COVERING ARRAYS AVOIDING FORBIDDEN EDGES AND EDGE CLIQUE COVERS

Elizabeth Maltais

Thesis Submitted to the Faculty of Graduate and Postdoctoral Studies
In partial fulfilment of the requirements for the degree of Master of Science in Mathematics [1]

Department of Mathematics and Statistics
Faculty of Science
University of Ottawa

[1]The M.Sc. program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Abstract

Covering arrays avoiding forbidden edges (CAFEs) are combinatorial designs which can be used to generate test suites for practical testing applications. CAFEs generate test suites in which all required pairwise interactions between any two factors are tested at least once each, with the property that a specified list of pairwise interactions, the so called forbidden interactions, are avoided by all tests generated by the CAFE. Consequently, CAFEs can be applied to testing applications wherein constraints are imposed on the factors of the tests, resulting in forbidden interactions.

In this thesis, we study CAFEs, as well as their relationship to the edge clique cover problem from graph theory. We give new results and bounds for uniform edge clique covers and CAFEs. We establish the computational complexity of several problems related to CAFEs and edge clique covers. In particular, we prove that finding an optimal CAFE (as well as finding an optimal error-locating array) for a graph is NP-hard, even for the case of binary alphabets.

Acknowledgements

For making my Master's experience a truly enjoyable quest, I must deeply thank the following people.

First, thank you Lucia! My supervisor, Dr. Lucia Moura, provided me with the opportunity to do my Master's on a fascinating topic. Thanks to her guidance, expertise, innovative ideas, and detailed edits, my thesis turned out to be an accomplishment that I am very proud of. Lucia's suggestions and hints led to many interesting results, and I feel very fortunate to have had such an excellent supervisor. Obrigada!

I would also like to thank my family for their support and love: Mom, Dad, Pierre and John. Thanks to Gryffin and Monty too—my office would not have been the same without the poodle entities near by. Thank you Chris Dionne, my love, the unexpected and most wonderful outcome of my master's experience. "Working in parallel on our respective theses truly was a blessing" simply does not do to describe the amazing journey we have begun together. At least once every day: you make me smile; you make me laugh; you make me breakfast or tea; you captivate me with your ideas; you challenge mine; you hold my hand; you help; you listen; you laugh at me; you smile back. You are like the sun on my heart and I am immeasurably grateful to be able to share this life adventure with you, my dearest beloved.

For their time, suggestions, and positive comments, I would like to thank my thesis examiners Mike Newman and Brett Stevens.

For financial support and for providing me with the invaluable opportunity to be a teaching assistant, I am very grateful to the University of Ottawa, and the Department of Mathematics and Statistics, as well as to my supervisor.

For providing me with some fun distractions from my work, and for always being encouraging, I must thank my best friends Heather Schulz and Jennifer Côté. Thanks to Éric Serré for convincing me that I should do a master's and forcing me to do stations to get stronger. Thanks to my supervisor's co-student, Patrick Niesink, for being a friend in class and for providing me with class notes whenever I needed them.

Lastly, I want to thank Monica Nevins and Mateja Šajna for writing reference letters for me. I am especially grateful to Mateja for introducing me to Lucia in the first place, and for being the Chair of my defense at the end.

Elizabeth Jane Maltais

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Combinatorial designs are mathematical objects, typically involving subsets of a finite set that satisfy specified properties of intersection and balance. These properties vary depending on the design, and often have applications to real life problems.

In this thesis, we look at a family of combinatorial designs called covering arrays which are useful for designing smaller than exhaustive test suites. In Section 1.1, we define the testing problem to which covering arrays can be applied. In Section 1.2, we look at three types of covering arrays and explain how they are used in testing problems. In Section 1.3, we give an overview of the thesis, followed by Section 1.4 which highlights the necessary graph theoretic definitions as well as the most frequently used notation. In Section 1.5, we review basic computational complexity concepts.

1.1 The Testing Problem

Before a company releases a new product, much testing needs to occur in order to ensure high quality standards. Whether the product is a software-based electronic device or a new prescription drug, there are often various components or factors involved, each having several options, which should be tested in some sensible way. To model the general situation, we use the following definition.

Remark 1.1.1 Given integers l and m, we write $[l, l+m]$ to denote the set $\{l, l+1, l+2, ..., l+m\}$.

Definition 1.1.2 A **testing problem** is a system with k components called **factors**, which we label by the indices $1, ..., k$. Each factor $i \in [1, k]$ has g_i possible options, called **values**. Typically, we use the alphabet $[0, g_i - 1]$ to denote the values of factor i. For convenience, we denote such a testing problem as a $\mathrm{TP}(k, (g_1, ..., g_k))$. If the alphabet size is constant, that is, if $g_1 = g_2 = \cdots = g_k = g$ for some $g \in \mathbb{Z}$, then we shorten the notation to a $\mathrm{TP}(k, g)$. We represent a **test** by a k-tuple $T = (a_1, ..., a_k) \in [0, g_1 - 1] \times \cdots \times [0, g_k - 1]$, to mean that value a_i has been selected for factor i for each $i \in [1, k]$.

For example, Table 1.1 shows a $\mathrm{TP}(5, (3, 3, 2, 2, 2))$ for possible home entertainment systems. We can refer to a specific choice concisely by using the corresponding 5-tuple, say $(2, 0, 1, 0, 1)$, rather than by its full description: a 25" LCD TV with a DVD/VHS combo player, basic cable, no speakers and a Play Station game system. For our purposes, a test of such a system is simply a choice of one value for each of the factors. Our goal is to find out how successful the outcome of this particular combination of values is. At this point we need to distinguish between a *test* and the *procedure* used to determine its outcome.

The methods used to determine the success of a test depend on the particular system itself. We assume that the same **procedure** is run for each test. For example, if the system involves mixtures of chemical substances, then the procedure to determine the outcome of a test might be to measure the melting point, boiling point, strength, and flexibility of the resulting substance. On the other hand, the procedure to determine the outcome of the home entertainment system might be to let a family

Factors	**Values**
1 = TV	0 = 50” LCD high def. 1 = 32” LCD high def. 2 = 25” LCD
2 = DVD player	0 = DVD/VHS combo 1 = DVD only 2 = no DVD player
3 = Cable	0 = satellite 1 = basic
4 = Surround Sound System	0 = no 1 = yes
5 = Game system	0 = Nintendo 1 = Play Station

Table 1.1: Home entertainment system testing problem

use the system corresponding to a test for three weeks, and then to have them fill out a survey to determine their satisfaction.

The procedure for determining the outcome of the test is not our focal point. Regardless of the particular procedure, we only care about the success of the outcomes of the tests. To simplify matters, we assume that the nature of the system is such that the outcome of the tests performed is either *pass* or *fail.* If a test fails, we may assume that a fault is present in the system and that this fault is responsible for the test's failure. Our goal is thus to design a suite of tests which can reveal to us the faults of the system.

In practice, exhaustively testing a $\mathrm{TP}(k,(g_1,...,g_k))$ is too costly. Even for a moderately small testing problem, say a $\mathrm{TP}(5,4)$, exhaustive testing would require $4^5 = 1024$ tests, which could be infeasible depending on budget and time. So we must look for more reasonably sized test suites, but at the same time, we want the tests to cover a wide range of possibilities and test each value or combination of values in a balanced manner. For example, suppose we generate a test suite of a $\mathrm{TP}(5,4)$ by choosing the first ten tests in lexicographic order. We certainly save money and time compared to the exhaustive 1024 tests, but we do not even cover every value of every factor at least once each.

Since the purpose of testing products is to eliminate problems, we have to consider the causes of problems. It may be that one specific value of one of the factors is faulty. However, with systems involving several components, faults are often due to unexpected interactions that occur between a specific combination of the options (see [6, 34]). Therefore, one alternative to exhaustive testing would be to design a suite of tests in which every t-way interaction between any t of the factors is covered. To be precise we use the following definition.

Definition 1.1.3 [11] Let $\mathrm{TP}(k, (g_1, ..., g_k))$ be a testing problem, and let t be a positive integer such that $1 \leq t \leq k$. A **t-way interaction** is a set of values assigned to t distinct factors. We denote a t-way interaction as $I = \{(f_1, a_{f_1}), ..., (f_t, a_{f_t})\}$ where $f_i \in [1, k]$, $f_i \neq f_j$ for $i \neq j$, and $a_{f_i} \in [0, g_{f_i} - 1]$ for $1 \leq i \leq t$. If $t = 2$, we refer to a 2-way interaction as a **pairwise** interaction. If $t = 1$, we refer to a 1-way interaction as a **pointwise** interaction. We say that a test $T = (T_1, ..., T_k) \in [0, g_1 - 1] \times \cdots \times [0, g_k - 1]$ **covers** interaction $I = \{(f_1, a_{f_1}), ..., (f_t, a_{f_t})\}$ if $T_{f_i} = a_{f_i}$ for each $i \in [1, t]$.

As an alternative to exhaustive testing, test suites designed to cover all t-way interactions for some small value of t can be applied. Indeed, research has shown that testing all pairwise interactions in a testing problem finds a large percentage of existing faults, thus offers a good compromise to exhaustive testing [3, 10, 23, 24]. In the following section, we focus on the combinatorial designs which have the desired properties for designing test suites that guarantee the coverage of all t-way interactions.

1.2 Orthogonal Arrays, Covering Arrays, and Mixed Covering Arrays

We begin with orthogonal arrays, which were first introduced in 1943 (see [7]).

Definition 1.2.1 An **orthogonal array**, denoted $\mathrm{OA}_\lambda(N;t,k,g)$, is an $N \times k$ array with entries from an alphabet with g elements, typically $[0, g-1]$, such that in every $N \times t$ subarray, each t-tuple occurs *exactly* λ times as a row. Thus $N = \lambda g^t$. The parameter N is called the **size** of the array; t is its **strength**; k is the number of factors; we call λ the **index**. If $\lambda = 1$ we omit the subscript and denote the array simply as an $\mathrm{OA}(N;t,k,g)$.

For example, the following array is an $\mathrm{OA}(4;2,3,2)$:

$$A = \boxed{\begin{array}{ccc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{array}}\,.$$

We see that A has $N = 4$ rows and $k = 3$ columns with entries from the alphabet $X = \{0,1\}$, thus $g = 2$. The strength parameter is $t = 2$, so we see that in any two columns of A, the pairs $(0,0), (0,1), (1,0)$ and $(1,1)$ are all present exactly $\lambda = 1$ time each.

Suppose an $\mathrm{OA}(N;t,k,g)$ exists, say A. Then we can use A to solve a testing problem, $\mathrm{TP}(k,g)$, resulting in a test suite in the following way. Each column of A represents the corresponding factor of the $\mathrm{TP}(k,g)$. Each row $R_i = (R_i(1),..,R_i(k))$ of A is a k-tuple representing one test, where $R_i(j) \in [0, g-1]$ indicates the value to be assigned to the j^{th} factor. Since A contains N rows, the test suite consists of N tests. Since A has strength t, every $N \times t$ subarray of A contains each possible t-tuple exactly once each. In terms of the testing problem, this translates to every t-way interaction of the $\mathrm{TP}(k,g)$ being covered by exactly one test (one row of A).

There is a vast number of known constructions for orthogonal arrays. See, for example [17] for many constructions and results. The use of orthogonal arrays in testing applications, however, is limited, since for many parameters N, t, k, g, there does not always exist an $\mathrm{OA}(N; t, k, g)$. For example, there does not exist an $\mathrm{OA}(4; 2, 4, 2)$.

An array very similar to an orthogonal array exists for all parameters λ, t, k, g when we relax the requirement that every t-tuple appear *exactly* λ times to the requirement that every t-tuple appear *at least* λ times. This array is called a covering array and we give the precise definition below.

Definition 1.2.2 [6] A **covering array**, denoted $\mathrm{CA}_\lambda(N; t, k, g)$, is an $N \times k$ array, A, with entries from an alphabet with g symbols, typically $[0, g-1]$, such that in every $N \times t$ subarray, consisting of t columns of A, say columns $i_1, ..., i_t$, we have each t-tuple of $[0, g_{i_1} - 1] \times \cdots \times [0, g_{i_t} - 1]$ occurring *at least* λ times as a row. The parameter N is the **size** of the array; t is the **strength** of coverage of interactions; k is the number of **factors**; g is the number of values for each factor, called the **order**; λ is the **index**; if $\lambda = 1$, the subscript is omitted.

We see that a covering array is very much like an orthogonal array, with the exception that a covering array can have t-wise interactions repeating more than λ times each, whereas an orthogonal array is more strict in that it requires each t-tuple to occur *exactly* λ times each. In particular, every $\mathrm{OA}_\lambda(N; t, k, g)$ is a $\mathrm{CA}_\lambda(N; t, k, g)$. Unlike orthogonal arrays, covering arrays exist for every number of factors $k \in \mathbb{Z}$ because we can always enumerate all the k-tuples of our alphabet λ times each.

From now on, we consider only the case when the index $\lambda = 1$, and for given strength, number of factors, and alphabet size, of interest to us is the minimum size N for which a $\mathrm{CA}(N; t, k, g)$ exists.

Definition 1.2.3 Given parameters t, k, and g, the **covering array number**, denoted by $\mathrm{CAN}(t, k, g)$, is the minimum integer N for which a $\mathrm{CA}(N; t, k, g)$ exists. A covering array CA$(N; t, k, g)$ of size $N = \text{CAN}(t, k, g)$ is called **optimal**.

Evidently, since we require all t-tuples of our alphabet to appear at least once each, a lower bound for the covering array number must be

$$g^t \leq \text{CAN}(t, k, g).$$

Furthermore, removing a factor (column) from a CA$(N; t, k, g)$ yields a CA$(N; t, k - 1, g)$, thus

$$\text{CAN}(t, k-1, g) \leq \text{CAN}(t, k, g).$$

Since a covering array of strength t is also a covering array of strength t' for $1 \leq t' \leq t$, we have

$$\text{CAN}(t', k, g) \leq \text{CAN}(t, k, g),$$

for all $t' \in [1, t]$. For our purposes, we focus only on covering arrays of strength $t = 2$, which can be applied to testing problems where pairwise coverage is used.

In the particular case where the alphabet size is $g = 2$ we refer to such covering arrays as **binary** covering arrays. For this case, the binary covering array number has been determined exactly, for all numbers of factors k. It was shown for the case N even by Rényi [32], and independently by Kleitman and Spencer [21] and Katona [20], for all N.

Theorem 1.2.4 [29] Let k be a positive integer. Then

$$\text{CAN}(2, k, 2) = \min \left\{ N | \binom{N-1}{\lfloor \frac{N}{2} \rfloor - 1} \geq k \right\}.$$

The following asymptotic result holds for covering arrays, indicating that for fixed alphabet size g, the covering array number grows as $\log k$.

Theorem 1.2.5 (Gargano, Körner, and Vaccaro [14]) Let $g \geq 2$ be a fixed integer. Then, as $k \to \infty$,

$$\text{CAN}(2, k, g) \sim \frac{g}{2} \log_2 k.$$

For more information on covering arrays, their bounds and constructions, see Colbourn's survey [6].

For a covering array, the alphabet size g is constant and thus $g_i = g$ for $1 \leq i \leq k$; however, practical testing problems do not necessarily have this property. In practice, it is unlikely that we fall into the convenient case where each factor to be tested has the exact same number of possible values. This naturally brings us to a generalization of a covering array where we allow for the possibility of multiple alphabet sizes.

Definition 1.2.6 A **mixed covering array** (MCA) is an $N \times k$ array, A, having the following property. Each column i has symbols from the alphabet $[0, g_i - 1]$, and for $\{i_1, ..., i_t\} \subseteq \{1, ..., k\}$, if we consider the $N \times t$ subarray of A obtained by selecting columns $i_1, ..., i_t$, then there are $\prod_{i=1}^{t} g_i$ distinct t-tuples that could appear as a row. An MCA requires that each appear at least λ times. We denote such an array as an $\text{MCA}_\lambda(N; t, k, (g_1, g_2, ..., g_k))$. If $\lambda = 1$ then we omit the subscript.
Here is an example of an $\text{MCA}(9; 2, 5, (3, 3, 3, 2, 2))$ taken from [11]:

$$M = \boxed{\begin{array}{ccccc} 0 & 2 & 2 & 0 & 0 \\ 1 & 1 & 2 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 2 & 2 & 0 & 1 & 0 \\ 1 & 2 & 1 & 0 & 1 \\ 0 & 0 & 2 & 0 & 1 \\ 2 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 2 & 0 & 0 & 0 & 0 \end{array}}.$$

Again, we are hoping to find an MCA with the minimum possible number of rows, in order to reduce testing expenses.

Definition 1.2.7 The minimum integer N for which an $\text{MCA}(N; t, k, (g_1, ..., g_k))$ exists is called the **MCA number** and we denote it by $\text{MCAN}(t, k, (g_1, ..., g_k))$.

Since for every t columns $i_1, ..., i_t$, there are $\prod_{i=1}^{t} g_i$ possible t-tuples, a lower bound on the MCA number is

$$\prod_{i=1}^{t} g_{i_t},$$

for the t largest alphabet sizes $g_{i_1}, ..., g_{i_t}$. It is easy to check that the array obtained by reordering the columns of an MCA, is also an MCA with the same size, strength, number of factors, and index. Therefore, we can assume without loss of generality that the factors of an MCA are ordered so that $g_1 \geq g_2 \geq \cdots \geq g_k$, in which case our lower bound is simply $\prod_{i=1}^{t} g_i$. For upper bounds and constructions of MCAs, see [6] and [30].

Unfortunately, in practice, testing problems are even more complicated, and they frequently come with extra constraints. For many reasons, in a given system, not every plausible test is valid, and the constraints imposed on a testing problem can limit our options. Constraints may simply be due the incompatibility of specific values. For example, some values may require the presence or absence of specific values of other factors, otherwise testing these combinations does not make sense. We can also have dangerous interactions, such as explosive chemical mixtures. We may also wish to avoid interactions which have been found to be faulty by previous testing. Whatever the reason, many testing problems have constraints which result in interactions that should be avoided by all tests.

In this thesis, we consider further generalizations of mixed covering arrays which are used to build test suites for more complicated systems wherein constraints are present. We also study the translations of these problems into the language of graph theory, and examine the respective computational complexity of several of the problems. In the following section, we give a more detailed overview of the thesis.

1.3 Overview of Thesis

We begin by giving a review of the required graph theory in Section 1.4, highlighting the frequently used notations and definitions. In Section 1.5, we review some basic computational complexity concepts.

In Chapter 2, we look at covering arrays avoiding forbidden edges (CAFEs), a recently defined generalization of covering arrays, which take into consideration invalid pairwise interactions in a given testing problem. In practice, we often have constraints on the parameters of a testing problem, and we would like a test suite which covers all the desired interactions, but which avoids the forbidden combinations. The contents of this chapter are based almost entirely on work by Danziger, Mendelsohn, Moura, and Stevens [11]. We also look briefly at error-locating arrays, a related combinatorial object used in some of the computational complexity results that follow in Chapter 5.

In Chapter 3, we focus on a problem from graph theory, namely the edge clique cover (ECC) problem. The goal of the ECC problem is to find a collection of cliques of a given graph, such that every edge of the graph is covered by (has both its ends in) at least one of the cliques. In this chapter, we present some basic results for the ECC problem. We also consider three variations of the ECC problem, namely edge-disjoint ECCs, uniform ECCs, and partial ECCs. Since the uniform ECC problem is the variation that is most relevant to covering arrays and the testing problem, we prove several small new results in this area. In the particular case of partial ECCs, we contribute two new types of partial ECCs based on the edge-disjoint ECC problem and the uniform ECC problem. We also give some new basic results for partial ECCs. At the end of Chapter 3, we present a small section on node clique covers, another graph theory problem that is related to covering arrays, as well as its equivalence to the graph-colouring problem of the complementary graph.

In Chapter 4, we tie together several covering array problems to the corresponding variations of ECC problems. We present proofs of the equivalence between edge-disjoint ECCs and orthogonal arrays, ECCs of complete k-partite graphs and mixed covering arrays, and uniform ECCs and CAFEs. In particular, we contribute new results for the existence and upper bounds of CAFEs, based on their relationship with the uniform ECC problem. We also define a new type of array which we call a partial covering array avoiding forbidden edges. We give a proof of the equivalence between partial CAFEs and the partial uniform ECC problem, as well as some initial basic results. Partial CAFEs are a generalization of CAFEs, but accommodate an even wider range of testing applications.

In Chapter 5, we present several results on the computational complexity of problems related to CAFEs, ECCs, and error-locating arrays. In particular, we resolve the open question about the complexity of the decision problem related to the language 2-CAFEN, which addresses the existence of a binary CAFE of a specified size. We do so by giving a reduction from the ECC problem, which takes a simple graph and transforms it into a binary forbidden edges graph. We also use this result to show that g-CAFEN is NP-complete for all $g \geq 2$. We prove that the decision problem related to UNIFORM-ECCN, the language which addresses the existence of a partial ECC of a given graph, is NP-complete as well, using a reduction from 2-CAFEN. Furthermore, we extend the construction we use to prove the NP-completeness of 2-CAFEN in order to prove the NP-completeness of g-ELAN, the language which addresses the existence of error-locating arrays having alphabet size g, for all $g \geq 2$. This covers the remaining open cases of $g = 2, 3, 4$ since the NP-completeness of g-ELAN for $g \geq 5$ has been previously established [28].

1.4 Required Graph Theory and Notation

We now present a brief review of graph theory, giving the definitions and notations we use within this thesis. We follow the text by Bondy and Murty [1] throughout.

1.4.1 Basic Definitions

A **graph** G is an ordered pair $(V(G), E(G))$. The **vertex set** $V(G)$ consists of a nonempty set of elements called **vertices**. The **edge set** $E(G)$ is a multiset that is disjoint from $V(G)$, and its elements are called **edges**. An edge $e \in E(G)$ consists of a 2-element subset of $V(G)$, so that $e = \{u, v\}$ for two vertices $u, v \in V(G)$. If $e = \{u, v\}$, then e is said to **join** u and v; the vertices u and v are called the **ends** of e. The ends of an edge are said to be **incident** with the edge and vice-versa. Two vertices which are incident with a common edge are **adjacent**. Similarly, two edges which are incident with a common vertex are **adjacent**. If an edge e has identical ends, we refer to e as a **loop**; otherwise we simply call e an edge. If for two distinct edges, say e_1 and e_2 we have $e_1 = \{u, v\}$ and $e_2 = \{u, v\}$, that is, if two distinct edges have the same ends, then such edges are referred to as **parallel** edges.

A graph is **simple** if it contains no loops and no parallel edges. In the case where we have a simple graph, we may refer to an edge simply by its ends, since any pair of vertices in a simple graph can be joined by at most one edge. That is, if an edge e of a simple graph G satisfies $e = \{u, v\}$ then we can refer to e as the unordered pair $\{u, v\}$.

For our purposes, we only consider **finite** graphs, that is, graphs whose vertex set and edge set are both finite. We denote the number of vertices of a graph G as $|V(G)|$, and the number of edges of G as $|E(G)|$. Moreover, from now on, all graphs throughout this paper are simple finite graphs, and for convenience, we denote by $\mathbb{G}$ the set of all finite simple graphs.

In a graph G, the **degree** of a vertex $v \in V(G)$ is the number of edges of G incident with v, and we denote it by $d_G(v)$, or simply $d(v)$ when the context is clear. The minimum degree of any vertex in a graph G is denoted $\delta(G)$ and the maximum degree of a vertex of G is denoted $\Delta(G)$. The following lemma is referred to as the handshaking lemma.

Lemma 1.4.1 In any graph G, $\sum_{v\in V(G)} d_G(v) = 2|E(G)|$.

A vertex $v \in V(G)$ is called **isolated** if $d_G(v) = 0$. We call an edge $e \in E(G)$ an **isolated edge** if $e = \{u, v\}$ and $d_G(u) = 1 = d_G(v)$. The **neighbourhood** of $v \in V(G)$ in G, denoted by $N_G(v)$, is the set of vertices adjacent to v in G. That is, $N_G(v) = \{u \in V(G) | \{u, v\} \in E(G)\}$.

1.4.2 Equality, Subgraphs, and Induced Subgraphs

We say that two graphs H and G are **identical** if $V(H) = V(G)$ and $E(H) = E(G)$, in which case we write $H = G$. A graph H is a **subgraph** of G, denoted $H \subseteq G$, if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. When $H \subseteq G$ but $H \neq G$, we say that H is a **proper subgraph** of G, and we write $H \subset G$.

For a graph G, and a nonempty subset $V' \subseteq V(G)$, the subgraph of G whose vertex set is V' and whose edge set is the set of those edges of G that have both ends in V' is called the **subgraph of** G **induced by** V' and is denoted by $G[V']$. Similarly, for a nonempty subset of edges $E' \subseteq E(G)$, the subgraph of G whose vertex set is the set of ends of edges in E' and whose edge set is E' is called the **subgraph of** G **induced by** E', and is denoted $G[E']$.

For a subset of vertices V' of a the vertex set of graph G we use the notation $G - V'$ to denote the subgraph of G induced by the vertices *not* in V'. That is $G - V' = G[V(G) \setminus V']$. For a nonempty subset of edges, $E' \subseteq E(G)$, we use the notation $G \setminus E'$ to denote the graph with vertex set $V(G \setminus E') = V(G)$ and edge set $E(G \setminus E') = E(G) \setminus E'$.

1.4.3 Some Special Simple Graphs

A graph with just one vertex and no edges is **trivial** and all other graphs are **nontrivial**. A graph in which each pair of distinct vertices is joined by an edge is a **complete graph**. Up to isomorphism, there is only one complete graph on n vertices and it is denoted by K_n. Conversely, the graph on n vertices which contains no edges is called an **empty graph**, and we denote it by $\overline{K_n}$.

If k is a positive integer then a **k-partite graph** is one whose vertex set can be partitioned into subsets so that no edge has both ends in any one subset. The subsets of vertices in the partition are called the **partite sets**. A k-partite graph in which each vertex is joined to every vertex that is not in the same partite set is called a **complete k-partite graph**. If the partition of the vertices of a complete k-partite graph has partite sets of sizes $g_1, ..., g_k$, we denote such a graph as $K_{(g_1,...,g_k)}$. If a complete k-partite graph has partite sets of equal size, say $g_1 = \cdots = g_k = g$, then we simplify the notation to $K_{k,g}$. In the case where $k = 2$, a 2-partite graph is referred to as a **bipartite** graph. In the case where all parts of a k-partite graph are equal in size, we call this graph **equipartite**.

The **complement** of a graph G, denoted by $\overline{G}$, is the simple graph with vertex set $V(G)$, and the property that two distinct vertices are adjacent in $\overline{G}$ if and only if they are not adjacent in G.

1.4.4 Cliques and Independent Sets

Given a graph G, a subset of vertices, $C \subseteq V(G)$, is a **clique** of G if the subgraph of G induced by the vertex set C is complete. In other words, a clique of a graph is a subset of vertices in which every pair of distinct vertices is joined by an edge. Conversely, a subset of vertices, $I \subseteq V(G)$, is an **independent set** of G if the subgraph of G induced by the vertex set I is empty.

Aside from having similar definitions, cliques and independent sets are fundamentally related to each other by taking the complement of the graph in which they belong. The following easy result makes this relationship precise.

Proposition 1.4.2 Let G be a simple graph. Then a subset of vertices $C \subseteq V(G)$ is a clique of G if and only if C is an independent set of $\overline{G}$.

If C is a clique of G containing k vertices, that is, if $|C| = k$, then we call C a **k-clique** of G. Moreover, a clique C of G is **maximum** if G contains no clique C' such that $|C'| > |C|$. The **clique number** of a graph G, written $\omega(G)$, is the number of vertices in a maximum clique of G. Similarly, if I is an independent set of G containing k vertices, then we call I an **independent set of size** k. An independent set I of G is **maximum** if G contains no independent set I' such that $|I'| > |I|$. The **independence number** of G, denoted by $\alpha(G)$, is the number of vertices in a maximum independent set of G. .

1.4.5 Connectivity

In a simple graph, a finite non-null sequence $P = v_0\{v_0, v_1\}v_1\{v_1, v_2\}v_2...\{v_{l-1}, v_l\}v_l$, whose terms are alternately vertices $v_i \in V(G)$ for $0 \leq i \leq l$ and edges $\{v_{i-1}, v_i\} \in E(G)$ for $1 \leq i \leq l$, is called a **(v_0, v_l)-walk**. The vertex v_0 is called the **origin** of P, and v_l is called the **terminus** of P. The **length** of P is l, the number of edges in the sequence. If, in particular, $v_i \neq v_j$ for $0 \leq i < j \leq l$, then P is called a **(v_0, v_l)-path**. A finite non-null sequence $P = v_0v_1...v_{l-1}v_l$ is called a **cycle** if $v_i \neq v_j$ for $i \neq j$, $\{v_i, v_{i+1}\} \in E(G)$ for $0 \leq i \leq l-1$ and $v_0 = v_l$. If $P = v_0...v_{l-1}v_0$ is a cycle then l is its length. If the length of a path P (or, cycle, respectively) is odd, then we call P an **odd path** (**odd cycle**, respectively). If the length of P is even, then we call P an **even path** (**even cycle**, respectively).

Two vertices $u, v \in V(G)$ are **connected** if there exists a (v_0, v_l)-path in G. Two vertices being connected in a graph G forms an equivalence relation on the vertex set $V(G)$. Thus, there exists a partition of $V(G)$ into nonempty subsets $V_1, ..., V_m$ such that two vertices u and v are connected in G if and only if both u and v belong to the same set V_i. The induced subgraphs $G[V_1], ..., G[V_m]$ are called the **connected components** of G. If G has exactly $m = 1$ connected component, then we say that G is **connected**; otherwise, G is **disconnected**.

1.4.6 Graph Colouring

A k**-vertex colouring** of a graph G is an assignment of k colours, 1,...,k, to the vertices of G. If no two adjacent vertices are assigned the same colour, then the colouring is called **proper**. A proper k-vertex colouring partitions $V(G)$ into k (possibly empty) sets $\{V_1, ..., V_k\}$, so that vertices in V_i are assigned colour i, for $1 \leq i \leq k$. If G admits a proper k-vertex colouring, then we call G k**-colourable**.

The minimum k for which a graph G is k-colourable is called the **chromatic number** of G, and is denoted by $\chi(G)$. In fact, a graph G is k-colourable if and only if it is k-partite, since the colour classes $V_1, ..., V_k$ have the property that no two vertices in a given colour class V_i are adjacent in G. A proper $\chi(G)$-vertex colouring of a graph G is called **optimal**.

For the particular case of 2-colourable graphs (bipartite graphs) the following result holds.

Theorem 1.4.3 A graph is 2-colourable (bipartite) if and only if it contains no odd cycles.

1.5 Required Computational Complexity Concepts

In this section, we give the definitions of the complexity classes P and NP, as well as the concept of NP-completeness and polynomial-time reducibility, following [9] throughout. We consider only *decision problems* here. That is, we do not look directly at the problem of finding an optimal solution to a given optimization problem, but rather, we look at a closely related problem for which the answer is either "yes" or "no." Let us be more precise. An **abstract decision problem** Q is a map from the set I of instances for the problem to the set of solutions, namely $\{0, 1\}$. Thus, for an instance $i \in I$ we have $Q(i) = 1$ if the answer for the decision is "yes" and $Q(i) = 0$ if the answer is "no."

Since computers are designed to use binary strings as encodings for abstract structures, as well as numbers and letters, it makes sense to translate instances for an abstract decision problem into binary strings. We do this using an **encoding** which is simply a map e from the set I of instances for an abstract decision problem to the set of binary strings. We denote by ε the empty string and let $\{0,1\}^*$ denote the set of all binary strings. That is, $\{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$. Thus, computers do not solve the abstract decision problem, but rather, they solve the encoded version. For convenience, given an abstract decision problem $Q : I \to \{0,1\}$ and an encoding $e : I \to \{0,1\}^*$, we denote by Q_e the encoded version of Q. That is, Q_e is a map from the set $\{0,1\}^*$ to the set of solutions $\{0,1\}$, so that for any instance $i \in I$, we have $Q(i) = 1$ if and only if $Q_e(e(i)) = 1$. Since an encoding e does not necessarily map surjectively onto $\{0,1\}^*$, we use the convention that if $x \in \{0,1\}^*$ and $x \neq e(i)$ for some $i \in I$, then $Q_e(x) = 0$. As a result, we may think of an abstract decision problem and its encoded version interchangeably.

Now, in order to represent an abstract decision problem concisely, we use *languages*. For our purposes, a **language** L is any set of strings made up of symbols from the alphabet $\{0,1\}$. Every language over $\{0,1\}$ is thus a subset of $\{0,1\}^*$. With this framework, an abstract decision problem Q can be viewed as a language L over $\{0,1\}$ where $L = \{x \in \{0,1\}^* | Q_e(x) = 1\}$. In practice, we more commonly define a language by the underlying abstract problem, rather than the encoded version. That is, when we have a decision problem Q defined as the language $L = \{x \in \{0,1\}^* | Q_e(x) = 1\}$, we more commonly write L as $L = \{i \in I | Q(i) = 1\}$.

A language L is **decided** by an algorithm A if for every $x \in L$, the algorithm's output is $A(x) = 1$, and for every $x \notin L$, the algorithm's output is $A(x) = 0$. A language L is **polynomial-time decidable** if there is an algorithm A and a constant c such that for any input $x \in \{0,1\}^*$ such that $|x| = n$, the algorithm A correctly decides whether $x \in L$ in time $O(n^c)$. A **verification algorithm** is a two-input algorithm A, where one argument is an input string x and the other is a binary string y called a **certificate**. We say that A **verifies** an input string x if there exists a certificate y such that $A(x, y) = 1$. The language **verified by** a verification algorithm A is

$$L = \{x \in \{0,1\}^* | \text{there exists } y \in \{0,1\}^* \text{ such that } A(x,y) = 1\}.$$

Informally, if we are given an instance $x \in L$, then there exists a certificate y that A can use as proof that $x \in L$. If there exists a polynomial-time algorithm A that verifies a language L, then we say that L is **polynomial-time verifiable**.

The **complexity class P** is the class of languages that are polynomial-time decidable. That is,

$$P = \{L \subseteq \{0,1\}^* | \text{ there exists an algorithm } A \text{ that decides } L \text{ in polynomial time}\}.$$

The **complexity class NP** is the class of languages that can be verified by a polynomial-time algorithm. We say that a language L **belongs to the class NP** if and only if there exists a two-input polynomial-time algorithm A and a constant c such that

$$L = \{x \in \{0,1\}^* | \text{ there exists a certificate } y \text{ with } |y| = O(|x|^c) \text{ such that } A(x,y) = 1\}.$$

It is clear that P $\subseteq$ NP since any decision problem that is polynomial-time decidable must also be polynomial-time verifiable. However, the answer to the question "does P equal NP?" is unknown. In particular, there is a special subset of decision problems in the class NP called NP-complete problems which are in some sense the hardest problems in the class NP. In order to give the definition of NP-complete, we need to be precise in what we mean by "a problem is at least as hard as another problem." We give the definition for reducibility below.

Definition 1.5.1 A language L_1 is **polynomial-time reducible** to a language L_2 if there exists a polynomial-time computable function f taking instances for the language L_1 to instances for the language L_2 such that for all x, we have $x \in L_1$ if and only if $f(x) \in L_2$. If such a function exists, we use the notation $L_1 \leq_P L_2$ to denote that L_1 is polynomial-time reducible to L_2. The function f is called the **reduction function**, and a polynomial-time algorithm A_f that computes $f(x)$ is called a **reduction algorithm**.

Thus, if we have $L_1 \leq_P L_2$ for languages L_1 and L_2 we can say that the decision problem for L_2 is at least as hard as the decision problem for L_1 since a polynomial-time algorithm that decides L_2 together with a reduction function f can be used to decide L_1 in polynomial time.

Lemma 1.5.2 If L_1 and L_2 are languages such that $L_1 \leq_P L_2$, then $L_2 \in P$ implies $L_1 \in P$.

Now we give the definition for an NP-complete language.

Definition 1.5.3 A language L is **NP-hard** if $L' \leq_P L$ for every language $L' \in$ NP. If L is NP-hard and we also have $L \in$ NP, then L is **NP-complete.**

If a language is NP-complete, we see that its decision problem is at least as hard as that of every other language in the class NP. In fact, no polynomial-time algorithm that solves the decision problem for an NP-complete language is known, and the following theorem tells us that if any NP-complete language admits a polynomial-time algorithm, then every language in NP does as well.

Theorem 1.5.4 If any NP-complete problem is polynomial-time decidable, then P = NP. Equivalently, if any problem in NP is not polynomial-time decidable, then no NP-complete problem is polynomial-time decidable.

Proof: Suppose that $L \in$ P and L is NP-complete. For any $L' \in$ NP, we have $L' \leq_P L$. By Lemma 1.5.2 we must have that $L' \in$ P also, for every $L' \in$ NP which implies NP $\subseteq$ P and so P = NP. ▣

Now, given a language that is NP-complete we can prove that another language is NP-complete via the following lemma.

Lemma 1.5.5 If L_0 is a language that is NP-complete and L is a language such that $L_0 \leq_P L$, then L is NP-hard. Moreover, if additionally we have $L \in$ NP, then L is NP-complete.

Proof: Since L_0 is NP-complete, for every language $L' \in$ NP we have $L' \leq_P L_0$. We also have $L_0 \leq_P L$ and by transitivity of $\leq_P$ for every language $L' \in$ NP we have $L' \leq_P L$. Thus L is NP-hard. If additionally we have $L \in$ NP, then L is also NP-complete. ▣

The usefulness of Lemma 1.5.5 relies on the existence of some initial NP-complete language. That is, we need some language that can be proven to be NP-complete from scratch, without already requiring the existence of an NP-complete problem. Indeed, in 1971, Cook [8] proved the existence of an NP-complete problem, the **satisfiability** problem. A **boolean formula** consists of the conjunction of clauses, wherein each clause is the disjunction of literals (a literal is a variable or the negation of a variable). A boolean formula φ is **satisfiable** if there exists a valuation V, that is an a assignment of truth values (T or F) to each of the variables, such that $V(\varphi) = T$. The satisfiability problem is given by the language

$$\text{SAT} = \{\varphi | \varphi \text{ is a satisfiable boolean formula}\}.$$

For general k, instances for k-SAT are boolean formulas consisting of the conjunction of clauses, wherein each clause consists of exactly k literals. Levin [25] also proved the existence of an NP-complete tiling problem, independently from Cook. In 1972, Karp [19] proved that several central combinatorial problems, including the clique problem, the graph colouring problem, and 3-SAT, are all NP-complete. In contrast to the NP-completeness of 3-SAT, it is known that 2-SAT $\in$ P. By revealing that such a large number of important problems are computationally intractable, Karp's work resulted in further study of NP-completeness and his paper provides many tools for proving NP-completeness via reductions from other NP-complete problems. In 1979, Garey and Johnson [13] catalogued many problems that were known at the time to be NP-complete, and provided an excellent guide to the theory of NP-completeness.

Chapter 2

Covering Arrays Avoiding Forbidden Edges

Using MCAs to generate test suites can be useful; however, these designs do not take into consideration the possibility that some interactions in a testing problem may be invalid, faulty, or even dangerous. Such interactions should be avoided (not covered) by all tests performed. Thus, we are looking for a suite of tests which do cover all the allowable interactions of a desired size, but which avoid all forbidden interactions. In this chapter, the object we study for this purpose is a covering array avoiding forbidden edges. Section 2.1 motivates the study of such objects. Section 2.2 looks at a family of graphs which are used to represent the forbidden interactions in a testing problem. In Section 2.3, we give the definition of covering arrays avoiding forbidden edges and some important properties. Sections 2.4 and 2.5 review bounds and a construction for covering arrays avoiding forbidden edges. Lastly, Section 2.6 looks at error-locating arrays, objects which are closely related to covering arrays avoiding forbidden edges.

2.1 Testing in the Presence of Forbidden Configurations

For some reason, a particular t-way interaction of a given testing problem may need to be forbidden from all tests. For example, some combinations of components in a highly-configurable software system can be invalid. Table 2.1 shows an example presented by Cohen et. al. [5], of a $\mathrm{TP}(5,(3,3,3,2,2))$ for a mobile phone product line. This system contains some inherent constraints. For example, video ringtones cannot be used without the presence of a video camera. Table 2.2 gives a list of constraints on the various options, as well as the resulting forbidden interactions. In this case, the system has seven forbidden pairwise interactions and one forbidden 3-way interaction. An $\mathrm{MCA}(N;2,5,(3,3,3,2,2))$ provides a suite of tests which guarantees the coverage of all pairwise interactions, but ignores the constraints completely. Consequently, some of the tests generated by the MCA simply cannot take place, resulting in wasted tests. Thus it is desirable to design a minimal suite of tests which cover all permitted interactions, but which avoid the forbidden interactions. Experiments involving material mixtures provide an example of a testing problem where ignoring forbidden interactions could be deadly. These types of experiments may combine materials in order to produce mixtures with improved properties such as strength and flexibility, but absolutely must avoid creating known explosive or toxic combinations. Cawse [4] supports the use of covering arrays for the design of such experiments, but the ability to avoid the dangerous combinations is essential.

Hartman and Raskin [16] address the need for forbidden configurations in some testing applications, although their proposed solution requires an exhaustive list of all invalid tests (not simply a list of the forbidden interactions themselves). We opt not to use their solution since its non-compact representation seems inefficient. Cohen et. al. [5] define constrained covering arrays and present a general technique for representing

Factors	**Values**
1 = display	0 = 16 million colours 1 = 8 million colours 2 = black and white
2 = email viewer	0 = graphical 1 = text 2 = none
3 = camera	0 = 2 megapixels 1 = 1 megapixel 2 = none
4 = video camera	0 = yes 1 = no
5 = video ringtones	0 = yes 1 = no

Table 2.1: Mobile phone product line

Constraints	**Forbidden Interactions**
(C1) graphical email viewer **requires** a colour display	$\{(1,2),(2,0)\}$
(C2) 2 megapixel camera **requires** a colour display	$\{(1,2),(3,0)\}$
(C3) graphical email viewer is **not supported** with 2 megapixel camera	$\{(2,0),(3,0)\}$
(C4) 8 million colour display **does not support** a 2 megapixel camera	$\{(1,1),(3,0)\}$
(C5) video camera **requires** a camera and a colour display	$\{(3,2),(4,0)\}$ $\{(1,2),(4,0)\}$
(C6) video ringtones **cannot occur** without a video camera	$\{(4,1),(5,0)\}$
(C7) the combination of 16 million colours, text, and 2 megapixel camera will **not be supported**	$\{(1,0),(2,1),(3,0)\}$

Table 2.2: Constraints on the mobile phone product line

constraints so that existing testing problem algorithms can now handle constraints. Danziger et. al. [11] use graphs to represent forbidden pairwise interactions of a testing problem and define covering arrays avoiding forbidden edges. The use of graphs allows the authors to establish links between a clique covering problem from graph theory and testing problems having constraints.

We follow the solution of Danziger et. al. [11] and study covering arrays avoiding forbidden edges (CAFEs). In general, the constraints imposed on a testing problem can result in forbidden interactions of any size (forbidden t-way interactions for any $t \in [1, k]$). For example, the constraint (C7) of Table 2.2 yields a forbidden 3-way interaction. These situations can be modeled using hypergraphs representing the forbidden interactions; however, we concentrate solely on the simpler case, where all forbidden interactions are pairwise interactions.

Remark 2.1.1 From now on, we only consider the problem of covering all pairwise interactions of a given testing problem. Moreover, if a testing problem has an associated forbidden interaction set, we assume all forbidden interactions are pairwise interactions. Unless otherwise stated, we refer to pairwise interactions simply as interactions.

2.2 The Forbidden Edges Graph

Given a testing problem $\mathrm{TP}(k, (g_1, ..., g_k))$ and an associated **forbidden (pairwise) interaction set**, say $\mathfrak{F} = \{I | I \text{ is a forbidden interaction of } \mathrm{TP}(k, (g_1, ..., g_k))\}$, we represent the forbidden interactions using a k-partite graph G that is a member of the following family of graphs.

Definition 2.2.1 The **family of forbidden edges graphs**, denoted by $\mathcal{G}_{(g_1,...,g_k)}$, is the family of k-partite graphs having parts of sizes $g_1, ..., g_k$. Furthermore the vertices of any $G \in \mathcal{G}_{(g_1,...,g_k)}$ are labeled v_{i,a_i} where $i \in [1, k]$ and $a_i \in [0, g_i - 1]$, so that the respective parts are of the form $P_i = \{v_{i,a_i} | a_i \in [0, g_i - 1]\}$ for each $i \in [1, k]$. The edge set of any $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ is a subset of the edge set of the complete k-partite graph $K_{(g_1,\ldots,g_k)}$ with vertices labeled likewise. In the particular case when $g_1 = g_2 = \cdots = g_k = g$, we denote the family of forbidden edges graphs with uniform alphabet size g as $\mathcal{G}_{k,g}$.

For each $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ there is a corresponding testing problem $\mathrm{TP}(k, (g_1, \ldots, g_k))$ with forbidden interaction set $\mathfrak{F}$ such that G contains one vertex for each value of each factor in the corresponding testing problem. Moreover, each partite set $P_i = \{v_{i,a_i} | a_i \in [0, g_i - 1]\}$ corresponds to the g_i values of factor i, and we sometimes refer to this vertex set simply as **factor** i. For $i \neq j$, two vertices v_{i,a_i} and v_{j,a_j} are joined by an edge in G if and only if the interaction between the values a_i and a_j between factors i and j, respectively, is forbidden. That is, if interaction $I = \{(i, a_i), (j, a_j)\} \in \mathfrak{F}$, then the graph G contains the edge $\{v_{i,a_i}, v_{j,a_j}\}$. In this case, we call G the **forbidden edges graph** for the testing problem $\mathrm{TP}(k, (g_1, \ldots, g_k))$ with forbidden interaction set $\mathfrak{F}$.

Although we never allow a test to assign two distinct values of a given factor simultaneously, we do not add these “implicity forbidden” interactions to the forbidden edges graph. However, it is sometimes convenient to consider the graph obtained from a forbidden edges graph G that does have these edges. We denote by $G^{|}$ the graph obtained from a forbidden edges graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ by adding to G all edges of the form $\{v_{i,a_i}, v_{i,b_i}\}$ for each factor $i \in [1, k]$ and for every two distinct values $a_i \neq b_i$ such that $a_i, b_i \in [0, g_i - 1]$. Figure 2.1 shows a forbidden edges graph G as well as the graph $G^{|}$. In the following section, we consider covering arrays associated to a forbidden edges graph.

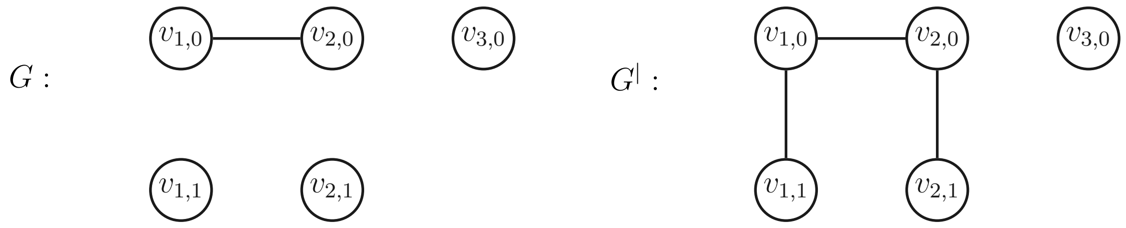


Figure 2.1: A forbidden edges graph G and the graph $G^{|}$

The graph $G \in \mathcal{G}_{(2,2,1)}$ *is associated to the testing problem TP*$(3, (2, 2, 1))$*, having one forbidden interaction between factors 1 and 2, namely* $\mathfrak{F} = \{\{(1, 0), (2, 0)\}\}$*. The graph* $G^{|}$ *contains edges joining vertices within each factor.*

2.3 Covering Arrays Avoiding Forbidden Edges

Now, we are looking for a suite of tests which avoids all forbidden interactions represented by a given forbidden edges graph, but which does cover all allowable interactions between factors. The structure that does the trick, if it exists, is called a covering array avoiding forbidden edges. First, let us consider the following definitions.

Definition 2.3.1 [11] Given a testing problem TP$(k, (g_1, ..., g_k))$, we say that a k-tuple $T = (T_1, ..., T_k)$ **avoids** interaction $I = \{(f_1, a_{f_1}), ..., (f_t, a_{f_t})\}$ if T does not cover I. A k-tuple $T = (T_1, ..., T_k) \in [0, g_1 - 1] \times \cdots \times [0, g_k - 1]$ is said to **avoid** the forbidden edges graph $G \in \mathcal{G}_{(g_1,...,g_k)}$ if for all $i, j \in [1, k]$, we have $\{v_{i,T_i}, v_{j,T_j}\} \notin E(G)$.

We should remark that when a k-tuple $T = (T_1, ..., T_k)$ avoids a graph G it must be that the set of vertices $\{v_{1,T_1}, ..., v_{k,T_k}\}$ is an independent set of $G^{|}$ (and therefore of G as well).

Lemma 2.3.2 Let $G \in \mathcal{G}_{(g_1,...,g_k)}$. Then $T = (T_1, ..., T_k)$ forms a k-tuple avoiding G if and only if $I = \{v_{1,T_1}, ..., v_{k,T_k}\}$ is an independent set of $G^{|}$.

Proof: Let $T = (T_1, ..., T_k)$ be a k-tuple avoiding G. By definition, for all $i, j \in [1, k]$ we have $\{v_{i,T_i}, v_{j,T_j}\} \notin E(G)$, which is equivalent to $I = \{v_{1,T_1}, ..., v_{k,T_k}\}$ being an independent set of $G^|$. Conversely, if $I = \{v_{1,T_1}, ..., v_{k,T_k}\}$ is an independent set of $G^|$, then for all $i, j \in [1, k]$, we have $\{v_{i,T_i}, v_{j,T_j}\} \notin E(G)$. Thus $T = (T_1, ..., T_k)$ forms a k-tuple avoiding G. ▣

Remark 2.3.3 It is sometimes convenient for us to refer to an interaction $I = \{(i, a_i), (j, a_j)\}$ simply as the pair of vertices $\{v_{i,a_i}, v_{j,a_j}\}$. Then, if $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$, we have a non-forbidden interaction, and if $\{v_{i,a_i}, v_{j,a_j}\} \in E(G)$, we have a forbidden interaction.

We now give the definition of a CAFE.

Definition 2.3.4 [11] A **covering array avoiding forbidden edges** (CAFE) of a graph $G \in \mathcal{G}_{(g_1,...,g_k)}$, is an $N \times k$ array A, with each column i having symbols from the alphabet $[0, g_i - 1]$, and denoted CAFE(N, G), such that:

1. each row of A forms a k-tuple avoiding G, and
2. for all $v_{i,a_i}, v_{j,a_j} \in V(G)$ with $i \neq j$, if $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$, then there exists a row $R_l = (R_l(1), ..., R_l(k))$ such that $R_l(i) = a_i$ and $R_l(j) = a_j$.

The **CAFE number** of a forbidden edges graph G, denoted by CAFEN(G), is the minimum integer N for which a CAFE(N, G) exists, if a CAFE of G exists, or $+\infty$ otherwise.

As depicted in Figure 2.2, we see that there does not always exist a CAFE(n, G) for every graph $G \in \mathcal{G}_{(g_1,...,g_k)}$. This brings us to the next definition where we consider the properties of forbidden edges graphs for which a CAFE exists.

Definition 2.3.5 [11] Let $G \in \mathcal{G}_{(g_1,...,g_k)}$. An interaction $I = \{(i, a_i), (j, a_j)\}$ is said to be **consistent** with G if there exists a k-tuple T with $T_i = a_i$ and $T_j = a_j$ that avoids G. The graph G is **consistent** if all interactions $\{(i, a_i), (j, a_j)\}$ such that $i \neq j$ and $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$ are consistent with G.

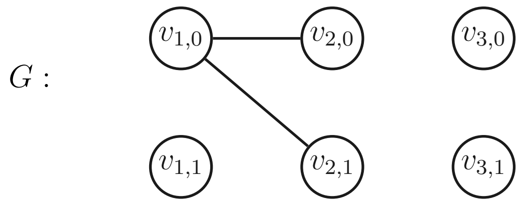


Figure 2.2: Example of a forbidden edges graph that is not consistent
The graph $G \in \mathcal{G}_{3,2}$ is associated to the testing problem TP$(3,2)$ with forbidden interaction set $\mathfrak{F} = \{\{(1,0),(2,0)\},\{(1,0),(2,1)\}\}$. If a CAFE$(n,G)$ exists then in particular there needs to be a row that covers the non-forbidden interaction $I = \{(1,0),(3,0)\}$; however, upon closer inspection, we see that such a row cannot exist since there is no value that can be assigned to the second factor within the same row that covers the point $v_{1,0}$.

Indeed, there exists a CAFE(n,G) for a forbidden edges graph G if and only if G is consistent. For a forbidden edges graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$, we denote by $\hat{G}$ the minimal supergraph of G that is consistent. We refer to $\hat{G}$ as the **avoidance closure** of G [11]. Indeed, $\hat{G}$ exists for every $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ since we can always add to G all possible edges and the complete k-partite graph $K_{(g_1,\ldots,g_k)}$ is trivially consistent.

For the particular case of **binary** forbidden edges graphs, that is, for graphs $G \in \mathcal{G}_{k,2}$ corresponding to CAFEs with binary alphabets, we have the following result by Danziger et. al. [11] which characterizes their consistency.

Proposition 2.3.6 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{k,2}$ be a forbidden edges graph with vertex set $V(G) = \{v_{i,a} | 1 \leq i \leq k, a \in \{0,1\}\}$. Then G is consistent if and only if

1. $\{v_{i,a}, v_{j,b}\} \in E(G)$ whenever $i \neq j$ and there exist vertices in the same factor, say $v_{l,c}$ and $v_{l,1-c}$, such that $l \neq i, l \neq j$ and $\{v_{i,a}, v_{l,c}\} \in E(G)$ and $\{v_{j,b}, v_{l,1-c}\} \in E(G)$ (see Figure 2.3-1.), and
2. $\{v_{i,a}, v_{j,b}\} \in E(G)$ for all $j \in [1,k] \setminus \{i\}$ whenever there exist vertices in the

same factor, say $v_{l,0}$ and $v_{l,1}$ such that $l \neq i$, $l \neq j$, and $\{v_{i,a}, v_{l,0}\} \in E(G)$ and $\{v_{i,a}, v_{l,1}\} \in E(G)$ (see Figure 2.3-2.).

Proof: Suppose we have vertices $v_{i,a}, v_{j,b}, v_{l,c}$ and $v_{l,1-c}$ from three distinct parts i, j and l, such that $\{v_{i,a}, v_{l,c}\} \in E(G)$, $\{v_{j,b}, v_{l,1-c}\}$, but $\{v_{i,a}, v_{j,b}\} \notin E(G)$. Then the non-forbidden interaction $\{(i,a),(j,b)\}$ would not be consistent with G and consequently G would not be consistent. Similarly, if edges $\{v_{i,a}, v_{l,0}\}$ and $\{v_{i,a}, v_{l,1}\}$ exist, but for $i \neq j$ we have $\{v_{i,a}, v_{j,b}\} \notin E(G)$, then $\{(i,a),(j,b)\}$ would not be consistent with G. Thus, necessity is clear.

For sufficiency, assume that 1. and 2. holds. Let $\{v_{i,a}, v_{j,b}\}$ be any non-edge of G with $i \neq j$. We give an algorithm that yields a k-tuple $T = (T_1, ..., T_k)$ avoiding G and covering $\{(i,a),(j,b)\}$. Initially set $F = \emptyset$. At the first iteration the algorithm sets $f_1 = i$ and $T_{f_1} = a$, and resets $F \leftarrow F \cup \{f_1\}$. At the second iteration, the algorithm sets $f_2 = j$ and $T_{f_2} = b$, and resets $F \leftarrow F \cup \{f_2\}$. At the m^{th} iteration, $F = \{f_1, f_2, ..., f_{m-1}\}$ and T_{f_i} has been set for all $f_i \in F$ so that the set of vertices $\{v_{f_i,T_{f_i}} | f_i \in F\}$ is an independent set of G. We then select $f_m \in [1,k] \setminus F$. If $\{v_{f_m,0}, v_{f_i,T_{f_i}}\}$ is an edge of G for some $f_i \in F$ then set $T_{f_m} = 1$. Otherwise, set $T_{f_m} = 0$.

At this point, if the algorithm fails it means that neither $T_{f_m} = 0$ nor $T_{f_m} = 1$ yields an independent set. That is, we must have $f_p, f_q \in F$ such that $\{v_{f_p,T_{f_p}}, v_{f_m,0}\} \in E(G)$ and $\{v_{f_q,T_{f_q}}, v_{f_m,1}\} \in E(G)$. If $f_p \neq f_q$, then 1. implies that $\{v_{f_p,T_{f_p}}, v_{f_q,T_{f_q}}\} \in E(G)$. This contradicts our assumption that $\{v_{f_i,T_{f_i}} | f_i \in F\}$ is an independent set, hence is impossible. If $f_p = f_q$ then 2. implies that $\{v_{f_i,T_{f_i}}, v_{f_p,T_{f_p}}\} \in E(G)$ for every $f_i \neq f_p$ which again, contradicts our assumption that $\{v_{f_i,T_{f_i}} | f_i \in F\}$ is an independent set. Therefore, the algorithm cannot fail at the m^{th} iteration.

The algorithm then sets $F \leftarrow F \cup \{f_m\}$ and iterates similarly until $|F| = k$, at which point T is a k-tuple avoiding G. Thus every interaction $\{(i,a),(j,b)\}$ such that $i \neq j$ and $\{v_{i,a}, v_{j,b}\} \notin E(G)$ is consistent with G and by definition G is consistent. ▣

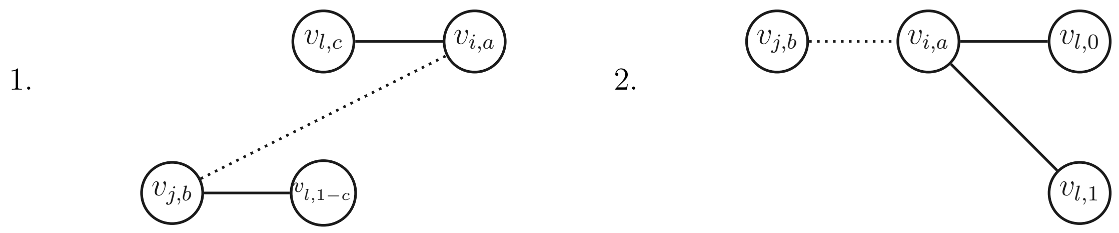


Figure 2.3: Forbidden induced subgraphs for binary forbidden edges graphs
The dotted edges indicate interactions that would not be consistent with G if non-forbidden. That is, the dotted edges must be present in order for the graph to be consistent.

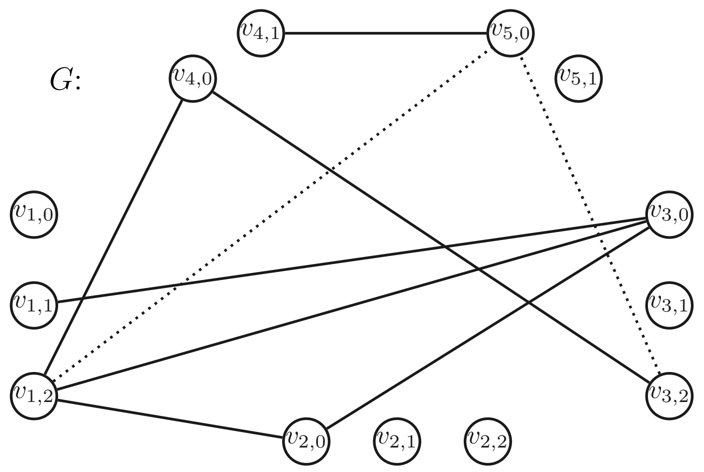
The above proposition is equivalent to stating that for $G \in \mathcal{G}_{k,2}$, $G = \hat{G}$ if and only if G does not contain either of the induced forbidden subgraphs shown in Figure 2.3. We now give the definition of a related structure which we sometimes find useful.

Definition 2.3.7 Let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$. A **pointwise CAFE**, denoted $\mathrm{CAFE}^1(N, G)$, is an $N \times k$ array A, with each column i having symbols from the alphabet $[0, g_i - 1]$, such that:

1. each row of A forms a k-tuple avoiding G, and
2. for all $v_{i,a_i} \in V(G)$ such that there exists $v_{j,a_j} \in V(G)$ with $i \neq j$ and $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$, there exists a row $R_l = (R_l(1), ..., R_l(k))$ that covers the pointwise interaction $\{(i, a_i)\}$.

The **pointwise CAFE number** of G, denoted by $\mathrm{CAFEN}^1(G)$, is the minimum integer N for which a $\mathrm{CAFE}^1(N, G)$ exists if a pointwise CAFE of G exists, or $+\infty$ otherwise.

Indeed, every $\mathrm{CAFE}(N, G)$ is a $\mathrm{CAFE}^1(N, G)$, so the pointwise CAFE number is always less than or equal to the CAFE number of a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$. That



a $\mathrm{CAFE}(10, \hat{G})$:

$$\boxed{\begin{array}{ccccc}
0 & 0 & 2 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 \\
0 & 2 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 \\
1 & 1 & 2 & 1 & 1 \\
1 & 2 & 1 & 0 & 1 \\
2 & 1 & 1 & 1 & 1 \\
2 & 2 & 2 & 1 & 1 \\
1 & 1 & 1 & 0 & 0
\end{array}}$$

Figure 2.4: The forbidden edges graph $G \in \mathcal{G}_{(3,3,3,2,2)}$ of the mobile phone product line testing problem given in Tables 2.1 and 2.2, and a $\mathrm{CAFE}(10, \hat{G})$

is $\mathrm{CAFEN}^1(G) \leq \mathrm{CAFEN}(G)$. Furthermore, a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ may not be consistent; nevertheless, there still may exist a pointwise CAFE for G.

We conclude this section with a concrete example based on the mobile phone testing problem given in Table 2.1, along with the constraints of Table 2.2, omitting constraint (C7) so that the only forbidden interactions are pairwise interactions. The graph in Figure 2.4 with the solid edges represents the forbidden edges graph $G \in \mathcal{G}_{(3,3,3,2,2)}$ of the mobile phone product line testing problem. The dotted edges $\{v_{1,2}, v_{5,0}\}$ and $\{v_{3,2}, v_{5,0}\}$ correspond to interactions which are not consistent with G. For example, if we try to cover the interaction $I = \{(1,2),(5,0)\}$ with a row $R = (R(1), ..., R(5))$, then we must have $R(1) = 2$ and $R(5) = 0$ and R must form a 5-tuple avoiding G. Since $\{v_{1,2}, v_{4,0}\} \in E(G)$, we cannot have $R(4) = 0$, otherwise R would not avoid G. On the other hand, since $\{v_{4,1}, v_{5,0}\} \in E(G)$, we cannot have

$R(4) = 1$ either. Thus, we have no way to cover I and still avoid G. Similarly, we cannot cover the interaction $\{(3,2),(5,0)\}$, so the avoidance closure $\hat{G}$ requires the dotted edges to be present. The array in Figure 2.4 is a CAFE$(10, \hat{G})$ and is taken from Danziger et. al. [11].

2.4 Bounds for CAFEs

In this section, we look at upper and lower bounds for the CAFE number of a consistent forbidden edges graph. We also examine the effects of adding or removing forbidden interactions.

Proposition 2.4.1 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ be a consistent forbidden edges graph. Let $E^{i,j}(G)$ denote the set of edges with one end in factor i and the other end in factor j. Then

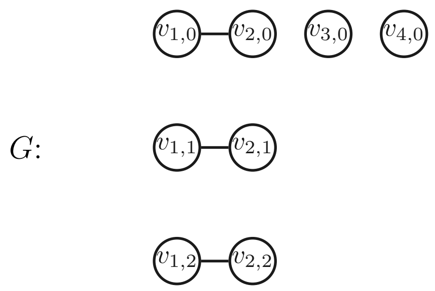
$$\max_{1\le i<j\le k} \left\{g_i g_j - |E^{i,j}(G)|\right\} \le \text{CAFEN}(G) \le \sum_{1\le i<j\le k} \left(g_i g_j - |E^{i,j}(G)|\right).$$

Proof: Every non-forbidden interaction between two factors i and j must be covered by distinct rows. This is equal to the number of non-edges between factors i and j, which equals $g_i g_j - |E^{i,j}(G)|$. So an optimal CAFE(N, G) requires at least $\max_{1\le i<j\le k}\{g_i g_j - |E^{i,j}(G)|\}$ rows.

On the other hand, an optimal CAFE(N, G) requires at most one row per non-forbidden interaction, which is equal to the total number of non-edges between every pair of distinct factors. Thus CAFEN$(G) \le \sum_{1\le i<j\le k}(g_i g_j - |E^{i,j}(G)|)$. ▣

The lower and upper bounds of Proposition 2.4.1 are attained for all forbidden edges graphs $G \in \mathcal{G}_{(g_1,g_2)}$ with only $k = 2$ factors, since in this case we have

$$\max_{1\le i<j\le k} \left\{g_i g_j - |E^{i,j}(G)|\right\} = g_1 g_2 - |E^{1,2}(G)| = \sum_{1\le i<j\le k} \left(g_i g_j - |E^{i,j}(G)|\right).$$



a CAFE$(6, G)$:
$$\boxed{\begin{array}{cccc} 0 & 1 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \end{array}}$$

Figure 2.5: A graph attaining the lower bound of Proposition 2.4.1

The graph $G \in \mathcal{G}_{(3,3,1,1)}$ attains the lower bound in Proposition 2.4.1. A CAFE for G requires at least $g_1g_2 - |E^{1,2}(G)| = 6$ rows, and this is sufficient.

In Section 4.3, we prove that the upper bound is never attained by any consistent forbidden edges graph with $k \geq 3$ factors. The lower bound, however, can be attained for all $k \geq 3$ by a consistent graph $G \in \mathcal{G}_{(g_1,g_2,1,\dots,1)}$ such that all of its forbidden interactions lie between factors 1 and 2. See Figure 2.5 for a particular example.

Now let us consider the effects of the number of edges in a forbidden edges graph. For a given family of forbidden edges graphs, $\mathcal{G}_{(g_1,\dots,g_k)}$, we show that increasing the number of forbidden edges may increase or decrease the CAFE number. Using the same notation as in Proposition 2.4.1, for a graph $G \in \mathcal{G}_{(g_1,\dots,g_k)}$ let $E^{i,j}(G)$ denote the set of edges with one end in factor i and the other end in factor j. Then the total number of forbidden interactions is

$$|E(G)| = \sum_{1\leq i<j\leq k} |E^{i,j}(G)| \leq \sum_{1\leq i<j\leq k} g_ig_j.$$

If $|E(G)| = 0$, that is, if G is empty, then we are reduced to a mixed covering array problem and $\text{CAFEN}(G) = \text{MCAN}(2, k, (g_1, \dots, g_k))$. If, on the other hand $G = K_{(g_1,\dots,g_k)}$, then we have no non-forbidden interactions to cover and so $\text{CAFEN}(K_{(g_1,\dots,g_k)}) = 0$. Thus, increasing the number of forbidden interactions can decrease the CAFE number.

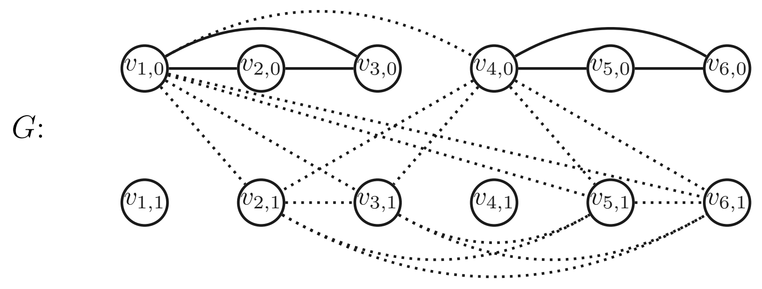


Figure 2.6: A graph $G \in \mathcal{G}_{6,2}$ with $\text{CAFEN}(G) \geq \lfloor k^2/4 \rfloor = 9$.
The graph $G \in G_{6,2}$ has two cliques $C_1 = \{v_{1,0}, v_{2,0}, v_{3,0}\}$ and $C_2 = \{v_{4,0}, v_{5,0}, v_{6,0}\}$. The dotted edges show the row that is forced when covering the zero-zero interaction $\{v_{1,0}, v_{4,0}\}$. Similarly, each interaction between vertices of C_1 and C_2 forces a distinct row which covers no other zero-zero interaction. Thus, we require $\lfloor k^2/4 \rfloor = 9$ rows simply to cover the zero-zero interactions of G.

For fixed alphabet size g, if $G \in \mathcal{G}_{k,g}$ is consistent, then the upper bound of Proposition 2.4.1 becomes

$$\text{CAFEN}(G) \leq g^2 \binom{k}{2} - |E(G)|.$$

At worst, this bound is quadratic on k. Indeed, Danziger et. al. [11] provide an example of a graph $G \in \mathcal{G}_{k,g}$ for which $\text{CAFEN}(G) \geq \lfloor \frac{k^2}{4} \rfloor$. The graph consists of two cliques, $C_1 = \{v_{i,0} \in V(G) | 1 \leq i \leq \lfloor k/2 \rfloor\}$ and $C_2 = \{v_{i,0} \in V(G) | \lfloor k/2 \rfloor + 1 \leq i \leq k\}$. Covering the non-forbidden interactions of the form $\{v_{i,0}, v_{j,0}\}$ where $v_{i,0} \in C_1$ and $v_{j,0} \in C_2$ requires $\lfloor \frac{k}{2} \rfloor \left(k - \lfloor \frac{k}{2} \rfloor\right) \geq \lfloor \frac{k^2}{4} \rfloor$ rows. See Figure 2.6 for a particular example of this construction. Note that for an empty graph $G' \in \mathcal{G}_{k,g}$ we have $\text{CAFEN}(G') = \text{CAN}(2, k, g) \sim \frac{g}{2} \log_2 k$, as $k \to \infty$ (see Theorem 1.2.5). Thus, the construction given above demonstrates how adding forbidden interactions to a $\text{TP}(k, g)$ can indeed greatly increase the number of tests required.

We now extend the construction of this example to yield a graph $G \in \mathcal{G}_{k,g}$ with $\text{CAFEN}(G) \geq (g-1)^2 \lfloor \frac{k^2}{4} \rfloor$, in order to demonstrate how an increase in the

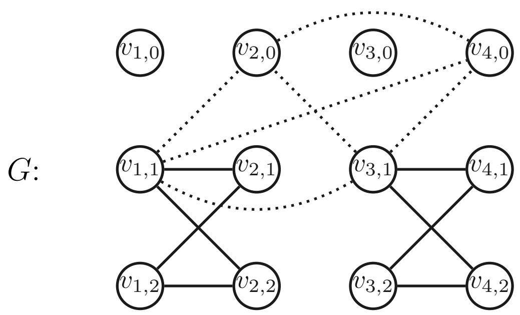


Figure 2.7: A graph $G \in \mathcal{G}_{4,3}$ with $\mathrm{CAFEN}(G) \geq (g-1)^2\lfloor k^2/4 \rfloor = 16$.
The graph $G \in \mathcal{G}_{4,3}$ has two complete equipartite graphs, G_1 and G_2, where $V(G_1) = \{v_{1,1}, v_{2,1}, v_{1,2}, v_{2,2}\}$ and $V(G_2) = \{v_{3,1}, v_{4,1}, v_{3,2}, v_{4,2}\}$. The dotted edges show the row that is forced when we cover interaction $\{(1,1),(3,1)\}$. Similarly, each interaction between vertices of G_1 and G_2 forces a distinct row which covers no other interaction between vertices of G_1 and vertices of G_2. Thus, we require $(g-1)^2\lfloor k^2/4 \rfloor = 16$ rows simply to cover the "G_1-G_2" interactions of G.

number of forbidden interactions can increase the CAFE number. Rather than G having two cliques, we let G have two complete equipartite graphs, G_1 and G_2 with $V(G_1) = \{v_{i,a_i} \in V(G) | 1 \leq i \leq \lfloor k/2 \rfloor\}$ and $V(G_2) = \{v_{i,a_i} \in V(G) | \lfloor k/2 \rfloor + 1 \leq i \leq k\}$. The partite sets of G_1 are of the form $P_i = \{v_{i,a_i} \in V(G_1) | a_i \in [1, g-1]\}$ for $1 \leq i \leq \lfloor k/2 \rfloor$. Similarly, the partite sets of G_2 are of the form $P_i = \{v_{i,a_i} \in V(G_2) | a_i \in [1, g-1]\}$ for $\lfloor k/2 \rfloor + 1 \leq i \leq k$. Then every row that covers an interaction $\{(i, a_i), (j, a_j)\}$ between vertices of G_1 and vertices of G_2 is forced to use the zero values for all $l \in [1,k] \setminus \{i,j\}$. There are $\lfloor \frac{k}{2} \rfloor (g-1) \left(k - \lfloor \frac{k}{2} \rfloor\right)(g-1) = (g-1)^2 \lfloor \frac{k^2}{4} \rfloor$ such interactions. Thus, $\mathrm{CAFEN}(G) \geq (g-1)^2 \lfloor \frac{k^2}{4} \rfloor$. See Figure 2.7 for a particular example.

2.5 A Recursive Construction for CAFEs

In the following theorem, Danziger et. al. [11] give a recursive construction for building CAFEs. Assuming that the forbidden edges graph can be broken up into several subgraphs of a particular form, the construction in Theorem 2.5.1 builds a CAFE based on these smaller subgraphs.

Theorem 2.5.1 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{(g_1,...,g_k)}$ be a forbidden edges graph that can be decomposed into s subgraphs $G_1, ..., G_s$ in the following way. Let $F_1, ..., F_s$ denote disjoint subsets of factors $1, ..., k$, so that $F_i \subseteq [1, k]$, and $F_i \cap F_j = \emptyset$ for $i \neq j$. We assume that the subgraph G_i is associated to the factor subset F_i, so that G_i contains all vertices from each of the factors in F_i. In other words, G_i is the induced subgraph $G_i = G[\{v_{f_i,a_{f_i}} \in V(G) | f_i \in F_i, a_{f_i} \in [0, g_{f_i} - 1]\}]$. Moreover, we assume that all edges of G lie within the subgraphs G_i for $1 \leq i \leq s$. Thus, if $e \in E(G)$, then both ends of e are contained in factors from some F_i. Let $k_i = |F_i|$ for $1 \leq i \leq s$. It is possible that $F_1 \cup \cdots \cup F_s \neq [1, k]$, so we let $l = |[1, k] \setminus (F_1 \cup \cdots \cup F_2)|$. If $l > 0$, then without loss of generality, we assume that $[1, l] = [1, k] \setminus (F_1 \cup \cdots \cup F_s)$, and we let $F_{s+1} = [1, l]$ and denote by G_{s+1} the subgraph of G induced by the vertices of factors $1, ..., l$. In this case, G_{s+1} is an empty graph. If the following designs exist:

1. $P_1, ..., P_s$, where each P_i is a CAFE$^1(p_i, G_i)$ with p_i rows and k_i columns;
2. $A_1, ..., A_s$, where each A_i is an $a_i \times k_i$ array such that the array C_i obtained by appending the rows of P_i and A_i forms a CAFE$(p_i + a_i, G_i)$;
3. P_{s+1}: an MCA$(p_{s+1}; 1, l, (g_1, ..., g_l))$ (possibly $p_{s+1} = 0$ and P is empty if $l = 0$);
4. A_{s+1}: an $a_{s+1} \times l$ array, such that the array C_{s+1} obtained by appending the rows of P_{s+1} and A_{s+1} is an MCA$(p_{s+1} + a_{s+1}; 2, l, (g_1, ..., g_l))$ (possibly $a_{s+1} = 0$ if $l = 0$);
5. M: an MCA$(m; 2, s+1, (p_1, ..., p_s, p_{s+1}))$ if $l > 0$; otherwise, if $l = 0$, then M is an MCA$(m; 2, s, (p_1, ..., p_s))$;

then there exists a CAFE(n, G) where $n = m + \max\{a_1, ..., a_s, a_{s+1}\}$.

Proof: We build an array C with its first set of rows consisting of the arrays $A_1, ..., A_{s+1}$, pasted side by side. We add arbitrary repeated rows to the subarrays in order to complete $\max\{a_1, ..., a_{s+1}\}$ rows of C.

We complete the array C by adding m rows in the following way. For each row $R_i = (R_i(1), ..., R_i(s+1))$ of M, we form row $R_{i+\max\{a_1,...,a_{s+1}\}}$ of C, by pasting side by side row $R_{R_i(1)}$ of P_1, row $R_{R_i(2)}$ of P_2,..., row $R_{R_i(s)}$ of P_s, and row $R_{R_i(s+1)}$ of P_{s+1}. Thus, C is an $n \times k$ array where $n = m + \max\{a_1, ..., a_{s+1}\}$.

First we show that the subarray C_i of C, corresponding to the columns of factors in F_i, forms a CAFE(n, G_i) for $i \in [1, s]$. By construction, C_i contains every row of A_i. Moreover, since column i of M covers every symbol from $[1, p_i]$ at least once each, we see that the last m rows of C_i must contain each row of P_i at least once each. By design of the arrays A_i and P_i, the subarray C_i forms a CAFE(n, G_i). Similarly, if $l > 0$, then subarray C_{s+1} is an MCA$(n; 2, l, (g_1, ..., g_l))$ by design, which is equivalent to a CAFE(n, G_{s+1}) since G_{s+1} is empty.

Since every edge of G is an edge of one of the subgraphs $G_1, ..., G_s$, and the subarrays $C_1, ..., C_s$ are CAFEs for $G_1, ..., G_s$, respectively, we see that every row of C avoids G. Moreover, the subarrays C_i cover all the non-forbidden interactions of G that lie entirely within any one of the subgraphs $G_1, ..., G_s$. Similarly, if $l > 0$, then C_{s+1} covers all the non-forbidden interactions of G that lie in G_{s+1}.

It remains to show that C covers all the non-forbidden interactions of the form $\{v_{i,a_i}, v_{j,a_j}\}$ where $i \in F_x$ and $j \in F_y$ and $x \neq y$. Indeed, we show that the last m rows of C are sufficient to cover all such interactions. Let $\{v_{i,a_i}, v_{j,a_j}\}$ be a non-forbidden interaction such that $i \in F_x$ and $j \in F_y$ for some $x \neq y$. Then $v_{i,a_i} \in V(G_x)$ so there is a row, say R_{i_0}, of P_x that covers the point v_{i,a_i}. Similarly, since $v_{j,a_j} \in V(G_y)$, there is a row, say row R_{j_0}, of P_y that covers v_{j,a_j}. We have $i_0 \in [1, p_x]$ and $j_0 \in [1, p_y]$, so there exists a row of M, say row R_w, such that $R_w(x) = i_0$ and $R_w(y) = j_0$.

Thus, row $R_{w+\max\{a_1,\ldots,a_{s+1}\}}$ of C contains sub-rows R_{i_0} and R_{j_0} simultaneously, hence covers both v_{i,a_i} and v_{j,a_j}. Since $\{v_{i,a_i}, v_{j,a_j}\}$ was arbitrary, we conclude that C is a CAFE(n, G). ▣

Remark 2.5.2 If $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ has subgraphs $G_1, \ldots, G_s$ as described in Theorem 2.5.1, then the arrays A_i and P_i of the construction exist so long as CAFEN$(G) \neq +\infty$. If we take a CAFE(n, G), say C, then for $i \in [1, s]$, the columns of C corresponding to the factors in F_i must form a CAFE(n, G_i). Furthermore, a CAFE(n, G_i) is also a CAFE$^1(n, G_i)$, possibly having some unnecessary rows. The MCAs required for the construction also always exist, though they may be empty depending on the value of l.

Indeed, Theorem 2.5.1 can be applied to any consistent graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$, using the following definition.

Definition 2.5.3 Let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$. We call G **factor-connected** if the graph $G^{|}$ is connected. The connected components of $G^{|}$ are called the **factor-connected components** of G.

Clearly, the non-trivial factor-connected components of $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ can be used in Theorem 2.5.1 as the subgraphs $G_1, \ldots, G_s$.

Corollary 2.5.4 (Danziger et. al. [11]) Let $g \geq 2$ be a fixed integer, and let $\mathscr{G}$ be a family of consistent forbidden edges graphs such that: for all $G \in \mathscr{G}$, we have $G \in \mathcal{G}_{k,g}$, and all the edges of G have both ends in factors in $F_1 \subseteq [1, k]$, where we let $k_1 = |F_1|$, and let G_1 be the subgraph of G on factors in F_1. Then as $k \to \infty$,

$$\frac{g}{2}\log_2(k - k_1) \leq \text{CAFEN}(G) \leq h(k_1, k) \sim \max\left\{\frac{g^2}{2}{k_1}^2, \frac{g}{2}\log_2(k - k_1)\right\}. \quad (2.5.1)$$

In particular,

1. CAFEN$(G) = O({k_1}^2 + \log k) = O(|E(G)|^2 + \log k)$.

2. If $k_1 = o(\sqrt{\log k})$ (or, if $|E(G)|^2 = o(\sqrt{\log k})$), then $\mathrm{CAFEN}(G) \sim \frac{g}{2}\log_2 k$.

Moreover, as $k \to \infty$, if $\mathrm{CAFEN}(G_1) = O(f(k_1))$, then $\mathrm{CAFEN}(G) = O(f(k_1) + \log k)$.

Proof: For each $G \in \mathscr{G}$ we apply Theorem 2.5.1 with $s = 1$. In this case, we have $l = k - k_1$. For the array P_1, we get $p_1 \leq gk_1$, since an upper bound on $\mathrm{CAFEN}^1(G_1)$ is the total number of vertices of G_1. As for P_2, an $\mathrm{MCA}(p_2; 1, l, (g, ..., g))$ is simply a $\mathrm{CA}(p_2; 1, l, g)$. Since $\mathrm{CAN}(1, l, g) = g$ we can take $p_2 = g$. For the array M, we have $m \leq g^2 k_1$, as $\mathrm{MCAN}(2, 2, (p_1, p_2)) = p_1 p_2 \leq (gk_1)g$.

Using the upper bound in Proposition 2.4.1, for the array A_1, we have

$$a_1 \leq \mathrm{CAFEN}(G_1) \leq g^2 \binom{k_1}{2} < g^2 \frac{{k_1}^2}{2}.$$

Finally, for the array A_2, we have $a_2 \leq \mathrm{MCAN}(2, l, (g, ..., g)) = \mathrm{CAN}(2, l, g)$. Since $l = k - k_1$, we have $a_2 \leq \mathrm{CAN}(2, k - k_1, g) \sim \frac{g}{2}\log_2(k - k_1)$, by Theorem 1.2.5. Therefore, as $k \to \infty$, we have

$$\begin{aligned}\mathrm{CAFEN}(G) &\leq m + \max\{a_1, a_2\} \\ &\leq g^2 k_1 + \max\left\{g^2 \frac{{k_1}^2}{2}, a_2\right\} \\ &\sim \max\left\{g^2 \frac{{k_1}^2}{2}, \frac{g}{2}\log_2(k - k_1)\right\},\end{aligned}$$

which proves the upper bound in Equation 2.5.1. The lower bound follows from the fact that $\frac{g}{2}\log_2(k - k_1) \sim a_2 \leq \mathrm{CAFEN}(G)$.

If $k_1 = o(\sqrt{\log k})$ (or, if $|E(G)|^2 = o(\sqrt{\log k})$), then $\max\left\{g^2\frac{{k_1}^2}{2}, \frac{g}{2}\log_2(k - k_1)\right\} \sim \frac{g}{2}\log_2 k$, which proves 2. Since G_1 can be chosen so that $k_1 - 1 \leq |E(G)|$, we have

$$\mathrm{CAFEN}(G) \sim \max\left\{g^2 \frac{{k_1}^2}{2}, \frac{g}{2}\log_2(k - k_1)\right\} = O({k_1}^2 + \log_2 k) = O(|E(G)|^2 + \log_2 k),$$

which proves 1. If $\mathrm{CAFEN}(G_1) = O(f(k_1))$, then using a similar proof as for the upper bound in Equation 2.5.1, as $k \to \infty$, we get

$$\mathrm{CAFEN}(G) = O\left(\max\left\{f(k_1), \frac{g}{2}\log_2(k - k_1)\right\}\right) = O(f(k_1) + \log_2 k),$$

which proves the last statement. ▣

2.6 Error-Locating Arrays

In this section, we look at error-locating arrays (ELAs), another type of array that is closely related to CAFEs and was introduced by Martinez et. al. [28]. As with MCAs, we can consider ELAs of various strengths $t \geq 1$; however, for our purposes, we restrict the definitions and results that follow to the case where $t = 2$.

Given a $\mathrm{TP}(k, (g_1, ..., g_k))$, suppose that all **errors** (interactions which are responsible for the failure of any test that they are covered by) correspond to faulty pairwise interactions. Our goal is to discover these errors, so we would like to design a test suite having the following properties. For each faulty interaction, $I = \{(i, a_i), (j, a_j)\}$, every test that covers I is a failing test. Moreover there exists a failing test, $T_I = (a_1, ..., a_k) \in [0, g_1 - 1] \times \cdots \times [0, g_k - 1]$, such that T_I covers I and for each pairwise interaction, $I' \neq I$, that is covered by T_I, there exists a passing test T' that covers I'. Such a test suite allows us to deduce which of the pairwise interactions of the $\mathrm{TP}(k, (g_1, ..., g_k))$ correspond to errors.

We use a graph $G \in \mathcal{G}_{(g_1,...,g_k)}$ to encode the faulty pairwise interactions. The edges of G correspond to pairwise interactions which are faulty and result in failing test outcomes whenever covered. Unlike the forbidden edges graphs associated to CAFEs, the edges of the error-encoding graphs associated to ELAs are *unknown* to us in advance. These edges can only be discovered if there exists a suite of passing/failing tests that form an ELA for G, for which we give the definition below.

Definition 2.6.1 Let $G \in \mathcal{G}_{(g_1,...,g_k)}$. A k-tuple $T = (T_1, ..., T_k) \in [0, g_i - 1] \times \cdots \times [0, g_k - 1]$ is said to **locate** interaction $I = \{(i, a_i), (j, a_j)\}$ if $T_i = a_i$ and $T_j = a_j$, and for every other interaction $\{(p, a_p), (q, a_q)\} \neq I$ that T covers we have $\{v_{p,a_p}, v_{q,a_q}\} \notin$

$E(G)$. In this case we say that interaction I is **located by** T.

An **error-locating array** (ELA) for a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$, denoted by $\mathrm{ELA}(n, G)$, is an $n \times k$ array A, with each column i having symbols from the alphabet $[0, g_i - 1]$, such that every interaction $\{(i, a_i), (j, a_j)\}$ corresponding to a pair of vertices $v_{i,a_i}, v_{j,a_j} \in V(G)$ with $i \neq j$ (corresponding to an edge, or a non-edge of G) is located by a k-tuple corresponding to some row of A.

If for some $n \in \mathbb{Z}$ there exists an $\mathrm{ELA}(n, G)$, then we say that G is **locatable**. The **ELA number** of G, denoted by $\mathrm{ELAN}(G)$, is the smallest N such that an $\mathrm{ELA}(N, G)$ exists, if there exists an ELA for G, or $+\infty$ otherwise.

As with CAFEs, ELAs require rows to cover all pairwise interactions corresponding to the non-edges of G. Equivalently, there is a subset of the rows of an $\mathrm{ELA}(n, G)$ that actually form a $\mathrm{CAFE}(n', G)$. In contrast to CAFEs, ELAs also require rows to locate each pairwise interaction corresponding to an edge of G. The next result gives the precise relationship.

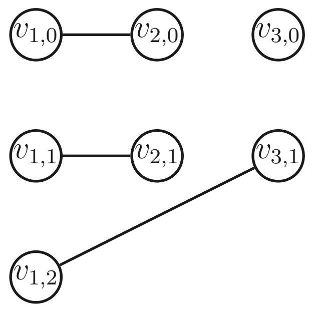
Theorem 2.6.2 (Danziger et. al. [11]) If $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ is locatable, then

$$\mathrm{ELAN}(G) = \mathrm{CAFEN}(G) + |E(G)|.$$

Proof: The rows of an optimal $\mathrm{ELA}(N, G)$ can be classified into two sets. One set of rows locates individually each interaction $\{v_{i,a_i}, v_{j,a_j}\} \in E(G)$. Thus we require exactly $|E(G)|$ such rows. The other set of rows locates all interactions $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$. Clearly, the second set of rows is equivalent to an optimal $\mathrm{CAFE}(N', G)$. ▣

It is possible for a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ to be consistent, but not locatable. Therefore, we can have $\mathrm{CAFEN}(G) = N \neq +\infty$, but $\mathrm{ELAN}(G) = +\infty$. The next result give us a sufficient condition for a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ to be locatable. It uses the definition of *safe values* given next.

G:



an ELA$(9, G)$:

0	1	0
1	0	0
1	0	1
0	1	1
2	0	0
2	1	0
0	0	0
1	1	1
2	1	1

Figure 2.8: An error-encoding graph $G \in \mathcal{G}_{(3,2,2)}$ and an ELA$(9, G)$.

Definition 2.6.3 [28] A graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ is said to have **safe values**, if for every factor $i \in [1, k]$ there exists a vertex $v_{i,s_i} \in V(G)$ such that v_{i,s_i} is an isolated vertex of G. The values $s_1, \ldots, s_k$ are called **safe values** of G.

Proposition 2.6.4 (Martinez et. al. [28]) Let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ have safe values $s_1, \ldots, s_k$. Then G is locatable.

Proof: Take any pair of vertices $v_{i,a_i}, v_{j,a_j} \in V(G)$ such that $i \neq j$. Then a k-tuple $T = (T_1, \ldots, T_k)$ defined by

$$T_p = \begin{cases} s_p, & \text{if } p \in [1, k] \setminus \{i, j\}; \\ a_i, & \text{if } p = i; \\ a_j, & \text{if } p = j; \end{cases}$$

clearly locates the interaction corresponding to v_{i,a_i}, v_{j,a_j}. If we build an array with one row like T per pair $v_{i,a_i}, v_{j,a_j} \in V(G)$, then this array forms an ELA for G and so G is locatable. ▣

In Figure 2.8, we give an example of an error-encoding graph $G \in \mathcal{G}_{(3,2,2)}$ and an

ELA$(9, G)$. As the graph G in this figure contains no safe values in factor 1, but is nonetheless locatable, we see that the converse of Proposition 2.6.4 is not true.

Chapter 3

Edge Clique Covers

In this chapter we look at edge clique covers of graphs. We give the definition of an edge clique cover and give some results on upper bounds of the edge clique cover number, as well as some other properties in Section 3.1. We investigate several other types of edge clique covers in Sections 3.2, 3.3, and 3.4, namely edge-disjoint, uniform, and partial edge clique covers, respectively. In particular, in Section 3.3, we give new results regarding uniform edge clique covers. In Section 3.4, we define two new types of partial edge clique covers and give some initial results for these problems. Finally, in Section 3.5, we look at the relationship between the node clique cover problem and the edge clique cover problem, as well as their relationship to the vertex colouring problem.

3.1 Edge Clique Covers

Here we study edge clique covers. In particular, we present several results regarding upper bounds on the minimum number of cliques required to form an edge clique cover, as well as various other properties of interest. We begin by giving the definition of an edge clique cover.

Definition 3.1.1 Let G be a simple graph. If C is a clique of G and e is an edge of G, we say that the clique C **covers** e if the ends of e belong to C. That is, if $e = \{u, v\}$ and $u, v \in C$, then C covers e. If $\mathfrak{C} = \{C_1, ..., C_N\}$ is a collection of N cliques of G such that for every edge $e \in E(G)$ there is at least one clique $C_i \in \mathfrak{C}$ that covers e, then we say that $\mathfrak{C}$ is an **edge clique cover** (ECC) of G. We say that an ECC of G, $\mathfrak{C}$, is **optimal** if there is no ECC of G, say $\mathfrak{C}'$, such that $|\mathfrak{C}'| < |\mathfrak{C}|$. The number of cliques in an optimal ECC of G is called the **ECC number** of G, and we denote it by $\theta'(G)$.

Remark 3.1.2 Clearly every graph G admits an edge clique cover since we can form an ECC of G by taking the collection of edges, namely $\mathfrak{C} = \{\{u, v\} \subseteq V(G) | \{u, v\} \in E(G)\}$. Clearly, every edge of G is covered by the 2-clique in $\mathfrak{C}$ containing its ends. Thus we always have $\theta'(G) \leq |E(G)|$.

The following result from Erdös, Goodman and Posá [12] gives us another upper bound on the ECC number of a graph. The proof requires the next lemma.

Lemma 3.1.3 For any positive integer n, we have $\lfloor (n+2)^2/4 \rfloor = \lfloor n^2/4 \rfloor + n + 1$.

Theorem 3.1.4 (Erdös, Goodman, and Pósa [12]) Let G be a simple graph on $n \geq 2$ vertices. Then $\theta'(G) \leq \lfloor n^2/4 \rfloor$ and an ECC attaining this bound need only contain 2-cliques and 3-cliques.

Proof: We prove the result by induction on n, the number of vertices of the graph. For the base case when n is even, let G be a graph on $n = 2$ vertices. Either $G = K_2$, in which case $\theta'(G) = 1 = \lfloor 2^2/4 \rfloor$; otherwise, $G = \overline{K_2}$, in which case $\theta'(G) = 0 < \lfloor 2^2/4 \rfloor$. For the odd base case, take $n = 3$. If $G = K_3$, then $\theta'(G) = 1 \leq 2 = \lfloor 3^2/4 \rfloor$. Otherwise, G is a graph on 3 vertices with at most 2 edges in which case $\theta'(G) \leq |E(G)| \leq 2 \leq \lfloor 3^2/4 \rfloor$.

Now for the induction step we assume the statement holds for all graphs on n vertices and prove that it must also hold for all graphs on $n + 2$ vertices. Let G be a graph on $n+2$ vertices. If G contains no edges, then $\theta'(G) = 0 \leq \lfloor n^2/4 \rfloor$. Otherwise, G must contain at least one edge, in which case we can label the vertices of G as $x_1, x_2, ..., x_{n+2}$, so that in particular, the vertices x_1 and x_2 are joined by an edge in G. Now, consider $G[V']$, the subgraph of G induced by the vertex set $V' = \{x_3, ..., x_{n+2}\}$. Clearly, $G[V']$ is a graph on n vertices. By our induction hypothesis, at most $\lfloor n^2/4 \rfloor$ cliques are required to cover all the edges of $G[V']$.

By Lemma 3.1.3, we need only show that the remaining edges of G can be covered by at most $n+1$ edges. Consider the vertex $x_i \in V' = \{x_3, ..., x_{n+2}\}$. If x_i is adjacent to both x_1 and x_2 in G, then we cover those edges by the 3-clique $C_i = \{x_i, x_1, x_2\}$. If x_i is adjacent to one of x_1 or x_2, but not both, then we cover the edge $\{x_i, x_1\}$ (or $\{x_i, x_2\}$) by the 2-clique $C_i = \{x_i, x_1\}$ (or $C_i = \{x_i, x_2\}$, respectively). If x_i is not adjacent to either x_1 or x_2 in G, then there is no edge to cover.

Hence, for $i = 3, ..., n+2$, we require at most n cliques, C_i, to cover the edges of G joining vertices of V' to x_1 or x_2. One last clique, namely $\{x_1, x_2\}$, may possibly need to be added in order to ensure that the edge $\{x_1, x_2\}$ is covered. In total, we require at most $\lfloor n^2/4 \rfloor + n + 1$ cliques. Therefore, by Lemma 3.1.3, we have $\theta'(G) \leq \lfloor (n+2)^2/4 \rfloor$. ▣

In fact, Erdös, Goodman, and Pósa [12] also prove that the number $\lfloor n^2/4 \rfloor$ in Theorem 3.1.4 cannot be replaced by any smaller number. For n even, consider the complete bipartite graph $K_{k,2}$ on $n = 2k$ vertices. It is easy to see that $K_{k,2}$ does not contain any clique larger than size 2, hence $\theta'(K_{k,2}) = |E(K_{k,2})| = k^2 = \left(\frac{n}{2}\right)^2 = \lfloor n^2/4 \rfloor$. Similarly, for n odd, the complete bipartite graph $K_{(k,k+1)}$ on $n = 2k+1$ vertices satisfies $\theta'(K_{(k,k+1)}) = k^2 + k = \lfloor n^2/4 \rfloor$.

A slight improvement to the above result is given by Lovász [26]. The improvement is given when we know the number of edges in the graph.

Theorem 3.1.5 (Lovász, [26]) Let G be a simple graph on n vertices. Suppose

$r = \binom{n}{2} - |E(G)|$ and t is the greatest natural number such that $t^2 - t \leq r$. Then $\theta'(G) \leq r + t$.

Proof: Let G be a simple graph on n vertices and ε edges. Clearly, if $\varepsilon = 0$, then the statement holds. Assume, therefore, that $\varepsilon > 0$. Let C_1 be a clique of G of maximum size, and for $i \geq 2$, let C_{i+1} be a clique of maximum size of the graph $G - C_1 - \cdots - C_i$. Let a_i denote the number of vertices of C_i and let $C_1, C_2, ..., C_p$ be the nonempty graphs defined above. Let q be the index for which $a_q \geq 2$ and $a_{q+1} < 2$ (this index exists by assumption that $\varepsilon > 0$).

For a given C_i and vertex $x \in V(G)$, if $x \notin C_i$, then let $S_{i,x}$ denote the set of vertices of C_i which are adjacent to x in G. Clearly $S_{i,x} \cup \{x\}$ is a clique of G since any two vertices in C_i are adjacent and x is adjacent to every vertex in $S_{i,x} \subseteq C_i$. Let $C_{i,x}$ denote the clique $S_{i,x} \cup \{x\}$. Then the collection of cliques

$$\mathfrak{C} = \{C_i | 1 \leq i \leq q\} \cup \{C_{i,x} | x \in C_j, 1 \leq i < j \leq p \text{ and } |S_{i,x}| > 0\}$$

forms an ECC of G. For a given index $i \geq 1$ there are at most $a_{i+1} + ... + a_p$ vertices in the cliques C_j such that $i < j \leq p$. Therefore, we have

$$|\{C_{i,x} | x \in C_j, i < j \leq p \text{ and } |S_{i,x}| > 0\}| \leq a_{i+1} + a_{i+2} + ... + a_p$$

for $1 \leq i \leq p$. Summing over all $i \in [1, p]$, there are at most $a_2 + 2a_3 + ... + (p-1)a_p$ cliques of the form $C_{i,x}$ as defined above. Therefore,

$$|\mathfrak{C}| \leq q + a_2 + 2a_3 + ... + (p-1)a_p.$$

By maximality of each C_i, if $i < j$ and $x \in C_j$ there must be at least one vertex of C_i that is not adjacent to x; otherwise, C_i could have been extended to include x in the first place. For every $j > i$ there are at least a_j non-edges between each vertex of C_j and the vertices of C_i. In other words the number of non-edges of G must be at least $a_2 + 2a_3 + ... + (p-1)a_p$. Since $r = \binom{n}{2} - \varepsilon$ is the number of non-edges of G we must have $r \geq a_2 + 2a_3 + ... + (p-1)a_p$. Therefore $|\mathfrak{C}| \leq q + r$.

Furthermore, since q is the index chosen so that $a_i \geq 2$ for $1 \leq i \leq q$ we have

$$\begin{aligned} r &\geq a_2 + 2a_3 + ... + (q-1)a_q && \text{since } p \geq q; \\ &\geq 2 + 2(2) + ... + (q-1)2 && \text{since } a_i \geq 2 \text{ for } 1 \leq i \leq q; \\ &= 2[1 + 2 + ... + (q-1)] \\ &= q(q-1). \end{aligned}$$

Clearly the choice of t to be the greatest positive integer such that $t^2 - t \leq r$ implies $q(q-1) \leq t(t-1) \leq r$ and so $q \leq t$. We conclude that $|\mathfrak{C}| \leq r+t$ so $\theta'(G) \leq r+t$. ▣

As a result, if we know for a graph G on n vertices that $|E(G)| \leq \lfloor n^2/4 \rfloor$, then it is easy to see that we get $r + t \geq |E(G)|$. Thus, the bound becomes $\theta'(G) \leq |E(G)|$ by Remark 3.1.2. Otherwise, if $|E(G)| > \lfloor n^2/4 \rfloor$, we have $r + t < \lfloor n^2/4 \rfloor$, hence the bound $\theta'(G) \leq r + t$ is an improvement on $\theta'(G) \leq \lfloor n^2/4 \rfloor$.

The next results give us a few basic properties of ECCs. They are taken from a paper by Orlin [31] in which the author refers to ECCs as *R-content decompositions of graphs* and the ECC number is referred to as the *R-content* of the graph. However, to be consistent with the notation used throughout this thesis, we paraphrase these results in terms of ECCs and $\theta'(G)$.

Proposition 3.1.6 (Orlin [31]) If G is a simple graph and $V' \subseteq V(G)$, then $\theta'(G) \geq \theta'(G[V'])$.

Proof: Let $\mathfrak{C} = \{C_1, C_2, ..., C_n\}$ be an edge clique cover of G. Let $\mathfrak{C}' = \{C_1 \cap V', C_2 \cap V', ..., C_n \cap V'\}$. It follows that $\mathfrak{C}'$ is an ECC of G and hence $\theta'(G[V']) \leq \theta'(G) \leq n$. ▣

Now we come to an easy yet important result which tells us that for the purpose of determining the ECC number of a graph, we may restrict our attention to ECCs in which each clique is maximal with respect to set inclusion.

Proposition 3.1.7 (Orlin [31]) Let G be a simple graph. Then there exists an optimal ECC of G, $\mathfrak{C}'$, such that every clique $C \in \mathfrak{C}'$ is maximal with respect to set inclusion.

Proof: Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an ECC of G. For each $C_i \in \mathfrak{C}$ take C_i' to be a clique of G with $C_i \subseteq C_i'$, and such that C_i' is maximal with respect to set inclusion. Then $\mathfrak{C}' = \{C_1', ..., C_N'\}$ is an ECC of G and $|\mathfrak{C}'| \leq |\mathfrak{C}|$. Moreover, if $|\mathfrak{C}| = \theta'(G)$, then $|\mathfrak{C}'| = \theta'(G)$. ▣

For convenience, if $\mathfrak{C} = \{C_1, ..., C_n\}$ is an ECC of a graph G with the added property that each clique $C_i \in \mathfrak{C}$ is maximal with respect to set inclusion, then we call $\mathfrak{C}$ a **clique-maximal** ECC of G.

Proposition 3.1.8 (Orlin [31]) Let G be a simple graph and suppose that edge $e = \{u, v\}$ belongs to a unique clique C_e of G that is maximal with respect to set inclusion. Then C_e occurs in every clique-maximal ECC of G.

Proof: The edge e must be covered by at least one clique in a clique-maximal ECC of G. By uniqueness, C_e is the only choice. ▣

Corollary 3.1.9 (Orlin [31]) Suppose a graph G has a subset of edges, $E' \subseteq E(G)$, with the property that every edge $e \in E'$ belongs to a unique maximal clique C_e. Furthermore, suppose that $\mathfrak{C} = \{C_e | e \in E'\}$ is an ECC of G. Then $\mathfrak{C}$ is the unique optimal clique-maximal ECC of G.

Proof: By Proposition 3.1.8, for each $e \in E'$ the clique C_e must be included in every clique-maximal ECC of G. Thus $\mathfrak{C} = \{C_e | e \in E'\} \subseteq \mathfrak{C}'$ where $\mathfrak{C}'$ is any optimal clique-maximal ECC of G. Consequently, we must have $|\mathfrak{C}| \leq |\mathfrak{C}'|$. Since we assume that $\mathfrak{C}$ is an ECC of G, we must have $\theta'(G) \leq |\mathfrak{C}| \leq |\mathfrak{C}'| = \theta'(G)$. We conclude that $\mathfrak{C} = \mathfrak{C}'$, thus $\mathfrak{C}$ is the unique optimal clique-maximal ECC of G. ▣

Here we give a simple lower bound on the ECC number of a graph. It is taken from Gyárfás [15] and it uses the following definition of **equivalent** vertices.

Definition 3.1.10 [15] Let G be a graph. Two distinct vertices $u, v \in V(G)$ are called **equivalent** if $\{u, v\} \in E(G)$ and if for all vertices $w \in V(G)$ such that $w \neq u$ and $w \neq v$, we have $\{w, u\} \in E(G)$ if and only if $\{w, v\} \in E(G)$.

If a graph G contains equivalent vertices, the following lemma tells us that we can identify equivalent vertices without changing the ECC number of the graph.

Proposition 3.1.11 (Gyárfás [15]) If u and v are equivalent vertices of a simple graph G and if the edge $\{u, v\}$ is not an isolated edge, then $\theta'(G) = \theta'(G - \{v\})$.

Proof: If u and v are equivalent vertices and edge $e = \{u, v\}$ is not an isolated edge, there must exist another vertex $x \in V(G)$ such that $x \neq u$, $x \neq v$, and x is adjacent to both u and v.

Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal clique-maximal ECC of the graph $G - \{v\}$. Observe that for $1 \leq i \leq N$, C_i is a clique of G. Furthermore, since $\{x, u\} \in E(G)$ and $x \neq v$, we must have $\{x, u\} \in E(G - \{v\})$. Therefore, there must exist some clique $C_j \in \mathfrak{C}$ that covers edge $\{x, u\}$. In particular, note that C_j contains vertex u.

Now, for $1 \leq i \leq N$ we define $C_i^* = \begin{cases} C_i \cup \{v\}, & \text{if } u \in C_i; \\ C_i, & \text{otherwise.} \end{cases}$

We claim that $\mathfrak{C}^* = \{C_1^*, ..., C_N^*\}$ forms an ECC of G. First observe that each C_i^* is a clique of G for $1 \leq i \leq N$. If $u \notin C_i$ then $C_i^* = C_i$ and all pairs of vertices in C_i^* are therefore adjacent in G since they are adjacent in $G - \{v\}$. Otherwise, if $u \in C_i$ then $C_i^* = C_i \cup \{v\}$ and we need to show that v must be adjacent to all vertices of C_i. Indeed, since $u \in C_i$, every other vertex in C_i is adjacent to u. By equivalence of u and v, we must have that v is adjacent to every vertex in C_i other than u, and by definition of equivalence, we must also have that v is adjacent to u. Thus every pair of vertices in C_i^* are adjacent in G, so C_i^* is a clique of G.

Now let e' be any edge of G. If v is not an end of e', then $e' \in E(G - \{v\})$. Therefore, there exists $C_i \in \mathfrak{C}$ that covers e', and consequently, $C_i^* \in \mathfrak{C}^*$ covers e'. If v is an end of e', then $e' = \{v, y\}$ for some $y \in V(G)$. If $y = u$, then clique $C_j \in \mathfrak{C}$ contains u. Consequently, $C_j^* = C_j \cup \{v\}$ covers e'. Otherwise, if $y \neq u$, we know that $e' = \{v, y\} \in E(G)$ if and only if $\{u, y\} \in E(G)$. Moreover, $\{u, y\} \in E(G - \{v\})$ so there exists some clique, say $C_l \in \mathfrak{C}$, that covers edge $\{u, y\}$. Clearly, $C_l^* = C_l \cup \{v\}$ covers edge $\{v, y\}$. We conclude that $\mathfrak{C}^*$ forms an ECC of G, so $\theta'(G) \leq N$.

By Proposition 3.1.6 we have $N = \theta'(G - \{v\}) \leq \theta'(G)$ since $V(G) \setminus \{v\} \subseteq V(G)$, and so $\theta'(G) = \theta'(G - \{v\})$. ▣

The next small result indicates that we may remove all isolated vertices of a graph without changing its ECC number.

Proposition 3.1.12 (Gyárfás [15]) Let G be a simple graph on q vertices and let n be the number of non-isolated vertices of G. Then $\theta'(G) = \theta'(G_n)$, where G_n denotes the graph obtained from G by removing all isolated vertices.

Proof: Let $G_n = G - \{v \in V(G) | d_G(v) = 0\}$. By Proposition 3.1.6, we have $\theta'(G_n) \leq \theta'(G)$. Now, let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal ECC of G_n. Let $e = \{u, v\}$ be any edge of G. Then, in particular, the ends of e are non-isolated vertices of G, so e must be an edge of G_n. Thus, there exists some clique, say $C_e \in \mathfrak{C}$, that covers e. Since e was an arbitrary edge of G, we see that $\mathfrak{C}$ is an ECC of G. Therefore, $\theta'(G) \leq N = \theta'(G_n)$. ▣

Now we come to Gyárfás' theorem which gives a lower bound on the ECC number of a graph containing no isolated vertices, and no pair of equivalent vertices.

Theorem 3.1.13 (Gyárfás [15]) Let G be a simple graph on n vertices. If G contains no isolated vertices and no equivalent vertices then $\theta'(G) \geq \log_2(n+1)$.

Proof: Let G be a simple graph on n vertices, and assume that G contains no isolated vertices and no equivalent vertices. Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an ECC of G. For each vertex $x \in V(G)$, let $I(x)$ denote the index set of the cliques of $\mathfrak{C}$ that cover the edges incident with x. Since G contains no isolated vertices, we must have $I(x) \neq \emptyset$ for all $x \in V(G)$. For $x, y \in V(G)$ such that $x \neq y$ we claim that $I(x) \neq I(y)$. If $\{x, y\} \notin E(G)$, then the claim is true since no clique in $\mathfrak{C}$ can contain both vertices x and y. Thus $I(x)$ and $I(y)$ must be disjoint. Otherwise, if $\{x, y\} \in E(G)$ then since x and y are not equivalent there exists a vertex $z \in V(G)$ such that z is adjacent to exactly one of the vertices x and y. Assume without loss of generality that z is adjacent to x but not to y. That is, $\{z, x\} \in E(G)$ and $\{z, y\} \notin E(G)$. Let $C_i \in \mathfrak{C}$ be a clique that covers edge $\{z, x\}$. Then $y \notin C_i$ since $\{z, y\} \notin E(G)$ and consequently, we have index $i \in I(x)$ but $i \notin I(y)$ which implies that $I(x) \neq I(y)$.

Therefore, the sets $I(x)$ are distinct non-empty subsets of $[1, N]$ and we have $|V(G)| \leq 2^N - 1$. In other words, $n + 1 \leq 2^N$ which proves that $\log_2(n+1) \leq N$. ▣

3.2 Edge-disjoint ECCs

There are several variations of the edge clique cover problem. Firstly, let us give the definition of an *edge-disjoint ECC* of a graph.

Definition 3.2.1 An **edge-disjoint** ECC of a graph G is an ECC of G, say $\mathfrak{C}$, such that every edge of G is covered by *exactly* one clique in $\mathfrak{C}$. If $\mathfrak{C}$ is an edge-disjoint ECC of a graph G such that for every edge-disjoint ECC, $\mathfrak{C}'$ of G we have $|\mathfrak{C}| \leq |\mathfrak{C}'|$, then we say that $\mathfrak{C}$ is **optimal**. For a given graph G, we denote the size of an optimal edge-disjoint ECC of G by $\theta^{ed}(G)$. This number is called the **edge-disjoint ECC number** of G.

As with ECCs, Remark 3.1.2 also applies to edge-disjoint ECCs since we can always cover any simple graph by its collection of edges. Obviously since each edge in such an ECC belongs only to the one clique containing its ends, we have a trivial edge-disjoint ECC for any graph. Thus, for any simple graph G we have

$$\theta^{ed}(G) \leq |E(G)|.$$

In the following theorem, Erdös, Goodman, and Pósa [12] give us another upper bound on the edge-disjoint ECC number. Their result uses the next little lemma.

Lemma 3.2.2 For any positive integer n we have $\lfloor n^2/4 \rfloor = \lfloor (n-1)^2/4 \rfloor + \lfloor n/2 \rfloor$.

Theorem 3.2.3 (Erdös, Goodman and Pósa, [12]) Any simple graph G on $n \geq 2$ vertices satisfies $\theta^{ed}(G) \leq \lfloor n^2/4 \rfloor$. Furthermore, an edge-disjoint ECC attaining this bound need only contain 2-cliques and 3-cliques.

Proof: We prove the statement by induction on n, the number of vertices. For $n = 2$ there are only two simple graphs on 2 vertices, either the compete graph K_2 or the empty graph $\overline{K_2}$. Clearly $\theta^{ed}(K_2) = 1 \leq \lfloor 2^2/4 \rfloor$, and $\theta^{ed}(\overline{K_2}) = 0 \leq \lfloor 2^2/4 \rfloor$. For the induction step, we assume the statement is true for all graphs on less than or equal to $n-1$ vertices, and we prove that it must also be true for graphs on n vertices. Let G be any simple graph on n vertices. We have two cases.

First, if G contains at least one vertex, say x_1, such that $d_G(x_1) \leq \lfloor n/2 \rfloor$, then by the induction hypothesis, the graph $G - \{x_1\}$ satisfies $\theta^{ed}(G - \{x_1\}) \leq \lfloor (n-1)^2/4 \rfloor$. The only other edges of G we need to cover are those incident with x_1. Therefore we can cover all edges of G with at most $\lfloor (n-1)^2/4 \rfloor + \lfloor n/2 \rfloor$ cliques which by Lemma 3.2.2 means $\theta^{ed}(G) \leq \lfloor n^2/4 \rfloor$.

Otherwise, if G contains no vertices of degree less than or equal to $\lfloor n/2 \rfloor$ then every vertex $v \in V(G)$ satisfies $d_G(v) > \lfloor n/2 \rfloor$. Let x_1 be the vertex of G of minimum degree $\delta(G) = d_G(x_1) = t$. Then $t = \lfloor n/2 \rfloor + r$ for some integer $r > 0$. Label the vertices of G that are adjacent to x_1 as $y_1, y_2, ..., y_t$ and let $Y = \{y_1, ..., y_t\}$. We claim that $G[Y]$ must contain r independent edges, that is, $G[Y]$ must contain r edges such that no two edges are adjacent. First, observe that $2r \leq t$ since $t = \lfloor n/2 \rfloor + r \leq n-1$. Now, assume by way of contradiction that $G[Y]$ has only $m \leq r-1$ independent edges which without loss of generality we label $\{y_1, y_2\}, \{y_3, y_4\}, ..., \{y_{2m-1}, y_{2m}\}$. By assumption, y_{2m+1} is incident with at least $\lfloor n/2 \rfloor + r$ edges in G since $\delta(G) = \lfloor n/2 \rfloor + r$. In particular, y_{2m+1} can be adjacent to at most $2m$ of the vertices $y_1, y_2, ..., y_{2m}$, and to at most $n-t$ vertices of G that are not in Y. Moreover, y_{2m+1} cannot be adjacent to any $y_q \in Y$ if $2m+2 \leq q \leq t$, since our assumption is that $G[Y]$ contains at most m independent edges. Thus,

$$\begin{aligned} d_G(y_{2m+1}) &\leq 2m + n - t \\ &\leq 2(r-1) + n - t, && \text{since } m \leq r-1; \\ &= 2r - 2 + n - (\lfloor n/2 \rfloor + r), && \text{since } t = \lfloor n/2 \rfloor + r; \\ &= n - \lfloor n/2 \rfloor + r - 2 \\ &< \lfloor n/2 \rfloor + r = \delta(G), \end{aligned}$$

contradicting our hypothesis that the minimum degree of G is $\delta(G) = \lfloor n/2 \rfloor + r$. Therefore, we must conclude that $G[Y]$ contains at least $m \geq r$ independent edges.

Now, without loss of generality, we label the r independent edges of $G[Y]$ as $E' = \{\{y_1, y_2\}, \{y_3, y_4\}, ..., \{y_{2r-3}, y_{2r-2}\}, \{y_{2r-1}, y_{2r}\}\}$. Denote by G' the subgraph of $(G - \{x_1\})$ with vertex set $V(G') = V(G - \{x_1\})$ and edge set $E(G') = E(G) \setminus E'$. Then G' is a graph on $n-1$ vertices. By the induction hypothesis, G' can be covered by at most $\lfloor (n-1)^2/4 \rfloor$ 2-cliques and 3-cliques that are pairwise edge-disjoint. The remaining edges of G left uncovered at this point are the r independent edges in E' and the edges incident with x_1. Indeed, these edges can be covered with the 3-cliques $C_1 = \{x_1, y_1, y_2\}, C_2 = \{x_1, y_3, y_4\}, ..., C_r = \{x_1, y_{2r-1}, y_{2r}\}$ and the 2-cliques $C_i = \{x_1, y_i\}$ where $i \in [2r+1, t]$. It is easy to see that for $1 \leq i \leq r + (t - 2r)$ the

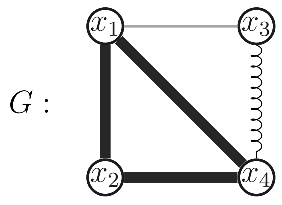


Figure 3.1: Example of an edge-disjoint ECC

The graph G satisfies $\theta'(G) = 2$ since $\mathfrak{C} = \{\{x_1, x_2, x_4\}, \{x_1, x_3, x_4\}\}$ is an ECC of G; however, $\theta^{ed}(G) = 3$ as is depicted by the three types of edges.

cliques C_i are pairwise edge-disjoint. Therefore,

$$\begin{aligned}\theta^{ed}(G) &\leq \lfloor (n-1)^2/4 \rfloor + r + t - 2r \\ &= \lfloor (n-1)^2/4 \rfloor - r + (\lfloor n/2 \rfloor + r), \quad \text{since } t = \lfloor n/2 \rfloor + r; \\ &= \lfloor n^2/4 \rfloor, \qquad \text{by Lemma 3.2.2.}\end{aligned}$$

Again, by considering $K_{k,2}$ for $n = 2k$ even, or $K_{(k,k+1)}$ for $n = 2k+1$ odd, we see that the number $\lfloor n^2/4 \rfloor$ in the above theorem cannot be replaced by any smaller number.

Remark 3.2.4 For any simple graph G, we always have $\theta'(G) \leq \theta^{ed}(G)$ since every edge-disjoint ECC of G is in particular an ECC of G. Equality holds, for example, for any graph G satisfying $\omega(G) = 2$, in which case the only ECC of G is the collection of edges. Equality also holds for any graph G that is the disjoint union of cliques. Strict inequality holds for example in the graph in Figure 3.1.

Remark 3.2.5 By the graph in Figure 3.1, we also see that Proposition 3.1.7 does not hold for edge-disjoint ECCs. We cannot assume without loss of generality that all cliques of an optimal edge-disjoint ECC of a graph be maximal with respect to set inclusion.

However, the following result from Orlin [31] tells us that if a graph has the property that all of its edges each belong to a unique maximal clique of the graph, then the graph has a unique edge-disjoint ECC containing all of the maximal cliques.

Proposition 3.2.6 (Orlin [31]) Let G be a simple graph with the property that every edge of G belongs to a unique maximal (with respect to set inclusion) clique. Then $\theta^{ed}(G) = N$, where N is the number of distinct maximal cliques of G. Furthermore, G has a unique optimal edge-disjoint ECC.

Proof: Let G be a simple graph with the property that every one of its edges belongs to a unique maximal clique. For each $e \in E(G)$, let C_e denote the unique maximal clique that covers edge e. If C is any clique that covers edge e we must have $C \subseteq C_e$, otherwise e would belong to more than one clique that is maximal with respect to set inclusion. Moreover, for any other edge e' such that the ends of e' belong to C_e, we must have that $C_e = C_{e'}$, for if $C_{e'}$ is the unique maximal clique that covers e' then we have $C_e \subseteq C_{e'}$; however, C_e is maximal with respect to set inclusion so we must have $C_e = C_{e'}$.

Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal edge-disjoint ECC of G. Suppose further that for some edge $e \in E(G)$ we have $C_e \notin \mathfrak{C}$. Choose a minimal index set $I \subseteq [1, N]$ such that for every edge e' joining two vertices of C_e, there is some $i \in I$ such that clique C_i covers e'. We can do this since every edge of G must be covered by some clique in $\mathfrak{C}$, so in particular, all edges of $G[C_e]$ must be covered by some collection of cliques of $\mathfrak{C}$. Thus $C_e \subseteq \bigcup_{i \in I} C_i$.

By maximality of C_e, every edge e' joining two vertices of C_e has the property that C_e is the unique maximal clique of G that covers e'. By minimality of I, for each $i \in I$ the clique C_i covers at least one edge e' that joins two vertices of C_e. By maximality of C_e, we must have $C_i \subseteq C_e$ for all $i \in I$. Therefore, $\bigcup_{i \in I} C_i \subseteq C_e$, and we can obtain a smaller edge-disjoint ECC of G by replacing the collection of cliques C_i such that $i \in I$ with the single clique C_e. This contradicts the optimality of $\mathfrak{C}$, so the assumption that $C_e \notin \mathfrak{C}$ is wrong. We conclude that for every edge $e \in E(G)$ the unique maximal clique C_e covering the edge e must belong $\mathfrak{C}$. Consequently, $\{C_e | e \in E(G)\}$ is the unique optimal edge-disjoint ECC of G and it contains N cliques, where N is the number of distinct maximal cliques. ▣

Again, we get the same result for edge-disjoint ECCs as we did for ECCs in Proposition 3.1.6.

Proposition 3.2.7 (Orlin [31]) If G is a graph and $V' \subseteq V(G)$, then $\theta^{ed}(G) \geq \theta^{ed}(G[V'])$.

3.3 Uniform ECCs

Another variation on the ECC problem is the restriction of size of all the cliques.

Definition 3.3.1 [11] Let G be a simple graph and k be an integer. An ECC of G, $\mathfrak{C}$, is said to be **k-uniform** if every clique in $\mathfrak{C}$ has size k. We call $\mathfrak{C}$ a k-ECC of G for short. We define the **k-uniform ECC number** of G to be the size of a minimum k-ECC of G if it exists, or $+\infty$ if one does not exist, and denote it by $\theta'_k(G)$. An ECC of G is said to be **uniform** if it is k-uniform for some integer k.

Indeed, every simple graph admits a uniform ECC. Simply take the collection of 2-cliques of the graph and we have a 2-uniform ECC. However, with the added constraint that an ECC need be k-uniform for some $k \geq 3$, we see that not every graph admits a k-ECC. Here we show two necessary conditions.

Proposition 3.3.2 Let k be a positive integer. If a graph G admits a k-uniform ECC, then the clique number of G must be greater than or equal to k. That is, if $\theta'_k(G) \neq +\infty$ then $\omega(G) \geq k$.

Proof: Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be a k-ECC of G. Then C_i is a k-clique of G, for $1 \leq i \leq N$. Hence, $\omega(G) \geq k$. ▣

Proposition 3.3.3 Let G be a simple graph and let $k \geq 2$ be an integer. If G admits a k-uniform edge clique cover then $|E(G)| \geq \frac{n(k-1)}{2}$, where n is the number of non-isolated vertices of G.

Proof: If G admits a k-uniform edge clique cover then in particular, every non isolated vertex of G must belong to at least one k-clique. Thus, $\sum_{v \in V(G)} d(v) \geq n(k-1)$. By Lemma 1.4.1 we have $n(k-1) \leq 2|E(G)|$. ▣

As with Proposition 3.1.8, when both ends of an edge belong to a unique k-clique, that k-clique must be present in every k-ECC of the graph.

Proposition 3.3.4 Let G be a simple graph and let k be a positive integer. If both ends of an edge $e \in E(G)$ belong to a unique k-clique of G, say C_e, then the clique C_e must be contained in every k-uniform ECC of G.

Comparing the k-uniform ECC number of a graph G to its ECC number, we always have $\theta'(G) \leq \theta'_k(G)$, since $\theta'_k(G) = +\infty$ if a k-ECC of G does not exist, and if $\theta'_k(G) \neq +\infty$, then every k-uniform ECC of G is in particular an ECC of G. Strict inequality holds, for example, in the case where $G = K_n$ for some $n > k$. Then $\theta'(K_n) = 1 < \theta'_k(K_n)$. Now, suppose for a graph G we know that G admits a k-uniform ECC, then when do we have equality? That is, when does $\theta'(G) = \theta'_k(G)$ hold? The following results answer a few cases for the above question.

Proposition 3.3.5 Let k be a positive integer, and let G be a simple graph. If for every clique of G, C, there exists a k-clique of G, C', such that $C \subseteq C'$, then $\theta'(G) = \theta'_k(G)$.

Proof: Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal ECC of G. By assumption, for each $C_i \in \mathfrak{C}$ there is a k-clique of G, say C'_i, such that $C_i \subseteq C'_i$. Clearly, $\mathfrak{C}' = \{C'_1, ..., C'_N\}$ forms an optimal k-uniform ECC of G. Thus, $\theta'(G) = \theta'_k(G)$. ▣

Corollary 3.3.6 Let k be a positive integer. Then $\theta'(K_{(g_1,...,g_k)}) = \theta'_k(K_{(g_1,...,g_k)})$.

Proof: Without loss of generality, label the vertices of $K_{(g_1,...,g_k)}$ as x_{i,j_i} where $1 \leq i \leq k$ and $j_i \in [0, g_i - 1]$, so that the partite sets are of the form $P_i = \{x_{i,j_i} | j_i \in [0, g_i - 1]\}$.

We claim that any clique C of $K_{(g_1,...,g_k)}$ sits inside a k-clique of $K_{(g_1,...,g_k)}$. If $C = \{x_{i_1,j_1}, ..., x_{i_l,j_l}\}$ is a clique of $K_{(g_1,...,g_k)}$, then observe that we must have $i_p \neq i_q$ whenever $p \neq q$, since any pair of adjacent vertices of $K_{(g_1,...,g_k)}$ must belong to two distinct partite sets. Now $C' = C \cup \{x_{i,0} | i \in [1,k] \backslash \{i_1, ..., i_l\}\}$ is a k-clique of $K_{(g_1,...,g_k)}$ since C' contains exactly one vertex from each of the partite sets of $K_{(g_1,...,g_k)}$, and clearly we have $C \subseteq C'$. By Proposition 3.3.5, $\theta'(K_{(g_1,...,g_k)}) = \theta'_k(K_{(g_1,...,g_k)})$. ▣

The next result holds for triangle-free graphs.

Proposition 3.3.7 If a simple graph G satisfies $\omega(G) = 2$, then $\theta'(G) = \theta'_2(G)$.

Proof: If $\omega(G) = 2$, then every edge of G is contained in a unique maximal clique, namely the 2-clique containing its ends. Thus $\theta'(G) = |E(G)| = \theta'_2(G)$. ▣

Even for graphs containing triangles, equality holds for 3-uniform ECCs whenever the clique number of the graph equals 3 provided the graph admits a 3-uniform ECC.

Proposition 3.3.8 Let G be a simple graph satisfying $\omega(G) = 3$. Furthermore, assume that G admits a 3-ECC, that is, assume $\theta'_3(G) \neq +\infty$. Then $\theta'(G) = \theta'_3(G)$.

Proof: Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal clique-maximal ECC of a graph G. Since $\omega(G) = 3$ then clearly we have $|C_i| \in \{2, 3\}$ for $1 \leq i \leq N$. By assumption, $\theta'_3(G) \neq +\infty$ so we know that each edge of G can be covered by some 3-clique of G. In other words, every 2-clique of G is contained in at least one 3-clique of G. Thus, every $C_i \in \mathfrak{C}$ must be a clique of size 3 at least. Since 3 is also the maximum possible size of a clique of G we must have that every $C_i \in \mathfrak{C}$ is a clique of size 3. Thus $\mathfrak{C}$ is a

3-uniform ECC of G and we conclude that $\theta'(G) = \theta'_3(G)$. ◻

The statement that a graph G satisfies $\theta'(G) = \theta'_k(G)$ whenever $\omega(G) = k$ and $\theta'_k(G) \neq +\infty$, however, does not hold in general.

Theorem 3.3.9 Let $k \geq 4$ and $m \geq 1$ be integers. Then there exists a graph G such that $\omega(G) = k$ and $\theta'_k(G) = \theta'(G) + m$.

Proof: For $k = 4$ we provide an example given in Figure 3.2. In fact, this example can be generalized for all values of $k \geq 5$ as follows. Form a $(k-1)$-clique on $k-1$ vertices which we label $u_1, u_2, ..., u_{k-1}$. Call this clique C_0. For each of the $\binom{k-1}{2}$ edges of C_0, add $k-2$ extra vertices and form a k-clique with the $k-2$ vertices added and the two ends of the corresponding edge of C_0. Call these k-cliques $C_1, C_2, ..., C_{\binom{k-1}{2}}$.

Assume without loss of generality that $C_1 = \{u_1, u_2, x_1, ..., x_{k-2}\}$, so that the vertices $x_1, ..., x_{k-2}$ correspond to the $k-2$ vertices added to form a k-clique with edge $\{u_1, u_2\}$ of C_0. Similarly assume $C_2 = \{u_3, u_4, y_1, ..., y_{k-2}\}$. Next, add vertex z_1 and join z_1 to vertices $u_2, x_1, ..., x_{k-2}$ of C_1, so that $C_{\binom{k-1}{2}+1} = \{z_1, u_2, x_1, ..., x_{k-2}\}$ is a k-clique. Similarly, add vertices z_2, z_3 and z_4 and join z_2 to each vertex in $C_1 \setminus \{u_2\}$, join z_3 to each vertex in $C_2 \setminus \{u_3\}$, and join z_4 to each vertex in $C_2 \setminus \{u_4\}$, so that $C_{\binom{k-1}{2}+2} = \{u_1, z_2, x_1, ..., x_{k-2}\}, C_{\binom{k-1}{2}+3} = \{z_3, u_4, y_1, ..., y_{k-2}\}$, and $C_{\binom{k-1}{2}+4} = \{u_3, z_4, y_1, ..., y_{k-2}\}$ are k-cliques. Call the resulting graph $G^{[k]}$. It is easy to check that $\chi(G^{[k]}) = k$, $\omega(G^{[k]}) = k$ and $\theta'_k(G^{[k]}) \neq +\infty$. Moreover, each k-clique C_i for $3 \leq i \leq \binom{k-1}{2}+4$ contains the ends of an edge that belongs to a unique maximal clique of the graph, namely the clique C_i itself. By Proposition 3.1.8 every optimal clique-maximal ECC of $G^{[k]}$ must contain the cliques $C_3, ..., C_{\binom{k-1}{2}+4}$. The only edges left uncovered are $\{u_1, u_2\}$ and $\{u_3, u_4\}$ which can be covered by a single $(k-1)$-clique C_0. Thus $\mathfrak{C} = \{C_0, C_3, C_4, ..., C_{\binom{k-1}{2}+4}\}$ is an optimal ECC of $G^{[k]}$ and $\theta'(G^{[k]}) = \binom{k-1}{2}+3$. However, in an optimal k-uniform ECC of $G^{[k]}$, edges $\{u_1, u_2\}$ and $\{u_3, u_4\}$ must be

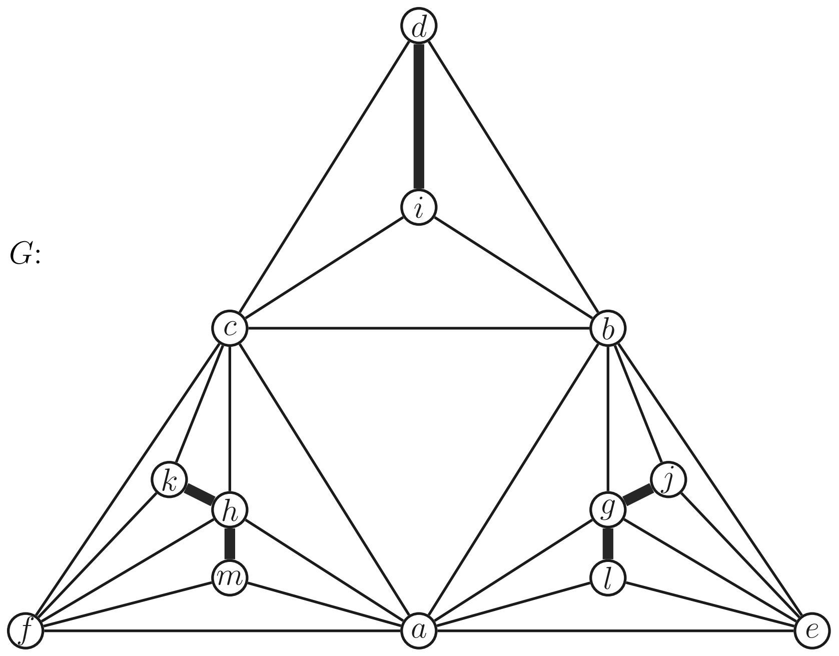


Figure 3.2: Example of a graph G with $\omega(G) = 4$, $\theta'_4(G) \neq +\infty$ and $\theta'_4(G) = \theta'(G)+1$. *The graph G satisfies $\omega(G) = 4$. Furthermore, G admits a 4-uniform ECC since each edge $e \in E(G)$ has the property that both its ends belong to some 4-clique of G. The five thick edges each belong to a unique maximal clique of G. By Proposition 3.1.8, all those unique maximal cliques must be present in every optimal clique-maximal ECC of G. The only edges of G left uncovered after all the thick edges are covered, are the two edges $\{a, b\}$, and $\{a, c\}$, which can be covered with a single extra 3-clique, namely, $\{a, b, c\}$. Thus, $\theta'(G) = 6$. However, the two edges $\{a, b\}$, and $\{a, c\}$ do not belong to a common 4-clique, hence they must be covered separately in an optimal 4-ECC of G. Thus, $\theta'_4(G) = 7$.*

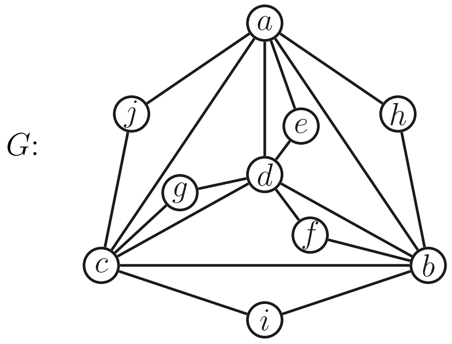


Figure 3.3: Example of a graph G with $\omega(G) = 4$, but $\theta'_3(G) = \theta'(G)$

covered by two separate k-cliques, namely C_1 and C_2. Thus $\theta'_k(G^{[k]}) = \theta'(G^{[k]}) + 1$.

For every integer $m \geq 1$ we can take the disjoint union of m copies of the graph $G^{[k]}$. Denote this union by $m \cdot G^{[k]}$. It is easy to see that the graph $m \cdot G^{[k]}$ satisfies $\theta'_k(m \cdot G^{[k]}) = \theta'(m \cdot G^{[k]}) + m$. ▣

In fact, regardless of a graph G having $\omega(G) > k$, equality between $\theta'(G)$ and $\theta'_k(G)$ can still hold. See Figure 3.3 for a particular example.

Theorem 3.3.10 Let $n > k \geq 3$ be integers. There exists a graph G such that $\omega(G) = n$ and $\theta'_k(G) = \theta'(G)$.

Proof: Take a complete graph on n vertices labeled $v_1, ..., v_n$. For each edge $\{v_i, v_j\}$, $1 \leq i < j \leq n$, add $k-2$ new vertices labeled $v_1^{i,j}, v_2^{i,j}, ..., v_{k-2}^{i,j}$, and add edges so that $C_{i,j} = \{v_i, v_j, v_1^{i,j}, ..., v_{k-2}^{i,j}\}$ forms a k-clique. Refer to the resulting graph as $G^{[k,n]}$. In each k-clique $C_{i,j}$, there is an edge whose ends belong to a unique maximal clique of $G^{[k,n]}$, namely the clique $C_{i,j}$ itself. It is easy to see that $\mathfrak{C} = \{C_{i,j} | 1 \leq i < j \leq n\}$ forms an optimal ECC of $G^{[k,n]}$. Moreover, $\omega(G^{[k,n]}) = n$. Since each $C_{i,j} \in \mathfrak{C}$ is a k-clique, we have $\theta'_k(G^{[k,n]}) = \theta'(G^{[k,n]})$. ▣

In the following proposition, we prove that if G admits a k-uniform ECC for some $k \geq 3$, then G also must admit an l-uniform ECC for all l such that $2 \leq l \leq k$.

Proposition 3.3.11 Let G be a simple graph and let $k \geq 3$ be a positive integer. If $\theta'_k(G) \neq +\infty$ then $\theta'_{k-1}(G) \neq +\infty$.

Proof: We prove the statement by proving the contrapositive. Suppose for $k \geq 3$ that G does not admit a $(k-1)$-uniform ECC. Then there exists an edge $e \in E(G)$ such that the ends of e do not both belong to a common $(k-1)$-clique of G. Consequently, the ends of e do not both belong to a common k-clique of G; hence, $\theta'_k(G) = +\infty$. ▣

In the following results, we give upper bounds on the k-uniform ECC number.

Proposition 3.3.12 Let $k \geq 4$ be a positive integer and let G be a simple graph such that $\omega(G) = k$ and $\theta'_k(G) \neq +\infty$. Then

$$\theta'_k(G) \leq \binom{k-1}{2}\theta'(G).$$

Proof: Let $\mathfrak{C} = \{C_1, ..., C_{\theta'(G)}\}$ be an optimal ECC of G. Without loss of generality, assume that $\mathfrak{C}$ is clique-maximal. Since $\omega(G) = k$, any clique $C_i \in \mathfrak{C}$ must be a clique of size k or smaller. That is, $|C_i| \leq k$ for all $C_i \in \mathfrak{C}$. Let

$$p := |\{C_i \in \mathfrak{C} | C_i \text{ is a } k\text{-clique}\}|$$

$$q := |\{C_i \in \mathfrak{C} | C_i \text{ is a clique of size } k-1 \text{ or smaller}\}|.$$

Then $p + q = \theta'(G)$ and we have $0 \leq q \leq \theta'(G)$, and $0 \leq p \leq \theta'(G)$. Since we can form a k-uniform ECC of G by taking the p k-cliques from $\mathfrak{C}$ and adding at most $\binom{k-1}{2}q$ extra k-cliques obtained by breaking up the q cliques of size $k-1$ or smaller, we have

$$\begin{aligned}
\theta'_k(G) &\le p + \binom{k-1}{2} q \\
&= (\theta'(G) - q) + \binom{k-1}{2} q && \text{since } p = \theta'(G) - q; \\
&= \theta'(G) + \left[\binom{k-1}{2} - 1\right] q \\
&\le \theta'(G) + \left[\binom{k-1}{2} - 1\right] \theta'(G) && \text{since } q \le \theta'(G); \\
&= \binom{k-1}{2} \theta'(G).
\end{aligned}$$

Below is another simple upper bound on the k-uniform ECC number, based on the number of edges of the graph.

Proposition 3.3.13 Let G be a simple graph and let $k \ge 2$ be a positive integer. If G admits a k-uniform ECC then

$$\theta'_k(G) \le |E(G)| - \binom{k}{2} + 1.$$

Proof: Let $\mathfrak{C} = \{C_1, ..., C_{\theta'_k(G)}\}$ be an optimal k-uniform ECC of G. Then C_1 covers $\binom{k}{2}$ edges of G and for $i = 2, ..., \theta'_k(G)$, clique C_i must cover at least one edge of G not already covered by any of the cliques $C_1, ..., C_{i-1}$. Otherwise, C_i would be unnecessary and this would violate the optimality of $\mathfrak{C}$. Therefore, we must have

$$|E(G)| \ge \binom{k}{2} + \theta'_k(G) - 1.$$

3.4 Partial ECCs

Another type of edge clique cover that we may wish to consider is that of a subset of the edge set of a given graph. In a proof on the complexity of various graph covering problems, Orlin [31] addressed the complexity of determining the minimum number of cliques of a graph that cover a specified subset of the edges. For convenience, we refer to this type of problem as a *partial* ECC. In this section, we give the definition of a partial ECC as well as a few properties. We also extend the definition to apply to other types of ECCs, namely edge-disjoint ECCs and uniform ECCs.

Definition 3.4.1 Let G be a simple graph and let $E' \subseteq E(G)$ be a subset of edges of G. We call a collection of cliques of G, $\mathfrak{C}$, a **partial ECC** of (G, E') if for every edge $e \in E'$ there is some clique $C_e \in \mathfrak{C}$ that covers e. A partial ECC of (G, E') is said to be **optimal** if it contains the minimum possible number of cliques, and the minimum number is denoted by $\theta'(G, E')$, and called the **partial ECC number** of (G, E').

Observe that a partial ECC of a graph G exists for every nonempty subset E' of edges of G and we have $\theta'(G, E') \leq |E'|$.

Proposition 3.4.2 For any simple graph G and any subset of edges $E' \subseteq E(G)$, we have $\theta'(G, E') \leq \theta'(G)$.

Proof: Let $E' \subseteq E(G)$ and let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal ECC of G. Then $\mathfrak{C}' = \{C_i \in \mathfrak{C} | C_i \text{ covers some edge } e \in E'\}$ is a partial ECC of (G, E') and we have $\theta'(G, E') \leq |\mathfrak{C}'| \leq |\mathfrak{C}| = \theta'(G)$. ▣

The notion of partial ECCs can also be extended to edge-disjoint ECCs as well as uniform ECCs.

Definition 3.4.3 Let G be a simple graph and let $E' \subseteq E(G)$. A **partial edge-disjoint ECC** of (G, E') is a collection of pairwise edge-disjoint cliques of G, $\mathfrak{C}$, such that for every edge $e \in E'$ there exists some clique $C_e \in \mathfrak{C}$ that covers e. We denote by $\theta^{ed}(G, E')$ the minimum possible number of cliques in a partial edge-disjoint ECC of (G, E'), and we call this number the **partial edge-disjoint ECC number** of (G, E').

Again, we always have $\theta^{ed}(G, E') \leq |E'|$ for every subset of edges $E' \subseteq E(G)$. Furthermore, it is easy to see that we have $\theta^{ed}(G, E') \leq \theta^{ed}(G)$ for every subset of edges E'.

Definition 3.4.4 Let k be a positive integer. For a simple graph G and a subset of edges $E' \subseteq E(G)$, a **partial k-uniform ECC** of (G, E') is a collection of k-cliques of G, say $\mathfrak{C}$, such that for every edge $e \in E'$ there is some k-clique $C_e \in \mathfrak{C}$ that covers e. We denote by $\theta'_k(G, E')$ the minimum possible number of k-cliques in a partial k-uniform ECC of (G, E') if it exists; otherwise, we let $\theta'_k(G, E') = +\infty$. This number is called the **partial k-uniform ECC number** of (G, E').

As with k-uniform ECCs, we see that not every graph admits a partial k-uniform ECC. However, whenever a graph G satisfies $\theta'_k(G) \neq +\infty$ we also have $\theta'_k(G, E') \neq +\infty$ for every subset of edges $E' \subseteq E(G)$. In the example shown in Figure 3.4, we see that a graph G can have the property that $\theta'_k(G) = +\infty$ but for some proper subset of edges E' we may have $\theta'_k(G, E') \neq +\infty$.

In general, we always have $\theta'_k(G, E') \leq \theta'_k(G)$ for every subset E' of edges of the graph.

Proposition 3.4.5 Let k be a positive integer and let G be a simple graph. For any subset of edges $E' \subseteq E(G)$, we have $\theta'_k(G, E') \leq \theta'_k(G)$.

Proof: First suppose $\theta'_k(G) \neq +\infty$. Then let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal k-ECC of G. The collection of k-cliques $\mathfrak{C}' = \{C_i \in \mathfrak{C} | C_i \text{ covers some edge } e \in E'\}$ is

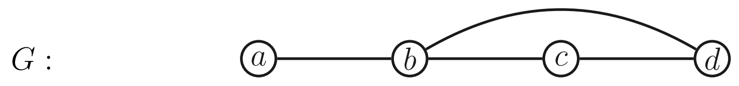


Figure 3.4: Example of graph with $\theta_3'(G, E') = 1$ and $\theta_3'(G) = +\infty$.

The graph G clearly does not admit a 3-uniform ECC since the ends of the edge $\{a, b\}$ *are not contained in any 3-clique of the graph. Thus we have* $\theta_3'(G) = +\infty$. *However, the subset of edges* $E' = \{b, c, d\}$ *satisfies* $\theta_3'(G, E') = 1$.

a partial k-ECC of (G, E') and we have $\theta_k'(G, E') \leq |\mathfrak{C}'| \leq |\mathfrak{C}| = \theta_k'(G)$.

Otherwise, we have $\theta_k'(G) = +\infty$ in which case we either have $\theta_k'(G, E') = L$ for some positive integer $L \leq +\infty$ or we also have $\theta_k'(G, E') = +\infty$. ⧆

If a partial k-ECC of (G, E') does exist, then we always have $\theta_k'(G, E') \leq |E'|$.

3.5 ECCs and Node Clique Covers

The ECC problem is concerned with covering all edges of a graph by cliques. Another closely related clique covering problem is called the node clique cover problem. Several authors have considered various aspects of the relationship between these problems (see for example [2, 22, 31]). In this section, we look at the relationships between ECCs and node clique covers, as well as the graph colouring problem.

Definition 3.5.1 Let G be a simple graph. A collection of cliques of G, say $\mathfrak{C} = \{C_1, ..., C_N\}$, is called a **node clique cover** (NCC) of G if for every vertex $v \in V(G)$ there is some clique $C_i \in \mathfrak{C}$ such that $v \in C_i$. An NCC of G is said to be **optimal** if it contains the least possible number of cliques. We denote by $\theta(G)$ the number of cliques in an optimal NCC of a graph G and call this number the **NCC number** of G.

It is easy to see that every simple graph G admits an NCC, and we have $\theta(G) \leq |V(G)|$ since the collection of 1-cliques of G forms a trivial NCC. The next result gives us another upper bound.

Proposition 3.5.2 Let G be a simple graph. Then $\theta(G) \leq |V(G)| - \omega(G) + 1$.

Proof: Let C_1 be an $\omega(G)$-clique of G. Then the collection of cliques $\mathfrak{C} = \{C_1\} \cup \{\{v\} | v \in V(G) \setminus C_1\}$ forms an NCC of G, and we have $|\mathfrak{C}| = 1 + |V(G)| - \omega(G)$. Therefore, $\theta(G) \leq |V(G)| - \omega(G) + 1$. ▣

The following proposition tells us that any optimal NCC of a graph G can always be assumed without loss of generality to have (vertex-)disjoint cliques.

Proposition 3.5.3 Let G be a simple graph. Then there exists an optimal NCC of G in which any two distinct cliques are disjoint.

Proof: Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal NCC of a simple graph G. We define a new collection $\mathfrak{C}' = \{C_1', ..., C_N'\}$ such that $C_1' = C_1$ and for $2 \leq i \leq N$, take

$$C_i' = C_i \setminus \{v \in V(G) | v \in C_j \text{ for some index } j < i\}.$$

Then $C_i' \subseteq C_i$ for $1 \leq i \leq N$ so each $C_i' \in \mathfrak{C}'$ is a clique of G. Moreover, if $v \in V(G)$, then v is contained in at least one clique in $\mathfrak{C}$. Choose $C_i \in \mathfrak{C}$ with the lowest index i such that $v \in C_i$. Then by construction we have $v \in C_i'$. Moreover, $v \notin C_j'$ for any index $j > i$ by construction of C_j'. Thus, $C_i' \cap C_j' = \emptyset$ if $i \neq j$, so $\mathfrak{C}'$ is a NCC of G in which any two distinct cliques are disjoint. Lastly, since $|\mathfrak{C}'| \leq |\mathfrak{C}| = \theta(G)$, we see that $\mathfrak{C}'$ is optimal. ▣

Brigham and Dutton [2] give several results relating the NCC number to the ECC number. In their results, it is assumed that all graphs are connected, simple graphs. The first result gives an upper bound on $\theta'(G)$ in terms of the number of vertices of G, the NCC number of G as well as the sizes of the cliques in a particular optimal NCC.

Theorem 3.5.4 (Brigham and Dutton [2]) Let G be a simple connected graph on n vertices. Let $\mathfrak{C} = \{C_1, ..., C_{\theta(G)}\}$ be an optimal NCC of G such that the cliques in $\mathfrak{C}$ are pairwise disjoint. For $1 \leq i \leq \theta(G)$, let $n_i = |C_i|$. Furthermore assume the cliques are indexed so that $n_i \leq n_{i+1}$ for $i = 1, 2, ..., \theta(G) - 1$. Then

$$\theta'(G) \leq (n+1)\theta(G) - \sum_{i=1}^{\theta(G)} i n_i.$$

Proof: For $1 \leq i < j \leq \theta(G)$, let $n_{i,j}$ be the number of edges having one end in C_i and the other end in C_j. Then we can cover all edges of G by the $\theta(G)$ cliques in $\mathfrak{C}$ and for $1 \leq i < j \leq \theta(G)$ we add $n_{i,j}$ 2-cliques to cover each edge with one end in C_i and the other end in C_j. Thus

$$\theta'(G) \leq \theta(G) + \sum_{i=1}^{\theta(G)-1} \sum_{j=i+1}^{\theta(G)} n_{i,j}.$$

Observe that the number of cliques necessary to cover the edges incident with a single vertex in C_i to any vertex in C_j is exactly one. Therefore we can cover the $n_{i,j}$ edges having one end in C_i and the other end in C_j by $\min\{n_i, n_j\}$ cliques. By assumption, the cliques are indexed so that $n_i \leq n_j$ whenever $i < j$. Consequently, we can replace

$n_{i,j}$ by n_i to obtain

$$\begin{aligned}
\theta'(G) &\leq \theta(G) + \sum_{i=1}^{\theta(G)-1} \sum_{j=i+1}^{\theta(G)} n_i \\
&= \theta(G) + \sum_{i=1}^{\theta(G)-1} n_i[\theta(G) - i], && \text{since } \sum_{j=i+1}^{\theta(G)} n_i = n_i[\theta(G) - i]; \\
&= \theta(G) + \theta(G) \sum_{i=1}^{\theta(G)-1} n_i - \sum_{i=1}^{\theta(G)-1} in_i \\
&\leq \theta(G) + n\theta(G) - \sum_{i=1}^{\theta(G)-1} in_i, && \text{since } \sum_{i=1}^{\theta(G)-1} n_i \leq n; \\
&= (n+1)\theta(G) - \sum_{i=1}^{\theta(G)-1} in_i.
\end{aligned}$$

▣

The next result is better in the sense that it gives an upper bound on the ECC number of a graph in terms of the number of vertices and the NCC number *only*. For it, we require the following lemma.

Lemma 3.5.5 Let $\{a_1, ..., a_l\}$ be a set of $l > 0$ positive real numbers indexed so that $a_1 \leq a_2 \leq \cdots \leq a_l$. Then

$$\sum_{i=1}^{l} ia_i \geq \frac{1}{2}(l+1) \sum_{i=1}^{l} a_i.$$

Proof: We prove the lemma by induction on l. For $l = 1$, the statement holds. Assume the equation holds for any positive integer $l > 0$. Now, let $a_1, ..., a_l, a_{l+1}$ be

$l+1$ positive real numbers indexed so that $a_1 \le a_2 \le \cdots \le a_l \le a_{l+1}$. Then

$$\begin{aligned}
\sum_{i=1}^{l+1} ia_i &= \sum_{i=1}^{l} ia_i + (l+1)a_{l+1} \\
&\ge \frac{1}{2}(l+1)\sum_{i=1}^{l} a_i + \sum_{i=1}^{l+1} a_{l+1} && \text{by induction hypothesis;} \\
&= \frac{1}{2}(l+1)\sum_{i=1}^{l} a_i + \frac{1}{2}\sum_{i=1}^{l+1} a_{l+1} + \frac{1}{2}\sum_{i=1}^{l+1} a_{l+1} \\
&\ge \frac{1}{2}(l+1)\sum_{i=1}^{l} a_i + \frac{1}{2}(l+1)a_{l+1} + \frac{1}{2}\sum_{i=1}^{l+1} a_i && \text{since } a_{l+1} \ge a_i \text{ for } 1 \le i \le l; \\
&= \frac{1}{2}(l+1)\sum_{i=1}^{l+1} a_i + \frac{1}{2}\sum_{i=1}^{l+1} a_i \\
&= \frac{1}{2}(l+2)\sum_{i=1}^{l+1} a_i.
\end{aligned}$$

▣

Using Lemma 3.5.5 as well as Theorem 3.5.4, Brigham and Dutton [2] give the following bound on $\theta'(G)$.

Theorem 3.5.6 (Brigham and Dutton [2]) Let G be a simple connected graph on n vertices. Then

$$\theta'(G) \le \theta(G) + \frac{1}{2}n(\theta(G)-1).$$

Proof: Using the same notation as in Theorem 3.5.4, we have

$$\theta'(G) \le (n+1)\theta(G) - \sum_{i=1}^{\theta(G)} in_i.$$

Since $n_1, ..., n_{\theta(G)}$ are positive real numbers and indexed so that $n_1 \le \cdots \le n_{\theta(G)}$, we have

$$\begin{aligned}
\sum_{i=1}^{\theta(G)} in_i &\ge \frac{1}{2}[\theta(G)+1]\sum_{i=1}^{\theta(G)} n_i && \text{by Lemma 3.5.5;} \\
&= \frac{1}{2}[\theta(G)+1]n && \text{since } \sum_{i=1}^{\theta(G)} n_i = n.
\end{aligned}$$

Therefore,

$$\theta'(G) \leq (n+1)\theta(G) - \frac{1}{2}n[\theta(G)+1] = \theta(G) + \frac{1}{2}n[\theta(G)-1].$$

▣

We now show that the problem of finding an optimal NCC of a simple graph G is equivalent to finding an optimal proper vertex colouring of the complement $\overline{G}$. Consequently, we have $\theta(G) = \chi(\overline{G})$ for all simple graphs G.

Proposition 3.5.7 For a simple graph G and its complement $\overline{G}$, we have $\theta(G) = \chi(\overline{G})$.

Proof: Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal NCC of a simple graph G. By Proposition 3.5.3, we may assume that the cliques in $\mathfrak{C}$ are pairwise disjoint. In the complement $\overline{G}$, each $C_i \in \mathfrak{C}$ is an independent set and each vertex of G is contained in exactly one of these independent sets. It is easy to see that $\mathfrak{C}$ forms a proper N-vertex colouring of $\overline{G}$ when we colour vertex v colour i whenever $v \in C_i$. Thus $\theta(G) = N \geq \chi(\overline{G})$. Similarly, if $\mathfrak{C}' = \{C_1', ..., C_M'\}$ is an optimal vertex colouring of $\overline{G}$, each $C_i' \in \mathfrak{C}'$ forms an independent set of $\overline{G}$, hence a clique of G. It is easy to see that $\mathfrak{C}'$ forms an NCC of G and so we have $M = \chi(\overline{G}) \geq \theta(G)$. ▣

Chapter 4

Covering Arrays as Edge Clique Cover Problems

In this chapter, we relate the edge clique cover problem to covering arrays. In particular, we show the equivalence between ECCs of k-partite graphs and the various covering array structures.

4.1 Orthogonal arrays

Here we show that an $\mathrm{OA}(N; 2, k, g)$ is equivalent to an edge-disjoint k-ECC of a complete k-partite graph having parts of size g, although this equivalence is known (see p. 161-162 in [7]).

Proposition 4.1.1 Let $K_{k,g}$ be a complete k-partite graph with k parts of size g each. An orthogonal array $A = \mathrm{OA}(N; 2, k, g)$, of strength $t = 2$, is equivalent to $\mathfrak{C}$, an edge-disjoint k-ECC of $K_{k,g}$, with $|\mathfrak{C}| = N$.

Proof: Let the vertices of $K_{k,g}$ be labeled as v_{i,a_i} where $i \in [1, k]$ and $a_i \in [0, g-1]$ so that the partite sets of $K_{k,g}$ are of the form $P_i = \{v_{i,a_i} | a_i \in [0, g-1]\}$ for $1 \leq i \leq k$.

First we show that the existence of an OA$(N; 2, k, g)$ implies the existence of an edge-disjoint k-ECC of $K_{k,g}$ having N cliques. Let A be an OA$(N; 2, k, g)$. Take any row of A, say $R_l = (R_l(1), ..., R_l(k))$, and consider the corresponding set of vertices $C_l := \{v_{1,R_l(1)}, v_{2,R_l(2)}, ..., v_{k,R_l(k)}\}$. For $1 \leq l \leq N$, the set of vertices C_l forms a k-clique of $K_{k,g}$ since C_l contains exactly one vertex from each of the partite sets of $K_{k,g}$. Hence, every pair of vertices in C_l is joined by an edge. Furthermore, we claim that $\mathfrak{C} = \{C_1, ..., C_N\}$ forms an edge-disjoint k-ECC of $K_{k,g}$. Let $\{v_{i,a_i}, v_{j,a_j}\}$ be any edge of $K_{k,g}$. Necessarily we have $i \neq j$. Consider the i^{th} and j^{th} columns of A. By the properties of A we know that there exists exactly one row of A, say R_l such that $R_l(i) = a_i$ and $R_l(j) = a_j$. Thus, exactly one k-clique in $\mathfrak{C}$, namely C_l, contains both the vertices v_{i,a_i} and v_{j,a_j}. Therefore, $\mathfrak{C}$ covers the edge $\{v_{i,a_i}, v_{j,a_j}\}$ exactly once. Since $\{v_{i,a_i}, v_{j,a_j}\}$ is an arbitrary edge of $K_{k,g}$, we see that $\mathfrak{C}$ forms an edge-disjoint k-ECC of G and we have $|\mathfrak{C}| = N$.

Conversely, we show that the existence of an edge-disjoint k-ECC of size N of $K_{k,g}$ implies the existence of an OA$(N; 2, k, g)$. If $\mathfrak{C} = \{C_1, ..., C_N\}$ is an edge-disjoint k-ECC of $K_{k,g}$, then each k-clique C_l such that $l \in [1, N]$, can contain at most one vertex from each partite set P_i since no two vertices from one partite set are adjacent. Moreover, each clique $C_l \in \mathfrak{C}$ must contain k vertices, hence exactly one vertex from each partite set. Thus, each clique $C_l \in \mathfrak{C}$ is of the form $C_l = \{v_{1,C_l(1)}, ..., v_{k,C_l(k)}\}$ where $C_l(i) \in [0, g-1]$ for $1 \leq i \leq k$. Now we form an $N \times k$ array A with N rows of the form $R_l = (C_l(1), ..., C_l(k))$. We claim that A forms an OA$(N; 2, k, g)$. Take any two distinct columns, i and j, and take any pair $(a, b) \in [0, g-1] \times [0, g-1]$. Then in $K_{k,g}$ there exists a unique clique, say $C_l \in \mathfrak{C}$, containing the vertices $v_{i,C_l(i)}$ and $v_{j,C_l(j)}$ where $C_l(i) = a$ and $C_l(j) = b$. Consequently, there exists a unique row of A, namely $R_l = (C_l(1), ..., C_l(k))$, such that $C_l(i) = a$ and $C_l(j) = b$. Therefore, in any $N \times 2$ subarray of A, every pair $(a, b) \in [0, g-1] \times [0, g-1]$ occurs in exactly one row. We conclude that A is an OA$(N; 2, k, g)$. ▣

an OA$(4;2,3,2)$: $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

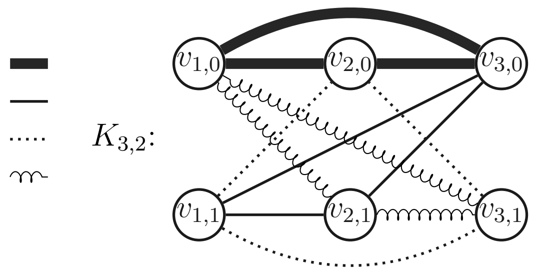


Figure 4.1: Equivalence of an OA$(4;2,3,2)$ and an edge-disjoint 3-ECC of $K_{3,2}$ *The vertices of $K_{3,2}$ are labeled as described in the proof of Proposition 4.1.1 and the edge-disjoint k-ECC of $K_{3,2}$ is pictured by the different types of edges. Each edge type shows the three edges covered by a given 3-clique.*

We illustrate this equivalence with an example shown in Figure 4.1.

4.2 Covering Arrays and Mixed Covering Arrays

Here we prove that the existence of an MCA$(N;2,k,(g_1,...,g_k))$ is equivalent to the existence of a k-uniform ECC, containing N cliques, of the complete k-partite graph $K_{(g_1,...,g_k)}$.

Theorem 4.2.1 There exists an MCA$(N;2,k,(g_1,...,g_k))$ if and only if there exists a k-uniform ECC, containing N cliques, of $K_{(g_1,...,g_k)}$.

Proof: Let A be an MCA$(N;2,k,(g_1,...,g_k))$. Label the vertices of $K_{(g_1,...,g_k)}$ as v_{i,a_i} where $i \in [1,k]$ and $a_i \in [0,g_i-1]$ for $1 \leq i \leq k$, so that the partite sets of $K_{(g_1,...,g_k)}$ are of the form $P_i = \{v_{i,a_i} | a_i \in [0,g_i-1]\}$. Take row $R_l = (R_l(1),...,R_l(k))$ of A and recall that $R_l(i) \in [0,g_i-1]$ for $1 \leq i \leq k$. For $1 \leq l \leq N$, let $C_l = \{v_{1,R_l(1)}, v_{2,R_l(2)},...,v_{k,R_l(k)}\}$. Clearly C_l forms a k-clique of $K_{(g_1,...,g_k)}$ since C_l contains exactly one vertex from each partite set of $K_{(g_1,...,g_k)}$. We claim that the collection of cliques $\mathfrak{C} = \{C_1,...,C_N\}$ is a k-uniform ECC of $K_{(g_1,...,g_k)}$. By construction, each clique $C_l \in \mathfrak{C}$ is a k-clique so we need only show that all edges of $K_{(g_1,\dots,g_k)}$ are covered. Let $e \in E(K_{(g_1,\dots,g_k)})$ be any edge. Then e is of the form $\{v_{i,a_i}, v_{j,a_j}\}$ for some $i \neq j$. By definition, the array A must contain some row, say $R_l = (R_l(1), \dots, R_l(k))$ such that $R_l(i) = a_i$ and $R_l(j) = a_j$. Therefore, the k-clique C_l contains the vertices v_{i,a_i} and v_{j,a_j} and hence covers edge e. Since e was arbitrary, all edges of $K_{(g_1,\dots,g_k)}$ are covered by some clique in $\mathfrak{C}$. Thus there exists $\mathfrak{C}$, a k-uniform ECC of $K_{(g_1,\dots,g_k)}$, such that $|\mathfrak{C}| = N$.

Now, let $\mathfrak{C} = \{C_1, \dots, C_N\}$ be a k-uniform ECC of $K_{(g_1,\dots,g_k)}$. Using the same notation as above we see that each $C_l \in \mathfrak{C}$ is of the form $C_l = \{v_{1,C_l(1)}, v_{2,C_l(2)} \dots, v_{k,C_l(k)}\}$ where $C_l(i) \in [0, g_i - 1]$, since any k-clique of $K_{(g_1,\dots,g_k)}$ must contain exactly one vertex from each partite set. Define an array A with N rows of the form $R_l = (C_l(1), C_l(2), \dots, C_l(k))$. Select any two distinct columns of A, say column i and column j, and choose any pair $(a_i, a_j) \in [0, g_i - 1] \times [0, g_j - 1]$. In $\mathfrak{C}$ there exists a clique C_l that covers edge $\{v_{i,a_i}, v_{j,a_j}\}$, in which case we have $C_l(i) = a_i$ and $C_l(j) = a_j$. Consequently, row R_l of A satisfies $R_l(i) = C_l(i) = a_i$ and $R_l(j) = C_l(j) = a_j$. Since the choice of (i, j) and (a_i, a_j) was arbitrary, A is an MCA$(N; 2, k, (g_1, \dots g_k))$. ⧈

As a consequence we see that finding MCAN$(2, k, (g_1, \dots, g_k))$ is in fact equivalent to finding $\theta'(K_{(g_1,\dots,g_k)})$.

Corollary 4.2.2 MCAN$(2, k, (g_1, \dots, g_k)) = \theta'_k(K_{(g_1,\dots,g_k)}) = \theta'(K_{(g_1,\dots,g_k)})$.

Proof: By Theorem 4.2.1 we get $\theta'_k(K_{(g_1,\dots,g_k)}) =$ MCAN$(2, k, (g_1, \dots, g_k))$. Moreover, by Corollary 3.3.6 we have $\theta'_k(K_{(g_1,\dots,g_k)}) = \theta'(K_{(g_1,\dots,g_k)})$. ⧈

The equivalence of constant alphabet covering arrays and k-uniform ECCs of $K_{k,g}$ is a direct consequence of Theorem 4.2.1 since a mixed covering array with constant alphabet sizes is a covering array.

Corollary 4.2.3 There exists a $\mathrm{CA}(N; 2, k, g)$ if and only if there exists a k-uniform ECC of $K_{k,g}$, say $\mathfrak{C}$, such that $|\mathfrak{C}| = N$.

Corollary 4.2.4 $\mathrm{CAN}(2, k, g) = \theta'_k(K_{k,g}) = \theta'(K_{k,g})$.

Given the equivalence between covering arrays and k-uniform ECCs of complete k-partite graphs we can translate results for ECCs into results for covering arrays and vice versa. The next result from Orlin [31] gives a recursive bound on the ECC number of the complete equipartite graph $K_{k,2}$.

Proposition 4.2.5 (Orlin [31]) Let $k \geq 2$ be an integer. Then

$$\theta'(K_{k,2}) \leq \theta'(K_{k-1,2}) + 1.$$

Proof: Denote the vertices of $K_{k,2}$ as $v_1, v_2, ..., v_k, u_1, u_2, ..., u_k$ so that the partite sets are of the form $P_i = \{u_i, v_i\}$ for $1 \leq i \leq k$. Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal clique-maximal ECC of $K_{k-1,2}$. Then each maximal (with respect to set inclusion) clique contains either vertex v_i or u_i but not both. Hence, each $C_i \in \mathfrak{C}$ contains exactly $k-1$ vertices. Since for each i, vertices v_i and u_i are symmetric, it is thus possible to relabel vertices so that $C_1 = \{v_1, v_2, ..., v_{k-1}\}$. Now let $\mathfrak{C}^* = \{C_1^*, ..., C_N^*, C_{N+1}^*\}$ where $C_1^* = \{v_1, v_2, ..., v_k\}$, $C_{N+1}^* = \{u_1, u_2, ..., u_{k-1}, v_k\}$, and for $i = 2, ..., N$ we let $C_i^* = C_i \cup \{u_k\}$. We claim that $\mathfrak{C}^*$ is an ECC of $K_{k,2}$. The cliques $C_1^*, ..., C_N^*$ cover all edges of the form $\{u_i, u_j\}$ for $1 \leq i < j \leq k-1$, $\{v_i, v_j\}$ for $1 \leq i < j \leq k-1$, and $\{u_i, v_j\}$ for $i \neq j$, $1 \leq i, j \leq k-1$ since $\mathfrak{C}$ is an ECC of $K_{k-1,2}$. Furthermore, $C_2^*, ..., C_N^*$ cover edges $\{u_i, u_k\}$ and $\{v_i, u_k\}$ for $1 \leq i \leq k-1$. Clique C_1^* covers edges $\{v_i, v_k\}$ for $1 \leq i \leq k-1$, and clique C_{N+1}^* covers edges $\{u_i, v_k\}$ for $1 \leq i \leq k-1$. Thus $\mathfrak{C}^*$ is an ECC of $K_{k,2}$ and we have $|\mathfrak{C}^*| = \theta'(K_{k-1,2}) + 1$. ▣

The above theorem gives the following upper bound on the ECC number of complete equipartite graphs having partite sets of size 2.

Corollary 4.2.6 (Orlin [31]) For $k \geq 3$ the graph $K_{k,2}$ satisfies $\theta'(K_{k,2}) \leq k+1$.

Proof: The statement is true by induction on k, the fact that $\theta'(K_{3,2}) = 4$ and Proposition 4.2.5. ▣

Indeed, we can now improve on Orlin's bound, which is linear in k, using the exact value for binary strength 2 covering arrays given by Rényi [32] for N even, and independently by Kleitman and Spencer [21] and Katona [20], for all N (see Theorem 1.2.4), which is known to be in $O(\log_2 k)$.

Corollary 4.2.7 Let k be a positive integer. Then

$$\theta'(K_{k,2}) = \theta'_k(K_{k,2}) = \min\left\{N \in \mathbb{Z} | \binom{N-1}{\lfloor \frac{N}{2} \rfloor - 1} \geq k\right\}.$$

4.3 Covering Arrays Avoiding Forbidden Edges

The next theorem gives us the equivalence between a CAFE(N, G), where $G \in \mathcal{G}_{(g_1,\dots,g_k)}$, and a k-uniform ECC of the complement of $G^|$ (recall that $G^|$ denotes the graph obtained from G by adding edges of the form $\{v_{i,a_i}, v_{i,b_i}\}$ for $1 \leq i \leq k$ and $a_i \neq b_i$ with $a_i, b_i \in [0, g_i - 1]$).

Theorem 4.3.1 (Danziger et. al. [11]) Let k be a positive integer and let $G \in \mathcal{G}_{(g_1,\dots,g_k)}$ be a forbidden edges graph. Then there exists a CAFE(N, G) if and only if there exists a k-uniform ECC, containing N cliques, of the graph $\overline{G^|}$.

Proof: Let A be a CAFE(N, G). Then, the N rows of A, $R_l = (R_l(1), \dots, R_l(k))$, $1 \leq l \leq N$, form k-tuples avoiding G. By Lemma 2.3.2, the set of vertices $C_l = \{v_{1,R_l(1)}, v_{2,R_l(2)}, \dots, v_{k,R_l(k)}\}$ is an independent set of G and of $G^|$. When we take the complement of $G^|$, any independent set of $G^|$ is a clique of $\overline{G^|}$ (see Proposition 1.4.2). We claim that the collection of k-cliques $\mathfrak{C} = \{C_1, \dots, C_N\}$ forms a k-uniform ECC of $\overline{G^|}$. Let $\{v_{i,a_i}, v_{j,a_j}\} \in E(\overline{G^|})$. Then $i \neq j$ and $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G^|)$, and consequently, we have $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$. Therefore, there exists a row of A, say $R_l = (R_l(1), ..., R_l(k))$ such that $R_l(i) = a_i$ and $R_l(j) = a_j$. Then the clique $C_l = \{v_{1,R_l(1)}, v_{2,R_l(2)}, ..., v_{k,R_l(k)}\}$ is a k-clique of $\overline{G^|}$ that covers edge $\{v_{i,a_i}, v_{j,a_j}\}$. We conclude that $\mathfrak{C}$ is a k-uniform ECC, containing N cliques, of $\overline{G^|}$.

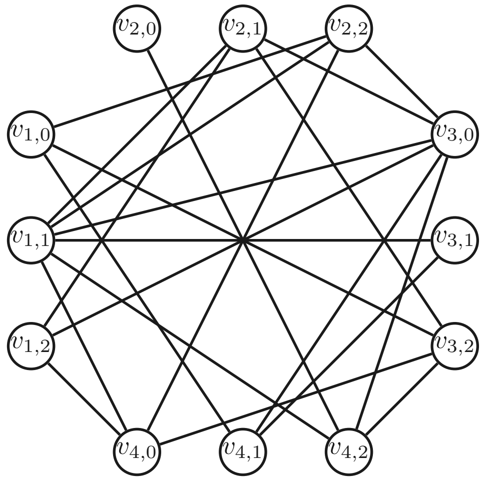
Conversely, let $\overline{G^|}$ be the complement of some forbidden edges graph $G \in \mathcal{G}_{(g_1,...,g_k)}$. Suppose $\mathfrak{C} = \{C_1, ..., C_N\}$ is a k-uniform ECC of $\overline{G^|}$. Each $C_l \in \mathfrak{C}$ is of the form $C_l = \{v_{1,C_l(1)}, v_{2,C_l(2)}, ..., v_{k,C_l(k)}\}$ where for $1 \leq i \leq k$, $C_l(i) \in [0, g_i - 1]$, since any k-clique of $\overline{G^|}$ can contain only one vertex from each factor. Now, define an array A with N rows of the form $R_l = (C_l(1), ..., C_l(k))$ for $1 \leq l \leq N$. Let $i, j \in [1, k]$ such that $i \neq j$. Suppose $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$. Then $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G^|)$ since $i \neq j$ and consequently, we have $\{v_{i,a_i}, v_{j,a_j}\} \in E(\overline{G^|})$. By definition, there exists some k-clique, say $C_l \in \mathfrak{C}$, that covers edge $\{v_{i,a_i}, v_{j,a_j}\}$. Thus, row R_l of A satisfies $R_l(i) = C_l(i) = a_i$ and $R_l(j) = C_l(j) = a_j$. Therefore A covers all non-forbidden interactions of G. Moreover, any row $R_l = (C_l(1), ..., C_l(k))$ forms a k-tuple avoiding G since C_l is clearly an independent set of G and $|C_l| = k$. Therefore, A is a CAFE(N, G). ▣

Corollary 4.3.2 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{(g_1,...,g_k)}$ be a forbidden edges graph. Then CAFEN$(G) = \theta'_k(\overline{G^|}) \geq \theta'(\overline{G^|})$.

In fact, Danziger et. al. [11] prove that the CAFE number of a *binary* forbidden edges graph, $G \in \mathcal{G}_{k,2}$, is equal to the *ECC number* of $\overline{G^|}$ provided that G is consistent. In the following section, we address the particular case of $G \in \mathcal{G}_{k,2}$. Equality in the bound of Corollary 4.3.2, however, is not necessarily the case for $g \geq 3$. In Figure 4.2, we give an example for $g = 3$, where CAFEN$(G) = \theta'_k(\overline{G^|}) > \theta'(\overline{G^|})$.

Subsequently, we give a necessary condition for a CAFE to exist, as well as some new upper bounds on the CAFE number, based on its relationship with the k-ECC number.

G :



a CAFE$(7, G)$:

$$\boxed{\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 1 \\ 2 & 2 & 2 & 1 \\ 2 & 2 & 1 & 2 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 2 & 0 & 2 & 1 \end{array}}$$

$\overline{G^{|}}$:

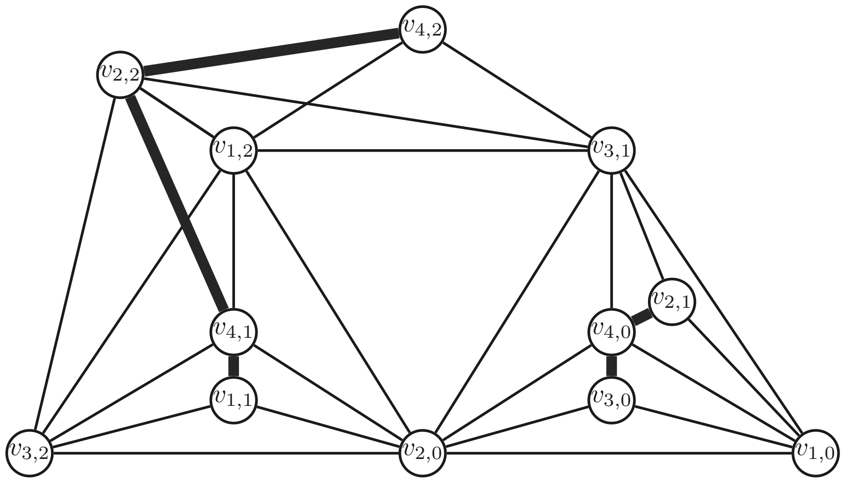


Figure 4.2: A graph $G \in \mathcal{G}_{4,3}$ such that CAFEN$(G) > \theta'(\overline{G^{|}})$.

In $\overline{G^{|}}$, the thick edges are edges whose ends belong to unique maximal cliques, corresponding to the first 5 rows of the CAFE$(7, G)$. There remain two edges of $\overline{G^{|}}$ left to cover, namely $\{v_{1,2}, v_{2,0}\}$ and $\{v_{2,0}, v_{3,1}\}$. These edges can be covered by a single extra 3-clique, $\{v_{1,2}, v_{2,0}, v_{3,1}\}$, so $\theta'(\overline{G^{|}}) = 6$. In a 4-ECC of $\overline{G^{|}}$, we require two separate 4-cliques to cover $\{v_{1,2}, v_{2,0}\}$ and $\{v_{2,0}, v_{3,1}\}$, namely $\{v_{1,2}, v_{2,0}, v_{3,2}, v_{4,1}\}$ and $\{v_{1,0}, v_{2,0}, v_{3,1}, v_{4,0}\}$, respectively. Thus $7 = \theta'_4(\overline{G^{|}}) = CAFEN(G) > \theta'(\overline{G^{|}}) = 6$.

Proposition 4.3.3 Let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ and assume that every vertex $v_{i,a_i} \in V(G)$ has at least one vertex $v_{j,a_j} \in V(G)$ such that $i \neq j$ and $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$. That is, every vertex of G is the "end" of at least one non-forbidden interaction. Then CAFEN$(G) \neq +\infty$ implies

$$|E(G)| \leq \sum_{1\leq i<j\leq k} g_i g_j - \left(\frac{k-1}{2}\right)\sum_{i=1}^{k} g_i.$$

Proof: We have CAFEN$(G) = \theta'_k(\overline{G^{|}})$ by Corollary 4.3.2. By Proposition 3.3.3, $\theta'_k(\overline{G^{|}}) \neq +\infty$ implies $|E(\overline{G^{|}})| \geq \frac{n(k-1)}{2}$, where n is the number of non-isolated vertices of $\overline{G^{|}}$. By our assumption, $n = \sum_{i=1}^{k} g_i$. Therefore,

$$|E(\overline{G^{|}})| = \sum_{1\leq i<j\leq k} g_i g_j - |E(G)| \geq \left(\frac{k-1}{2}\right)\sum_{i=1}^{k} g_i.$$

$$\text{Equivalently, } |E(G)| \leq \sum_{1\leq i<j\leq k} g_i g_j - \left(\frac{k-1}{2}\right)\sum_{i=1}^{k} g_i.$$

The following result gives an upper bound on the CAFE number, and for $k \geq 3$, it is a strict improvement on the upper bound given by [11] (see Proposition 2.4.1).

Proposition 4.3.4 Let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$. If CAFEN$(G) \neq +\infty$, then

$$\text{CAFEN}(G) \leq \sum_{1\leq i<j\leq k} g_i g_j - |E(G)| - \binom{k}{2} + 1.$$

Proof: If CAFEN$(G) \neq +\infty$, then we have

$$\begin{aligned}
\text{CAFEN}(G) &= \theta'_k(\overline{G^{|}}), && \text{by Corollary 4.3.2;}\\
&\leq |E(\overline{G^{|}})| - \binom{k}{2} + 1, && \text{by Proposition 3.3.13;}\\
&= \sum_{1\leq i<j\leq k} g_i g_j - |E(G)| - \binom{k}{2} + 1.
\end{aligned}$$

Here we translate the upper bound of Proposition 3.3.12 into another upper bound for the CAFE number.

Proposition 4.3.5 Let $k \geq 4$ and let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$. If $\mathrm{CAFEN}(G) \neq +\infty$ then,

$$\mathrm{CAFEN}(G) \leq \binom{k-1}{2}\theta'(\overline{G^{|}}).$$

Proof:

$$\mathrm{CAFEN}(G) = \theta'_k(\overline{G^{|}}) \leq \binom{k-1}{2}\theta'(\overline{G^{|}}) \text{ by Proposition 3.3.12.}$$

▣

The following proposition, tells us that for consistent forbidden edges graphs $G \in \mathcal{G}_{(g_1,g_2,g_3)}$, having $k = 3$ factors, the CAFE number of G is equal to the ECC number of $\overline{G^{|}}$.

Proposition 4.3.6 Let $G \in \mathcal{G}_{(g_1,g_2,g_3)}$ be a consistent forbidden edges graph with $k = 3$ factors. Then $\mathrm{CAFEN}(G) = \theta'_3(\overline{G^{|}}) = \theta'(\overline{G^{|}})$.

Proof: By Corollary 4.3.2, we have $\mathrm{CAFEN}(G) = \theta'_3(G)$. If G contains no non-forbidden edges, that is, if $\overline{G^{|}}$ is empty, then $\mathrm{CAFEN}(G) = \theta'_3(G) = \theta'(\overline{G^{|}}) = 0$. It is easy to see that for $G \in \mathcal{G}_{(g_1,g_2,g_3)}$, if G is consistent and has at least one non-forbidden interaction, then $\omega(\overline{G^{|}}) = 3$. In this case, by Proposition 3.3.8, we have $\theta'_3(\overline{G^{|}}) = \theta'(\overline{G^{|}})$, so $\mathrm{CAFEN}(G) = \theta'(\overline{G^{|}})$. ▣

4.4 Binary CAFEs

In this section, we look at a particular type of forbidden edges graphs, namely those in $\mathcal{G}_{k,2}$ having CAFEs with *binary* alphabets ($g = 2$). We present a theorem by

Danziger et. al. [11] which tells us that the CAFE number of a consistent graph $G \in \mathcal{G}_{k,2}$ is equal to the ECC number of the graph $\overline{G^{|}}$. We also look at a special case of consistent binary forbidden edges graphs, namely those that are bipartite, and give several results by Danziger et. al. [11].

Theorem 4.4.1 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{k,2}$ be a binary forbidden edges graph. If G is consistent, then $\text{CAFEN}(G) = \theta'(\overline{G^{|}})$.

Proof: Given any clique $C = \{v_{i_1,a_1}, v_{i_2,a_2}, ..., v_{i_q,a_q}\}$ in $\overline{G^{|}}$, we show that if $q < k$ then C can be extended into a k-clique. Let $j \in [1,k] \setminus \{i_1, ..., i_q\}$. Suppose neither vertex from factor j can be added to C and still induce a clique. Then we must have vertices $v_{i_l,a_l}, v_{i_m,a_m} \in C$ such that $\{v_{i_l,a_l}, v_{j,0}\} \notin E(\overline{G^{|}})$ and $\{v_{i_m,a_m}, v_{j,1}\} \notin E(\overline{G^{|}})$. This would mean that in G we have $\{v_{i_l,a_l}, v_{j,0}\} \in E(\overline{G^{|}})$ and $\{v_{i_m,a_m}, v_{j,1}\} \in E(\overline{G^{|}})$ but $\{v_{i_l,a_l}, v_{i_m,a_m}\} \notin E(\overline{G^{|}})$. By Proposition 2.3.6, this would mean that G is not consistent, a contradiction. Therefore, we must always be able to add to C at least one vertex from each factor $j \in [1,k] \setminus \{i_1, ..., i_q\}$ and still induce a clique. In other words, we can extend any clique of $\overline{G^{|}}$ into a k-clique. Consequently, we have $\theta'(\overline{G^{|}}) = \theta'_k(\overline{G^{|}}) = \text{CAFEN}(G)$ for any consistent binary forbidden edges graph G. ▣

The above theorem is an improvement on Corollary 4.3.2 since the ECC number of a graph is always less than or equal to its k-uniform ECC number.

We now look at a special case of consistent binary forbidden edges graphs, namely those that are bipartite. The following results tell us that if $G \in \mathcal{G}_{k,2}$ is consistent and bipartite, then there is a $\text{CAFE}(n, G)$ with two disjoint rows.

Lemma 4.4.2 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{k,2}$ be consistent. Then G is bipartite if and only if $G^{|}$ is bipartite.

Proof: Clearly, if $G^{|}$ is bipartite, then with the same partition of vertices G is also bipartite, since $G = G^{|} \setminus \{\{v_{i,0}, v_{i,1}\} \in E(G^{|}) | 1 \leq i \leq k\}$.

We need to show that if G is bipartite, then so is $G^{|}$. Assume G is bipartite. Then G is 2-colourable and contains no odd cycles. We claim that there exists a 2-colouring of G which has the desired property that within each factor the two vertices are given different colours.

Suppose by way of contradiction that there is a factor j where both its vertices are forced to have the same colour. This means that there must be an even length path in G from $v_{j,0}$ to $v_{j,1}$. Let $v_{j,0}q_1q_2\cdots q_{2r+1}v_{j,1}$ be the shortest even path connecting $v_{j,0}$ to $v_{j,1}$. Then $\{v_{j,0},q_1\}\in E(G)$ and $\{q_{2r+1},v_{j,1}\}\in E(G)$. Because G is consistent, we must have $\{q_1,q_{2r+1}\}\in E(G)$ also. This, however, implies that $q_1q_2\cdots q_{2r+1}$ is an odd cycle of G, contradicting the fact that G is bipartite. Thus, it must be possible to colour the vertices of each factor j differently. Consequently, $G^{|}$ is bipartite. ▣

Corollary 4.4.3 (Danziger et. al. [11]) Let $G\in\mathcal{G}_{k,2}$ be consistent. Then G is bipartite if and only if for some n, G admits a CAFE(n,G) containing two disjoint rows.

Proof: Assume G is bipartite. By Lemma 4.4.2, $G^{|}$ is bipartite, so there exists a 2-colouring of $G^{|}$, say $\{C_1,C_2\}$, and we have exactly one vertex from each factor in C_1 and the other vertex from that factor in C_2. Furthermore, C_1 and C_2 correspond to independent sets of G of size k, hence they correspond to k-tuples avoiding G. Consequently, a CAFE(n,G) exists with the rows corresponding to C_1 and C_2, which are disjoint.

Next, if G admits a CAFE(n,G) with two disjoint rows, then these rows correspond to two disjoint independent sets of size k each. In other words, we can colour G with two colours, according to these independent sets, so G is bipartite. ▣

The following result for bipartite graphs is based on an algorithm given in [11], which produces a CAFE for a consistent binary forbidden edges graph. We omit the algorithm and the proof of its correctness here. The algorithm first removes "forced" or "redundant" factors of G, yielding the *reduced* graph, for which we give the definition below.

Definition 4.4.4 Let $G \in \mathcal{G}_{k,2}$ be consistent. We call G **reduced** if

1. G has no factor i and value $a \in [0,1]$ such that $\{v_{i,a}, v_{j,b}\} \in E(G)$ for all $j \in [1,k] \setminus \{i\}$ and all $b \in [0,1]$, and
2. G has no pair of factors i and j and values $a, b \in [0,1]$ such that $\{v_{i,a}, v_{j,b}\} \in E(G)$ and $\{v_{i,1-a}, v_{j,1-b}\} \in E(G)$ (factors i and j with **parallel edges**).

Note that part 1. in the definition of reduced is equivalent to G having no factor i such that $\{v_{i,a}, v_{j,0}\} \in E(G)$ and $\{v_{i,a}, v_{j,1}\} \in E(G)$ for *some* $j \in [1,k] \setminus \{i\}$, since G is assumed to be consistent.

Indeed, a CAFE(n, G) of a consistent graph $G \in \mathcal{G}_{k,2}$ that is not reduced has columns which are forced and/or redundant. Consider a graph $G \in \mathcal{G}_{k,2}$ with a vertex $v_{i,a}$ such that $\{v_{i,a}, v_{j,b}\} \in E(G)$ for all $j \in [1,k] \setminus \{i\}$ and for all $b \in [0,1]$. Then column i of a CAFE(n, G) is forced to take the value $1-a$ in every row since $v_{i,a}$ is incompatible with all other factors. We can reduce G by simply removing factor i. In this case we can focus on finding a CAFE$(n, G - \{v_{i,0}, v_{i,1}\})$ and we can always add back the "missing" column i consisting of the forced values of factor i, later in order to obtain a CAFE(n, G).

Similarly, if G contains two factors i and j with parallel edges, then in each row of a CAFE(n, G), the value in column i forces the value in column j and vice versa, essentially making column j redundant whenever column i is already present. In this case, we can reduce G by removing factor j and focus on finding a CAFE$(n, G - \{v_{j,0}, v_{j,1}\})$. We could then add back the "missing" column j by looking at column i and deducing the forced values, thus yielding a CAFE(n, G).

Theorem 4.4.5 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{k,2}$ be consistent and let $k' \leq k$ be the number of factors left after G is reduced. Let T be a k-tuple avoiding G, and let

G_1 be the graph obtained by removing all edges incident to the vertices corresponding to T. Then, $\text{CAFEN}(G) \leq k' + 1 + m \leq k + 1 + m$, where m is the size of an ECC of $\overline{G_1}$.

As a consequence of this theorem and the properties given above for bipartite binary forbidden edges graphs, we get the following result.

Corollary 4.4.6 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{k,2}$ be consistent and bipartite. Let $k' \leq k$ be the number of factors left after G is reduced. Then $\text{CAFEN}(G) \leq k' + 2 \leq k + 2$.

Proof: Since G is bipartite, so is $G^{|}$ by Lemma 4.4.2, so there is a 2-colouring, $\{C_1, C_2\}$, of $G^{|}$. In Theorem 4.4.5, we can choose T to be the k-tuple corresponding to the vertices of one of these colour classes, in which case, G_1 will be empty, and $m = \theta'(\overline{G_1}) = 1$. By Theorem 4.4.5, $\text{CAFEN}(G) \leq 1 + k' + m = k' + 2 \leq k + 2$. ▣

Corollary 4.4.7 (Danziger et. al. [11]) Let $G \in \mathcal{G}_{k,2}$ be consistent, and let $k' \leq k$ be the number of factors left after G is reduced. Suppose the factor-connected components of the reduced version of G are $G_1, ..., G_s$, with k'_i factors in G_i. If each G_i is bipartite, then

$$\text{CAFEN}(G) \leq \text{CAN}(2, s, 2) + \max_{1 \leq i \leq s}\{k'_i\} - 1.$$

Proof: Let $i \in [1, s]$. By Corollary 4.4.6, we can build C_i, a $\text{CAFE}(k'_i + 2, G_i)$, since G_i is bipartite. Moreover, C_i contains two disjoint rows corresponding to the 2-colouring of G_i. Indeed, these two particular rows form a $\text{CAFE}^1(2, G_i)$. Therefore, in Theorem 2.5.1, we can take P_i to be an array with these two rows, and we can take A_i to be the array obtained by removing these two rows from the array C_i, so that $a_i = k'_i$. Then, by Theorem 2.5.1, we have $\text{CAFEN}(G) \leq \text{CAN}(2, s, 2) + \max\{k'_1, ..., k'_s\}$. Since without loss of generality the first row of the array M, the $\text{CA}(m; 2, s, 2)$, is a row of all zeros, we see that the construction given here produces a $\text{CAFE}(n, G)$

with the first row repeated. Removing this unnecessary row, we get $\text{CAFEN}(G) \leq \text{CAN}(2, s, 2) + \max\{k'_1, ..., k'_s\} - 1$. ▣

4.5 Partial Covering Arrays Avoiding Forbidden Edges and Testing Applications

Now that we have seen the problems of covering arrays translated into the language of k-ECCs, we may wish to consider some new generalizations of covering arrays based on k-ECC problems. Perhaps of interest is the following problem.

Given a testing problem $\text{TP}(k; (g_1, ..., g_k))$ we may specify a list of forbidden pairwise interactions to the problem as well as a list of non-forbidden interactions for which we do not care whether or not we cover those interactions in any test. The idea for this generalization stems from the so called covering arrays on graphs which are motivated by the fact that not every pair of factors in a given testing problem interact (see [29]). Consequently, there is no need to cover every interaction between two non-interacting or independent factors. Thus we can reduce the number of tests required by allowing certain interactions to be labeled for optional coverage.

We now define a new structure given the same motivation as above. However, the interactions for which we deem coverage as optional need not occur between all values of two given factors. That is, we do not require knowledge that factor i and factor j do not interact; instead, we simply want a list of pairwise interactions for which coverage is necessary.

Definition 4.5.1 Let $G \in \mathcal{G}_{(g_1,...,g_k)}$, and let $\mathcal{I}$ be a subset of the non-forbidden interactions (non-edges) of G, referred to as **the necessary interaction set**. A **partial covering array avoiding forbidden edges** of $(G, \mathcal{I})$, denoted $\text{PCAFE}(N; G, \mathcal{I})$, is an $N \times k$ array A such that

1. every row of A forms a k-tuple avoiding G;
2. for every interaction $I \in \mathcal{I}$ there exists a row of A that covers I.

The **partial CAFE number**, denoted PCAFEN$(G, \mathcal{I})$, is the minimum integer N such that a PCAFE$(N; G, \mathcal{I})$ exists, if a partial CAFE of $(G, \mathcal{I})$ exists, or $+\infty$ otherwise.

In other words, a PCAFE$(N; G, \mathcal{I})$ is equivalent to a partial k-uniform ECC of $(\overline{G^{|}}, E_{\mathcal{I}})$ of size N, where $E_{\mathcal{I}} = \{\{v_{i,a_i}, v_{j,a_j}\} | \{(i, a_i), (j, a_j)\} \in \mathcal{I}\} \subseteq E(\overline{G^{|}})$. We can also think of a partial CAFE graph as a complete k-partite graph $K_{(g_1,...,g_k)}$ with its edge set partitioned into three classes: the edges that must be covered, the forbidden edges, and the edges for which coverage is optional.

Theorem 4.5.2 Let k be a positive integer and let $G \in \mathcal{G}_{(g_1,...,g_k)}$. There exists a PCAFE$(N; G, \mathcal{I})$ if and only if there exists a partial k-ECC of $(\overline{G^{|}}, E_{\mathcal{I}})$ where $E_{\mathcal{I}} = \{\{v_{i,a_i}, v_{j,a_j}\} | \{(i, a_i), (j, a_j)\} \in \mathcal{I}\} \subseteq E(\overline{G^{|}})$.

Proof: Let A be a PCAFE$(N; G, \mathcal{I})$. Then every row $R_i = (R_i(1), ..., R_i(k))$ corresponds to an independent set, $C_i = \{v_{1,R_i(1)}, ..., v_{k,R_i(k)}\}$, of G. Since C_i contains exactly one vertex from each factor, C_i forms a k-clique of $\overline{G^{|}}$. Let $e = \{v_{i,a_i}, v_{j,a_j}\} \in E_{\mathcal{I}}$. Then, there is an interaction $I = \{(i, a_i), (j, a_j)\} \in \mathcal{I}$ and there exists a row of A, say R_l, such that $R_l(i) = a_i$ and $R_l(j) = a_j$. Consequently, the corresponding k-clique, $C_l = \{v_{1,R_l(1)}, ..., v_{k,R_l(k)}\}$, covers e. Therefore, $\mathfrak{C} = \{C_1, ..., C_N\}$ is a partial k-ECC of $(\overline{G^{|}}, E_{\mathcal{I}})$.

Now, let $\mathfrak{C} = \{C_1, ..., C_N\}$ be a partial k-ECC of $(\overline{G^{|}}, E_{\mathcal{I}})$. Define an array A, with rows $R_1, ..., R_N$ such that $R_l(i) = a_i$ whenever $v_{i,a_i} \in C_l$. Since each $C_l \in \mathfrak{C}$ contains exactly one vertex from each factor, it is easy to see that each row R_l of A forms a k-tuple avoiding G. Let $I = \{(i, a_i), (j, a_j)\} \in \mathcal{I}$. Then $e = \{v_{i,a_i}, v_{j,a_j}\} \in E_{\mathcal{I}}$ and there must exist a clique, say $C_l \in \mathfrak{C}$, that covers e. Consequently, row R_l covers I. Since I was arbitrary, we see that A is a PCAFE$(N; G, \mathcal{I})$. ▣

Corollary 4.5.3 Let $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$, let $\mathcal{I}$ be the necessary interaction set for G, and let $E_{\mathcal{I}} = \{\{v_{i,a_i}, v_{j,a_j}\} | \{(i,a_i),(j,a_j)\} \in \mathcal{I}\} \subseteq E(\overline{G^{|}})$. Then,

1. $\mathrm{PCAFEN}(G,\mathcal{I}) = \theta'_k(\overline{G^{|}}, E_{\mathcal{I}})$;
2. $\mathrm{PCAFEN}(G,\mathcal{I}) \leq \mathrm{CAFEN}(G)$;
3. if $\mathrm{PCAFEN}(G,\mathcal{I}) \neq +\infty$, then $\mathrm{PCAFEN}(G,\mathcal{I}) \leq |\mathcal{I}|$.

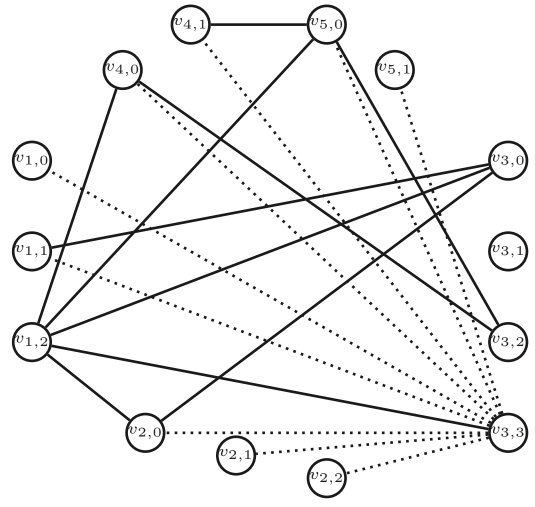
Proof:

1. Immediate from Theorem 4.5.2.
2. Since $\theta'_k(\overline{G^{|}}, E_{\mathcal{I}}) \leq \theta'_k(\overline{G^{|}})$ by Proposition 3.4.2, and since $\theta'_k(\overline{G^{|}}) = \mathrm{CAFEN}(G)$ by Corollary 4.3.2, we get $\mathrm{PCAFEN}(G,\mathcal{I}) \leq \mathrm{CAFEN}(G)$.
3. We have $\theta'_k(\overline{G^{|}}, E_{\mathcal{I}}) = \mathrm{PCAFEN}(G,\mathcal{I}) \neq +\infty$, so $\theta'_k(\overline{G^{|}}, E_{\mathcal{I}}) \leq |E_{\mathcal{I}}| = |\mathcal{I}|$.

$\square$

Hartman and Raskin [16] proposed the need for covering array extension in testing applications. A set of regression tests may already exist for a given testing problem and we would like to add the minimum number of extra tests needed in order to achieve pairwise coverage. In this case, we can check all the pairwise interactions which are already covered by the regression tests, and let $\mathcal{I}$ be the set of remaining pairwise interactions. Although forbidden interactions are not specifically mentioned in this particular application, we can still use a PCAFE having no forbidden interactions, and with necessary interaction set $\mathcal{I}$ as indicated. One other application to motivate the study of such an object was indicated in [18]. In some software testing applications, the tester runs tests on a testing problem, say $\mathrm{TP}(k;(g_1,...,g_k))$ to cover all pairwise interactions. After these tests are *already run*, new values are then added to certain factors of the problem. Thus we would like to cover the new interactions added to the problem with as few tests as possible, but coverage of the "old" interactions is optional. In this case, the necessary interaction set is the set of "new"

G:



a PCAFE$(3; \hat{G}, \mathcal{I})$:

$$\boxed{\begin{array}{ccccc} 0 & 0 & 3 & 0 & 0 \\ 1 & 1 & 3 & 1 & 1 \\ 0 & 2 & 3 & 0 & 0 \end{array}}$$

Figure 4.3: The forbidden edges graph $G \in \mathcal{G}_{(3,3,4,2,2)}$ of the augmented mobile phone product line testing problem, with the dotted edges corresponding to $\mathcal{I}$, and a PCAFE$(3; G, \mathcal{I})$

interactions. The set of forbidden interactions of the augmented testing problem is simply a superset of the forbidden interactions of the original problem.

Let us illustrate partial CAFEs with the testing problem of the mobile phone product line given in Table 2.1. Assume that software testers have run the ten tests of the mobile phone product line testing problem, corresponding to the CAFE of the graph $\hat{G}$, given in Figure 2.4. After these tests are run, a new value for factor 3 has been added, say value 3: "4 megapixels" camera. Moreover, the 4 megapixels camera **requires** a colour display, resulting in a new forbidden interaction $\{(1,2),(3,3)\}$. We now have an augmented testing problem, a TP$(5,(3,3,4,2,2))$. Rather than redoing the tests of this problem from scratch, we only care about the new interactions corresponding to the added value, namely the necessary interaction set $\mathcal{I} = \{\{(i, a_i),(3,3)\} | i \in [1,5], i \neq 3, a_i \in [0, g_i - 1]\}$. The forbidden edges graph $G \in \mathcal{G}_{(3,3,4,2,2)}$ corresponding to the augmented testing problem of the mobile phone product line is shown in Figure 4.3. We have PCAFEN$(G, \mathcal{I}) \geq 3$ because the three dotted edges joining vertices of factor 2 to $v_{3,3}$ can only be covered by three distinct rows. Consequently, the PCAFE$(3; G, \mathcal{I})$ given in Figure 4.3 is optimal.

Chapter 5

Computational Complexity of Covering Array Problems

In this chapter, we look at the complexity of the underlying decision problems related to covering array structures. In Section 5.1, we consider the decision problems most closely related to covering array structures and the previously known results on their complexity. In Sections 5.2 and 5.3, we prove the NP-completeness of g-CAFEN, for $g \geq 2$. In Section 5.4, we prove that the partial CAFE problem is NP-complete. In Section 5.5, we prove the NP-completeness of the uniform and partial uniform ECC problems. Section 5.6 gives a review of complexity results related to error-locating arrays. In Sections 5.7 and 5.8, we show that the decision problem for the language g-ELAN is NP-complete for $g \geq 2$.

5.1 Decision Problems Related to CAFEs and Previous Results

Since we are mainly concerned with the problem of finding an optimal CAFE(n, G) for a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$, we consider the complexity of the decision problems associated to the following languages defined in [11]:

$$\begin{aligned}
&\text{AVOID} = \{G \in \mathcal{G}_{(g_1,\ldots,g_k)} \mid \text{there exists a } k\text{-tuple avoiding } G\},\\
&\text{ONE-COVER\&AVOID} = \{G \in \mathcal{G}_{(g_1,\ldots,g_k)} \mid \text{for some } n \text{ there exists a CAFE}^1(n,G)\},\\
&\text{COVER\&AVOID} = \{G \in \mathcal{G}_{(g_1,\ldots,g_k)} \mid \text{for some } n \text{ there exists a CAFE}(n,G)\},\\
&\text{CAFEN} = \{(G,N) \in \mathcal{G}_{(g_1,\ldots,g_k)} \times \mathbb{Z} \mid \text{there exists a CAFE}(N,G)\}.
\end{aligned}$$

Furthermore, for each language L defined above, we use the notation g-L to describe the language where the graph input G is of the particular form $G \in \mathcal{G}_{k,g}$. For example, 2-AVOID $= \{G \in \mathcal{G}_{k,2}|$ there exists a k-tuple avoiding $G.\}$

Based on the close relationship between the ECC problem and covering array problems described in the last chapter, we also explore the complexity of the decision problem associated to ECCs, given by the language ECCN defined below. We also consider the language associated to the NCC problem. Recall that we use $\mathbb{G}$ to denote the set of all finite simple graphs.

$$\begin{aligned}
&\text{ECCN} = \{(G,N) \in \mathbb{G} \times \mathbb{Z} \mid \theta'(G) \leq N\}\\
&\text{NCCN} = \{(G,N) \in \mathbb{G} \times \mathbb{Z} \mid \theta(G) \leq N\}
\end{aligned}$$

We now look at some results regarding the complexities of these languages. The following theorem from Danziger et. al. [11] establishes that some of the decision problems for the binary (i.e. $g = 2$) versions of the above languages are in the class P, whereas, others where $g \geq 3$ are shown to be NP-complete. We follow the main steps of their proof, and fill in the details.

Theorem 5.1.1 (Danziger et. al. [11])

1. 2-AVOID $\in P$.

2. g-AVOID is NP-complete for $g \geq 3$ and so AVOID is NP-complete.
3. 2-ONE-COVER&AVOID and 2-COVER&AVOID are in P.
4. $(g+1)$-ONE-COVER&AVOID and $(g+2)$-COVER&AVOID are NP-complete for $g \geq 3$, and so are ONE-COVER&AVOID and COVER&AVOID.
5. g-CAFEN is NP-complete for $g \geq 5$, and so CAFEN is NP-complete.

For convenience, we separate the individual results into the next five propositions for which we present detailed proofs.

Proposition 5.1.2 2-AVOID $\in$ P.

Proof: In order to prove that 2-AVOID belongs to the class P, by Lemma 1.5.2 it is enough to show that 2-AVOID $\leq_P$ 2-SAT, since 2-SAT $\in P$. Let $G \in \mathcal{G}_{k,2}$ be a graph that is an instance for 2-AVOID. Associate to each vertex $v_{i,a_i} \in V(G)$ a literal l_{i,a_i} such that $l_{i,a_i} = x_i$ if $a_i = 0$, or else $l_{i,a_i} = \neg x_i$ if $a_i = 1$. Associate to each edge $\{v_{i,a_i}, v_{j,a_j}\} \in E(G)$ a clause $(\neg l_{i,a_i} \vee \neg l_{j,a_j})$. Now let f be the reduction function so that $f(G) = \varphi$ is the conjunction of these clauses. That is, let

$$f(G) = \varphi = \bigwedge_{\{v_{i,a_i}, v_{j,a_j}\} \in E(G)} (\neg l_{i,a_i} \vee \neg l_{j,a_j}).$$

Clearly $f(G)$ can be computed in polynomial time with respect to the size of the input graph G. Furthermore, we claim that $G \in$ 2-AVOID if and only if $f(G) = \varphi \in$ 2-SAT. First, let $G \in$ 2-AVOID. Then there exists a k-tuple, say $T = (a_1, a_2, ..., a_k)$, that avoids G, in which case the set of vertices $I = \{v_{1,a_1}, v_{2,a_2}, ..., v_{k,a_k}\}$ is an independent set of G. Let V be a valuation for φ such that $V(l_{i,a_i}) = T$ if and only if $v_{i,a_i} \in I$. Now suppose $\{v_{i,a_i}, v_{j,a_j}\} \in E(G)$. Then it is not possible for both vertices v_{i,a_i} and v_{j,a_j} to belong to I simultaneously. Assume without loss of generality that $v_{i,a_i} \notin I$. Then $V(l_{i,a_i}) = F$ by definition of V and consequently we have $(\neg V(l_{i,a_i}) \vee \neg V(l_{j,a_j})) \equiv (T \vee \neg V(l_{j,a_j})) \equiv T$. Since $\{v_{i,a_i}, v_{j,a_j}\} \in E(G)$ was an arbitrary edge of G, we see that each clause of φ is true under V. Thus, $V(\varphi) = T$ so $f(G) = \varphi \in$ 2-SAT.

Now let $f(G) = \varphi \in$ 2-SAT. So there exists a valuation V such that $V(\varphi) = T$. Let $I = \{v_{i,a_i} | V(l_{i,a_i}) = T\}$. Suppose v_{i,a_i} and v_{j,a_j} both belong to I and furthermore assume by way of contradiction that $e = \{v_{i,a_i}, v_{j,a_j}\} \in E(G)$. Then the clause associated to e would be false under the valuation V, and consequently we would have $V(\varphi) = F$, a contradiction. Therefore, every pair of vertices in I are not adjacent in G. Hence I is an independent set of G and thus $(a_1, ..., a_k)$ forms a k-tuple avoiding G. Therefore, $G \in$ 2-AVOID. ▣

Proposition 5.1.3 g-AVOID is NP-complete for $g \geq 3$ and so AVOID is NP-complete.

Proof: It is easy to see that 3-AVOID $\in$ NP. To prove that 3-AVOID is NP-complete, we show that 3-SAT $\leq_P$ 3-AVOID, since 3-SAT is NP-complete [19]. Let $\varphi = (l_{1,0} \vee l_{1,1} \vee l_{1,2}) \wedge \cdots \wedge (l_{k,0} \vee l_{k,1} \vee l_{k,2})$ be a formula that is an instance for 3-SAT with k clauses with three literals each. Build a k-partite graph $f(\varphi) = G$ with three vertices labeled $v_{i,0}, v_{i,1}$ and $v_{i,2}$ per partite set for $1 \leq i \leq k$. Let $\{v_{i,a_i}, v_{j,a_j}\}$ be an edge of G if and only if $i \neq j$ and $l_{i,a_i} = \neg l_{j,a_j}$. If $\varphi \in$ 3-SAT then there exists a valuation, say V, such that $V(\varphi) = T$. So for each $i \in [1, k]$ there is l_{i,a_i} such that $V(l_{i,a_i}) = T$. Select one vertex v_{i,a_i} per part such that $V(l_{i,a_i}) = T$ to obtain a collection $I = \{v_{1,a_1}, ..., v_{k,a_k}\}$ of k vertices of G. For $i \neq j$, if v_{i,a_i} and v_{j,a_j} both belong to I then $l_{i,a_i} \neq \neg l_{j,a_j}$ since $V(l_{i,a_i}) = V(l_{j,a_j}) = T$. Therefore, $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$, so I is an independent set of G. Consequently, $(a_1, ..., a_k)$ forms a k-tuple avoiding G, so $f(\varphi) = G \in$ 3-AVOID.

Now suppose $f(\varphi) = G \in$ 3-AVOID. Then there exists a set of vertices $I = \{v_{1,a_1}, ..., v_{k,a_k}\}$ that is an independent set of size k of G. Thus, for every pair of vertices $v_{i,a_i}, v_{j,a_j} \in I$ we have $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$ which means that $l_{i,a_i} \neq \neg l_{j,a_j}$. Then the valuation V such that $V(l_{i,a_i}) = T$ whenever $v_{i,a_i} \in I$ satisfies φ. Thus $\varphi \in$ 3-SAT. ▣

Proposition 5.1.4 2-ONE-COVER&AVOID and 2-COVER&AVOID are in P.

Proof: First we show that 2-COVER&AVOID is in P. Let $G \in \mathcal{G}_{k,2}$ and let $\{v_{i,a_i}, v_{j,a_j}\}$ be any pair of vertices of G such that $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$ and $i \neq j$. Define a new graph $G^{\{v_{i,a_i},v_{j,a_j}\}}$ to be the graph obtained from G by replacing each vertex in factors i and j with two copies of v_{i,a_i} and v_{j,a_j}, respectively, as well as a copy of their incident edges. Then we claim that $G^{\{v_{i,a_i},v_{j,a_j}\}} \in$ 2-AVOID if and only if there exists a k-tuple avoiding G that covers $\{v_{i,a_i}, v_{j,a_j}\}$.

Suppose that $(a_1, ..., a_k)$ is a k-tuple avoiding G that covers $\{v_{i,a_i}, v_{j,a_j}\}$. Then clearly $I = \{v_{1,a_1}, ..., v_{k,a_k}\}$ is an independent set of $G^{\{v_{i,a_i},v_{j,a_j}\}}$ and hence $G^{\{v_{i,a_i},v_{j,a_j}\}} \in$ 2-AVOID. Conversely, if $G^{\{v_{i,a_i},v_{j,a_j}\}} \in$ 2-AVOID then there exists a k-tuple $(a_1, ..., a_k)$ avoiding $G^{\{v_{i,a_i},v_{j,a_j}\}}$. Thus $I = \{v_{1,a_1}, ..., v_{k,a_k}\}$ is an independent set of $G^{\{v_{i,a_i},v_{j,a_j}\}}$. By construction of $G^{\{v_{i,a_i},v_{j,a_j}\}}$, the only choice for vertices from factors i and j are v_{i,a_i} and v_{j,a_j} respectively. Therefore, I is an independent set of $G^{\{v_{i,a_i},v_{j,a_j}\}}$ that covers $\{v_{i,a_i}, v_{j,a_j}\}$. It is easy to see that I is also an independent set of G.

We conclude that if $G^{\{v_{i,a_i},v_{j,a_j}\}} \in$ 2-AVOID for every non-edge $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$, then for every non-edge $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$ there exists a k-tuple avoiding G that covers $\{v_{i,a_i}, v_{j,a_j}\}$, in which case we have $G \in$ 2-COVER&AVOID. If, on the other hand, $G^{\{v_{i,a_i},v_{j,a_j}\}} \notin$ 2-AVOID for some non-edge $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$, then there is no k-tuple avoiding G that covers $\{v_{i,a_i}, v_{j,a_j}\}$ in which case $G \notin$ 2-COVER&AVOID. Since 2-AVOID $\in$ P we have shown that a polynomial number of calls (at most $|E(\overline{G})| \leq 2k(k-1)$ calls) to a polynomial-time algorithm that solves 2-AVOID solves 2-COVER&AVOID, so 2-COVER&AVOID $\in$ P.

Now, we can similarly show that 2-ONE-COVER&AVOID is in P. Let $G \in \mathcal{G}_{k,2}$ be an instance for 2-ONE-COVER&AVOID. Let $G^{\{v_{i,a_i}\}}$ be the graph obtained from G by replacing both vertices of the i^{th} factor with copies of v_{i,a_i} and a copy of the edges incident with v_{i,a_i} in G. In a similar way, we can show that $G \in$ 2-ONE-COVER&AVOID if and only if $G^{\{v_{i,a_i}\}} \in$ 2-AVOID for every vertex $v_{i,a_i} \in V(G)$ such that there exists v_{j,a_j} with $i \neq j$ such that $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$. From this it follows that 2-ONE-COVER&AVOID $\in$ P. ▣

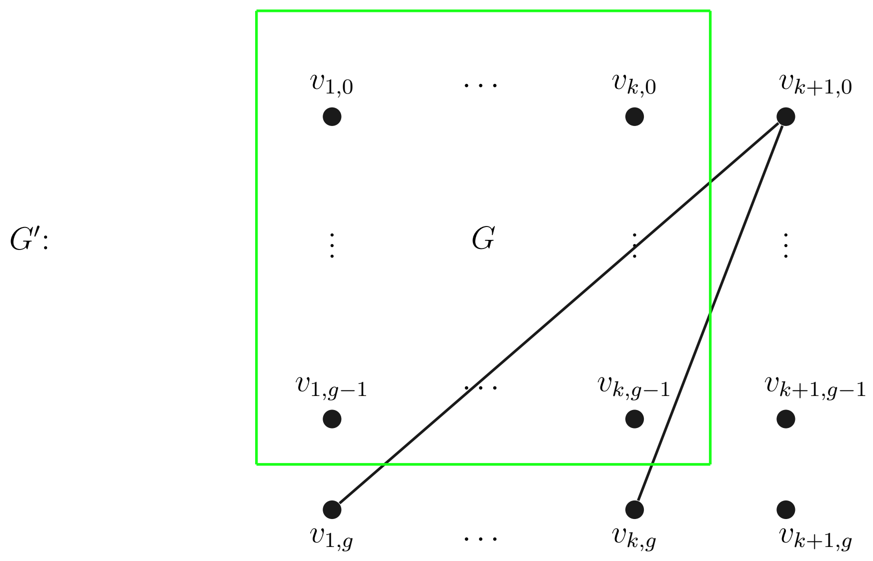


Figure 5.1: The graph G' of Proposition 5.1.5

Proposition 5.1.5 $(g+1)$-ONE-COVER&AVOID and $(g+2)$-COVER&AVOID are NP-complete for $g \geq 3$, and so are ONE-COVER&AVOID and COVER&AVOID.

Proof: Let $g \geq 3$ be a positive integer. It is easy to see that $(g+1)$-ONE-COVER&AVOID belongs to the class NP. To prove that $(g+1)$-ONE-COVER&AVOID is NP-complete, we show that g-AVOID $\leq_P$ $(g+1)$-ONE-COVER&AVOID in the following way. Let G be an instance for g-AVOID. Append to G one new factor indexed by $k+1$ with vertices $v_{k+1,0}, v_{k+1,1}, ..., v_{k+1,g-1}$. Add a new vertex $v_{i,g}$ per factor $i \in [1, k+1]$. Add edges $\{v_{k+1,0}, v_{i,g}\}$ for $1 \leq i \leq k$. Refer to the new graph as G' (see Figure 5.1).

In G' we can cover v_{i,a_i} for $1 \leq i \leq k$ and $a_i \in [0, g]$ by a $(k+1)$-tuple T where $T_i = a_i$ and for $l \in [1, k+1] \setminus \{i\}$ we let $T_i = g$. Similarly we can cover $v_{k+1,i}$ for $1 \leq i \leq g$. To cover $v_{k+1,0}$ we are forced to use vertices of G that form a k-tuple avoiding G, in order to form a $(k+1)$-tuple avoiding G' that covers $v_{k+1,0}$. Thus $G' \in$ (g+1)-ONE-COVER&AVOID if and only if we can find a $(k+1)$-tuple T

covering $v_{k+1,0}$ and avoiding G' which in turn is equivalent to the first k components of T avoiding G.

Let $g \geq 4$ be a positive integer. We prove that $(g+1)$-COVER&AVOID is NP-complete by showing that g-ONE-COVER&AVOID $\leq_P$ $(g+1)$-COVER&AVOID. Let $G \in \mathcal{G}_{k,g}$ be an instance for g-ONE-COVER&AVOID. Build G' to be an instance for $(g+1)$-COVER&AVOID by appending to G one new factor indexed by $k+1$ with vertices $v_{k+1,0}, v_{k+1,1}, ..., v_{k+1,g-1}$. Add a new vertex $v_{i,g}$ to each factor $i \in [1,k]$. Furthermore, add edges joining $v_{k+1,0}$ to each $v_{i,g}$ for $1 \leq i \leq k$.

By construction, covering a pair $\{v_{i,a_i}, v_{j,a_j}\}$ for $1 \leq i < j \leq k+1$ where $(j, a_j) \neq (k+1, 0)$ while avoiding G' can be easily done using a $(k+1)$-tuple T with $T_i = a_i$, $T_j = a_j$ and $T_l = g$ for $l \in [1, k+1] \setminus \{i, j\}$.

To cover a pair of the form $\{v_{i,a_i}, v_{k+1,0}\}$ it is necessary to find a $(k+1)$-tuple T such that $T_i = a_i$, $T_j \neq g$ for all $j \in [1,k]$, and $T_{k+1} = 0$. This is equivalent to finding a k-tuple avoiding G that covers vertex v_{i,a_i}. Therefore, $G' \in (g+1)$-COVER&AVOID if and only if $G \in g$-ONE-COVER&AVOID. It is easy to verify that $(g+1)$-COVER&AVOID belongs to the class NP, and since g-ONE-COVER&AVOID is NP-complete for $g \geq 4$, we conclude that $(g+1)$-COVER&AVOID is NP-complete for $g \geq 4$. ▣

Proposition 5.1.6 g-CAFEN is NP-complete for $g \geq 5$, and so CAFEN is NP-complete.

Proof: It is easy to see that g-CAFEN belongs to the class NP. Therefore, to show that g-CAFEN is NP-complete for $g \geq 5$ it is sufficient to prove that g-COVER&AVOID $\leq_P$ g-CAFEN. Let G be an instance for g-COVER&AVOID. Then $G \in g$-COVER&AVOID if and only if there exists a CAFE(n, G) for some positive integer n. Since by Proposition 2.4.1 we have CAFEN$(G) \leq g^2\binom{k}{2}$ for any graph $G \in \mathcal{G}_{k,g}$ such that CAFEN$(G) \neq +\infty$, we must have $G \in g$-COVER&AVOID if and only if there exists a CAFE$(g^2\binom{k}{2}, G)$ if and only if $(G, g^2\binom{k}{2}) \in g$-CAFEN. ▣

The complexity results given in the following section use a reduction from the decision problem ECCN which Kou, Stockmeyer and Wong [22] prove to be NP-complete. Their result is given in the following proposition, and is based on a reduction from the NCC problem. As already shown, an NCC of a given graph is simply a vertex colouring of its complement, and is known to be NP-complete [19]. In the proof of the next proposition we show that NCCN polynomially reduces to ECCN, establishing the NP-completeness of ECCN.

Theorem 5.1.7 (Kou, Stockmeyer, and Wong [22]) ECCN is NP-complete.

Proof: It is easy to see that ECCN belongs to the class NP. We show that NCCN $\leq_P$ ECCN. Let (G, N) be an instance for NCCN. Suppose that G has ε edges and ν vertices. Label the vertices of G as $V(G) = \{v_1, ..., v_\nu\}$. Let G' be the graph obtained from G by adding $\varepsilon+1$ new vertices, which we label $u_1, u_2, ..., u_{\varepsilon+1}$, and edges joining each vertex u_i to all the vertices of the original graph G, for $1 \leq i \leq \varepsilon + 1$. Moreover, let $N' = N(\varepsilon + 1) + \varepsilon$. Then we claim that $(G, N) \in$ NCCN if and only if $(G', N') \in$ ECCN.

First, let $(G, N) \in$ NCCN. Then there is an NCC of G, say $\mathfrak{C} = \{C_1, ..., C_N\}$, that covers all the vertices of G. Then in G', we can cover the ν edges incident with vertex u_i by the cliques $C_1 \cup \{u_i\}, C_2 \cup \{u_i\}, ..., C_N \cup \{u_i\}$, for $1 \leq i \leq \varepsilon + 1$. The edges of G' having both ends in $V(G)$ can be covered by ε 2-cliques, namely, the ends of each edge of the original graph G. Therefore, $\theta'(G') \leq N(\varepsilon + 1) + \epsilon = N'$ and hence $(G', N') \in$ ECCN.

Next we show that if $(G, N) \notin$ NCCN then $(G', N') \notin$ ECCN. Suppose $(G, N) \notin$ NCCN. Then $\theta(G) \geq N + 1$. Then for each vertex u_i of G' where $1 \leq i \leq \varepsilon + 1$, we require enough cliques to cover all the edges incident with u_i. Since G' is constructed so that each u_i is adjacent to all of the vertices in $V(G) = \{v_1, ..., v_\nu\}$, and so that u_i is not adjacent to any other u_j for $i \neq j$, the only clique of G' which covers the edge $\{u_i, v_j\}$ must be of the form $\{u_i\} \cup C_j$ where C_j is a clique of the original graph G containing the vertex v_j. Indeed, since u_i is adjacent to every $v_j \in V(G)$ the fewest number of C_j's which do the trick is $\theta(G) \geq N + 1$. Thus we require at least $(\varepsilon + 1)(\theta(G)) \geq (\varepsilon + 1)(N + 1)$ cliques in order to cover all the edges of the form $\{u_i, v_j\}$ in G'. We conclude that $\theta'(G') \geq (\varepsilon+1)(N+1) = N(\varepsilon+1)+\varepsilon+1 \geq N'+1$ and so $(G', N') \notin$ ECCN. ▣

Orlin [31] also published another proof of the NP-completeness of the ECCN decision problem in 1977, which was discovered independently. Moreover, "the proofs are similar in nature although far from identical" [31]. Orlin's result addresses the complexity of determining the R-bicontent of a bipartite graph G, which is equal to the minimum number of complete bipartite subgraphs of G (with edge repetitions allowed) whose union includes all of the edges of G, and the R-content of a graph G, which is equal to its ECC number.

Theorem 5.1.8 (Orlin [31]) The following five problems are all NP-complete.

1. Determine the fewest number of cliques which cover all of the vertices of graph a G (decision problem: NCCN).
2. Determine the fewest number of complete bipartite subgraphs of a bipartite graph G which are sufficient to cover a specified subset of edges of G.
3. Determine the R-bicontent of a bipartite graph G.
4. Determine the R-content of a graph G, that is, determine $\theta'(G)$ (decision problem: ECCN).
5. Determine the minimum number of cliques of a graph G that cover a specified subset of edges $E' \subseteq E(G)$. That is, find the partial ECC number of (G, E').

5.2 NP-Completeness of 2-CAFEN

In contrast to the results by Danziger et. al. [11] that the decision problems related to the feasibility of CAFEs with binary alphabets, namely 2-AVOID, 2-ONE-COVER&AVOID, and 2-COVER&AVOID, all belong to the class P, we now show that the decision problem related to the optimization of CAFEs for $g = 2$, 2-CAFEN, is NP-complete. We use the closely related ECC problem, which was shown to be NP-complete by Kou, Stockmeyer, and Wong [22] (see Theorem 5.1.7), and independently by Orlin [31]. Our proof polynomial-time reduces 2-CAFEN from ECCN, using Construction 1 defined below.

Remark 5.2.1 Throughout this chapter we always assume that the original graph G is nonempty. That is, we assume that G contains at least one edge. Since an empty graph, $\overline{K_\nu}$, satisfies $\theta'(\overline{K_\nu}) = 0$, we can verify that $(\overline{K_\nu}, N) \in$ ECCN in polynomial time, for any positive integer N. It is therefore less trivial to consider nonempty graphs.

Construction 1 Let G be a simple nonempty graph on ν vertices, and let $k \geq 2$ be the number of non-isolated vertices of G. We construct another simple graph, G_{UV}, on $2(k+2)$ vertices such that $\theta'(G) + 2 = \text{CAFEN}(G_{UV})$.

Step 1. If G contains any isolated vertices, remove them to obtain a new graph G_k on k non-isolated vertices, which we denote by $\{v_1, v_2, ..., v_k\}$. If G contains no isolated vertices, then $\nu = k$ and $G = G_k$.

Step 2. Take the complement, $\overline{G_k}$, of G_k.

Step 3. Add two extra vertices, v_{k+1} and v_{k+2}, and add edges joining v_{k+1} to each v_i for $1 \leq i \leq k$. Moreover, join the vertex v_{k+2} to all vertices v_i for $1 \leq i \leq k+1$. Refer to the resulting graph as G_V and denote its vertex set as $V = \{v_1, v_2, ..., v_k, v_{k+1}, v_{k+2}\}$.

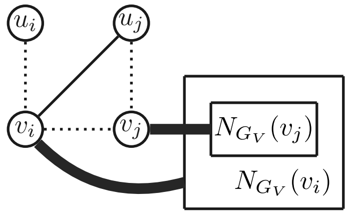


Figure 5.2: Across Edge Rule

The across edge $\{v_i, u_j\} \in E(G_{UV})$ if and only if $N_{G_V}(v_j) \subseteq N_{G_V}(v_i)$. The dotted lines represent non-edges of G_{UV} and the thick edges represent all the edges incident to v_i and v_j, respectively, joining v_i and v_j to their respective neighbours. Whenever $N_{G_V}(v_j) \subseteq N_{G_V}(v_i)$, the constraints imposed on the point v_i are stricter than, or equal to the constraints imposed on the point v_j. Thus, if a k-tuple exists that covers the point v_i (the one value of the i^{th} factor), we can always choose v_j for the value of the j^{th} factor.

Step 4. Construct the graph G_{UV} from G_V by adding the vertex set $U = \{u_1, u_2, ..., u_{k+2}\}$ to G_V, and adding edges according to the **Across Edge Rule**.

For convenience we also use the following definition.

Definition 5.2.2 Any edge joining a vertex in U to a vertex in V, we refer to as an **across edge**. Any pair of vertices $u_i \in U$ and $v_j \in V$ which are not joined to each other by an edge we refer to as an **across non-edge**.

Across Edge Rule: We add the across edge $\{v_i, u_j\}$ to G_{UV} if and only if $i \neq j$ and the neighbourhood of v_j in G_V is a subset of the neighbourhood of v_i in G_V. That is, $\{v_i, u_j\} \in E(G_{UV})$ if and only if $i \neq j$ and $N_{G_V}(v_j) \subseteq N_{G_V}(v_i)$. See Figure 5.2.

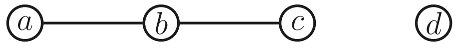
In summary, from an arbitrary simple graph G, Construction 1 gives G_{UV}, having vertex set $V(G_{UV}) = U \cup V$ and the edge set $E(G_{UV})$ comprised of the edges of $\overline{G_k}$ as well as the edges joining v_{k+1} to each v_i for $1 \leq i \leq k$, the edges joining v_{k+2} to each v_i, $1 \leq i \leq k+1$, and any across edge added according to the above rule. This is done via a sequence of graphs

$$G \to G_k \to \overline{G_k} \to G_V \to G_{UV}$$

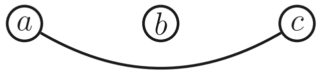
where the notation $G \to G'$ denotes that G' is constructed from G, by some step of Construction 1.

We illustrate this process with a simple example.

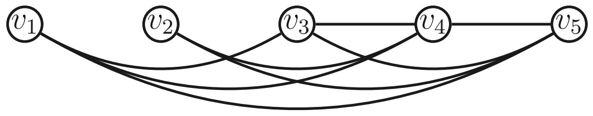
Example 1 Consider the following graph, G, on $\nu = 4$ vertices.

G :


In this case $k = 3$. We remove the isolated vertex d. Then we take the complement and obtain $\overline{G_k}$.

$\overline{G_k}$:


We then add the vertices v_4 and v_5 and relabel the vertices a, b, and c to v_1, v_2, and v_3, respectively. We join the vertex v_4 to each of the vertices v_1, v_2, v_3, and we join the vertex v_5 to each of the vertices v_1, v_2, v_3, v_4 which results in the graph G_V.

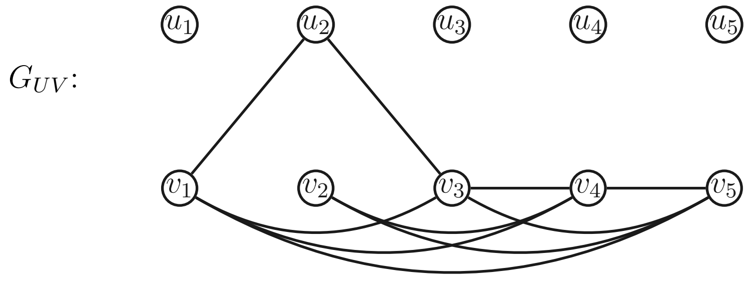
G_V:


We list the neighbourhoods of the vertices in G_V:

- $N_{G_V}(v_1) = \{v_3, v_4, v_5\}$
- $N_{G_V}(v_2) = \{v_4, v_5\}$
- $N_{G_V}(v_3) = \{v_1, v_4, v_5\}$
- $N_{G_V}(v_4) = \{v_1, v_2, v_3, v_5\}$
- $N_{G_V}(v_5) = \{v_1, v_2, v_3, v_4\}$

Finally we add the vertex set $U = \{u_1, ..., u_5\}$ and add all the required across edges, which yields the final graph G_{UV}. According to the Across Edge Rule, we add the following edges.

- $N_{G_V}(v_2) \subseteq N_{G_V}(v_1) \Rightarrow \{v_1, u_2\} \in E(G_{UV})$
- $N_{G_V}(v_2) \subseteq N_{G_V}(v_3) \Rightarrow \{v_3, u_2\} \in E(G_{UV})$



Now, let us observe a few properties of G_{UV}.

Proposition 5.2.3 Let G be a simple nonempty graph, and let G_{UV} be the graph obtained from G using Construction 1. Let $i \in [1, k+2]$. Then,

1. $\{v_i, u_{k+1}\} \notin E(G_{UV})$, and $\{v_i, u_{k+2}\} \notin E(G_{UV})$,
2. $\{u_i, v_{k+1}\} \notin E(G_{UV})$ and $\{u_i, v_{k+2}\} \notin E(G_{UV})$,
3. if $i \notin \{k+1, k+2\}$, then $N_{G_V}(v_i) \neq V \setminus \{v_i\}$.

Proof: If $\{v_i, v_j\} \in E(G_{UV})$ then $v_j \in N_{G_V}(v_i)$ and $v_j \notin N_{G_V}(v_j)$. So, $N_{G_V}(v_i) \nsubseteq N_{G_V}(v_j)$, which implies $\{v_j, u_i\} \notin E(G_{UV})$. Since $\{v_i, v_{k+1}\} \in E(G_{UV})$ for each $i \in [1, k+2] \setminus \{k+1\}$ and $\{v_i, v_{k+2}\} \in E(G_{UV})$ for each $i \in [1, k+1]$, we obtain 1. and 2. The fact that G_k contains no isolated vertices implies 3. ▣

The following result gives an equivalent statement for the Across Edge Rule.

Corollary 5.2.4 For two distinct vertices $v_i, v_j \in V$, $\{v_i, u_j\} \in E(G_{UV})$ if and only if $\{v_i, v_j\} \in E(G_k)$ and $N_{G_k}(v_i) \setminus \{v_j\} \subseteq N_{G_k}(v_j) \setminus \{v_i\}$.

Lemma 5.2.5 Let G be a simple graph. Then $I = \{u_1, u_2, ..., u_k, u_{k+1}, v_{k+2}\}$ is the only independent set of size $k+2$ of G_{UV} that contains both vertices u_{k+1} and v_{k+2}, and $I' = \{u_1, u_2, ..., u_k, v_{k+1}, u_{k+2}\}$ is the only independent set of size $k+2$ of G_{UV} that contains the vertices v_{k+1} and u_{k+2}.

Proof: By construction, $\{u_i, u_j\} \notin E(G_{UV})$ for $1 \leq i < j \leq k+2$. Hence $\{u_1, u_2, ..., u_k, u_{k+1}\}$ is an independent set of G_{UV}. By Proposition 5.2.3, we know that $\{v_{k+2}, u_i\} \notin E(G_{UV})$ for $1 \leq i \leq k+1$. Thus, $I = \{u_1, u_2, ..., u_k, u_{k+1}, v_{k+2}\}$ is an independent set of size $k+2$. To show that I is the only such independent set, simply observe that v_{k+2} is adjacent to all v_i for $1 \leq i \leq k+1$. The argument for I' is the same. ▣

Remark 5.2.6 The graph G_{UV} corresponds to a forbidden edges graph for an instance of the binary CAFE problem having $k+2$ factors with two values each. More specifically, we have $G_{UV} \in \mathcal{G}_{k+2,2}$, and the vertex u_i of G_{UV} represents the zero value for the i^{th} factor of the CAFE, and the corresponding vertex, v_i, represents the one value for the i^{th} factor. According to the notation used in [11] and in Definition 2.2.1, the vertex v_i of G_{UV} represents the vertex $v_{i,1}$ and the vertex u_i represents $v_{i,0}$ where $1 \leq i \leq k+2$. Given an interaction $I = \{(i, a_i), (j, a_j)\}$, we refer to I as a **zero-zero, zero-one,** or **one-one** interaction, if $a_i = a_j = 0$, $\{a_i, a_j\} = \{0, 1\}$, or $a_i = a_j = 1$, respectively. Moreover, if I is not forbidden, that is, if $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G_{UV})$, then we call I a **required** interaction.

Lemma 5.2.7 Let G be a simple graph and let A be a CAFE(n, G_{UV}). Then each row of A, R_i, corresponds to an independent set of G_{UV}, namely $I_i = \{v_j | R_i(j) = 1\} \cup \{u_j | R_i(j) = 0\}$ and $|I_i| = k+2$.

Proof: By Remark 5.2.6, the vertex u_j is equivalent to the vertex $v_{j,0}$, and v_j is equivalent to $v_{j,1}$. If $R_i = (R_i(1), ..., R_i(k+2))$ is a row of a CAFE(n, G_{UV}), then by definition, R_i forms a $(k+2)$-tuple avoiding G_{UV}. By Lemma 2.3.2, $I_i = \{v_j | R_i(j) =$

$1\} \cup \{u_j | R_i(j) = 0\}$ is an independent set of G_{UV} and $|I_i| = k+2$. ▣

Lemma 5.2.8 Let G be a simple graph. A required one-one interaction of the graph G_{UV} corresponds to an edge of the original graph G.

Proof: Let $\{(i,1),(j,1)\}$ be a required one-one interaction of G_{UV}. Then $\{v_i, v_j\} \notin E(G_{UV})$ and $i \neq j$. Since v_{k+1} is adjacent to each vertex in $V \setminus \{v_{k+1}\}$, and similarly, v_{k+2} is adjacent to every vertex in $V \setminus \{v_{k+2}\}$, we must have $i, j \notin \{k+1, k+2\}$. Therefore, the non-edge $\{v_i, v_j\}$ corresponds to a non-edge between two vertices in $V \setminus \{v_{k+1}, v_{k+2}\}$, which in turn, corresponds exactly to a non-edge of the graph $\overline{G_k}$. Consequently, $\{v_i, v_j\} \in E(G_k)$, and so $\{(i,1),(j,1)\}$ corresponds to an edge of the original graph G. ▣

We now show that Construction 1 produces G_{UV}, a forbidden edges graph for an instance of the binary CAFE problem, that is consistent.

Proposition 5.2.9 Let G be a simple graph. Then the graph G_{UV} is consistent.

Proof: By Proposition 2.3.6, there are only two possible forbidden induced subgraphs for binary CAFEs. Since G_{UV} is a loopless binary CAFE graph and also has the property that none of its zero vertices u_i are joined by any edge, there are only three possibilities for an inconsistency to occur.

First, by condition 2. of Proposition 2.3.6, if for some $i \neq j$ we have $\{v_i, u_j\} \in E(G_{UV})$ and $\{v_i, v_j\} \in E(G_{UV})$. However, this would never occur because the across edge $\{v_i, u_j\}$ would not be added when computing G_{UV} since $N_{G_V}(v_j) \nsubseteq N_{G_V}(v_i)$.

By condition 1. of Proposition 2.3.6, G_{UV} would not be consistent, if for three distinct indices $i, j, l \in [1, k+2]$ we have $\{v_i, u_l\} \in E(G_{UV})$, $\{v_j, v_l\} \in E(G_{UV})$, and $\{v_i, v_j\} \notin E(G_{UV})$. However, since the across edge $\{v_i, u_l\}$ is an edge of G_{UV}, we know that $N_{G_V}(v_l) \subseteq N_{G_V}(v_i)$. This is a contradiction since $v_j \in N_{G_V}(v_l)$ but $v_j \notin N_{G_V}(v_i)$. Thus, G_{UV} cannot contain such an inconsistency.

By condition 1. of Proposition 2.3.6, we could also have an inconsistency if for three distinct indices $i, j, l \in [1, k+2]$ we have $\{v_i, u_l\} \in E(G_{UV})$, $\{v_l, u_j\} \in E(G_{UV})$, but $\{v_i, u_j\} \notin E(G_{UV})$. By the Across Edge Rule, we have $N_{G_V}(v_l) \subseteq N_{G_V}(v_i)$ and $N_{G_V}(v_j) \subseteq N_{G_V}(v_l)$. Therefore, we have $N_{G_V}(v_j) \subseteq N_{G_V}(v_i)$, and so $\{v_i, u_j\}$ must be an edge of G_{UV}. Therefore, G_{UV} is consistent. ▣

Now that we know G_{UV} is consistent, it remains to show that CAFEN$(G_{UV}) = \theta'(G) + 2$. For this purpose we need the following corollary which tells us that it is always possible to build a row for the CAFE by taking ones for the values of the factors corresponding to the vertices in a maximal clique of G and zeros for all the remaining factors.

Proposition 5.2.10 Let C be a clique of G that is maximal with respect to set inclusion such that $|C| > 1$. Then the set of vertices $I = \{v_i \in V | v_i \in C\} \cup \{u_i \in U | v_i \notin C\}$ forms an independent set of size $k+2$ of the graph G_{UV}.

Proof: Let C be clique of G that is maximal (with respect to set inclusion) such that $|C| > 1$. Since $|C| > 1$, C must contain the ends of at least one edge of G, thus, C is a clique of G_k, the graph obtained from G by removing all isolated vertices (see Step 1. of Construction 1), that is maximal. Therefore, in the complement, $\overline{G_k}$, C is an independent set that is maximal. From $\overline{G_k}$, we obtain G_V by adding the two vertices v_{k+1} and v_{k+2}, which are joined by an edge to every other vertex in V (see Step 3. of Construction 1). In particular, this means that v_{k+1} and v_{k+2} are both adjacent to each of the vertices in C. Thus, the independent set C of G_k cannot be extended to include either v_{k+1} or v_{k+2} in G_V. Thus C is a maximal independent set of G_V. To show that the set of vertices $I = \{v_i \in V | v_i \in C\} \cup \{u_i \in U | v_i \notin C\}$ is an independent set of G_{UV}, we need to prove that there are no across edges of the form $\{v_i, u_j\}$ such that $v_i \in C$ and $v_j \notin C$. Such an edge would prevent us from extending C to include the vertices u_j such that $v_j \notin C$.

Let $v_i \in C$ and let $v_j \notin C$. Since $v_j \notin C$ and C is maximal there must be an edge in G_V joining v_j to at least one vertex in C. Suppose v_i is joined by an edge to v_j. Then if $\{v_i, u_j\}$ is an across edge of G_{UV}, then G_{UV} would contain an inconsistency, contradicting Proposition 5.2.9. Suppose that v_j is adjacent to some other vertex $v_l \in C$, $v_l \neq v_i$. We know that $\{v_i, v_l\} \notin E(G_{UV})$ because $v_i, v_l \in C$ and C is an independent set. Thus, if $\{v_i, u_j\}$ is an across edge of G_{UV} we would again have an inconsistency in G_{UV}, contradicting Proposition 5.2.9. We conclude that for every $v_i \in C$ and each $v_j \in V$ such that $v_j \notin C$ the graph G_{UV} does not contain the across edge $\{v_i, u_j\}$. Therefore, we can extend the independent set C to include each such vertex $u_j \in U$ whenever $v_j \notin C$. It is easy to see that I as defined forms an independent set of G_{UV} and $|I| = k + 2$ since $|V| = k + 2$. ▣

Theorem 5.2.11 Let G be a simple graph and let G_{UV} be the graph obtained from G by applying Construction 1. Then $\theta'(G) + 2 = \text{CAFEN}(G_{UV})$.

Proof: First we show that $\text{CAFEN}(G_{UV}) \leq \theta'(G) + 2$. Suppose for some positive integer N, G satisfies $\theta'(G) = N$. Let $\mathfrak{C} = \{C_1, ..., C_N\}$ be an optimal clique-maximal ECC of G. By Proposition 5.2.10, for $1 \leq i \leq N$, we can build a row, $R_i = (R_i(1), R_i(2), ..., R_i(k+2))$, corresponding to each clique $C_i \in \mathfrak{C}$, by taking $R_i(j) = 1$ whenever $v_j \in C_i$ and $R_i(j) = 0$ whenever $v_j \notin C_i$.

Since G_{UV} is constructed so that v_{k+1} and v_{k+2} are both joined by an edge to every other vertex in V, we see that covering all interactions of the form $\{v_i, v_j\}$ where $i, j \in [1, k]$ is sufficient to cover all the required one-one interactions of G_{UV}. Since the required one-one interactions of G_{UV} correspond exactly to the edges of the original graph G, we see that the N rows, $R_1, ..., R_N$ do indeed cover the required one-one interactions of G_{UV}.

Note that every row R_i corresponding to the clique $C_i \in \mathfrak{C}$ must also cover the interaction $\{u_{k+1}, u_{k+2}\}$ since $v_{k+1} \notin C_i$ and $v_{k+2} \notin C_i$ for each $C_i \in \mathfrak{C}$.

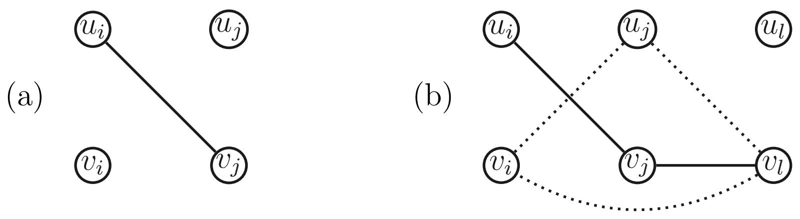


Figure 5.3: Theorem 5.2.11: Case 1: $\{v_i, v_j\} \notin E(G_{UV})$

Now, we build another row, R_{N+1}, corresponding to the independent set $I_{N+1} = \{u_1, ..., u_{k+1}, v_{k+2}\}$, possible by Lemma 5.2.5. The row R_{N+1} is sufficient to cover all the required zero-zero interactions between the vertices $u_1, ..., u_{k+1}$, as well as all the required interactions of the form $\{u_i, v_{k+2}\}$ where $1 \leq i \leq k+1$.

We also build the row R_{N+2} corresponding to the independent set $I_{N+2} = \{u_1, ..., u_k, v_{k+1}, u_{k+2}\}$, possible by Lemma 5.2.5. The row R_{N+2} is sufficient to cover all the required zero-zero interactions of the form $\{u_i, u_{k+2}\}$, as well as all the required interactions of the form $\{u_i, v_{k+1}\}$ where $1 \leq i \leq k$ or $i = k+2$.

Up to this point we have $N+2$ rows sufficient to cover all required zero-zero interactions and all required one-one interactions of G_{UV}. Therefore, the only concern left is that some required zero-one interaction has been missed. That is, some of the across non-edges of the form $\{v_i, u_j\}$ such that $1 \leq i \leq k$ and $1 \leq j \leq k+2$ might be left uncovered at this point. We claim that the rows $R_1, R_2, ..., R_N, R_{N+1}, R_{N+2}$ are sufficient to cover all the across non-edges of G_{UV}. Let $v_i \in V$ and $u_j \in U$ be two vertices that are not joined to each other by an edge in G_{UV}. We have two possible cases.

Case 1: $\{v_i, v_j\} \notin E(G_{UV})$ (see Figure 5.3 (a)). By the Across Edge Rule, we know that $N_{G_V}(v_j) \nsubseteq N_{G_V}(v_i)$, otherwise $\{v_i, u_j\}$ would be an edge of G_{UV}. Hence, there exists a vertex $v_l \in N_{G_V}(v_j)$ such that $v_l \notin N_{G_V}(v_i)$ (see Figure 5.3 (b)). This means that $\{v_i, v_l\}$ is a non-edge corresponding to a required one-one interaction

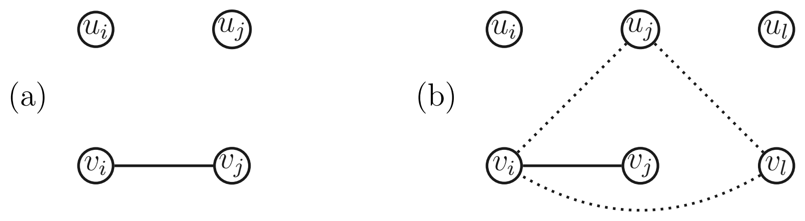


Figure 5.4: Theorem 5.2.11: Case 2: $\{v_i, v_j\} \in E(G_{UV})$

and therefore was already covered by some row R_p, $p \in [1, N]$ (the dotted edges in Figure 5.3 (b)). The row R_p must also cover $\{v_i, u_j\}$.

Case 2: $\{v_i, v_j\} \in E(G_{UV})$ (see Figure 5.4 (a)). If $i = k+1$, we are done since row R_{N+2} covers all required interactions of the form $\{v_{k+1}, u_j\}$ where $1 \leq j \leq k$ or $j = k+2$. Similarly, if $i = k+2$ we are done since the row R_{N+1} covers all required interactions of the form $\{v_{k+2}, u_j\}$ where $1 \leq j \leq k+1$.

If $1 \leq i \leq k$ then by Proposition 5.2.3 we know that $N_{G_V}(v_i) \neq V \setminus \{v_i\}$. So there is a vertex $v_l \in V$ such that $v_l \notin N_{G_V}(v_i)$. Thus $\{v_i, v_l\} \notin E(G_{UV})$ is a required one-one interaction and its coverage by one of the rows R_p, $p \in [1, N]$ forces the across non-edge $\{v_i, u_j\}$ to be covered (see Figure 5.4 (b)).

In summary, in both cases we see that the $N+2$ rows $R_i, 1 \leq i \leq N+2$ cover all the across non-edges of the graph G_{UV} as well as all the non-edges between vertices in V, and the non-edge $\{u_{k+1}, u_{k+2}\}$. The single row R_{N+1} covers required zero-zero interactions of the form $\{u_i, u_{k+1}\}$ where $1 \leq i \leq k$ and is the only row that covers the required interaction $\{v_{k+2}, u_{k+1}\}$. The single row R_{N+2} covers required zero-zero interactions of the form $\{u_i, u_{k+2}\}$ where $1 \leq i \leq k$ and is the only row that covers the required interaction $\{v_{k+1}, u_{k+2}\}$. Together these $N+2$ rows are sufficient to produce a CAFE$(N+2, G_{UV})$. Hence, we have CAFEN$(G_{UV}) \leq N+2$ whenever $\theta'(G) = N$.

Next we show that $\theta'(G) + 2 \leq \text{CAFEN}(G_{UV})$. Suppose we have an optimal CAFE(N, G_{UV}) with N rows. By Lemma 5.2.5, the only row which can cover the interaction $\{u_{k+1}, v_{k+2}\}$ is the one corresponding to the independent set $I_1 = \{u_1, ..., u_{k+1}, v_{k+2}\}$. Call this row R_1. Furthermore, the only row which can cover the interaction $\{v_{k+1}, u_{k+2}\}$ is the one corresponding to the independent set $I_2 = \{u_1, ..., u_k, v_{k+1}, u_{k+2}\}$. Call this row R_2. Since neither R_1 nor R_2 cover any one-one interactions, we observe that the remaining $N-2$ rows of the optimal CAFE(n, G_{UV}) must be sufficient to cover all the one-one interactions of G_{UV}. Call these remaining $N-2$ rows $R_3, ..., R_N$, and name the corresponding independent sets of G_{UV} of size $k+2$, $I_3, ..., I_N$, respectively. Then for $3 \leq i \leq N$, we have an independent set $C_i \subseteq I_i$ where $C_i = \{v_j | v_j \in I_i \text{ and } j \in [1, k]\}$. Thus, each C_i is an independent set of G_{UV} containing only vertices from the set $\{v_1, ..., v_k\}$. In other words, each C_i corresponds to a clique of the original graph G, and $C_3, ..., C_N$ cover all the edges of G. Therefore, $N - 2 \geq \theta'(G)$. Equivalently, $\theta'(G) + 2 \leq \text{CAFEN}(G_{UV})$. ▣

Corollary 5.2.12 2-CAFEN is NP-complete.

Proof: Let N be a positive integer and let G be a simple graph. By Theorem 5.2.11, we have $(G, N) \in$ ECCN if and only if $(G_{UV}, N+2) \in$ 2-CAFEN. Since it is easy to see that 2-CAFEN $\in$ NP, and ECCN is NP-complete, we need only observe that G_{UV} can be computed in polynomial time with respect to the size of the input graph G. Removing any isolated vertices takes time $O(\nu)$. Adding the vertices v_{k+1} and v_{k+2} and joining v_{k+1} and v_{k+2} to each other vertex in V takes time $O(k)$. It takes time $O(k)$ to add the vertex set U. For each of the k vertices $v_i \in V$, it takes time $O(k^2)$ to find $N_{G_V}(v_i)$. Then, there are $O(k^2)$ pairs of vertices in V for which we must compare their neighbourhoods, taking time $O(k^2)$ each. Since $k \leq \nu$, the overall time required to compute G_{UV} is $O(\nu^4)$. Therefore, ECCN $\leq_P$ 2-CAFEN, so 2-CAFEN is NP-complete. ▣

Given the intractability of 2-CAFEN, the next step is to look for decent approximation algorithms. For example, given a graph $G \in \mathcal{G}_{k,2}$, is there a polynomial-time algorithm A such that the output of A, denoted by $A(G)$, is guaranteed to satisfy $A(G) \leq c \cdot \text{CAFEN}(G)$ for a constant c ? Unfortunately, the answer is no. We use the following result given by Lund and Yannakakis [27], to relate the approximability of the ECC number to that of the binary CAFE number.

Theorem 5.2.13 (Lund and Yannakakis [27]) There exists a $\delta > 0$ such that there does not exist a polynomial-time approximation algorithm A that satisfies $A(G) \leq \nu^{\delta}\theta'(G)$ for all simple graphs G on ν vertices, unless P = NP.

Proposition 5.2.14 There exists a $\delta' > 0$ such that there does not exist a polynomial-time approximation algorithm A' that satisfies $A'(G) \leq k^{\delta'}\text{CAFEN}(G)$ for all $G \in \mathcal{G}_{k,2}$ and for all $k \geq 4$, unless P = NP.

Proof: Let $\delta > 0$ be the constant from Theorem 5.2.13, let $\delta' = \frac{\delta}{3} > 0$ and suppose there exists a polynomial-time approximation algorithm A' such that $A'(G) \leq k^{\delta'}\text{CAFEN}(G)$ for all $G \in \mathcal{G}_{k,2}$. Let G be a graph for which we want to approximate $\theta'(G)$. We can assume without loss of generality that $\nu = |V(G)| \geq max\{2, 2^{\frac{3}{\delta}}\}$ and $\theta'(G) \geq 2$, as the other cases can be dealt with in polynomial time. Then, we can apply A' to the graph G_{UV}, obtained from G by Construction 1, and we obtain a polynomial-time approximation algorithm A defined by

$$\begin{aligned}
A(G) &= A'(G_{UV}) \\
&\leq k^{\delta'}\text{CAFEN}(G_{UV}) && \text{by assumption;} \\
&\leq k^{\delta'}(\theta'(G)+2) && \text{by Theorem 5.2.11;} \\
&\leq k^{\delta'}(2\theta'(G)) && \text{since } \theta'(G) \geq 2; \\
&\leq 2(\nu+2)^{\delta'}\theta'(G) && \text{since } k \leq \nu+2; \\
&\leq 2(\nu^2)^{\delta'}\theta'(G) && \text{since } \nu+2 \leq \nu^2 \text{ for all } \nu \geq 2; \\
&\leq (\nu^{\frac{\delta}{3}})(\nu^{2\delta'})\theta'(G) && \text{since } \nu \geq 2^{\frac{3}{\delta}} \text{ by assumption;} \\
&= (\nu^{2\delta'+\frac{\delta}{3}})\theta'(G) \\
&= \nu^{\delta}\theta'(G) && \text{since } \delta' = \frac{\delta}{3}.
\end{aligned}$$

By Theorem 5.2.13, we must have P = NP. ▣

5.3 NP-Completeness of g-CAFEN

In, Danziger et. al. [11] the NP-completeness of the decision problems for 2-CAFEN, 3-CAFEN, and 4-CAFEN are left unresolved (see Theorem 5.1.1). In the previous section, we showed 2-CAFEN is NP-complete. The following result proves the NP-completeness of g-CAFEN, for $g \geq 3$. Recall that the decision problem g-CAFEN is defined

$$g\text{-CAFEN} := \{(G, N) \in \mathcal{G}_{k,g} \times \mathbb{Z} \mid \text{CAFEN}(G) \leq N\}.$$

Proposition 5.3.1 For $g \geq 2$ we have g-CAFEN $\leq_P$ $(g+1)$-CAFEN.

Proof: Let $G \in \mathcal{G}_{k,g}$ be a graph that is an instance for g-CAFEN. Without loss of generality, assume the vertices of G are labeled as $v_{i,a}$ where $i \in [1, k]$ and $a \in [0, g-1]$.

Construct a new graph G' from G as follows. Add a new vertex, $v_{i,g}$, to each factor $i \in [1, k]$. Add edges of the form $\{v_{i,g}, v_{j,a}\}$ for all $i \neq j$, $i, j \in [1, k]$ and for all $a \in [0, g-1]$. Moreover, add edges of the form $\{v_{i,g}, v_{j,g}\}$ for all $i \neq j$, $i, j \in [1, k]$. So $G' \in \mathcal{G}_{k,g+1}$.

Clearly the non-forbidden interactions of G' correspond exactly to the non-forbidden interactions of G, thus CAFEN(G) = CAFEN(G'). Therefore $(G, N) \in$ g-CAFEN if and only if $(G', N) \in (g+1)$-CAFEN. It is easy to see that G' can be computed from G in polynomial time with respect to the size of the graph G, and thus, g-CAFEN $\leq_P$ $(g+1)$-CAFEN. ▣

Corollary 5.3.2 For $g \geq 2$ the decision problem associated with the language g-CAFEN is NP-complete.

Proof: It is easy to see that for any $g \geq 2$, g-CAFEN belongs to the class NP. We prove the claim by induction on g, using Corollary 5.2.12 for the base case $g = 2$, and Proposition 5.3.1 for the induction step. ▣

5.4 NP-Completeness of PARTIAL-CAFEN

Using the NP-completeness of CAFEN and g-CAFEN for $g \geq 2$, established in the previous sections, we now address the complexity of the following language, which addresses the existence of a partial CAFE of a specified size. Recall that we use the notation $E_{\mathcal{I}}$ to denote the subset of edges of $\overline{G^{|}}$ corresponding to the necessary interaction set $\mathcal{I}$. That is, $E_{\mathcal{I}} = \{\{v_{i,a_i}, v_{j,a_j}\} | \{(i, a_i), (j, a_j)\} \in \mathcal{I}\} \subseteq E(\overline{G^{|}})$.

$$\text{PARTIAL-CAFEN} = \{(G, N, E_{\mathcal{I}}) \in \mathcal{G}_{(g_1,\ldots,g_k)} \times \mathbb{Z} \times \mathscr{P}(E(\overline{G^{|}})) | \text{ PCAFEN}(G, \mathcal{I}) \leq N\}$$

We also define the language g-PARTIAL-CAFEN to be the subset of PARTIAL-CAFEN, where the graph input G belongs to the family $\mathcal{G}_{k,g}$.

Proposition 5.4.1 g-PARTIAL-CAFEN is NP-complete for $g \geq 2$, and so is PARTIAL-CAFEN.

Proof: Let $g \geq 2$. We can easily verify that PARTIAL-CAFEN and g-PARTIAL-CAFEN belong to the class NP. Moreover, g-CAFEN $\leq_P$ g-PARTIAL-CAFEN since $(G, N) \in g$-CAFEN if and only if $(G, N, E(\overline{G^{|}})) \in g$-PARTIAL-CAFEN. Since, g-CAFEN is NP-complete we have that g-PARTIAL-CAFEN is also NP-complete. ▣

5.5 NP-Completeness of UNIFORM-ECCN

We now look at the language that decides if a given graph admits a k-uniform ECC, defined below:

$$\text{UNIFORM-ECCN} = \{(G, k, N) \in \mathbb{G} \times \mathbb{Z} \times \mathbb{Z} \mid \theta'_k(G) \leq N\}.$$

Proposition 5.5.1 UNIFORM-ECCN is NP-complete.

Proof: Let (G, N) be a graph-integer pair that is an instance for 2-CAFEN. So $G \in G_{k,2}$ for some positive integer $k \geq 2$. If $(G, N) \in$ 2-CAFEN then CAFEN$(G) = \theta'_k(\overline{G^{|}}) \leq N$. Consequently, we have $(\overline{G^{|}}, k, N) \in$ UNIFORM-ECCN. Otherwise, if $(G, N) \notin$ 2-CAFEN then CAFEN$(G) > N$ hence $\theta'_k(G) > N$ and so $(\overline{G^{|}}, k, N) \notin$ UNIFORM-ECCN. Thus 2-CAFEN $\leq_P$ UNIFORM-ECCN. It is easy to verify that $(\overline{G^{|}}, k, N)$ can be computed from (G, N) in polynomial time with respect to the size of the input graph G. Moreover, we have UNIFORM-ECCN $\in$ NP, hence UNIFORM-ECCN is NP-complete. ▣

We may also wish to consider the following language that decides if a given subset of edges E' of a graph G admits a partial k-uniform ECC of (G, E').

$$\text{PARTIAL-UNIFORM-ECCN} = \{(G, k, N, E') \in \mathbb{G} \times \mathbb{Z} \times \mathbb{Z} \times \mathscr{P}(E(G)) | \theta'_k(G, E') \leq N\}$$

Proposition 5.5.2 The decision problem for the language PARTIAL-UNIFORM-ECCN is NP-complete.

Proof: It is easy to check that PARTIAL-UNIFORM-ECCN belongs to the class NP. Let (G, k, N) be an instance for UNIFORM-ECCN. Clearly $(G, k, N) \in$ UNIFORM-ECCN if and only if $(G, k, N, E(G)) \in$ PARTIAL-UNIFORM-ECCN. Therefore UNIFORM-ECCN $\leq_P$ PARTIAL-UNIFORM-ECCN. ▣

5.6 Previous Results on the Complexity of ELAs

We now address the complexity of a related problem, namely the determination of the minimum size of an ELA. Martinez et. al. [28] address the complexity of the existence of an ELA, given its associated graph. This is not the same as the testing problem for which an ELA is to be built from an unknown graph, but rather, we look at a graph and would like to know if it has the properties required for an ELA to exist. The next theorem from [28] relates the languages AVOID and COVER&AVOID to the language LOCATE, defined below.

$$\text{LOCATE} = \{G \in \mathcal{G}_{(g_1,\ldots,g_k)} \mid G \text{ is locatable (an ELA}(n, G) \text{ exists for some } n \in \mathbb{Z}) \ \}$$

Recall that G is locatable if for every pair of vertices $\{v_{i,a_i}, v_{j,a_j}\}$ such that $i \neq j$, if $e = \{v_{i,a_i}, v_{j,a_j}\} \in E(G)$ then there exists a k-tuple T avoiding $G \setminus \{e\}$ such that $T_i = a_i$ and $T_j = a_j$. Otherwise, if $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G)$ then there exists a k-tuple avoiding G such that $T_i = a_i$ and $T_j = a_j$.

Theorem 5.6.1 (Martinez et. al. [28])

1. 2-LOCATE $\in P$.
2. g-LOCATE is NP-complete for $g \geq 5$, and so LOCATE is NP-complete.

For convenience in presenting its proof, we break up this theorem into two separate propositions.

Proposition 5.6.2 2-LOCATE $\in$ P.

Proof: By Theorem 5.1.1, we have 2-COVER&AVOID $\in$ P, so there is a polynomial-time algorithm A that decides 2-COVER&AVOID. Define an algorithm A' having instances for 2-LOCATE as input, and values from $\{0, 1\}$ as output as follows.

$$A'(G) = \begin{cases} 1, & \text{if } A(G) = 1 \text{ and for every edge } e \in E(G) \text{ we have } A(G \setminus \{e\}) = 1; \\ 0, & \text{otherwise.} \end{cases}$$

It is clear that $G \in$ 2-LOCATE if and only if $G \in$ 2-COVER&AVOID and for every edge $e = \{v_{i,a_i}, v_{j,a_j}\} \in E(G)$ there exists a k-tuple T avoiding $G \setminus \{e\}$ such that $T_i = a_i$ and $T_j = a_j$. In other words, $G \in$ 2-LOCATE if and only if $G \in$ 2-COVER&AVOID and for every edge $e \in E(G)$ the graph $G \setminus \{e\}$ is in 2-COVER&AVOID. Thus, A' decides 2-LOCATE. Since A' runs in polynomial time, we have 2-LOCATE $\in$ P. ▣

Proposition 5.6.3 g-LOCATE is NP-complete for $g \geq 5$, and so LOCATE is NP-complete.

Proof: First, we show that g-AVOID $\leq_P$ $(g+2)$-LOCATE. Given $G \in \mathcal{G}_{k,g}$, an instance for g-AVOID, we build G', an instance for $(g+2)$-LOCATE as follows. Append to G two new factors indexed by $k+1$ and $k+2$ with vertices $v_{k+1,0}, ..., v_{k+1,g-1}$ and $v_{k+2,0}, ..., v_{k+2,g-1}$, respectively. Add two new vertices $v_{i,g}$ and $v_{i,g+1}$ to each factor $i \in [1, k+2]$. Add edges joining vertex $v_{k+1,0}$ to each $v_{i,g}$ for factors $1 \leq i \leq k$ and similarly add edges joining vertex $v_{k+2,0}$ to $v_{i,g+1}$ for factors $1 \leq i \leq k$. Denote the graph obtained as G'.

We claim that $G \in g$-AVOID if and only if $G' \in (g+2)$-LOCATE. Let $i, j \in [1, k+2]$ be two distinct indices and let $(a_i, a_j) \in [0, g+1] \times [0, g+1]$. If $\{(i, a_i), (j, a_j)\} \neq \{(k+1, 0), (k+2, 0)\}$ then define a $(k+2)$-tuple T with $T_i = a_i$

and $T_j = a_j$, and for all $l \in [1, k+2] \setminus \{i, j\}$ take $T_l = g$ if $(k+1, 0) \notin \{(i, a_i), (j, a_j)\}$; otherwise, if $(k+1, 0) \in \{(i, a_i), (j, a_j)\}$ then take $T_l = g+1$. Then T avoids $G' \setminus \{e\}$ if $e = \{v_{i,a_i}, v_{j,a_j}\} \in E(G')$ and T avoids G' if $\{v_{i,a_i}, v_{j,a_j}\} \notin E(G')$. In this way, we can cover every edge, or non-edge of G' except the non-edge between vertices $v_{k+1,0}$ and $v_{k+2,0}$.

In the case where $\{(i, a_i), (j, a_j)\} = \{(k+1, 0), (k+2, 0)\}$, there exists a $(k+2)$-tuple T avoiding G' and such that $T_{k+1} = 0 = T_{k+2}$ if and only if there exists a k-tuple avoiding G, since T cannot use any vertex $v_{i,g}$ or $v_{i,g+1}$ for $1 \leq i \leq k$. Therefore $G' \in (g+2)$-LOCATE if and only if $G \in g$-AVOID. Since G' can be computed from G in polynomial time, we have g-AVOID $\leq_P$ $(g+2)$-LOCATE. By Theorem 5.1.1, since g-AVOID is NP-complete for $g \geq 3$ we have that g-LOCATE is NP-complete for $g \geq 5$. This also implies that LOCATE is NP-complete. ▣

Given a graph G and a positive integer N, we now look at the language ELAN which addresses the existence of an ELA$(n; 2, G)$ with $n \leq N$.

$$\text{ELAN} = \{(G, N) \in \mathcal{G}_{(g_1,\ldots,g_k)} \times \mathbb{Z} \mid \text{ELAN}(G) \leq N\}$$

Let g-ELAN denote the subset of ELAN where $G \in G_{k,g}$ and the corresponding ELA has uniform alphabet size g.

Now, observe that if a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ is locatable then

$$\text{ELAN}(G) \leq \sum_{1 \leq i < j \leq k} g_i g_j$$

since this is the total number of pairs of vertices from any two distinct factors of G, and an ELA$(n; 2, G)$ requires at most one row per non-edge between two distinct factors as well as one row per edge joining vertices from two distinct factors. From this upper bound we get the following result.

Proposition 5.6.4 g-ELAN is NP-complete for $g \geq 5$, so ELAN is NP-complete.

Proof: Let $g \geq 2$ and let $G \in \mathcal{G}_{k,g}$. Then $G \in g$-LOCATE if and only if there exists an ELA$(n; 2, G)$ for some positive integer n, if and only if ELAN$(G) \leq \Sigma_{1 \leq i < j \leq k}(g_i g_j) = \binom{k}{2}g^2$, if and only if $(G, \binom{k}{2}g^2) \in$ ELAN. Clearly $(G, \binom{k}{2}g^2)$ can be computed from G in polynomial time with respect to the size of G. By Theorem 5.6.1, we know that g-LOCATE is NP-complete for $g \geq 5$. Thus, g-ELAN is NP-complete for $g \geq 5$. This trivially implies that ELAN is NP-complete. ▣

5.7 NP-Completeness of 2-ELAN

We now give a proof for the NP-completeness of the binary ELA decision problem, 2-ELAN, by reducing from ECCN using a similar construction as for the 2-CAFEN problem. The proof is based on Theorem 2.6.2 which relates ELAN(G) to CAFEN(G) for locatable graphs $G \in \mathcal{G}_{(g_1,\dots,g_k)}$. Although one might think to use the NP-completeness of 2-CAFEN and Theorem 2.6.2 in order to obtain the NP-completeness of 2-ELAN, this does not work as there are graphs in $\mathcal{G}_{(g_1,\dots,g_k)}$ that are consistent but which are not locatable. Instead, in Construction 2 we reduce from ECCN and create locatable instances for 2-ELAN.

Construction 2 Let G be a simple graph on ν vertices and let n be the number of non-isolated vertices of G. We obtain from G another graph $(H_G)_{UV}$ on $2(2n+2)$ vertices as follows.

Step 1. Remove all isolated vertices of G to obtain a new graph G_n on n vertices. Throughout this section, n denotes the number of non-isolated vertices in G. Denote the vertices of G_n as $V(G_n) = \{v_1, v_2, ..., v_n\}$.

Step 2. Add a new set of vertices $V' = \{v_{n+1}, v_{n+2}, ..., v_{2n}\}$ and join by an edge each v_i to the corresponding vertex $v_{n+i} \in V'$ for $1 \leq i \leq n$. In addition, form an

n-clique between all the vertices of V' by adding the edges $\{v_{n+i}, v_{n+j}\}$ to the graph for $1 \leq i < j \leq n$. Denote this graph by H_G.

Step 3. Apply Construction 1 to the graph H_G.

In summary, we obtain a sequence of graphs

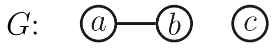
$$G \to G_n \to H_G \to (H_G)_k \to \overline{(H_G)_k} \to (H_G)_V \to (H_G)_{UV}$$

where $G \to G'$ denotes that G' was obtained from G by some step of Construction 2.

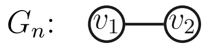
Remark 5.7.1 Since H_G contains $2n$ non-isolated vertices by construction, when applying Construction 1, we have $H_G = (H_G)_k$, and in this case $k = 2n$. Moreover, the vertices of H_G are already labeled as $v_1, ..., v_{2n}$, so without loss of generality, we assume that in Step 3. of Construction 2, when we obtain the graph $(H_G)_{UV}$ from H_G, the vertices $v_1, ..., v_{2n}$ retain their original labeling.

Let us illustrate Construction 2 with a concrete example.

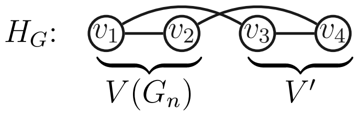
Example 2 Consider the following graph G.



We remove the isolated vertex c and relabel the vertices a and b to v_1 and v_2, respectively, to obtain the graph G_n. In this case, the number of non-isolated vertices of G is $n = 2$.

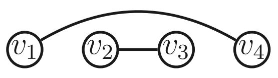


Next, we obtain the graph H_G from G_n by adding vertices v_3 and v_4 as well as the edges $\{v_1, v_3\}$, $\{v_2, v_4\}$, and an edge to form the 2-clique $\{v_3, v_4\}$.



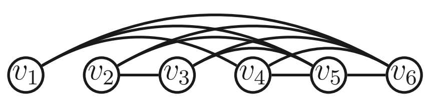
Since H_G contains no isolated vertices by construction, we have $(H_G)_k = (H_G)$ and $k = 2n$. We then take the complement, to obtain $\overline{(H_G)_k} = \overline{H_G}$.

$\overline{H_G}$:



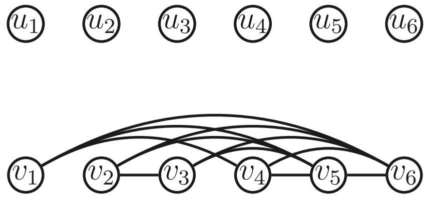
We obtain $(H_G)_V$ by adding vertices v_5 and v_6, as well as the edges joining v_5 to each v_i for $1 \leq i \leq 4$, and the edges joining v_6 to each v_i for $1 \leq i \leq 5$.

$(H_G)_V$:



Lastly, we obtain $(H_G)_{UV}$ by adding the vertex set $U = \{u_1, u_2, u_3, u_4, u_5, u_6\}$ and following the Across Edge Rule.

$(H_G)_{UV}$:



The following proposition gives the relationship between the ECC number of a graph G and the ECC number of the graph H_G, obtained from G by Construction 2.

Proposition 5.7.2 For any simple graph G, the graph H_G obtained in Step 2. of Construction 2 satisfies: $\theta'(H_G) = \theta'(G)+n+1$, where n is the number of non-isolated vertices in G.

Proof: Suppose for some positive integer N we have $\theta'(H_G) = N$. Take an optimal clique-maximal ECC of H_G, say $\mathfrak{C} = \{C_1, ..., C_N\}$. Let $C_j \in \mathfrak{C}$ be a clique that covers the edge $\{v_i, v_{n+i}\}$ for some index $i \in [1, n]$. Since by construction, v_i is not adjacent to any other vertex in V' aside from v_{n+i}, the clique C_j cannot contain any vertex of V' other than v_{n+i}. Since by construction, the vertex v_{n+i} is not adjacent to any other vertex in $V(G_n)$ aside from v_i, the clique C_j cannot contain any vertex of $V(G_n)$ other than v_i. Therefore, any clique that covers the edge $\{v_i, v_{n+i}\}$ is a 2-clique containing only the two ends v_i and v_{n+i}. Since there are n edges of this form, we must have n cliques of $\mathfrak{C}$ which each cover only one such edge.

Now, let $C_y \in \mathfrak{C}$ be a clique that covers the edge $\{v_{n+i}, v_{n+j}\}$ for two distinct indices $i, j \in [1, n]$. We assume C_y is maximal with respect to set inclusion, thus C_y must include as many vertices as possible. Since by construction, the vertex v_{n+i} is not adjacent to any vertex in $V(G_n)$ aside from v_i, the clique C_y cannot contain any vertex of $V(G_n)$ except possibly v_i. However, v_{n+j} is not adjacent to v_i, so there is no vertex in $V(G_n)$ that belongs to C_y. Since the vertices of V' are all adjacent to one another, C_y must include all such vertices. Thus, the clique C_y that covers any edge of the form $\{v_{n+i}, v_{n+j}\}$ for two distinct indices $i, j \in [1, n]$ must by its maximality cover the entire clique induced by the vertices of V'.

The remaining edges of H_G correspond exactly to the edges of G, and since the cliques above only cover edges with at most one end in $V(G_n)$, we conclude that $\theta'(H_G) - (n + 1) = \theta'(G)$. ▣

Now, we have only to prove that the graph $(H_G)_{UV}$ is locatable. It is sufficient to show that $(H_G)_{UV}$ has no across edges. For this purpose, we show the following result.

Proposition 5.7.3 Let v_i and v_j be two distinct vertices of the graph H_G. Then $\{v_i, v_j\} \in E(H_G)$ implies $N_{H_G}(v_i) \setminus \{v_j\} \nsubseteq N_{H_G}(v_j) \setminus \{v_i\}$.

Proof: Let v_i and v_j be two distinct vertices of the graph H_G, and let $\{v_i, v_j\} \in E(H_G)$. Then there are four cases to consider.

Case 1: $1 \leq i, j \leq n$

By the construction of H_G, for the vertices v_i and v_j we must have the corresponding vertices v_{n+i} and v_{n+j} in V'. Thus we have $\{v_i, v_{n+i}\} \in E(H_G)$ and

$\{v_j, v_{n+j}\} \in E(H_G)$. Moreover, $\{v_i, v_{n+j}\} \notin E(H_G)$ and $\{v_j, v_{n+i}\} \notin E(H_G)$. Thus $v_{n+i} \in N_{H_G}(v_i) \backslash \{v_j\}$ but $v_{n+i} \notin N_{H_G}(v_j) \backslash \{v_i\}$. Therefore $N_{H_G}(v_i) \backslash \{v_j\} \nsubseteq N_{H_G}(v_j) \setminus \{v_i\}$.

Case 2: $1 \leq i \leq n$ and $j = n + i$

Since the graph H_G is obtained from the graph G_n on the n non-isolated vertices of the original graph G, we know that the vertex v_i has at least one neighbour, say v_l such that $1 \leq l \leq n$. Then $v_j = v_{n+i}$ corresponds to the vertex in V' associated to v_i and we have $v_l \in N_{H_G}(v_i) \setminus \{v_{n+i}\}$ but $v_l \notin N_{H_G}(v_{n+i}) \setminus \{v_i\}$. Therefore $N_{H_G}(v_i) \setminus \{v_j\} \nsubseteq N_{H_G}(v_j) \setminus \{v_i\}$.

Case 3: $1 \leq j \leq n$ and $i = n + j$

Since the graph H_G is obtained from the graph G_n, we know that the vertex v_j has at least one neighbour, say v_l such that $1 \leq l \leq n$. By construction, v_l has a corresponding vertex $v_{n+l} \in V'$ and v_{n+l} is joined by an edge to each vertex in V'. In particular, $v_i = v_{n+j}$ corresponds to the vertex in V' associated to v_j and we have $v_{n+l} \in N_{H_G}(v_{n+j}) \setminus \{v_j\}$ but $v_{n+l} \notin N_{H_G}(v_j) \setminus \{v_{n+j}\}$. Therefore $N_{H_G}(v_i) \setminus \{v_j\} \nsubseteq N_{H_G}(v_j) \setminus \{v_i\}$.

Case 4: $n + 1 \leq i, j \leq 2n$

In this case, the vertices v_i and v_j both belong to V', and thus correspond to two of the vertices of $V(G_n)$, say $v_i = v_{n+l}$ for some l such that $1 \leq l \leq n$, and similarly $v_j = v_{n+m}$ for some m such that $1 \leq m \leq n$. Then $v_l \in N_{H_G}(v_{n+l}) \setminus \{v_{m+l}\}$, but $v_l \notin N_{H_G}(v_{n+m}) \backslash \{v_{n+l}\}$. Therefore, $N_{H_G}(v_i) \backslash \{v_j\} \nsubseteq N_{H_G}(v_j) \backslash \{v_i\}$.

▣

Indeed, the motivation for the construction of H_G lies in the result of Proposition 5.7.3. In the following corollary, we see that the construction of H_G forces no across edges to be added to the graph $(H_G)_{UV}$ when following the Across Edge Rule. As a result, $(H_G)_{UV}$ has safe values for each of its factors, which in turn, means that it is locatable.

Corollary 5.7.4 Let G be any simple graph. Then the graph $(H_G)_{UV}$ has no across edges.

Proof: By Corollary 5.2.4, we know that for two distinct vertices $v_i, v_j \in V$, the across edge $\{v_i, u_j\} \in E((H_G)_{UV})$ if and only if $\{v_i, v_j\} \in E((H_G)_k)$ and $N_{(H_G)_k}(v_i) \setminus \{v_j\} \subseteq N_{(H_G)_k}(v_j) \setminus \{v_i\}$. Since H_G has no isolated vertices by construction, we have $H_G = (H_G)_k$ in this case. By Proposition 5.7.3, we know $\{v_i, v_j\} \in E(H_G)$ implies $N_{H_G}(v_i) \setminus \{v_j\} \not\subseteq N_{H_G}(v_j) \setminus \{v_i\}$, which means $\{v_i, u_j\} \notin E((H_G)_{UV})$. Since v_i, v_j are arbitrary, we see that $(H_G)_{UV}$ contains no across edges. ▣

Corollary 5.7.5 The graph $(H_G)_{UV}$ is locatable.

Proof: Since $(H_G)_{UV}$ has the property that it is loopless and that none of its zero vertices are joined to one another, and since by Corollary 5.7.4, we know that $(H_G)_{UV}$ has no across edges, we see that the zero vertices of $(H_G)_{UV}$ are isolated. Thus $(H_G)_{UV}$ has safe values $0, ..., 0$. By Proposition 2.6.4, $(H_G)_{UV}$ is locatable. ▣

Given that $(H_G)_{UV}$ is locatable, we now get the following result.

Proposition 5.7.6 Let G be a simple graph, and let n be the number of non-isolated vertices of G. Then the graph $(H_G)_{UV}$, obtained by Construction 2, satisfies

$$\mathrm{ELAN}((H_G)_{UV}) = \theta'(G) - |E(G)| + \frac{3n^2}{2} + \frac{7n}{2} + 4.$$

Proof:

$$\begin{aligned}
\mathrm{ELAN}((H_G)_{UV}) &= \mathrm{CAFEN}((H_G)_{UV}) + |E((H_G)_{UV})|, && \text{by Theorem 2.6.2;}\\
&= \theta'(H_G) + 2 + |E((H_G)_{UV})|, && \text{by Theorem 5.2.11;}\\
&= \theta'(G) + n + 1 + 2 + |E((H_G)_{UV})|, && \text{by Proposition 5.7.2.}
\end{aligned}$$

It is easy to show that

$$|E((H_G)_{UV})| = \frac{3n^2}{2} + \frac{5n}{2} + 1 - |E(G)|.$$

Therefore, $\mathrm{ELAN}((H_G)_{UV}) = \theta'(G) - |E(G)| + \frac{3n^2}{2} + \frac{7n}{2} + 4$. ▣

Corollary 5.7.7 2-ELAN is NP-complete.

Proof: Let (G, N) be an instance for the decision problem ECCN. Then by Proposition 5.7.6, $(G, N) \in$ ECCN if and only if $\left((H_G)_{UV}, N - |E(G)| + \frac{3n^2}{2} + \frac{7n}{2} + 4\right) \in$ 2-ELAN. Moreover, it is easy to see that $(H_G)_{UV}$ can be computed in polynomial time with respect to the size of the input graph G. Therefore, we have ECCN $\leq_P$ 2-ELAN. By the NP-completeness of ECCN, and the fact that 2-ELAN belongs to the class NP, it follows that 2-ELAN is NP-complete. ▣

5.8 NP-Completeness of g-ELAN for $g \geq 3$

Intuitively, we now expect that g-ELAN is NP-complete for all $g \geq 2$. In this section, we extend the construction for showing the NP-completeness of 2-ELAN, in order to prove that 3-ELAN is also NP-complete. We further extend this construction to prove that g-ELAN is NP-complete for all $g \geq 2$, thereby filling in the gaps left by Proposition 5.6.4.

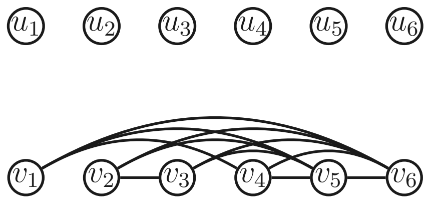
Construction 3 Let G be any simple graph. We construct another graph X_G as follows.

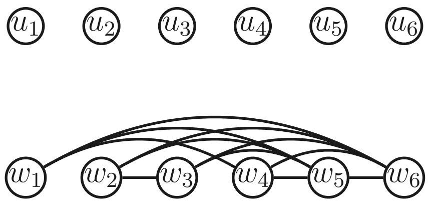
Step 1. Using Construction 2, obtain from G two copies of the graph $(H_G)_{UV}$. Refer to the vertices of the first copy as $V = \{v_1, ..., v_{k+2}\}$ and $U = \{u_1, ..., u_{k+2}\}$, and rename the vertex set $V = \{v_1, ..., v_{k+2}\}$ of the second copy as $W = \{w_1, ..., w_{k+2}\}$, so that each $w_i \in W$ corresponds exactly to the vertex $v_i \in V$ of the second copy of $(H_G)_{UV}$.

Step 2. Identify the two vertex sets U of the two copies so that the resulting graph contains three distinguished sets of vertices V, W, and U, and add new edges to the graph: for every $i \neq j$ such that $1 \leq i, j \leq k+2$, add an edge joining vertex $v_i \in V$ to $w_j \in W$. Refer to this graph as X_G.

Let us illustrate Construction 3 with an example.

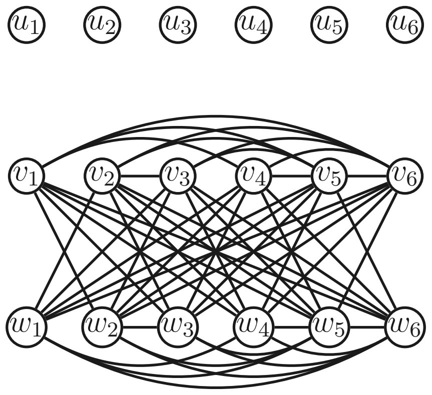
Example 3 For convenience, let us use the same graph G as in Example 2. We take two copies of $(H_G)_{UV}$ and relabel the vertex set V of the second copy, as $W = \{w_1, w_2, w_3, w_4, w_5, w_6\}$.





We then identify the two copies of U, and add edges of the form $\{v_i, w_j\}$ for $v_i \in V$ and $w_j \in W$ whenever $i \neq j$. This results in the final graph of the construction, X_G.

X_G:



We consider the graph X_G as an instance for the 3-ELAN problem having $k+2$ factors. That is, $X_G \in \mathcal{G}_{k+2,3}$, and we let the vertices of U represent the zero values, the vertices of V represent the one values, and the vertices of W represent the two values, respectively, of each of the $k+2$ factors.

Now observe that the non-edges between two distinct factors of X_G occur only between vertices of V and U, or between vertices of W and U, since X_G is constructed so that $i \neq j$ implies that $\{v_i, w_j\} \in E(X_G)$ for vertices $v_i \in V$ and $w_j \in W$. Furthermore, the vertex set U provides safe values for each of the $k+2$ factors of X_G, since by the properties of $(H_G)_{UV}$, X_G inherently has no across edges joining vertices of V to vertices of U, and similarly, X_G has no across edges joining vertices of W to vertices of U. Thus we conclude that the graph X_G is locatable. By Theorem 2.6.2, we know that $\text{ELAN}(X_G) = \text{CAFEN}(X_G) + |E(X_G)|$.

The next result relates the ECC number of the original graph G to the number of rows required for an optimal CAFE of X_G.

Proposition 5.8.1 For a simple graph G, the graph X_G obtained as described in this section satisfies $\text{CAFEN}(X_G) = 2 \times [\text{CAFEN}((H_G)_{UV})]$.

Proof: First we show that $2 \times [\text{CAFEN}((H_G)_{UV})] \leq \text{CAFEN}(X_G)$. An optimal CAFE of X_G, in particular, requires rows to cover all interactions of the form $\{v_i, v_j\}$ where $i \neq j$ and $v_i, v_j \in V$. By construction, any row that covers a vertex $v_i \in V$ cannot also cover a vertex $w_l \in W$ for $l \neq i$. Therefore, covering the $\{v_i, v_j\}$ interactions mentioned is equivalent to covering the $\{v_i, v_j\}$ interactions in $(H_G)_{UV}$, and thus requires $\text{CAFEN}((H_G)_{UV}) - 2$ rows. Furthermore, these rows cannot cover any interaction involving a vertex $w_l \in W$. By Lemma 5.2.5, we also require two additional rows to cover the particular interactions $\{v_{k+1}, u_{k+2}\}$ and $\{v_{k+2}, u_{k+1}\}$. We also require enough rows to cover all $\{w_i, w_j\}$ interactions for $i \neq j$ and $w_i, w_j \in W$. By the same reasoning above, this requires $\text{CAFEN}((H_G)_{UV}) - 2$ additional rows. Again, we require two additional rows to cover the particular interactions $\{w_{k+1}, u_{k+2}\}$ and $\{w_{k+2}, u_{k+1}\}$. Thus, an optimal $\text{CAFE}(n, X_G)$ requires at least $2 \times [\text{CAFEN}((H_G)_{UV})]$ rows in order to cover the interactions mentioned. Consequently, $\text{CAFEN}(X_G) \geq 2 \times [\text{CAFEN}((H_G)_{UV})]$.

Next, we show that $2 \times [\text{CAFEN}((H_G)_{UV})] \geq \text{CAFEN}(X_G)$. Cover the non-edges between distinct factors of the first copy of $(H_G)_{UV}$ with $\text{CAFEN}((H_G)_{UV})$ rows and similarly, cover the non-edges between distinct factors of the second copy of $(H_G)_{UV}$. It is easy to see that by the construction of X_G these $2 \times [\text{CAFEN}((H_G)_{UV})]$ rows are sufficient to cover and avoid X_G. We conclude that $2 \times [\text{CAFEN}((H_G)_{UV})] \geq \text{CAFEN}(X_G)$. ▣

Corollary 5.8.2 Let G be any simple graph. Then

$$\text{ELAN}(X_G) = 2\,[\theta'(G) - |E(G)|] + 7n^2 + 13n + 10.$$

Proof: Since X_G is constructed so that it has safe values for each of its factors, we know that X_G is locatable as an instance for the decision problem 3-ELAN. Therefore, we get the following results, where n is the number of non-isolated vertices of G.

$$\begin{aligned} \text{ELAN}(X_G) &= \text{CAFEN}(X_G) + |E(X_G)|, && \text{by Theorem 2.6.2;} \\ &= 2[\text{CAFEN}((H_G)_{UV})] + |E(X_G)|, && \text{by Proposition 5.8.1.} \end{aligned}$$

It is easy to see that

$$|E(X_G)| = 2|E((H_G)_{UV})| + (2n+2)(2n+1) = 7n^2 + 11n + 4 - 2|E(G)|.$$

Therefore,

$$\begin{aligned} \text{ELAN}(X_G) &= 2[\text{CAFEN}((H_G)_{UV})] + 7n^2 + 11n + 4 - 2|E(G)| \\ &= 2[\theta'(H_G) + 2] + 7n^2 + 11n + 4 - 2|E(G)|, && \text{by Theorem 5.2.11;} \\ &= 2[\theta'(G) + n + 1 + 2] + 7n^2 + 11n + 4 - 2|E(G)|, && \text{by Proposition 5.7.2;} \\ &= 2[\theta'(G) - |E(G)|] + 7n^2 + 13n + 10. \end{aligned}$$

▣

Corollary 5.8.3 3-ELAN is NP-complete.

Proof: Let (G, N) be an instance for ECCN, and let n be the number of non-isolated vertices of G. By the above corollary, the pair (G, N) is in ECCN if and only if the pair $(X_G, 2 \times [N - |E(G)|] + 7n^2 + 13n + 10)$ is in 3-ELAN. It is easy to see that X_G can be computed in polynomial time with respect to the original graph G. Again, since we know that ECCN is NP-complete, and we have reduced ECCN to 3-ELAN, we have shown the NP-completeness of 3-ELAN. ▣

In fact, in a similar way we can show that ECCN $\leq_P$ g-ELAN, for $g \geq 3$.

Theorem 5.8.4 g-ELAN is NP-complete, for $g \geq 3$.

Proof: Here we give a sketch of the proof. Let G be any graph. This time take $(g-1)$ copies of the graph $(H_G)_{UV}$ and identify the $g-1$ copies of the U vertex subsets. Denote by $V_1, V_2, ..., V_{g-1}$ the $g-1$ copies of the vertex subsets V of each respective copy. Label the vertices of V_i as $v_{1,i}, v_{2,i}, ..., v_{k,i}$ for $i = 1, 2, ..., g-1$. Join by an edge each vertex $v_{a,i} \in V_i$ to every $v_{b,j} \in V_j$ whenever $a \neq b$ and $i \neq j$, $i, j \in \{1, 2, ..., g-1\}$. Denote this graph as Y_G.

Similarly to X_G, we can show that Y_G is locatable based on the safe values in the vertex set U. Thus ELAN(Y_G) = CAFEN$(Y_G) + |E(Y_G)|$. Moreover we can show CAFEN$(Y_G) = (g-1)[\text{CAFEN}((H_G)_{UV})]$. It is easy to see that

$$|E(Y_G)| = (g-1)|E((H_G)_{UV})| + \binom{g-1}{2}(2n+1)(2n+2).$$

Therefore,

$$\text{ELAN}(Y_G) = (g-1)[\theta'(G) - |E(G)|] + (g-1)\left[\frac{3n^2}{2} + \frac{7n}{2} + 4\right] + \binom{g-1}{2}(2n+1)(2n+2)$$

and so $(G, N) \in$ ECCN if and only if $(Y_G, N') \in g$-ELAN, where

$$N' = (g-1)[N - |E(G)|] + (g-1)\left[\frac{3n^2}{2} + \frac{7n}{2} + 4\right] + \binom{g-1}{2}(2n+1)(2n+2).$$

Again, we can easily verify that g-ELAN belongs to the class NP, and clearly, Y_G can be computed in polynomial time from (G, N). ▣

Chapter 6

Conclusion

The motivation for the combinatorial designs studied in this thesis is to accommodate a wider range of practical testing problems. In particular, we focused on the problem of covering all pairwise interactions in the presence of constraints yielding forbidden pairwise interactions, using a relatively new design called a CAFE. We now summarize the contributions of this thesis in three main areas and discuss open problems.

Study of k-Uniform Edge Clique Covers

In light of the equivalence between CAFEs and k-uniform ECCs, we studied k-uniform ECCs and established several small results, including some simple upper bounds and necessary conditions for their existence. In particular we proved that the k-ECC number of a given graph can be strictly greater than its ECC number, by providing an example for each k; we also showed that the difference between these numbers can be arbitrarily large for any given k (see Theorem 3.3.9). On the other hand, we also provided an upper bound (see Proposition 3.3.12), indicating that for $k \geq 4$, a graph G such that $\omega(G) = k$, and such that G admits a k-ECC satisfies

$$\theta'_k(G) \leq \binom{k-1}{2} \theta'(G).$$

In all examples of graphs we looked at, which satisfied the hypotheses of Proposition

3.3.12, we found a much tighter upper bound. We always found that $\theta'_k(G) \leq 2\theta'(G)$, perhaps indicating that a better upper bound exists relating the k-ECC number and the ECC number, such as one that is linear in k, rather than quadratic. So, we formulate the following conjecture.

Conjecture 1 Let $k \geq 4$ and let G be a simple graph. If G admits a k-uniform ECC, then $\theta'_k(G) \leq k\theta'(G)$.

We also proved that if a graph G admits a k-uniform ECC for some $k \geq 3$, then G admits a $(k-1)$-uniform ECC as well (see Proposition 3.3.11); indeed, we do believe that the following stronger result holds.

Conjecture 2 Let $k \geq 3$ be a positive integer and let G be a simple graph. If G admits a k-uniform ECC, then $\theta'_k(G) \leq \theta'_{k-1}(G)$.

Although the CAFE number of a graph $G \in \mathcal{G}_{(g_1,\ldots,g_k)}$ is equal to the k-uniform ECC number of $\overline{G^{|}}$, we feel that further study of the ECC number would be beneficial. Better upper bounds relating the k-ECC number to the ECC number would give us more tools to estimate the CAFE number.

Partial CAFEs and Other Generalizations to Benefit Testing Applications

We generalized CAFEs to partial CAFEs, based on the corresponding generalization of k-ECCs to partial k-ECCs. We defined partial CAFEs in order to address the problem of covering a specified subset of the pairwise interactions of a testing problem.

Partial CAFEs can be used to solve a much broader spectrum of testing problems. For example, if we extend a testing problem by increasing the number of values of some factors *after* the test suite of the original testing problem has already been performed, we can use a partial CAFE to find the appropriate new tests to perform in order to cover the newly added interactions. Partial CAFEs can also be used in the case of regression tests (standard tests fixed a priori) and in the case of pairs of values known a priori to not interact (as with covering arrays on graphs). Thus, partial CAFEs allow more freedom in testing. We established some initial basic results for partial CAFEs, but more research in this area seems promising.

As considered in this thesis, partial CAFEs can only cover and avoid *pairwise* interactions. In reality, the constraints imposed on testing problems can yield forbidden t-way interactions for any $t \in [1, k]$. For example, constraint (C7) of Table 2.2 imposes a forbidden 3-way interaction for the mobile phone product line testing problem. Extending partial CAFEs to incorporate forbidden t-way interactions of various values of t would accommodate many more testing problems.

A natural generalization of partial CAFEs would be partial CAFEs which cover all necessary t-way interactions, for strength $t \geq 2$. Although, pairwise coverage is considered to be a relatively good compromise to exhaustive testing, there may be particular applications where covering all t-way interactions for some $t \geq 2$ is desirable. In fact, we may wish to consider a generalization of partial CAFEs that allow us to specify necessary interaction sets containing interactions of any given size.

Computational Complexity of Finding Optimal CAFEs

Perhaps the single most important contribution of this thesis was to establish that 2-CAFEN is NP-complete (see Corollary 5.2.12). The construction used in this proof also allowed us to establish the NP-completeness of g-CAFEN, g-PARTIAL-CAFEN and g-ELAN, for all $g \geq 2$. We also established a hardness of approximation result for binary CAFEs (see Proposition 5.2.14).

The closely related ECC decision problems ECCN, PARTIAL-ECCN, UNIFORM-ECCN, and PARTIAL-UNIFORM-ECCN are all NP-complete as well. Thus, viewing CAFE problems by their equivalent graph theoretical problems is not necessarily any easier. However, the correspondences established between ECC problems and CAFE problems allows us to bring graph theoretical results and algorithms in order to approach CAFE problems.

In light of the computational intractability and hardness of approximation for g-CAFEN, future research should focus on heuristic algorithms for general graphs or the study of special families of graphs. On the application side, studies to determine which classes of forbidden graphs occur most often in practical applications would certainly be useful.

Appendix A

Asymptotic Definitions

In order to estimate the growth of a function, we use the following definitions. The definitions here are taken from [33].

Definition A.0.5 Let f and g be functions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x) = O(g(x))$ if there are constants c and n_0 such that $|f(x)| \leq c|g(x)|$ for all $x > n_0$. We read this as "f is **big-oh** of g." The constants c and n are called **witnesses** to the relationship $f(x) = O(g(x))$.

Definition A.0.6 We say that $f(x) = o(g(x))$ when

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = 0.$$

We read this relationship as "f is **little-oh** of g."

Definition A.0.7 We say that the functions f and g are **asymptotically equivalent** if

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = 1.$$

In this case we write $f(x) \sim g(x)$.

Bibliography

[1] J. A. Bondy; U. S. R. Murty. *Graph Theory With Applications*. North-Holland, New York, 1982.

[2] Robert C. Brigham; Ronald D. Dutton. *On Clique Covers and Independence Numbers of Graphs*. Discrete Mathematics 44 (1983), 139–144.

[3] K. Burr; W. Young. *Combinatorial Test Techniques: Table-based Automation, Test Generation and Code Coverage*. Proc. of the Intl. Conf. on Software Testing Analysis & Review (San Diego), 1998.

[4] J. N. Cawse (editor). *Experimental Design for Combinatorial and High Throughput Materials Development*. John Wiley & Sons, New York, 2003.

[5] Myra B. Cohen; Matthew B. Dwyer; Jiangfan Shi. *Interaction Testing of Highly-Configurable Systems in the Presence of Constraints*, International Symposium on Software Testing and Analysis (ISSTA) (London), 2007, 129–139.

[6] Charles J. Colbourn. *Combinatorial aspects of covering arrays*, *Le Matematiche* (*Catania*) 58 (2004), 121–167.

[7] Charles J. Colbourn and Jeffrey H. Dinitz (editors). *Handbook of Combinatorial Designs, Second Edition*. Chapman and Hall/CRC, 2007.

[8] Stephen A. Cook. *The Complexity of Theorem-Proving Procedures*. Proceedings Third Annual ACM Symposium on Theory of Computing, May 1971, 151–158.

[9] Thomas H. Cormen; Charles E. Leiserson; Ronald L. Rivest; Clifford Stein. *Introduction to Algorithms, Second Edition.* MIT Press, Montréal, 2007.

[10] S. R. Dalal; A. Jain; N. Karunanithi; J. M. Leaton; C. M. Lott; G. C. Patton; B. M. Horowitz. *Model-based Testing in Practice.* Proc. of the Intl. Conf. on Software Engineering, (ICSE '99) (New York), 1999, 285–294.

[11] Peter Danziger; Eric Mendelsohn; Lucia Moura; Brett Stevens. *Covering arrays avoiding forbidden edges (CAFE).* Theoretical Computer Science, 410 (2009), 5403–5414.

[12] Paul Erdös; A. W. Goodman; Louis Pósa. *The Representation of a Graph by Set Intersections.* Canad. J. Math. 18 (1966), 106–112.

[13] Michael R. Garey; David S. Johnson. *Compupters and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[14] L. Gargano; J. Körner; U. Vaccaro. *Sperner Capacities.* Graphs Combin. 9 (1993), 31–46.

[15] A. Gyárfás. *A Simple Lower Bound on Edge Coverings by Cliques.* Discrete Mathematics 85 (1990) 103–104.

[16] Alan Hartman; Leonid Raskin. *Problems and Algorithms for Covering Arrays.* Discrete Mathematics 284 (2004), 149–156.

[17] A. Hedayat; N. Sloane; J. Stufken. *Orthogonal Arrays.* Springer-Verlag, New York, 1999.

[18] Dan Hoffman. *A Perfect Marriage: Covering Arrays and Grammar-based Test Generation*, a talk given for the Ottawa-Carleton Discrete Mathematics Group Seminar, December 3, 2008.

[19] Richard Karp. *Reducibility Among Combinatorial Problems. In* Raymond E. Miller; James W. Thatcher (editors). Complexity of Computer Computations. Plenum Press, New York, 1972, 85–104.

[20] G. Katona. *Two applications (for search theory and truth functions) of Sperner type theorems.* Periodica Math. 3 (1973), 19–26.

[21] D. Kleitman; J. Spencer. *Families of k-independent sets.* Discrete Math. 6 (1973), 255–262.

[22] L. T. Kou; L.J. Stockmeyer; C.K. Wong. *Covering Edges by Cliques with Regard to Keyword Conflicts and Intersection Graphs.* Communications of the ACM 21 (1978), 135–139.

[23] D. Kuhn; M. Reilly. *An Investigation into the Applicability of Design of Experiments to Software Testing.* Proc. 27^{th} Annual NASA/IEEE Software Engineering Workshop, NASA Goddard Space Flight Center, 2002, 91–95.

[24] R. Kuhn; D. Wallace; A. Gallo. *Software Fault Interactions and Implications for Software Testing.* IEEE Transactions of Software Engineering 30 (2004), 418–421.

[25] L. A. Levin. Universal sorting problems *Problemy Peredachi Informatsii* 9(1973), 265–266. *In Russian.*

[26] L. Lovász. *On Covering of Graphs*, Theory of Graphs (Proc. Colloq., Tihany, 1966), Academic Press, New York, 1968, 231–236.

[27] Carsten Lund; Mihalis Yannakakis. *On the Hardness of Approximating Minimization Problems.* Journal of the Association for Computing Machinery 41 (1994), 960–981.

[28] Conrado Martinez; Lucia Moura; Daniel Panario; Brett Stevens. *Locating Errors Using ELAs, Covering Arrays, and Adaptive Testing Algorithms*. SIAM Journal on Discrete Mathematics, *to appear*, 24 pages.

[29] Karen Meagher. *Covering Arrays on Graphs: Qualitative Independence Graphs and Extremal Set Partition Theory*. PhD thesis, University of Ottawa, Ottawa, 2005.

[30] Lucia Moura; J. Stardom; Brett Stevens; A. Williams. *Covering arrays with mixed alphabet sizes*. Journal of Combinatorial Designs 11 (2003), 413–432.

[31] James Orlin. *Contentment in graph theory: covering graphs with cliques*. Nederl. Akad. Wetensch. Proc. Ser. A, 80 (1977), 406–424.

[32] A. Rényi. *Foundations of Probability*. Wiley, New York, 1971.

[33] Kenneth H. Rosen. *Discrete Mathematics and Its Applications, Fifth Edition*. McGraw-Hill, Montreal, 2003.

[34] A. W. Williams and R. L. Probert. *A Measure for Component Interaction Test Coverage*. Proc. ACS/IEEE International Conference on Computer Systems and Applications, 2001, 301–311.