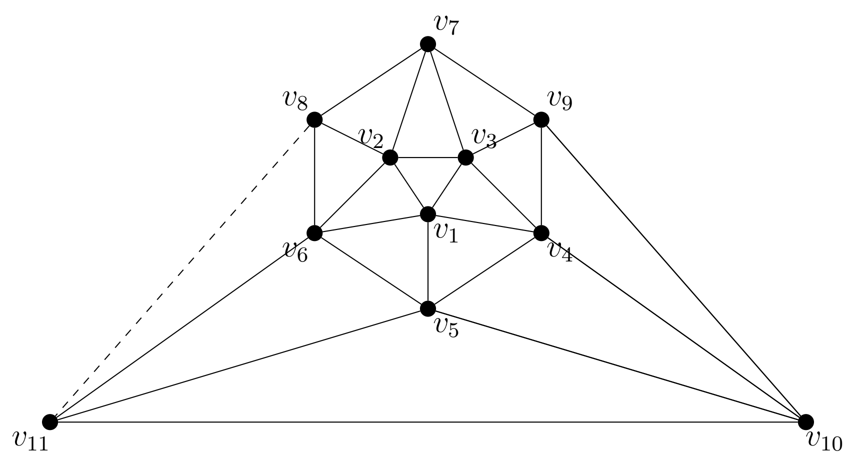
Similarly, the last remaining neighbor of v_4 is v_{10}, and edges $\{v_9, v_{10}\}$ and $\{v_5, v_{10}\}$ are added to complete the C_5 for the neighborhood of v_4, so $N_G(v_4) = \{v_1, v_3, v_5, v_9, v_{10}\}$. This establishes the neighborhood of v_4, so that v_1, v_2, v_3, v_4 have established neighborhoods.



The last neighbor of v_5 should be v_{11} so that $N_G(v_5) = \{v_1, v_4, v_6, v_{10}, v_{11}\}$. Add the edges $\{v_6, v_{11}\}$ and $\{v_{10}, v_{11}\}$ to the graph, and add v_5 to the list of vertices with established neighborhoods.



Note that v_6 is currently of degree 5 with neighbors $\{v_1, v_2, v_5, v_8, v_{11}\}$. The addition of edge $\{v_8, v_{11}\}$ will complete the C_5 and establish the neighborhood of v_6.

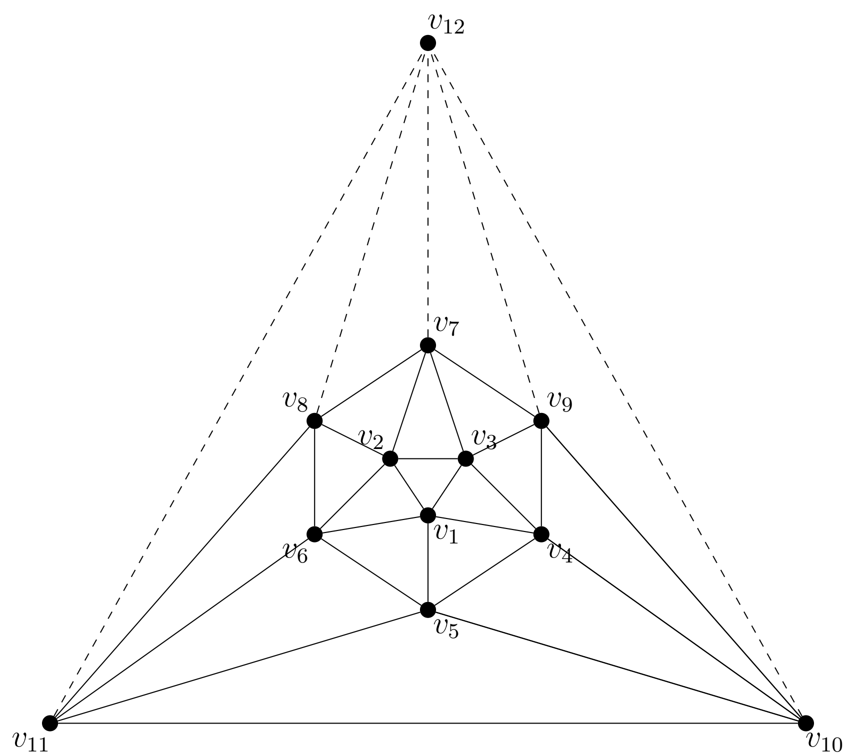


Now vertices v_7, v_8, v_9, v_{10} and v_{11} are all of degree 4. The neighborhoods of all prior vertices are already established, so there must be another vertex v_{12} adjacent to all five of $v_7, v_8, v_9, v_{10}, v_{11}$.

Suppose instead that v_{12} is added and has some new neighbor(s) yet to be identified; for the sake of illustration, suppose more specifically that $N_G(v_{12}) = \{v_{11}, v_{13}, v_{14}, v_{15}, v_{16}\}$. There must be a C_5 among these vertices, but v_{11}'s neighborhood is already established, so it can't be adjacent to any of these new vertices.

This reasoning would apply with any number of "new" neighbors introduced. Hence it must be the case that $N_G(v_{12}) = \{v_7, v_8, v_9, v_{10}, v_{11}\}$.

Observe that the resulting graph is 5-regular and is also locally C_5. Furthermore, this graph is isomorphic to the icosahedron.



No more vertices or edges can be added while maintaining regularity, so this graph is indeed the unique connected graph which is locally C_5. □

4.7.2 Realizations of H_3

The $R(4, 4; 17)$ graph, shown in Figure 2.2, is locally H_3, where H_3 is a critical Ramsey(3, 4) graph shown in Figure 2.1. While this is clear through simple observation, a constructive proof is in Section 6.3.2. The tree search procedure described in Section 4.5 produces another realization of H_3. This second realization is of order 21 and it shown in Figure 4.1. The second realization is also a Cayley graph and is constructed in Section 6.3.2. Based on the tree search, we conjecture that these two known realizations of H_3 are in fact the only finite realizations:

Conjecture 2. *Only two finite realizations of H_3 exist, specified in Table 4.4.*

Automorphism groups of the realizations are computed using GAP (Appendix I.2). A quick summary of the H_3 realizations is given in Table 4.4. The summary includes graph6 specifications for these graphs; graph6 is a text-based format for graph specification. For more information regarding the graph6 codes, see Appendix II.

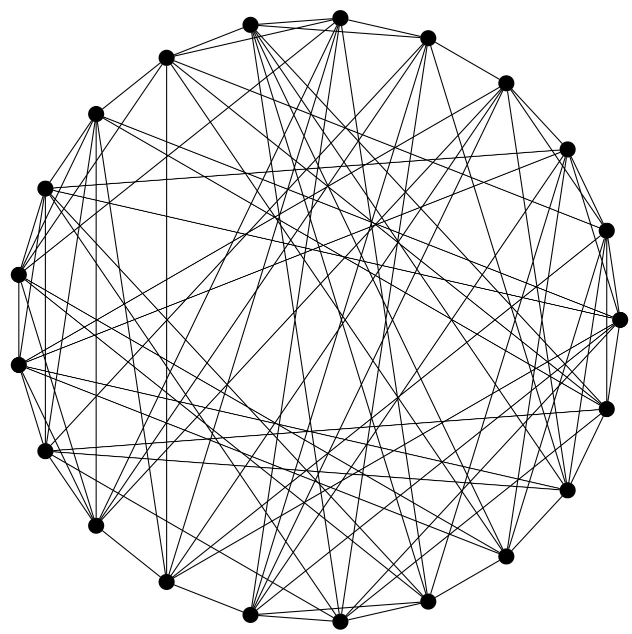


Figure 4.1: 21-vertex realization of H_3

4.7.3 Realizations of H_2

The graph H_2 is a critical Ramsey(3, 4) graph shown in Figure 2.1. The tree search produces three realizations of H_2. A summary of these graphs is given in Table 4.5.

	Realization 1	Realization 2
Order	17	21
Aut(G)	$\mathbb{Z}_{17} \rtimes \mathbb{Z}_8$	$PGL_2(\mathbb{F}_7)$
Vertex transitive	Yes	Yes
Cayley graph*	Yes	Yes

* constructions in Chapter 6

- Realization 1 graph6 specification: PsOihr~lEW{OeRSuLIhM]Fdg
- Realization 2 graph6 specification: TsOihr~sd?uGoBQEhoPYQHBgQCgaagKo{HFe
- *Note: See Appendix II for adjacency matrices.*

Table 4.4: Summary of known H_3 realizations

Automorphism groups are computed via GAP (Appendix I.2). Note that Realization 2 has an automorphism group of order 8. Vertex-transitive graphs of order n have automorphism groups of order at least n, so it follows that this graph is not vertex-transitive due to its automorphism group being too small.

	Realization 1	Realization 2	Realization 3
Order	21	24	24
Aut(G)	$\mathbb{Z}_7 \rtimes \mathbb{Z}_6$	$\mathbb{Z}_4 \times \mathbb{Z}_2$	$\mathbb{Z}_2 \times A_4$
Vertex transitive	Yes	No (!)	Yes
Cayley graph*	Yes	No	Yes

* constructions in Chapter 6

- Realization 1 graph6 specification:
  T@hZCf~KDOkPIcRBP_QghDSqPKoEN]Cdb@XH
- Realization 2 graph6:
  W@hZCf~N@_CRiACSA`KOaR?hCSSEBe?TU@BSOpBICWm?@|E
- Realization 3 graph6:
  W@hZCf~N@_CRiACSA`KOaR?hCSSABm?TUBBOOr@qCGx?@|D
- *Note: See Appendix II for adjacency matrices.*

Table 4.5: Summary of known H_2 realizations

CHAPTER 5
UNDECIDABILITY OF THE TRAHTENBROT-ZYKOV PROBLEM

The Trahtenbrot-Zykov problem and terminology associated with it (e.g. *realization*, *link graph*, *locally* F) are addressed in Chapter 4.

Theorem 15 (Bulitko, 1973 [12])**.** *There is no algorithm that can determine, given any graph F, whether there exists a graph G in which the subgraph induced by the open neighborhood of every vertex is isomorphic to F.*

In 1973, Bulitko proved that the link problem is undecidable [12], i.e. that there is no general algorithm that can determine whether a particular graph F is realizable or not. The result is based on another famous undecidable problem – the domino problem, detailed in Section 5.1. Bulitko's paper, published in Russian, is not available in English. This chapter contains a translation of the first section of Bulitko's paper, which establishes the undecidability of the Trahtenbrot-Zykov problem. We make some modifications to the notation and structure of Bulitko's proof, but the ideas presented here are largely due to Bulitko. The second section of Bulitko's paper addresses classes of graphs for which the link problem is decidable; that section is not translated in this work.

5.1 Introduction to domino problem

A domino is a square with edges colored. All dominoes are the same size. Each domino edge has a particular color. For every domino, there is an unlimited set of copies; these copies have a certain domino *type*. See Figure 5.1 for examples of domino types.

The problem is to cover the plane (quadrant) using copies of a specific set of types of dominoes under the following restrictions:

(D1) Dominoes may not be rotated.

(D2) Dominoes may not be reflected.

(D3) Dominoes may not overlap.

(D4) Adjacent edges between two dominoes must be the same color.

Tiles like the ones shown in 5.1 are sometimes called *Wang tiles*, due to the following famous result.

Theorem 16 (Wang [38]). *There is no general algorithm that can correctly determine whether any set of domino types can be used to tile the plane.*

Let P be a finite set of domino types. The pair (Q, P) with $Q \subseteq P$ is *solvable* on the plane if there is a covering of the plane by means of dominoes whose types belong to P and in the covering there is a domino whose type belongs to Q. Similarly, (Q, P) is said to be *solvable in the quadrant* (first quadrant) if it is possible to tile the first quadrant in such a way that the leftmost domino in the bottom row is from Q.

The domino types in Figure 5.1 will be used for some examples. Let $P = \{d_1, d_2, d_3, d_4, d_5, d_6\}$ from the figure. The set $Q_0 = P$ is solvable. The set $Q_1 = \{d_1, d_3, d_5\}$ is also solvable. Note that the set $Q_2 = \{d_5, d_6\}$ is not solvable, because there is no tiling of the plane using dominoes of type $d_i \in P$ in which d_5 or d_6 will appear.

A finite set P of domino types is *strongly solvable* on the plane if, for every $d \in P$, the pair $(\{d\}, P)$ is solvable on the plane. That is, a set of domino types is

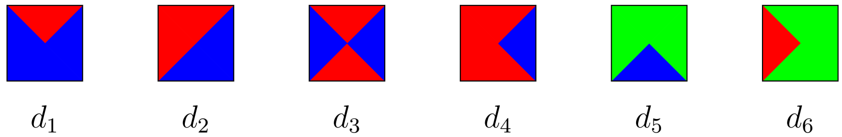


Figure 5.1: Some domino types

strongly solvable if each type in the set gets used in *some* tiling of the plane, i.e. there exists some tiling of the plane using dominoes from P where a domino of type d is used, for every $d \in P$. Hence the set P from 5.1 is not strongly solvable.

5.2 Construction of the graph $L(P)$

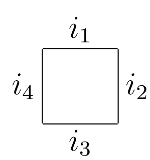
Let P be a finite set of domino types. This section constructs a graph $L(P)$ to serve as a model of these domino types. The graph $L(P)$ is a disconnected graph consisting of $|P| + 3$ components: one component for each domino type, labeled $L(d_i)$ for $d_i \in P$, and the components L_A, L_B, and L_F, which will be specified later. The goal is to construct a graph with the following properties:

(L1) For each $d_i \in P$, $L(d_i)$ has no nontrivial automorphisms.

(L2) For $d_i, d_j \in P$, with $i \neq j$, no supergraph of $L(d_i)$ is isomorphic to any subgraph of $L(d_j)$.

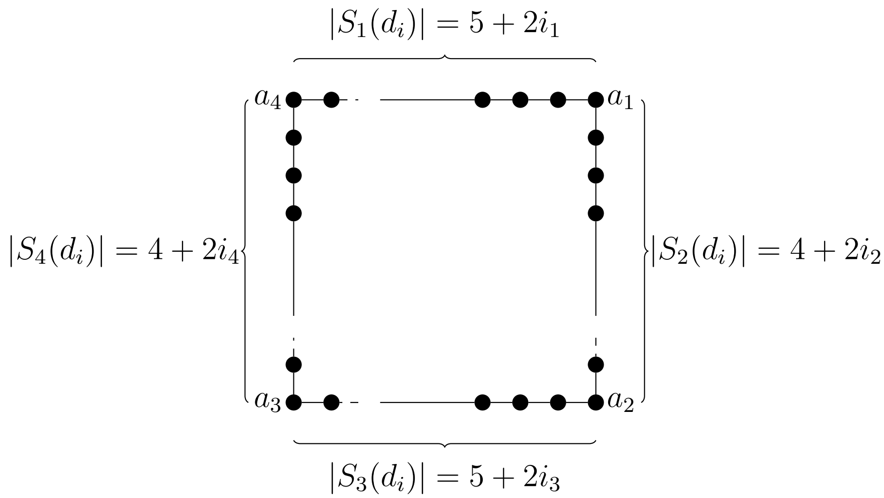
(L3) For each $d_i \in P$, no supergraph of L_A is isomorphic to any subgraph of $L(d_i)$.

To begin constructing this graph, let $d_i \in P$ be a domino type as shown below:



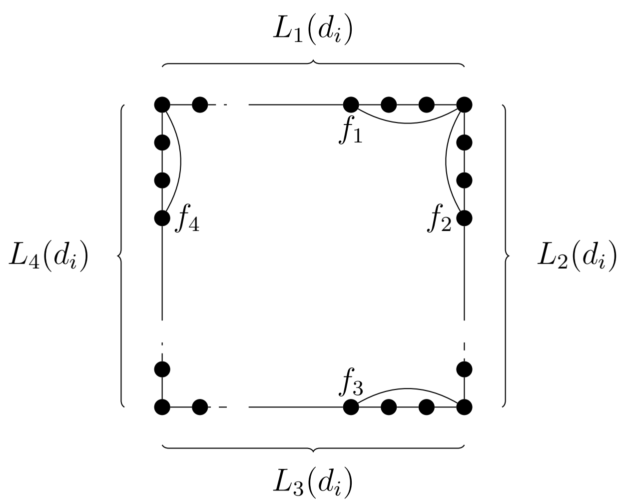
The first component of $L(P)$ constructed is the graph $L(d_i)$, corresponding to a single domino type. The graph $L(d_i)$ is assembled in three phases:

Phase 1:



The vertices a_1, a_2, a_3, a_4 are called *corner vertices.*

Phase 2:



The vertices f_1, f_2, f_3, f_4 are called *fulcrum vertices.* These serve as a way to "lock" adjacent dominoes together, as the first and third sides have compatible fulcrums, as well as the second and fourth sides. See Figure 5.2 for an illustration. The diagram above also draws attention to the subgraphs $L_k(d_i)$, $k \in \{1, 2, 3, 4\}$, where $L_k(d_i)$ coincides with the side of the domino colored i_k.

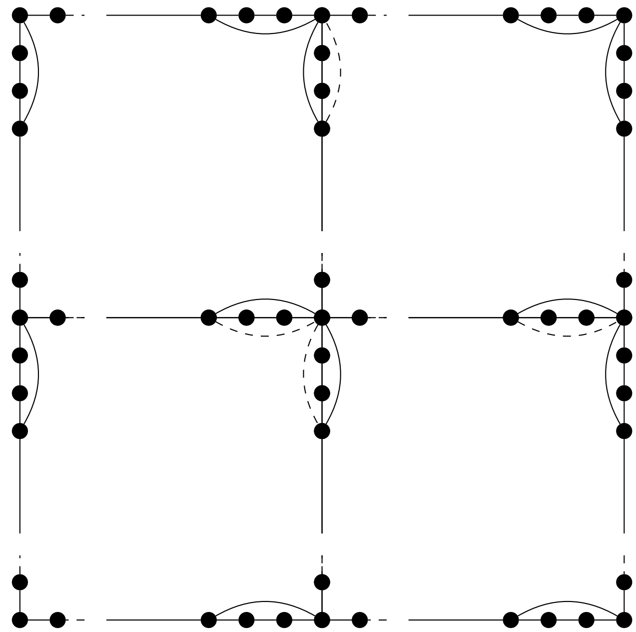
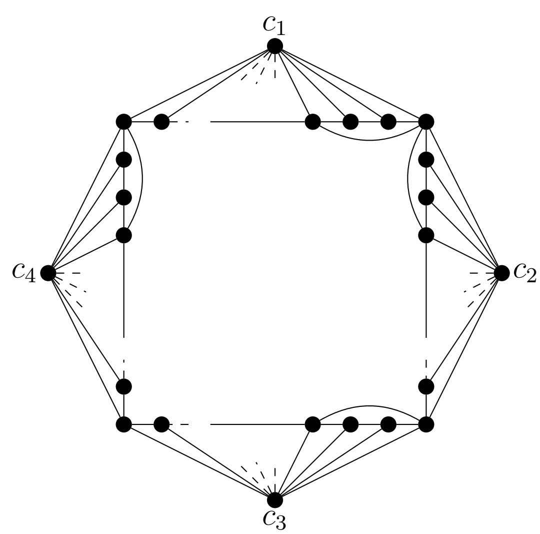


Figure 5.2: Fulcrum vertices serve as a locking mechanism.

Phase 3:



The vertices c_1, c_2, c_3, c_4 are called *apex vertices.*

This completes the construction of $L(d_i)$. Based on this construction of $L(d_i)$, the following properties of $L(P)$ are established:

(L4) Any triangle in $L(d_i)$ contains an apex vertex.

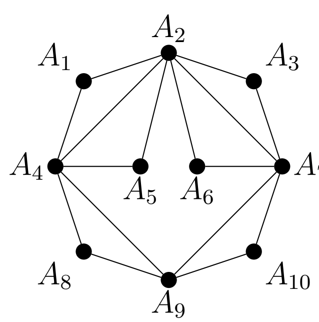
(L5) Any vertex that is adjacent to two distinct apex vertices must be a corner vertex.

(L6) No corner vertex is adjacent to any other corner vertex. No apex vertex is adjacent to any other apex vertex.

(L7) Corner vertices have degree 4, 5, or 6. Fulcrum vertices have degree 4. Apex vertices are of degree at least 7. All other vertices of $L(d_i)$ are of degree 3.

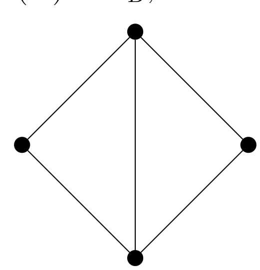
(L8) The open neighborhood in $L(d_i)$ of each fulcrum vertex is isomorphic to $K_{1,3}$.

Next, the three other components of the graph $L(P)$ are specified. The graph L_A is shown below:



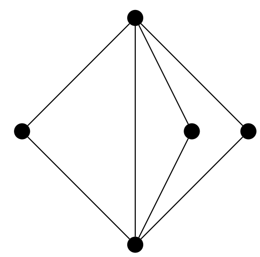
A result later in this chapter (Lemma 5) shows that this graph L_A is related to the corner vertices in $L(d_i)$.

The next component of $L(P)$ is L_B, shown below:



The graph L_B is isomorphic to the subgraph of $L(d_i)$ induced by the open neighborhood of any degree 3 vertex, with every vertex connected to an additional universal vertex.

The last component of $L(P)$ is L_F, shown below:



The graph L_F is isomorphic to the subgraph of $L(d_i)$ induced by the open neighborhood of any fulcrum vertex, with every vertex connected to an additional universal vertex.

The graph $L(P)$ is defined in terms of the above components:

Definition 51 ($L(P)$)**.** *Let P be a finite set of domino types. The graph $L(P)$ is defined by*

$$L(P) := L_A \cup L_B \cup L_F \cup \bigcup_{d_i \in P} L(d_i),$$

where $\cup$ denotes the disjoint union of graphs.

With all components specified, the following property of $L(P)$ is also observed:

(**L9**) Any vertex of $L(P)$ whose neighborhood contains a “long” path (a path of order at least 4) must be an apex vertex of some $L(d_i)$.

The properties of $L(P)$ outlined in this section are clear through observation of the graph. The next section establishes less pronounced properties of $L(P)$.

5.3 More properties of $L(P)$

Let G be a graph that is locally $L(P)$ for some finite set P of domino types. Let $x \in V(G)$ be arbitrary. As in Chapter 4, G_x denotes the subgraph of G induced by the open neighborhood of x. Let $G_x(d_i)$ denote the subgraph (component) of G_x

which is isomorphic to $G(d_i)$ for some domino d_i. Similarly, let $G_x[d_i]$ denote the graph $G_x(d_i)$ along with the vertex x as a universal vertex.

Lemma 4. *Let P be a finite set of domino types. Let G be a graph that is locally $L(P)$. Let $u \in V(G)$ and $d_j \in P$ both be arbitrary. If $v \in V(G)$ is an apex vertex of $G_u(d_j)$, then u is an apex vertex of $G_v(d_m)$ for some $d_m \in P$.*

Proof. Let G be locally $L(P)$. Let $u \in V(G)$ and $d_j \in P$ both be arbitrary. Let $v \in V(G)$ be an apex vertex of $G_u(d_j)$. Note that $G_{uv} = G_{vu}$ has some component which contains a long path. This long path coincides with some $G_v(d_m)$ for $d_m \in P$. In particular, each vertex of this long path is adjacent to u, so u is an apex vertex of $G_v(d_m)$ by (**L9**). $\square$

Lemma 5. *Let P be a finite set of domino types. Let G be a graph that is locally $L(P)$. Let $u \in V(G)$ and $d_j \in P$ both be arbitrary. If $v \in V(G)$ is a corner vertex of $G_u(d_j)$, then $(G_u[d_j])_v$ is isomorphic to some subgraph of L_A.*

Proof. Let $G_u[d_j]$ be labeled as shown in Figure 5.3. First consider vertex a_3; as shown later, the cases for the other corner vertices can be resolved similarly to this one. The current neighborhood of a_3 in $G_u[d_j]$ is shown in Figure 5.7. Note that $(G_u[d_j])_{a_3}$ is not isomorphic to any subgraph of L_B or L_F, so it must be isomorphic to a subgraph of either L_A or $L(d_p)$ for some $d_p \in P$. Suppose, to the contrary, that $(G_u[d_j])_{a_3}$ is a subgraph of $L(d_p)$.

Based on its degree, u must be either a corner or apex vertex in $G_{a_3}(d_p)$. It cannot be a fulcrum vertex because it has two distinct pairs of adjacent neighbors, a violation of (**L8**). Consider each case.

First, suppose u is an apex vertex in $G_{a_3}(d_p)$. Its neighborhood in $G_{a_3}(d_p)$ must contain a long path (**L9**). Note that in $G_u[d_j]$, the graph $G_u \cap G_{a_3} = G_{ua_3}$ is completely specified; that is, there can be no more vertices adjacent to u in $G_{a_3}(d_p)$, as these vertices would be adjacent to a_3 also. Similar restrictions prevent adding

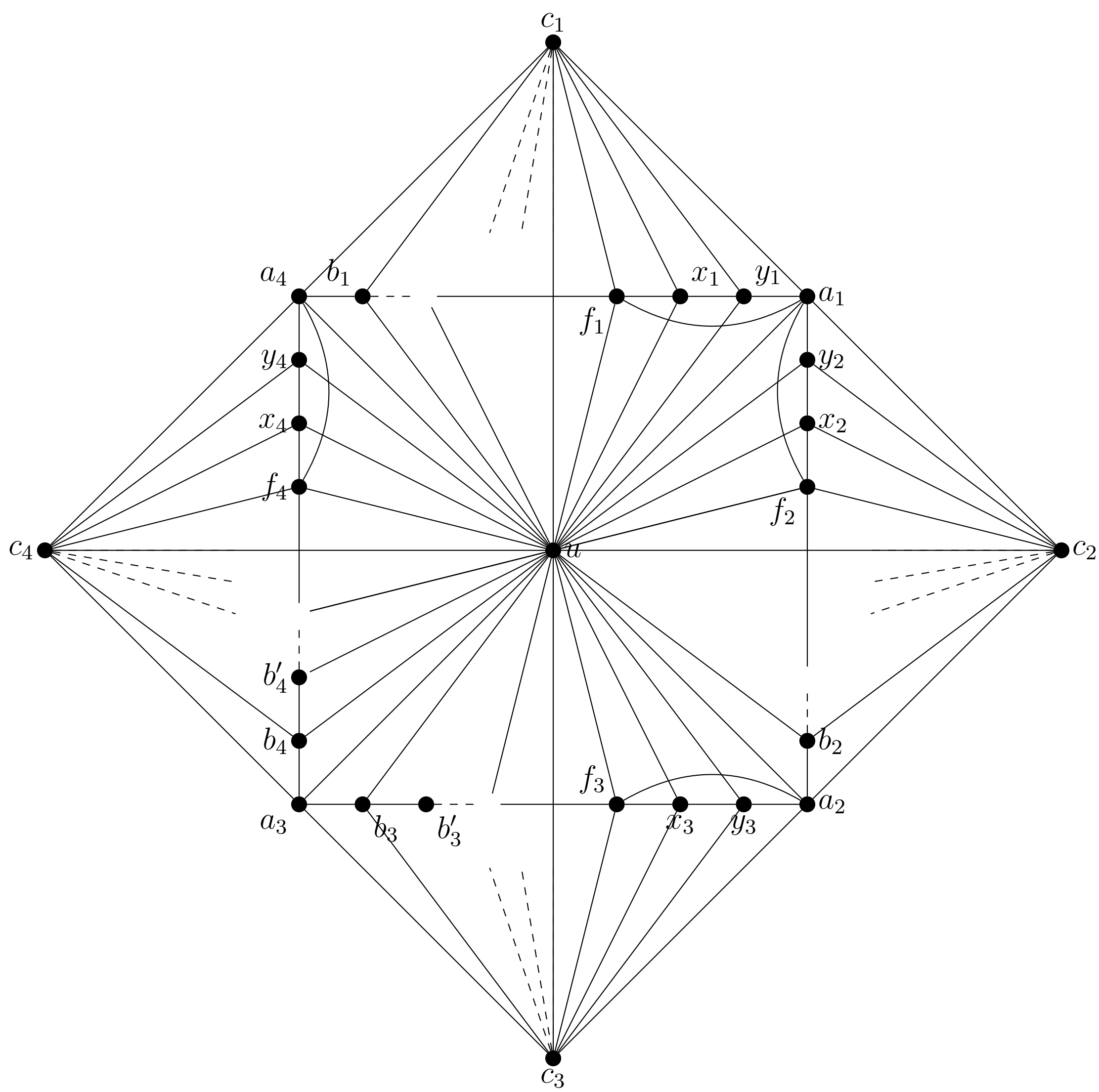


Figure 5.3: $G_u[d_j]$

any new edges to $G_{a_3}(d_p)$. It is therefore impossible to create the long path required in $(G_{a_3}(d_p))_u$.

Next, suppose u is a corner vertex in $G_{a_3}(d_p)$. Consider the triangles $\{u, b_3, c_3\}$ and $\{u, b_4, c_4\}$ in $G_{a_3}(d_p)$. Each of these triangles must contain an apex vertex (**L4**). Since u is a corner vertex, it cannot be an apex vertex. Thus, consider two cases: b_3 is an apex vertex, or c_3 is an apex vertex.

Case 1. Suppose b_3 is an apex vertex. Consider its neighborhood, shown in Figure 5.4. Since b_3 is an apex vertex of $G_{a_3}(d_p)$, it follows by Lemma 4 that a_3 is an apex vertex of some $G_{b_3}(d_r)$, $d_r \in P$. Furthermore, since $\{u, c_3, b_3'\}$ forms a triangle in $G_{b_3}(d_r)$, b_3' must also be an apex vertex (otherwise, there would be two adjacent apex vertices, a violation of (**L6**)). However, any common neighbors between two apex vertices must be corner vertices (**L5**), and corner vertices are not adjacent to other corner vertices (**L6**), so this is a contradiction.

Case 2. Suppose c_3 is an apex vertex. Since c_3 is an apex vertex of $G_{a_3}(d_p)$, a_3 must be an apex vertex of some $G_{c_3}(d_q), d_q \in P$ (Lemma 4). Similarly, since c_3 is an apex vertex of $G_u(d_j)$, u must be an apex vertex of some $G_{c_3}(d_{q'})$. In fact, since u is adjacent to a_3, u and a_3 are in the same component of G_{c_3}, so $q = q'$. This is a contradiction, as no apex vertex is adjacent to any other apex vertex (**L6**). A similar argument can be used to reach a contradiction when c_4 is an apex vertex.

Consider a_1, a_2, and a_4, whose neighborhoods are shown in Figures 5.5, 5.6, and 5.8, respectively. Observe that each neighborhood contains $G_u[d_j]_{a_3}$ as a subgraph. It can thus be similarly argued that each of these neighborhoods must be a subgraph of L_A, since the previous arguments regarding degree and vertex roles would be the same.

Thus, $(G_u[d_j])_{a_k}$ is a subgraph of L_A for each $k \in \{1, 2, 3, 4\}$. □

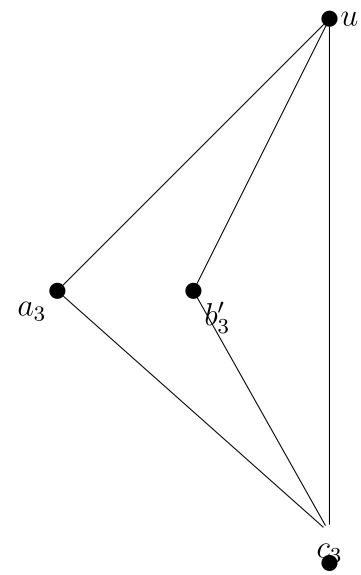


Figure 5.4: Neighborhood of b_3 in $G_u[d_j]$

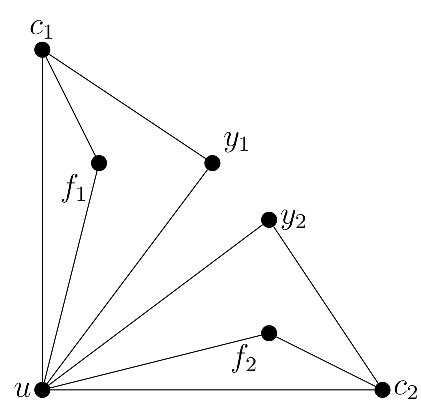


Figure 5.5: Neighborhood of a_1 in $G_u[d_j]$

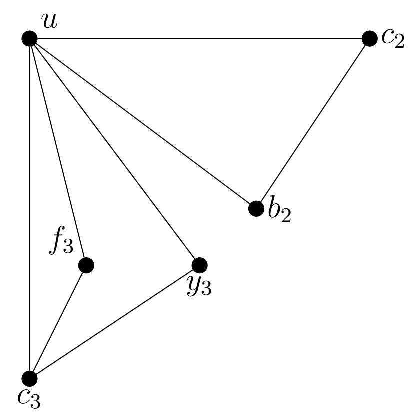


Figure 5.6: Neighborhood of a_2 in $G_u[d_j]$

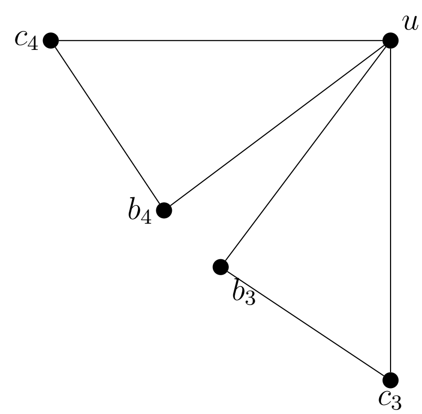


Figure 5.7: Neighborhood of a_3 in $G_u[d_j]$

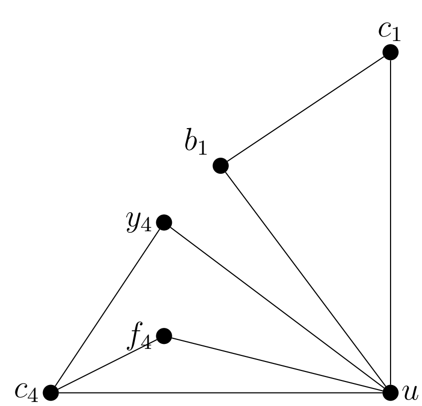


Figure 5.8: Neighborhood of a_4 in $G_u[d_j]$

Theorem 17 (Lemma 1 of [12])**.** *If $L(P)$ is a link graph, then P is strongly solvable in the plane.*

Proof. The proof begins by constructing a graph that is locally $L(P)$. It is then verified that this local graph coincides with a strongly solvable P.

Let G be locally $L(P)$. Let $u \in V(G)$ and $d_j \in P$ both be arbitrary. Let $G_u[d_j]$ be labeled according to Figure 5.3. The first goal is to complete the neighborhood component of each corner vertex. Consider a_1, whose current neighborhood is shown in Figure 5.5. By Lemma 5, $(G_u(d_j))_{a_1}$ must be a subgraph of L_A, i.e. u is in the L_A component of G_{a_1}.

Because u currently has degree 6, it must play the role of A_2 in L_A (see L_A on page 83). Since c_1 and c_2 each have two common neighbors with u, they must play the roles of A_4 and A_7, respectively. Next, f_1 and y_1 play the roles of A_1 and A_5 (interchangeably) and f_2 and y_2 play the roles of A_3 and A_6 (interchangeably). At this point, all vertices in the current neighborhood have assigned roles, so add three new vertices, t_1, t_2, and t_3, to the graph G to fulfill the roles of A_8, A_{10}, and A_9 respectively in $G_{a_1}(L_A)$. A "snapshot" of this portion of G is shown in Figure 5.9.

Next, consider $G_{a_2}(L_A)$. Since u is of degree 5, it must play the role of A_4 (or, equivalently, A_7). It cannot be A_2 because that would require more neighbors, but the neighborhood of u in this component is already fully specified (a consequence of $G_u \cap G_{a_3}$ being fully specified by $G_u[d_j]$). Next, since c_3 has two shared neighbors with u, it must play the role of A_2. Then f_3 and y_3 are, without loss of generality, A_1 and A_5. Whichever of b_2 and c_2 is assigned to A_9 will eventually share a neighbor with c_3 and a_2. For ease of the drawing, let c_2 be assigned to A_9. Now, add new

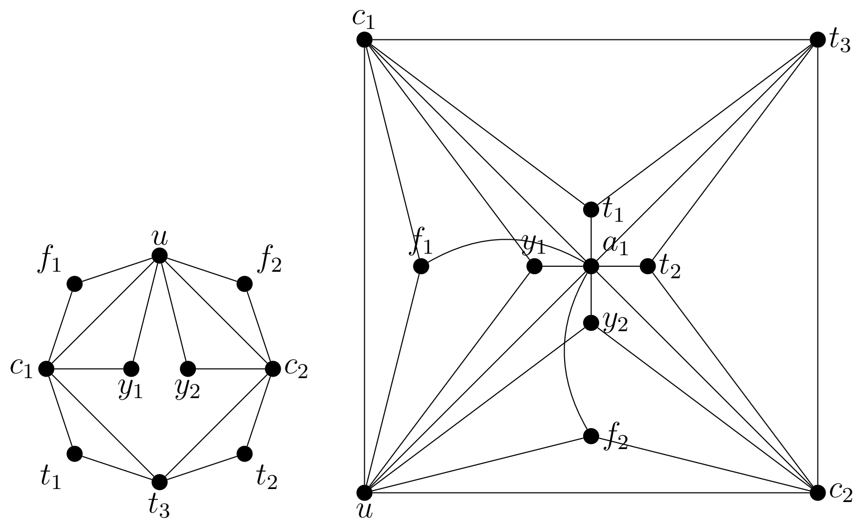


Figure 5.9: $G_{a_1}(L_A)$ and an updated partial view of G

vertices t_4, t_5, t_6, and t_7 to G to respectively play the roles of A_3, A_6, A_7, and A_{10} within $G_{a_2}(L_A)$. See Figure 5.10 for an updated view of the graph.

Since the specification of $G_{a_4}(L_A)$ will be similar to the procedure for a_2, that is next. The vertices f_4, c_4, u, y_4, b_1, and c_1 will play the roles of A_1, A_2, A_4, A_5, A_8, and A_9, respectively. New vertices t_8, t_9, t_{10}, and t_{11} are introduced to fulfill the roles of A_3, A_6, A_7, and A_{10}, respectively. See Figure 5.11.

Finally, consider $G_{a_3}(L_A)$. Let c_3 and c_4 play the roles of A_4 and A_7 respectively. Let b_3 and b_4 play the roles of A_8 and A_{10}, and let u play the role of A_9. Introduce vertices t_{12}, t_{13}, t_{14}, t_{15}, and t_{16} to fulfill the roles of A_1, A_2, A_3, A_5, and A_6, respectively. See Figure 5.12.

Now that all of the corner vertices have been addressed, an updated view of G is in Figure 5.13.

Next, consider c_1. Since the degree of c_1 exceeds the order of each of L_A, L_B, and L_F, it follows that $(G_u)_{c_1}$ must be a subgraph of some $L(d_p)$, $d_p \in P$. Consider $G_u(d_j) \cap G_{c_1}(d_p)$. On one hand, this graph is isomorphic to $L_1(d_j)$; on the other hand, it is isomorphic to some $L_k(d_p)$. Since $L_1(d_j)$ has an odd number of vertices,

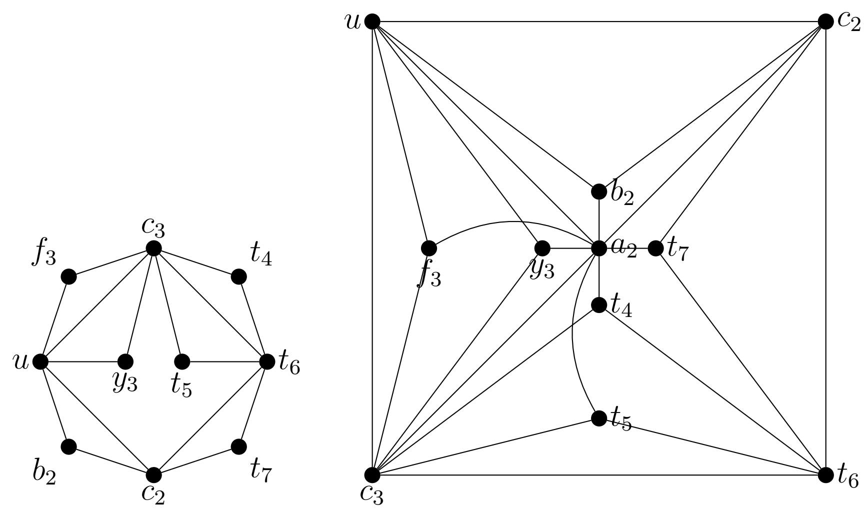


Figure 5.10: $G_{a_2}(L_A)$ and an updated partial view of G

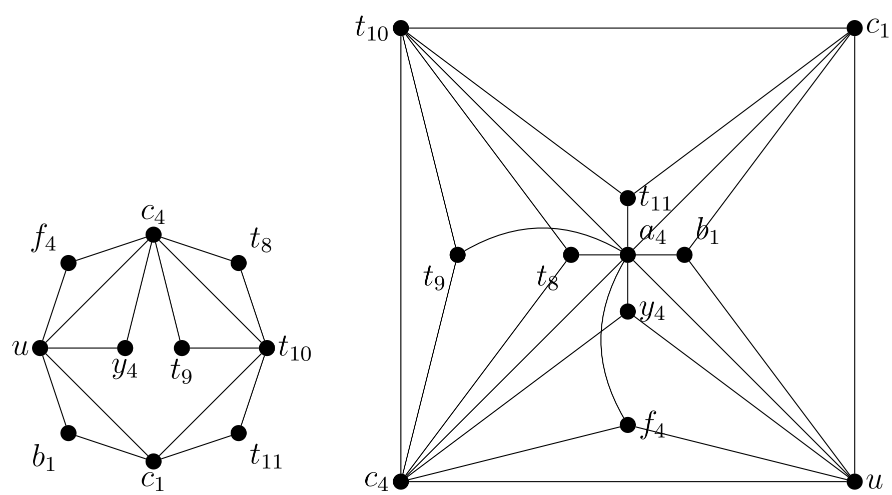


Figure 5.11: $G_{a_4}(L_A)$ and an updated partial view of G

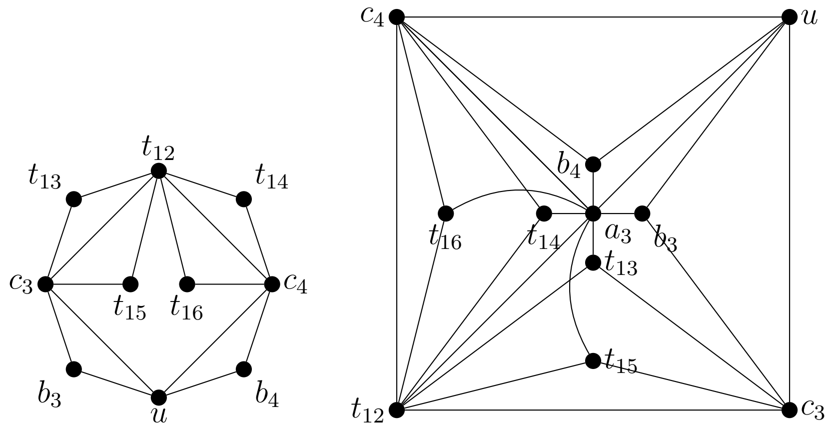


Figure 5.12: $G_{a_3}(L_A)$ and an updated partial view of G

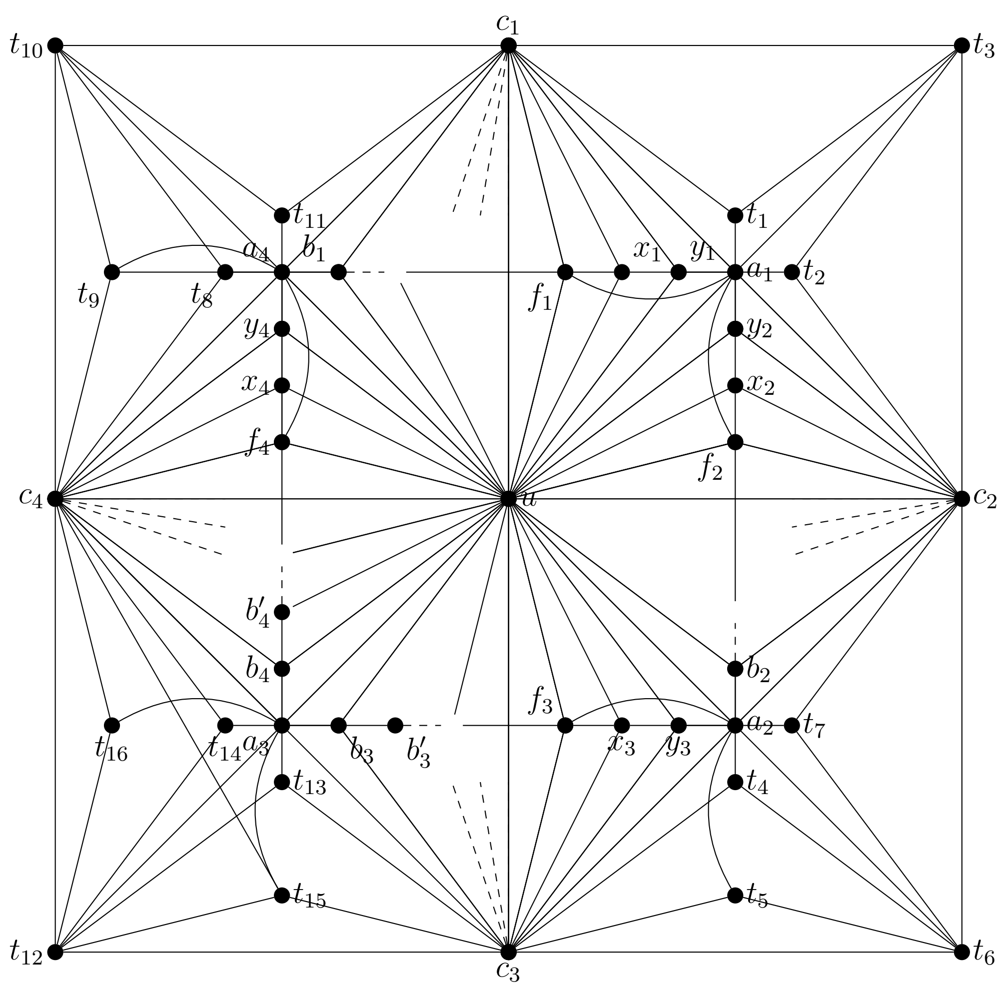


Figure 5.13: Updated view of G after specifying neighborhoods of corner vertices in $G_u(d_j)$

it follows that k must be 1 or 3. Suppose $k = 1$. Then, without loss of generality, there must be some fulcrum vertex f_5 such that $\{a_1, f_5\}$ and $\{c_1, f_5\}$ are edges in $G_{c_1}[d_p]$. This disrupts the already established $G_{a_1}(L_A)$ component, however. Hence k must be 3, and it follows that $G_u(d_j) \cap G_{c_1}(d_p) \cong L_1(d_j) \cong L_3(d_p)$.

Next, consider c_2. Again, $(G_u[d_j])_{c_2}$ must be a subgraph of some $L(d_q)$, $d_q \in P$. Consider $G_u(d_j) \cap G_{c_2}(d_q)$. This graph is, on the one hand, isomorphic to $L_2(d_j)$, and on the other, $L_k(d_q)$. Since $L_2(d_j)$ has an even number of vertices, it follows that k must be 2 or 4. Suppose $k = 2$. Then, as before, there must be some fulcrum vertex f_6 such that $\{a_1, f_6\}$ and $\{c_2, f_6\}$ are edges of $G_{c_2}(d_q)$. Again, this disrupts the $G_{a_1}(L_A)$ component, so conclude that $k = 4$ and thus $G_u(d_j) \cap G_{c_2}(d_q) \cong L_2(d_j) \cong L_4(d_q)$.

Observe then, that the color i_1 in d_j must match the color i_3 in d_p, and the color i_2 in d_j is the same as i_4 in d_q. This coincides with a tiling of dominoes in the plane, starting with a domino of type d_j in the leftmost place of the bottom row with a domino of type d_p above it and a domino of type d_q to the right of the domino of type d_j. As d_j and consequently d_p and d_q were chosen arbitrarily, it is thus shown that P is strongly solvable in the plane, as the tiling can be continued using similar techniques. $\square$

CHAPTER 6
CAYLEY GRAPHS

Chapter 4 addresses local graphs. Local graphs are closely related to two well-known graph classes addressed in this chapter: vertex-transitive graphs, and Cayley graphs. Section 6.1 reviews the relationship between vertex-transitive graphs and Cayley graphs, including a well-known theorem of Sabidussi. Section 6.2 explores Cayley graphs for cyclic groups, which form the famous class of graphs known as the circulant graphs. Section 6.3 exhibits Cayley realizations of some Ramsey graphs and includes a conjecture regarding the realizability of the $R(4,4;17)$ graph.

In other chapters, G typically denotes a graph. The reader is cautioned that in this chapter, G is used to denote groups, while Γ is used to denote graphs.

6.1 Introduction to Cayley graphs

Cayley graphs are graphs that represent group structures.

Definition 52 (Cayley graph)**.** *Let G be a group. Let S be a subset of elements of G such that the identity is not in S, S generates G, and S is closed under taking inverses. The* Cayley graph $\Gamma(G,S)$ *is the graph with vertex set*

$$V(\Gamma) = \{g : g \in G\}$$

and edge set

$$E(\Gamma) = \{\{g, gs\} : g \in G, s \in S\}.$$

Definition 52 requires that S generate G so the resulting Cayley graph is connected. The requirement that S be closed under taking inverses is so the resulting graph is also undirected. Directed Cayley graphs do not require inverse closure for S, but only undirected Cayley graphs are considered in this work.

Cayley graphs are closely related to vertex-transitive graphs, or graphs Γ such that $\text{Aut}(\Gamma)$ acts transitively on $V(\Gamma)$. Recall from Definition 14 that a *transitive action* is a group action such that for each x, y in the set X which a group G acts on, there is some group element g which sends x to y, i.e. there is some g such that $g \cdot x = y$. If this definition is further restricted to require the uniqueness of g, this corresponds to a *simply transitive action*:

Definition 53 (Simply transitive action)**.** *Let G be a group acting on a set X. The action is* simply transitive *if it is transitive and if for each $x, y \in X$ there exists a unique $g \in G$ such that $g \cdot x = y$.*

Sabidussi [61] characterizes Cayley graphs through the notion of simply transitive actions.

Theorem 18 (Sabidussi's Theorem [61])**.** *A graph Γ is a Cayley graph of a group G if and only if it admits a simply transitive action of G by graph automorphisms from Aut(Γ).*

An important corollary of Sabidussi's theorem is that every Cayley graph is a vertex-transitive graph. The Petersen graph (Figure 6.1) is a well-known example of a graph that is vertex-transitive but not a Cayley graph.

6.2 Circulant graphs

Circulant graphs are a well-studied class of graphs. Definition 16 is widely considered a typical definition of circulant graphs. Several equivalent definitions of circulant graphs exist, including a definition rooted in Cayley graphs.

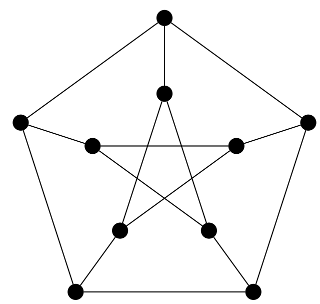


Figure 6.1: The Petersen graph

Definition 54 (Circulant graph as a Cayley graph)**.** *A graph G is a* circulant *graph if it is a Cayley graph for some cyclic group.*

Recall from Chapter 4 that a graph Γ is a realization of some graph L if, for every vertex v in $V(\Gamma)$, the subgraph induced by the open neighborhood of v in Γ is isomorphic to L. The next result concerns circulant realizations of certain graphs.

Proposition 3. *If L is a connected graph of odd order at least 3, and L is triangle-free, then L does not admit a circulant realization.*

Proof. Suppose Γ is a circulant realization of some connected L, $|L| \geq 5$. Note that since Γ is circulant, Γ is a Cayley graph for some cyclic group G with generating set S. That is, $\Gamma = \Gamma(Z, S)$, where Z is a cyclic group. We show that either L has a triangle or L has even order.

First, suppose L has odd order; show that L must then contain a triangle. Note that since Γ is regular of odd degree (as a realization of L), Γ must be of even order (since $|E(\Gamma)| = \frac{mn}{2}$). Thus $Z \cong \mathbb{Z}_{2k}, k \in \{2, 3, 4, \ldots\}$.

Since L has odd order, $|S|$ is odd. Since S is closed under taking inverses, it follows that some element of S must be its own inverse, i.e. S has an element of order 2, so $k \in S$.

Next, consider $N_\Gamma(0) = S$. Since L is connected, k has some neighbor $s \in N_\Gamma(0)$, where $-s \in S$ also. Since k is adjacent to s, it follows that $s = k + s'$

for some $s' \in S$, where $s' = s - k$. Note that $s - k \neq k$ since this would imply $s = 2k = 0$. Thus either $s - k = -s$, or $s - k$ is some other element in S. Consider each case.

1. First, suppose $s - k = -s$, i.e. $k = s + s$. It follows that

$$\begin{aligned} -s + k &= -s + (s + s) \\ &= s, \end{aligned}$$

   so $-s$ is adjacent to s. Also,

$$\begin{aligned} k + s &= s + k && \text{(Cyclic, abelian)} \\ &= s - k && \text{(since } k = -k\text{)} \\ &= s - (s + s) \\ &= -s, \end{aligned}$$

   so k is adjacent to $-s$ also. Hence there is a triangle in Γ_0.

2. Suppose $s - k$ is some other element of S. Next, $s + k = s - k$, so s is adjacent to $s - k$. Also,

$$\begin{aligned} k + s &= s + k \\ &= s - k, \end{aligned}$$

   so k is adjacent to $s - k$. Hence Γ_0 contains a triangle.

Thus if L is of odd order, then it must contain a triangle.

It remains to be shown that if L is triangle-free, then L must then be of even order. Thus, suppose L is triangle free. Note that if $|\Gamma|$ is odd, then $|L|$ must be even, since $|E(\Gamma)| = \frac{1}{2}|L||G|$. Thus consider only the case where $|\Gamma|$ is even. As previously shown, an element of order 2 in S forces a triangle in Γ_0. Since S is closed under taking inverses, and S contains no elements of order 2, $|S|$ is even so $|L|$ is indeed even. $\square$

Section 4.1 mentions the tension regarding cycles and the T-Z problem. Here is a different proof of the fact that C_6 is indeed realizable and, in particular, has infinitely many circulant realizations.

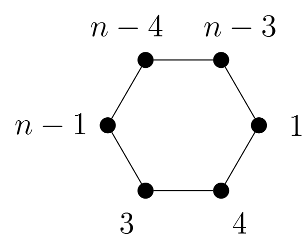
Proposition 4. *If G is a cyclic group of order at least 13, i.e. $G \cong \mathbb{Z}_n$, $n \geq 13$, and $S = \{1, -1, 3, -3, 4, -4\}$, then $\Gamma(G, S)$ is locally C_6.*

Proof. Let $\Gamma(G, S)$ be the Cayley graph for $G \cong Z_n, n \geq 13$ and

$S = \{1, -1, 3, -3, 4, -4\}$. Note the following addition table in $\mathbb{Z}_n$:

g	$g+1$	$g+3$	$g+4$	$g-4$	$g-3$	$g-1$
0	1	3	4	$n-4$	$n-3$	$n-1$
1	2	4	5	$n-3$	$n-2$	0
3	4	6	1	$n-1$	0	2
4	5	7	8	0	1	3
$-4 = n-4$	$n-3$	$n-1$	0	$n-8$	$n-7$	$n-5$
$-3 = n-3$	$n-2$	0	1	$n-7$	$n-6$	$n-4$
$-1 = n-1$	0	2	1	$n-5$	$n-4$	$n-2$

The goal is to establish $\Gamma_0 \cong C_6$, where $N_\Gamma(0) = S$. Note that from the table, $1 - 4 = n - 3$, so 1 is adjacent to $n - 3$ in Γ. By similarly using the table above, Γ_0 is as shown below:



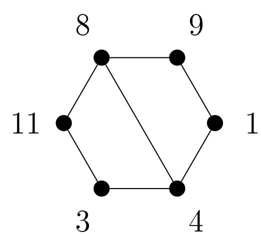
It must be verified that there are no more edges in Γ_0, i.e. that no vertex in S has any more neighbors within S. Observe that since $n \geq 13$, the table can be updated as follows:

g	$g+1$	$g+3$	$g+4$	$g-4$	$g-3$	$g-1$
0	1	3	4	$n-4 \geq 9$	$n-3 \geq 10$	$n-1 \geq 12$
1	2	4	5	$n-3 \geq 10$	$n-2 \geq 11$	0
3	4	6	1	$n-1 \geq 12$	0	2
4	5	7	8	0	1	3
$n-4 \geq 9$	$n-3 \geq 10$	$n-1 \geq 12$	0	$n-8 \geq 5$	$n-7 \geq 6$	$n-5 \geq 8$
$n-3 \geq 10$	$n-2 \geq 11$	0	1	$n-7 \geq 6$	$n-6 \geq 7$	$n-4 \geq 9$
$n-1 \geq 12$	0	2	1	$n-5 \geq 8$	$n-4 \geq 9$	$n-2 \geq 11$

So Γ_0 is precisely as specified above. □

The following example demonstrates the importance of $n \geq 13$ in Proposition 4.

Example 26. Let $\mathbb{Z}_{12}$ be the cyclic group of order 12, and let $S = \{1, -1, 3, -3, 4, -4\}$. Consider the Cayley graph $\Gamma(G, S)$. The neighborhood subgraph Γ_0 is as shown below:



Informally speaking, requiring $n \geq 13$ in Proposition 4 allows the neighborhood of $n-4$ to "clear" the neighborhood of 0 in Γ. △

6.3 Cayley realizations of Ramsey graphs

Section 4.7 identifies realizations of some Ramsey graphs. For each Ramsey graph for which at least one realization has been found, at least one of those realizations is a Cayley graph. This section provides constructions of currently known Cayley realizations of Ramsey graphs found by the tree search in Section 4.5.

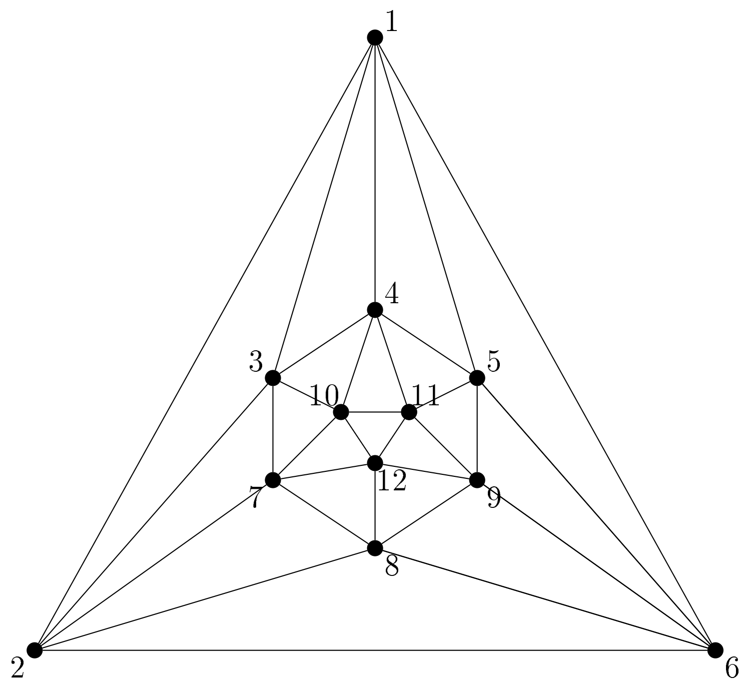


Figure 6.2: Labeled icosahedron

6.3.1 Unique Cayley realization of $R(3,3;5)$

Section 4.7.1 establishes the icosahedron as a realization of the unique $R(3,3;5)$ graph. What follows below is a construction of the icosahedron as a Cayley realization of $R(3,3;5)$ that includes many details. It seems possible that perhaps the uniqueness of the icosahedron as a realization of C_5 could be understood from a group theory perspective, though this section does not yet contain such a result.

Let Γ be the icosahedron graph, labeled as in Figure 6.2. Color the faces of this icosahedron using 5 colors in such a way that each vertex is incident with only one face of each color, as shown in Figure 6.3. Consider the faces colored yellow and label them arbitrarily as follows:

- $1 \to \{1, 2, 6\}$
- $2 \to \{4, 5, 11\}$

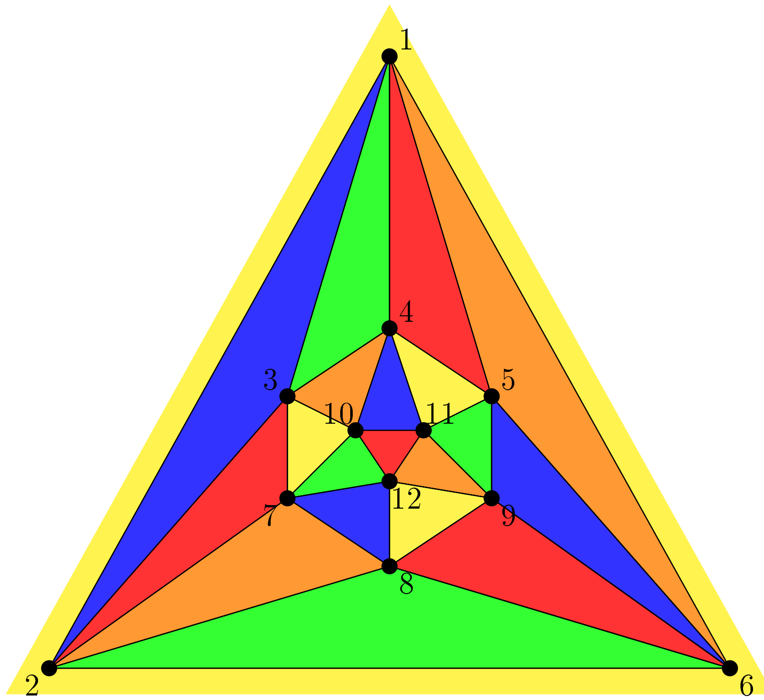


Figure 6.3: Icosahedron with faces colored

- $3 \to \{3, 7, 10\}$
- $4 \to \{8, 9, 12\}$

The goal is to form a subgroup of $\mathrm{Aut}(\Gamma)$ consisting of 12 automorphisms, sending vertex 1 to each of $1, 2, \ldots, 12$. Such a subgroup satisfies Sabidussi's Theorem (Theorem 18). For the element sending vertex 1 to vertex 1, select the identity element of $\mathrm{Aut}(\Gamma)$. Next, consider a rotation of each yellow face and the resulting automorphisms:

Rotate Face 1 (i.e. "Fix 1")	(1 2 6)(3 8 5)(4 7 9)(10 12 11)
	(1 6 2)(3 5 8)(4 9 7)(10 11 12)
Rotate Face 2 (i.e. "Fix 2")	(1 9 10)(2 8 7)(3 6 12)(4 5 11)
	(1 10 9)(2 7 8)(3 12 6)(4 11 5)
Rotate Face 3 (i.e. "Fix 3")	(1 11 8)(2 4 12)(3 10 7)(5 9 6)
	(1 8 11)(2 12 4)(3 7 10)(5 6 9)
Rotate Face 4 (i.e. "Fix 4")	(1 3 4)(2 10 5)(6 7 11)(8 12 9)
	(1 4 3)(2 5 10)(6 11 7)(8 9 12)

Now, consider ways to swap Face 1 with each of the other yellow faces, i.e. swap Face 1 with each of Faces 2, 3, and 4. Since automorphisms mapping vertex 1 to vertices 5, 7, or 12 have yet to be selected, choose the following:

Swap Faces 1 & 2	(1 5)(2 11)(3 9)(4 6)(7 12)(8 10)	(Note: Also swaps Faces 3 & 4)
Swap Faces 1 & 3	(1 7)(2 3)(4 8)(5 12)(6 10)(9 11)	(Note: Also swaps Faces 2 & 4)
Swap Faces 1 & 4	(1 12)(2 9)(3 11)(4 10)(5 7)(6 8)	(Note: Also swaps Faces 2 & 3)

The 12 automorphisms selected thus far do indeed form a subgroup of $\mathrm{Aut}(\Gamma)$. More specifically, this subgroup is isomorphic to A_4, as shown next. To specify the bijection $\varphi : A_4 \to \mathrm{Aut}(\Gamma)$, begin by assigning

$$\varphi((2\ 3\ 4)) = (1\ 2\ 6)(3\ 8\ 5)(4\ 7\ 9)(10\ 12\ 11),$$

since this automorphism was previously described as an element that "fixes" Face 1. Next, its inverse is

$$\varphi((2\ 4\ 3)) = (1\ 6\ 2)(3\ 5\ 8)(4\ 9\ 7)(10\ 11\ 12).$$

By swapping different yellow faces, it also naturally follows that

$$\varphi((1\ 2)(3\ 4)) = (1\ 5)(2\ 11)(3\ 9)(4\ 6)(7\ 12)(8\ 10)$$

$$\varphi((1\ 3)(2\ 4)) = (1\ 7)(2\ 3)(4\ 8)(5\ 12)(6\ 10)(9\ 11)$$

$$\varphi((1\ 4)(2\ 3)) = (1\ 12)(2\ 9)(3\ 11)(4\ 10)(5\ 7)(6\ 8)$$

To make remaining assignments, consider that in A_4,

$$(2\ 3\ 4)(1\ 2)(3\ 4) = (1\ 2\ 4)$$

$$(2\ 3\ 4)(1\ 3)(2\ 4) = (1\ 3\ 2)$$

$$(2\ 3\ 4)(1\ 4)(2\ 3) = (1\ 4\ 3)$$

Thus the bijection is fully specified as follows:

$g \in A_4$	$\varphi(x) \in \mathrm{Aut}(\Gamma)$
(1)	e
(2 3 4)	(1 2 6)(3 8 5)(4 7 9)(10 12 11)
(2 4 3)	(1 6 2)(3 5 8)(4 9 7)(10 11 12)
(1 4 3)	(1 9 10)(2 8 7)(3 6 12)(4 5 11)
(1 3 4)	(1 10 9)(2 7 8)(3 12 6)(4 11 5)
(1 2 4)	(1 11 8)(2 4 12)(3 10 7)(5 9 6)
(1 4 2)	(1 8 11)(2 12 4)(3 7 10)(5 6 9)
(1 3 2)	(1 3 4)(2 10 5)(6 7 11)(8 12 9)
(1 2 3)	(1 4 3)(2 5 10)(6 11 7)(8 9 12)
(1 2)(3 4)	(1 5)(2 11)(3 9)(4 6)(7 12)(8 10)
(1 3)(2 4)	(1 7)(2 3)(4 8)(5 12)(6 10)(9 11)
(1 4)(2 3)	(1 12)(2 9)(3 11)(4 10)(5 7)(6 8)

Next, return to the original labeling of Γ presented in Figure 6.2. Arbitrarily relabel vertex 1 using $(1) \in A_4$. Next, label remaining vertices according to the element $g \in A_4$ such that $\varphi(g)$ yields a permutation sending 1 that neighbor; for example, vertex 8 is relabeled (1 4 2) since $\varphi(1\ 4\ 2) = (1\ 8\ 11)(2\ 12\ 4)(3\ 7\ 10)(5\ 6\ 9)$. The resulting relabeling of Γ is shown in Figure 6.4. The set S consists of the elements that send vertex 1 to its neighbors:

$$S = \{(2\ 3\ 4), (1\ 3\ 2), (1\ 2\ 3), (1\ 2)(3\ 4), (2\ 4\ 3)\}.$$

Hence the icosahedron graph is isomorphic to the Cayley graph

$$\Gamma(A_4, \{(2\ 3\ 4), (1\ 3\ 2), (1\ 2\ 3), (1\ 2)(3\ 4), (2\ 4\ 3)\}).$$

The automorphism group of the icosahedron is $A_5 \times \mathbb{Z}_2$, a group of order 120. The alternating group A_4 is a subgroup of this group.

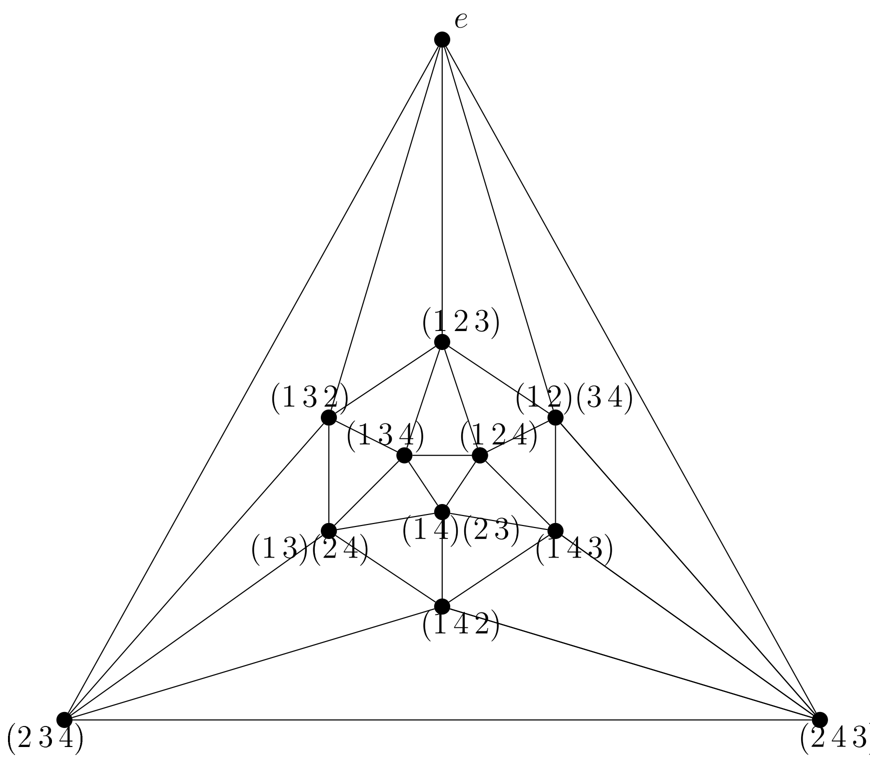


Figure 6.4: Icosahedron as a Cayley graph

6.3.2 Two Cayley realizations of H_3

Realizations of the critical Ramsey(3, 4) graph H_3 (Figure 2.1) are established in Section 4.7.2. The first known realization of H_3 is the well-known $R(4,4;17)$ graph. The second realization is a 21-vertex graph. Both of these realizations of H_3 are Cayley graphs, as shown next.

The first realization: $R(4,4;17)$

The $R(4,4;17)$ graph (Figure 2.2) realizes H_3 (Figure 2.1). The $R(4,4;17)$ graph is circulant and is therefore a Cayley graph for some cyclic group. More specifically, $R(4,4;17) \cong \Gamma(\mathbb{Z}_{17}, \{1,2,4,8,9,13,15,16\})$.

The automorphism group of the $R(4,4;17)$ graph is $\mathbb{Z}_{17} \rtimes \mathbb{Z}_8$, which contains $\mathbb{Z}_{17}$ as a subgroup.

The second realization

Let G be the group of order 21 with the following presentation:

$$G = \langle a, b : a^3 = b^7 = 1, aba^{-1} = b^4 \rangle.$$

Let

$$S = \{a, a^2, b, b^6, ab, ab^2, a^2b^3, a^2b^6\}.$$

The set S is of order 8, does not contain the identity element, and is closed under taking inverses:

$$a \cdot a^2 = e,$$

$$b \cdot b^6 = e,$$

$$\begin{aligned} ab \cdot a^2b^3 &= b^4 \cdot b^3 \\ &= e, \end{aligned}$$

$$
\begin{aligned}
ab^2 \cdot a^2b^6 &= ab^2a \cdot ab \cdot b^5 \\
&= ab^2a \cdot b^4a \cdot b^5 \\
&= ab^2 \cdot ab \cdot b^3 \cdot ab \cdot b^4 \\
&= ab^2 \cdot b^4a \cdot b^3 \cdot b^4a \cdot b^4 \\
&= ab^6a^2b^4 \\
&= ab^6a \cdot ab \cdot b^3 \\
&= ab^6a \cdot b^4a \cdot b^3 \\
&= ab^6 \cdot ab \cdot b^3 \cdot ab \cdot b^2 \\
&= ab^6 \cdot b^4a \cdot b^3 \cdot b^4a \cdot b^2 \\
&= ab^3a^2b^2 \\
&= ab^3a \cdot ab \cdot b \\
&= ab^3a \cdot b^4a \cdot b \\
&= ab^3 \cdot ab \cdot b^3 \cdot ab \\
&= ab^3 \cdot b^4a \cdot b^3 \cdot b^4a \\
&= a \cdot a \cdot a \\
&= e.
\end{aligned}
$$

Table 6.1 shows a multiplication table for S, where an entry of "-" denotes a product that is not in S. Consider the Cayley graph $\Gamma(G, S)$. This is an 8-regular graph of order 21. The neighborhood of the identity element in $\Gamma(G, S)$ is shown in Figure 6.5. Since $\Gamma_0 \cong H_3$, it follows that Γ is locally H_3. This group G is isomorphic to $\mathbb{Z}_7 \rtimes \mathbb{Z}_3$, which is a subgroup of $PGL_2(\mathbb{F}_7) \cong \mathrm{Aut}(\Gamma)$ as noted in Table 4.5.

	a	a^2	b	b^6	ab	ab^2	a^2b^3	a^2b^6
a	a^2	$-$	ab	$-$	$-$	$-$	$-$	b^6
a^2	$-$	a	$-$	a^2b^6	b	$-$	$-$	$-$
b	ab^2	$-$	$-$	$-$	$-$	$-$	a^2	a^2b^3
b^6	$-$	a^2b^3	$-$	$-$	$-$	a	a^2b^6	$-$
ab	$-$	$-$	ab^2	a	a^2b^3	$-$	$-$	$-$
ab^2	$-$	b	$-$	ab	$-$	a^2b^6	$-$	$-$
a^2b^3	b^6	$-$	$-$	$-$	$-$	b	ab	$-$
a^2b^6	$-$	$-$	a^2	$-$	b^6	$-$	$-$	ab^2

Table 6.1: Multiplication table for S

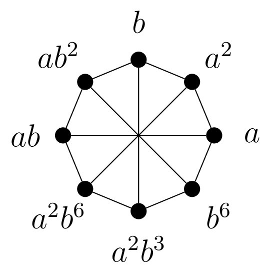


Figure 6.5: $\Gamma_0 \cong H_3$

6.3.3 Two Cayley realizations of H_2

As described in Section 4.7.3, the critical Ramsey(3, 4) graph H_2 (Figure 2.1) has three realizations, two of which are Cayley graphs. The Cayley graph constructions are presented in this section.

The first realization

Let G be the group of order 21 with the following presentation:

$$G = \langle a, b : a^3 = b^7 = 1, aba^{-1} = b^4 \rangle.$$

Let

$$S = \{a, a^2, b^2, b^3, b^4, b^5, ab^3, a^2b^2\}.$$

Note that S does not contain the identity, has order 8, and is closed under taking inverses:

$$a \cdot a^2 = b^2 \cdot b^5 = b^3 \cdot b^4 = ab^3 \cdot a^2b^2 = e,$$

where the last equality follows from

$$\begin{aligned} ab^3 \cdot a^2b^2 &= ab^3 \cdot a \cdot ab \cdot b \\ &= ab^3a \cdot b^4a \cdot b \quad (ab = b^4a) \\ &= ab^3a \cdot b^4 \cdot ab \\ &= ab^3ab^4 \cdot b^4a \\ &= ab^3 \cdot ab \cdot a \\ &= ab^3 \cdot b^4a \cdot a \\ &= a^3 \\ &= e. \end{aligned}$$

A multiplication table for S is given in Table 6.2, where an entry of "$-$" indicates a product that is not in S.

The neighborhood of the identity element in $\Gamma(G, S)$ is thus as shown in Figure 6.6. This graph is isomorphic to H_2.

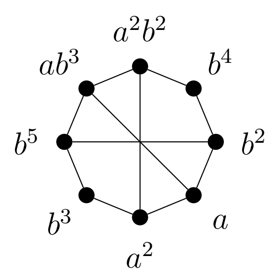


Figure 6.6: $\Gamma_0 \cong H_2$

	a	a^2	b^2	b^3	b^4	b^5	ab^3	a^2b^2
a	a^2	–	–	ab^3	–	–	–	b^2
a^2	–	a	a^2b^2	–	–	–	b^3	–
b^2	–	–	b^4	b^5	–	–	a	–
b^3	–	–	b^5	–	–	–	–	a^2
b^4	–	a^2b^2	–	–	–	–	ab^4	a^2b^4
b^5	ab^3	–	–	b^2	b^3	–	–	
ab^3	–	b^5	–	–	a	–	a^2b^2	–
a^2b^2	b^4	–	–	–	–	a^2	–	ab^3

Table 6.2: Multiplication table for S

This group G is isomorphic to $\mathbb{Z}_7 \rtimes \mathbb{Z}_3$, which is a subgroup of $\mathbb{Z}_7 \rtimes \mathbb{Z}_6 \cong \mathrm{Aut}(\Gamma)$ as noted in Table 4.5.

The second realization

Let G be the group of order 24 which has the following presentation:

$$G = \langle a, b, c, d : a^2 = b^2 = c^2 = d^3 = 1, ab = ba, ac = ca, ad = da, dbd^{-1} = bc = cb, dcd^{-1} = b\rangle.$$

Let

$$S = \{d, d^2, ab, bc, bd^2, abd^2, bcd, abcd\}.$$

Note that S is closed under taking inverses:

$$d \cdot d^2 = e,$$

$$\begin{aligned} ab \cdot ab &= ab \cdot ba \\ &= a \cdot a \\ &= e, \end{aligned}$$

$$\begin{aligned} bc \cdot bc &= bc \cdot cb \\ &= b \cdot b \\ &= e, \end{aligned}$$

$$\begin{aligned} bd^2 \cdot bcd &= d^2bc \cdot bcd \\ &= d^2 \cdot d \\ &= e, \end{aligned}$$

$$\begin{aligned} abd^2 \cdot abcd &= abd^2a \cdot bc \cdot d \\ &= abd^2a \cdot dbd^2 \cdot d \\ &= abd^2 \cdot ad \cdot b \\ &= abd^2 \cdot da \cdot b \\ &= ab \cdot ab \\ &= ab \cdot ba \\ &= a \cdot a \\ &= e. \end{aligned}$$

A multiplication table for S is given in Table 6.3, where an entry of "$-$" indicates a product that is not in S.

The neighborhood of the identity element in $\Gamma(G, S)$ is thus as shown in Figure 6.7. This graph is isomorphic to H_2.

	d	d^2	ab	bc	bd^2	abd^2	bcd	$abcd$
d	d^2	$-$	$abcd$	$-$	bc	$-$	$-$	$-$
d^2	$-$	d	$-$	bd^2	$-$	$-$	$-$	ab
ab	$-$	abd^2	$-$	$-$	$-$	d^2	$-$	$-$
bc	bcd	$-$	$-$	$-$	$-$	$-$	d	$-$
bd^2	$-$	$-$	$-$	d^2	bcd	$abcd$	$-$	$-$
abd^2	ab	$-$	$-$	$-$	$abcd$	bcd	$-$	$-$
bcd	$-$	bc	$-$	$-$	$-$	$-$	bd^2	abd^2
$abcd$	$-$	$-$	d	$-$	$-$	$-$	abd^2	bd^2

Table 6.3: Multiplication table for S

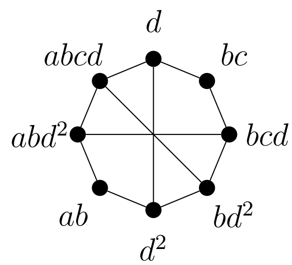


Figure 6.7: $\Gamma_0 \cong H_2$

The group G is isomorphic to $\mathbb{Z}_2 \times A_4$, which is itself the automorphism group of Γ, as noted in Table 4.5.

6.3.4 Realizability of $R(4,4;17)$

The Cayley graph constructions so far in this section lead to a natural conjecture regarding the $R(4,4;17)$ graph.

Conjecture 3. *The $R(4,4;17)$ graph is realizable and, in particular, is f-realizable by some Cayley graph.*

CHAPTER 7
CONCLUSIONS

Chapter 3 includes recommendations for continuing the reinforcement learning project for Ramsey graph construction. In those recommendations, there is a focus on making the Ramsey game work, and the game is centered solely around graphs as the object to work with. Given the content of later chapters, it seems reasonable to instead consider neighborhoods or groups.

This interest in groups is inspired by the Cayley realizations of two $R(3,4;8)$ graphs in Chapter 6. The same group ($\mathbb{Z}_7 \rtimes \mathbb{Z}_3$) with different generating sets leads to Cayley realizations of the critical Ramsey(3, 4) graphs, H_2 and H_3 (Figure 2.1). In retrospect, it seems that perhaps this group should have given enough information from the start to construct realizations. All constructions presented in Chapter 6 are derived from already knowing the realization and its automorphism group, but it seems that having a particular subgroup (the group for the Cayley graph) should be enough. A result similar to Proposition 4 seems within reach for $R(3,4;8)$ graphs. Such a result might also help determine whether or not realizations of the remaining critical Ramsey(3, 4) graph H_1 (Figure 2.1) exist.

If groups do not give enough information, perhaps they might still be used to boost the realization construction program in some way. This is of particular interest in attempting to construct realizations of the $R(4,4;17)$ graph.

Connections might also be drawn between Ramsey graphs and Cayley graphs by considering the following conjecture of Alon:

Conjecture 4 ([2]). *There is a constant c such that, for every finite group G of order $n > 1$, there is an inverse-closed generating set S for G such that the Cayley graph $\Gamma(G, S)$ has neither a clique nor an independent set of order $c \log n$.*

REFERENCES

[1] *Theory of graphs and its applications*, Publishing House of the Czechoslovak Academy of Sciences, Prague, 1964. MR 0172259

[2] *Research problems*, Discrete Mathematics **138** (1995), no. 1, 405–411, 14th British Combinatorial Conference.

[3] S. Ja. Agakišieva, *Graphs with prescribed vertex environments*, Mat. Zametki **3** (1968), 211–216. MR 236043

[4] Vigleik Angeltveit and Brendan D. McKay, $R(5,5) \leq 48$, J. Graph Theory **89** (2018), no. 1, 5–13. MR 3828124

[5] Egon Balas and Chang Sung Yu, *On graphs with polynomially solvable maximum-weight clique problem*, Networks **19** (1989), no. 2, 247–253. MR 984569

[6] Markus Baumeister and Anna M. Limbach, *Clique dynamics of locally cyclic graphs with* $\delta \geq 6$, Discrete Math. **345** (2022), no. 7, Paper No. 112873, 23. MR 4394715

[7] Itai Benjamini and Tom Hutchcroft, *Large, lengthy graphs look locally like lines*, Bull. Lond. Math. Soc. **53** (2021), no. 2, 482–492. MR 4239190

[8] Andreas Blass, Frank Harary, and Zevi Miller, *Which trees are link graphs?*, J. Combin. Theory Ser. B **29** (1980), no. 3, 277–292. MR 602420

[9] Morton Brown and Robert Connelly, *On graphs with a constant link*, New directions in the theory of graphs (Proc. Third Ann Arbor Conf., Univ. Michigan, Ann Arbor, Mich., 1971), 1973, pp. 19–51. MR 0347685

[10] ———, *On graphs with a constant link. II*, Discrete Math. **11** (1975), 199–232. MR 364016

[11] Francis Buekenhout and Xavier Hubaut, *Locally polar spaces and related rank 3 groups*, J. Algebra **45** (1977), no. 2, 391–434. MR 460155

[12] V. K. Bulitko, *Graphs with prescribed environments of the vertices*, Trudy Mat. Inst. Steklov. **133** (1973), 78–94, 274, Mathematical logic, theory of algorithms and theory of sets (dedicated to P. S. Novikov on the occasion of his seventieth birthday). MR 0434882

[13] Gary Chartrand and Raymond E. Pippert, *Locally connected graphs*, Časopis Pěst. Mat. **99** (1974), 158–163. MR 0398872

[14] Gary Chartrand and Ping Zhang, *Ramsey sequences of graphs*, AKCE Int. J. Graphs Comb. **17** (2020), no. 2, 646–652. MR 4169782

[15] Bruce L. Chilton, Ronald Gould, and Albert D. Polimeni, *A note on graphs whose neighborhoods are n-cycles*, Geometriae Dedicata **3** (1974), 289–294. MR 357220

[16] François Chollet, *Deep learning with Python*, Manning Publications Co, Shelter Island, New York, 2018 (en), OCLC: ocn982650571.

[17] Václav Chvátal and Frank Harary, *Generalized Ramsey theory for graphs. II. Small diagonal numbers*, Proc. Amer. Math. Soc. **32** (1972), 389–394. MR 332559

[18] L. H. Clark, R. C. Entringer, J. E. McCanna, and L. A. Székely, *Extremal problems for local properties of graphs*, vol. 4, 1991, Combinatorial mathematics and combinatorial computing (Palmerston North, 1990), pp. 25–31. MR 1129266

[19] P. Erdős and C. A. Rogers, *The construction of certain graphs*, Canadian J. Math. **14** (1962), 702–707. MR 141612

[20] P. Erdös, *Some remarks on the theory of graphs*, Bull. Amer. Math. Soc. **53** (1947), 292–294. MR 19911

[21] P. Erdös and G. Szekeres, *A combinatorial problem in geometry*, Compositio Math. **2** (1935), 463–470. MR 1556929

[22] Geoffrey Exoo, *A lower bound for* $R(5,5)$, J. Graph Theory **13** (1989), no. 1, 97–98. MR 982871

[23] Dalibor Fronček, *Locally linear graphs*, Math. Slovaca **39** (1989), no. 1, 3–6. MR 1016323

[24] Dennis P. Geoffroy and David P. Sumner, *An upper bound on the size of a largest clique in a graph*, J. Graph Theory **2** (1978), no. 3, 223–230. MR 505816

[25] A. W. Goodman, *On sets of acquaintances and strangers at any party*, Amer. Math. Monthly **66** (1959), 778–783. MR 107610

[26] Ronald L. Graham, Bruce L. Rothschild, and Joel H. Spencer, *Ramsey theory*, second ed., Wiley-Interscience Series in Discrete Mathematics and Optimization, John Wiley & Sons, Inc., New York, 1990, A Wiley-Interscience Publication. MR 1044995

[27] Jack E. Graver and James Yackel, *An upper bound for Ramsey numbers*, Bull. Amer. Math. Soc. **72** (1966), 1076–1079. MR 199120

[28] ———, *Some graph theoretic results associated with Ramsey's theorem*, J. Combinatorial Theory **4** (1968), 125–175. MR 225685

[29] R. E. Greenwood and A. M. Gleason, *Combinatorial relations and chromatic graphs*, Canadian J. Math. **7** (1955), 1–7. MR 67467

[30] J. I. Hall, *Locally Petersen graphs*, J. Graph Theory **4** (1980), no. 2, 173–187. MR 570352

[31] ———, *Graphs with constant link and small degree or order*, J. Graph Theory **9** (1985), no. 3, 419–444. MR 812408

[32] J. I. Hall and E. E. Shult, *Locally cotriangular graphs*, Geom. Dedicata **18** (1985), no. 2, 113–159. MR 792576

[33] Frank Harary and Edgar M. Palmer, *Graphical enumeration*, Academic Press, New York-London, 1973. MR 0357214

[34] Heiko Harborth and Stefan Krause, *Ramsey numbers for circulant colorings*, Proceedings of the Thirty-Fourth Southeastern International Conference on Combinatorics, Graph Theory and Computing, vol. 161, 2003, pp. 139–150. MR 2050525

[35] ———, *Distance Ramsey numbers*, Util. Math. **70** (2006), 197–200. MR 2238441

[36] Pavol Hell, *Graphs with given neighborhoods. I*, Problèmes combinatoires et théorie des graphes (Colloq. Internat. CNRS, Univ. Orsay, Orsay, 1976), Colloq. Internat. CNRS, vol. 260, CNRS, Paris, 1978, pp. 219–223. MR 539979

[37] Jiayi Huang, Mostofa Patwary, and Gregory Diamos, *Coloring big graphs with alphagozero*, 2019.

[38] A. S. Kahr, Edward F. Moore, and Hao Wang, *Entscheidungsproblem reduced to the* $\forall\exists\forall$ *case*, Proc. Nat. Acad. Sci. U.S.A. **48** (1962), 365–377. MR 169777

[39] J. G. Kalbfleisch, *Construction of special edge-chromatic graphs*, Canad. Math. Bull. **8** (1965), 575–584. MR 193026

[40] ______, *On an unknown Ramsey number*, Michigan Math. J. **13** (1966), 385–392. MR 201343

[41] ______, *Upper bounds for some Ramsey numbers*, J. Combinatorial Theory **2** (1967), 35–42. MR 211919

[42] F. Larrión, M. A. Pizaña, and R. Villarroel-Flores, *Small locally* nK_2 *graphs*, Ars Combin. **102** (2011), 385–391. MR 2867738

[43] R. C. Laskar and Henry Martyn Mulder, *Path-neighborhood graphs*, Discuss. Math. Graph Theory **33** (2013), no. 4, 731–745. MR 3117052

[44] R. C. Laskar, Henry Martyn Mulder, and B. Novick, *Maximal outerplanar graphs as chordal graphs, path-neighborhood graphs, and triangle graphs*, Australas. J. Combin. **52** (2012), 185–195. MR 2917926

[45] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, *Deep learning*, Nature **521** (2015), no. 7553, 436–444 (en).

[46] Bernard Lidický and Florian Pfender, *Semidefinite programming and Ramsey numbers*, SIAM J. Discrete Math. **35** (2021), no. 4, 2328–2344. MR 4321243

[47] A. Márquez, A. de Mier, M. Noy, and M. P. Revuelta, *Locally grid graphs: classification and Tutte uniqueness*, vol. 266, 2003, The 18th British Combinatorial Conference (Brighton, 2001), pp. 327–352. MR 1991727

[48] Brendan McKay, *Combinatorial data*.

[49] Brendan D. McKay, *Transitive graphs with fewer than twenty vertices*, Math. Comp. **33** (1979), no. 147, 1101–1121, loose microfiche suppl. MR 528064

[50] Brendan D. McKay and Zhang Ke Min, *The value of the Ramsey number $R(3,8)$*, J. Graph Theory **16** (1992), no. 1, 99–105. MR 1147807

[51] Brendan D. McKay and Stanisław P. Radziszowski, *A new upper bound for the Ramsey number $R(5,5)$*, Australas. J. Combin. **5** (1992), 13–20. MR 1165791

[52] ———, *Linear programming in some Ramsey problems*, J. Combin. Theory Ser. B **61** (1994), no. 1, 125–132. MR 1275272

[53] ———, $R(4,5) = 25$, J. Graph Theory **19** (1995), no. 3, 309–322. MR 1324481

[54] ———, *Subgraph counting identities and Ramsey numbers*, J. Combin. Theory Ser. B **69** (1997), no. 2, 193–209. MR 1438619

[55] T. D. Parsons and Tomaž Pisanski, *Graphs which are locally paths*, Combinatorics and graph theory (Warsaw, 1987), Banach Center Publ., vol. 25, PWN, Warsaw, 1989, pp. 127–135. MR 1097642

[56] Erich Prisner, *Graphs with few cliques*, Graph theory, combinatorics, and algorithms, Vol. 1, 2 (Kalamazoo, MI, 1992), Wiley-Intersci. Publ., Wiley, New York, 1995, pp. 945–956. MR 1405872

[57] Max Pumperla and Kevin Ferguson, *Deep learning and the game of Go*, Manning, Shelter Island, 2019 (en), OCLC: on1033778541.

[58] Stanisław P. Radziszowski, *Small Ramsey numbers*, Electron. J. Combin. **1** (1994), Dynamic Survey 1, 30. MR 1670625

[59] F. P. Ramsey, *On a Problem of Formal Logic*, Proc. London Math. Soc. (2) **30** (1929), no. 4, 264–286. MR 1576401

[60] Mark A. Ronan, *On the second homotopy group of certain simplicial complexes and some combinatorial applications*, Quart. J. Math. Oxford Ser. (2) **32** (1981), no. 126, 225–233. MR 615196

[61] Gert Sabidussi, *On a class of fixed-point-free graphs*, Proc. Amer. Math. Soc. **9** (1958), 800–804. MR 97068

[62] J. Sedláček, *On local properties of finite graphs*, Graph theory (Łagów, 1981), Lecture Notes in Math., vol. 1018, Springer, Berlin, 1983, pp. 242–247. MR 730654

[63] Jiří Sedláček, *Local properties of graphs*, Časopis Pěst. Mat. **106** (1981), no. 3, 290–298. MR 629727

[64] ———, *On local properties of graphs again*, Časopis Pěst. Mat. **114** (1989), no. 4, 381–390. MR 1027234

[65] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis, *Mastering the game of Go with deep neural networks and tree search*, Nature **529** (2016), no. 7587, 484–489 (en).

[66] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis, *A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play*, Science **362** (2018), no. 6419, 1140–1144 (en).

[67] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis, *Mastering the game of Go without human knowledge*, Nature **550** (2017), no. 7676, 354–359 (en).

[68] Z. Skupień, *Locally Hamiltonian graphs and Kuratowski theorem*, Bull. Acad. Polon. Sci. Sér. Sci. Math. Astronom. Phys. **13** (1965), 615–619. MR 193029

[69] ______, *Locally Hamiltonian and planar graphs*, Fund. Math. **58** (1966), 193–200. MR 195764

[70] ______, *On the locally Hamiltonian graphs and Kuratowski's theorem*, Prace Mat. **11** (1968), 255–264. MR 0227045

[71] Joel Spencer, *Ramsey's theorem—a new lower bound*, J. Combinatorial Theory Ser. A **18** (1975), 108–115. MR 366726

[72] L. Szamkoł owicz, *On a classification of graphs with respect to the properties of neighbourhood graphs*, Finite and infinite sets, Vol. I, II (Eger, 1981), Colloq. Math. Soc. János Bolyai, vol. 37, North-Holland, Amsterdam, 1984, pp. 675–678. MR 818269

[73] Lucjan Szamkoł owicz, *A note on a generalization of the Trachtenbrot-Zykov problem*, Graph theory (Łagów, 1981), Lecture Notes in Math., vol. 1018, Springer, Berlin, 1983, pp. 257–259. MR 730656

[74] Walter Vogler, *Graphs with given group and given constant link*, J. Graph Theory **8** (1984), no. 1, 111–115. MR 732024

[75] Adam Zsolt Wagner, *Constructions in combinatorics via neural networks*, 2021.

[76] K. Walker, *Dichromatic graphs and Ramsey numbers*, J. Combinatorial Theory **5** (1968), 238–243. MR 231751

[77] ———, *An upper bound for the Ramsey number $M(5, 4)$*, J. Combinatorial Theory Ser. A **11** (1971), 1–10. MR 274340

[78] Peter M. Winkler, *Existence of graphs with a given set of r-neighborhoods*, J. Combin. Theory Ser. B **34** (1983), no. 2, 165–176. MR 703601

[79] Huijuan Yu and Baoyindureng Wu, *Graphs in which $G - N[v]$ is a cycle for each vertex v*, Discrete Math. **344** (2021), no. 9, Paper No. 112519, 7. MR 4278078

[80] Bohdan Zelinka, *Edge neighbourhood graphs*, Czechoslovak Math. J. **36(111)** (1986), no. 1, 44–47. MR 822865

[81] ———, *Locally snake-like graphs*, Math. Slovaca **38** (1988), no. 1, 85–88. MR 945083

[82] ———, *Small directed graphs as neighbourhood graphs*, Czechoslovak Math. J. **38(113)** (1988), no. 2, 269–273. MR 946295

[83] ———, *Two local properties of graphs*, Časopis Pěst. Mat. **113** (1988), no. 2, 113–121. MR 949039

[84] ———, *The least connected non-vertex-transitive graph with constant neighbourhoods*, Czechoslovak Math. J. **40(115)** (1990), no. 4, 619–624. MR 1084898

[85] ———, *Locally regular graphs*, Math. Bohem. **125** (2000), no. 4, 481–484. MR 1802296

[86] Bo Zhang and Baoyindureng Wu, *Graphs G in which $G - N[v]$ has a prescribed property for each vertex v*, Discrete Appl. Math. **318** (2022), 13–20. MR 4432999

[87] A. A. Zykov, *Graph-theoretical results of Novsibirsk mathematicians*, Theory of Graphs and its Applications (Proc. Sympos. Smolenice, 1963), Publ. House Czech. Acad. Sci., Prague, 1963, pp. 151–153. MR 0172277

APPENDIX I
PROGRAMMING TOOLS

The accompanying code for this dissertation is publicly available at `https://github.com/ehawb/diss`.

I.1 Python

Python was chosen because of the vast number of resources available for it. The book our Ramsey graph bot is based on, *Deep Learning and the Game of Go*, is coded in Python. There is also a Python interface for Gurobi, the optimizer used in our linear programming subgraph checker.

I.1.1 Keras

Keras is a Python library for artificial neural networks. According to its website (`https://keras.io/`), Keras is one of the most widely used machine learning frameworks. Keras is designed to be easy to learn and use.

I.1.2 Gurobi

Gurobi is a powerful mathematical optimization solver. Free academic licenses are available, as well as other types of licenses. Gurobi has a variety of interfaces available, though we only use the Python `gurobipy` library. Gurobi's website (`https://www.gurobi.com/` has several resources available for those interested in getting started with Gurobi.

I.2 GAP

GAP (Group, Algorithms, Processing) is a system for computational discrete algebra. GAP has large data libraries of algebraic objects, including the groups addressed in this dissertation. GAP is freely available at `gap-system.org`.

GRAPE (GRaph Algorithms using PErmutation groups) is a GAP package for computing with graphs and groups. GRAPE is an interface to the well-known `nauty` (No AUTomorphisms, Yes?) package developed by Brendan McKay.

APPENDIX II
SPECIFICATION OF CERTAIN GRAPHS

This section includes information for graphs presented in this dissertation. Each graph is specified using a graph6 code and an adjacency matrix. The graph6 codes are also on GitHub (Appendix I).

II.1 Ramsey graphs

- $R(3, 3; 5)$ graph
  - graph6: `Dhc`
  - Adjacency matrix:

    ```
    01001
    10100
    01010
    00101
    10010
    ```

- H_1 (Figure 2.1)
  - graph6: `G@hZCc`
  - Adjacency matrix:

    ```
    00001001
    00000110
    00011010
    00100101
    10100100
    01011000
    01100001
    10010010
    ```

- H_2 (Figure 2.1)

- graph6:
- Adjacency matrix:

```
01001000
10000001
00010100
00100010
10000110
00101001
00011001
01000110
```

- H_3 (Figure 2.1)
  - graph6: `G`_gqK`
  - Adjacency matrix:

```
01110000
10001010
10000101
10000011
01000101
00101010
01010100
00111000
```

II.2 Realizations of Ramsey graphs

II.2.1 $R(3,3;5)$ realization

(Icosahedral graph)

- graph6: `KhFKFCrEk[n_`
- Adjacency matrix:

```
010001110010
101001010001
010100011001
001010001101
000101000111
110010000011
100000011110
111000101000
```

```
001100110100
000110101010
100011100100
011111000000
```

II.2.2 H_2 realizations

1. Realization 1

   - graph6: `T@hZCf~KDOkPIcRBP_QghDSqPKoEN]Cdb@XH`
   - Adjacency matrix:

   ```
   000010011010001001011
   000001101001010100011
   000110101111100000000
   001001011100000111000
   101001001010111000000
   010110001000011110000
   011000011001100000110
   100100101000000011101
   111111110000000000000
   001100000010100110011
   101010000101010001001
   011000100010010101100
   001010100100001010110
   010011000011000010101
   100011000000100101110
   010101000101001001010
   000101010100110000101
   100100010011001100100
   000000110001111011000
   110000100100101100001
   110000010110010010010
   ```

2. Realization 2

   - graph6: `W@hZCf~N@_CRiACSA`KOaR?hCSSEBe?TU@BSOpBICWm?@|E`
   - Adjacency matrix:

   ```
   000010011000100100111000
   000001101000000011010110
   000110101111100000000000
   001001011110011000000000
   101001001100100110000000
   010110001100001011000000
   011000011001000000100110
   ```

```
100100101000010000101100
111111110000000000000000
001111000001010101000000
001100000001101000010011
001000100110000001000111
101010000010000010011001
000100010100001100101001
000101000010010010110001
100010000100010001101010
010011000000101000010101
010001000101000100001110
100000110000011100010010
110000000010101010100010
100000010000110101000101
010000011000100001100100 1
010000010001100010111000 0
000000000011111010001100
```

3. Realization 3

    - graph6: W@hZCf~N@_CRiACSA`KOaR?hCSSABm?TUBBOOr@qCGx?@|D
    - Adjacency matrix:

```
000010011000100100111000
000001101000000011010110
000110101111100000000000
001001011110011000000000
101001001100100110000000
010110001100001011000000
011000011001000000001110
100100101000010000101100
111111110000000000000000
001111000001010101000000
001100000001101000010101
001000100110000001000111
101010000010000010011001
000100010100001100101001
000101000010010010110001
100010000100010001101010
010011000000101000010011
010001000101000100100110
100000010000011101010100
110000000010101010100100
100000110000110100000011
010000011001100000111000 0
```

```
0100001000010001110 01001
000000000011111010001010
```

II.2.3 H_3 realizations

1. Realization 1

   - graph6: `PsOihr~lEW{OeRSuLIhM]Fdg`
   - Adjacency matrix:

```
01110000111110000
10001010101101100
10000101110100110
10000011110011001
01000101101010101
00101010111000011
01010100100101011
00111000100011110
11111111000000000
10110100001001101
11001100010011010
11100010000010111
10011001001100011
01010011011000110
01101001010101001
00100111001111000
00011110010110100
```

2. Realization 2

   - graph6: `TsOihr~sd?uGoBQEhoPYQHBgQCgaagKo{HFe`
   - Adjacency matrix:

```
011100001111100000000
100010101100111000000
100001011011000110000
100000111101000001100
010001011000011100001
001010101000001110010
010101001100001001010
001110001001000100101
111111110000000000000
110100100010010000110
101000000100110110010
101100010000100011001
110000000011011010001
```

```
010010000110100100101
010011100000100011001
001011010010010000110
001001000011101001010
000100100001001010111
000100010100010101011
000001100110000111100
000010010001111001100
```

INDEX

CURRICULUM VITAE

Emily S. Hawboldt

emily.hawboldt@gmail.com | `github.com/ehawb`

EDUCATION	2017-2023	University of Louisville *Ph.D., Applied and Industrial Mathematics*
	2017-2019	University of Louisville *MA, Mathematics*
	2012-2017	University of Louisville *BM, Music Education* Band emphasis – horn Minor in Mathematics University of Louisville Honors Program Helen Boswell Award in Music Education *(Senior Award for Academic Achievement)*
TEACHING	2023-	Jefferson County Public Schools Hudson Middle School, 6th grade mathematics
	2023	Bellarmine University Bridge to BU
	2023	Jefferson County Public Schools Substitute teacher
	2017-2023	University of Louisville *2021-2022 Faculty Favorite* <u>Main instructor</u> Mathematics for Elementary Educators I Mathematics for Elementary Educators II College Algebra Contemporary Mathematics <u>Teaching assistant</u> Elements of Calculus Elementary Statistics College Algebra

		Contemporary Mathematics
	2017	Jefferson County Public Schools Student teacher <u>Jefferson County Traditional Middle School</u> Band, grades 6-8 <u>Foster Traditional Academy</u> General music, grades K-5
	2014	University of Louisville Summer Wind Band Institute Horn: Grades 6-12 Music theory: Grades 6-8
RESEARCH	2022	Presentation: University of Louisville American Mathematical Society Chapter *A machine-learning approach to Ramsey graphs leads to the Trahtenbrot-Zykov problem*
	2022	Presentation: University of Louisville Department of Mathematics *Ph.D. Candidacy Exam: Ramsey Theory and the Trahtenbrot-Zykov problem*
	2021	Presentation: University of Louisville Graduate Student Regional Research Conference *Ramsey theory: A reinforcement learning based approach*
	2019	Grant: University of Louisville Graduate Student Council Research Grant *Awarded $500 for graphics processing unit*
	2019	Presentation: Bluegrass Open Problems in Combinatorics Workshop *Antimagic Labelings*
	2017-2022	Independent studies *Deep learning and combinatorics* *Ramsey theory* *Programming for graph theory* *Research in combinatorics* *Algebraic graph theory* *Graph minors seminar*
INVOLVEMENT	2022-2023	Qualifying exams study group Started a weekly study group for mathematics Ph.D. students preparing for qualifying exams
	2021-2022	Mathematics graduate students walking club

	Started a biweekly walking club to build community among mathematics graduate students
2019-2021	Graduate Student Council Representative for Mathematics Department
2018-2020	General Education Committee Attended monthly meetings of the General Education Committee within the Department of Mathematics to discuss resources and content for general education courses in the department
2017-2023	American Mathematical Society University of Louisville Graduate Chapter Secretary, 2020-2022
2017-2022	Chamber Winds Louisville & Louisville Concert Band Horn player
2012-2016	University of Louisville Wind Ensemble Horn player
2012-2014	Cardinal Marching Band Mellophone player Section leader, 2013-2014