Decomposition and Domination of Some Graphs

Fairouz Beggas

N° d'ordre NNT : 2017LYSE1051

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON

opérée au sein de

l'Université Claude Bernard Lyon 1

École Doctorale ED 512
Informatique et Mathématiques (InfoMaths)

Spécialité de doctorat : Informatique

Soutenue publiquement le 28/03/2017, par :

Fairouz BEGGAS

Decomposition and Domination of Some Graphs

Devant le jury composé de :

Jean-Luc Baril, Professeur, Université de Bourgogne	Rapporteur
Lhouari Nourine, Professeur, Université Blaise Pascal Clermont-Ferrand	Rapporteur
Daniela Grigori, Professeur, Université de Paris Dauphine	Examinatrice
Salima Benbernou, Professeur, Université de Paris Descartes	Examinatrice
Norma Zagaglia Salvi, Professeur, École polytechnique de Milan	Examinatrice
Hamamache Kheddouci, Professeur, Université Lyon 1	Directeur de thèse
Mohammed Haddad, Maitre de Conférences, Université Lyon 1	Co-directeur de thèse

Acknowledgments

I would like to express my deepest gratitude to my advisor, Pr. Hamamache Kheddouci. His guidance helped me in all the time of research during this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

My thanks also go to my co-advisor, Dr. Mohammed Haddad, for him helpful advice and for the valuable comments and suggestions that he provided, especially on my presentation and the writing of this thesis.

I would also like to thank Pr. Volker Turau for his successful collaboration and for guiding my research and helping me to develop my background in a way of proving and writing. I also thank Pr. Norma Zagalia for her fruitful discussions and collaboration especially during my stay in Poletichnico di milano.

I wish to express my special thanks to all the personel of liris laboratory: Hacid, Isabelle, Brigitte, Saida, Catherine, Helene, Jean pierre...

Last but not least, I would like to thank my family for all their love and encouragement. For my parents who raised me with a love of science and supported me in all my pursuits. For my sister sarah for supporting me and pushing me all the time and who have given me constant support and love during the completion of the thesis. And most of all for my loving, supportive, encouraging, and patient sweetheart Mus whose faithful support during all this thesis and all my life is so appreciated. Many thanks my angel.

Fairouz

Abstract: Graph theory is considered as a field exploring a large variety of proof techniques in discrete mathematics. Thus, the various problems treated in this theory have applications in a lot of other scientific fields such as computer science, physics, sociology, game theory, etc. In this thesis, three major problems are considered: the *multidecomposition* of multigraphs, the $[1,2]$-*domination* and the *edge monitoring*. The fact that these three problems are of different nature allowed us to explore several proof techniques in this thesis.

The first part of this thesis deals with a popular aspect of research in graph theory called graph decomposition. Intuitively, a decomposition into subgraphs allows us to describe the original graph with a set of copies of these subgraphs. In this part, we give a particular interest to the multidecomposition of a complete multigraph into edge disjoint stars and cycles. Thus, we investigate the problem of (S_k, C_k)-multidecomposition of the complete multigraph and give necessary and sufficient conditions for such a multidecomposition to exist.

The second and third parts are the most important parts in terms of effort and spent time. They are devoted to problems related to domination in graphs. The original domination problem is to find a minimum set of vertices such that every vertex outside the dominating set is adjacent to at least one vertex from the dominating set. Many variants of theoretical and practical interest have been studied in the literature.

The second studied problem is called the $[i,j]$-domination in graphs. This problem was introduced by Chellali *et al.* in 2013. In addition to the properties of domination, this variant has the particularity that each non-dominating vertex should be adjacent to at least i dominating vertices but also to at most j of them. We particularly focus on the $[1,2]$-domination. It has been shown that the problem remains NP-complete. We are interested to study this problem on a particular graph namely the generalized Petersen graph. This graph was introduced by Watkins and has a lot of interesting properties. Moreover, several graph theoretical parameters have been studied on this graph class because of it unique structure. In addition, a study of the $[1,2]$-total domination is also proposed at the end of this part.

The last problem is a new variant called edge monitoring problem and was introduced by Dong *et al.* in 2008. It consists to find a set of vertices that monitors (dominates) the edge set of a graph such as a vertex monitors an edge if it forms a triangle with it *i.e.* it dominates both extremities of the edge. An edge can be monitored by one or more vertices. Three variants of the problem are considered in this part namely the *edge monitoring*, *uniform edge monitoring* and *weighted edge monitoring*. The essence of this problem lies on its combinatorial aspect and its range of applications in networks; especially wireless sensor networks. This problem is known to be NP-hard. Given the complexity of this kind of problems, we are first interested by a theoretical study: variants of the problem, bounds, characterizations, etc. We give more in depth studies of the problem for several graph classes.

Keywords: Graph decomposition, multidecomposition, $[1,2]$-dominating set, $[1,2]$-total dominating set, edge monitoring problem, weighted monitoring problem, k-uniform edge monitoring.

Résumé : La théorie des graphes est considérée comme un vaste champ qui permet d'explorer différentes techniques de preuve des mathématiques discrètes. Ainsi, les différents problèmes traités dans cette théorie ont plein d'applications dans d'autres domaines scientifiques tels que l'informatique, la physique, la sociologie, la théorie des jeux, etc.

Dans cette optique, nous proposons, dans cette thèse, de mettre l'accent sur trois problèmes de graphes, à savoir la *multidécomposition* de multigraphes, la $[1, 2]$-*domination* et le *monitoring des arêtes*. Ainsi, le fait d'explorer, dans ce travail de thèse, trois problèmes de graphes relativement distincts dans des classes de graphes différentes, nous a permis de développer plusieurs techniques de preuve ainsi qu'une multitude de façon à aborder un problème.

La première partie de cette thèse touche un aspect très important de la théorie des graphes, appelé la décomposition des graphes. Intuitivement, une décomposition en sous-graphe permet de représenter le graphe d'origine par un ensemble de copies du sous-graphe, où chaque arête du graphe initial appartient à une et une seule copie du sous-graphe. Dans cette partie, on s'intéresse plus particulièrement à la décomposition multiple d'un multigraphe complet en étoiles et cycles de même taille, c.à.d. générer à partir d'un multigraphe, plusieurs composantes disjointes (étoiles et cycles). Dans ce sens, des preuves formelles sont présentées pour déterminer les conditions nécessaires et suffisantes que doit avoir le multigraphe complet pour qu'une telle décomposition existe.

Les deux autres parties de cette thèse, les parties les plus consistantes, abordent un problème suscitant beaucoup d'attention actuellement, qui est l'étude de la domination dans les graphes. Le problème original de domination consiste à trouver un ensemble de sommets (de taille minimum) dominant le reste des sommets d'un graphe. De nombreuses variantes d'intérêts à la fois théoriques et pratiques ont été proposées et étudiées dans la littérature. Dans cette partie de thèse et celle qui suit, nous nous sommes intéressés à deux variantes de domination.

La première variante, appelée $[i, j]$-domination dans les graphes, a été introduite par Chellali *et al.* en 2013. En plus de ses propriétés de domination, la particularité de cette variante est que chaque sommet non dominant doit être adjacent à au moins i et au plus j sommets dominants. Plus particulièrement, nous nous somme interéssés à la $[1, 2]$-domination. Il convient de souligner qu'il a été démontré que le problème reste NP-complet. Dans ce sens, nous avons étudié ce paramètre dans des graphes particuliers, tels que les graphes de Petersen généralisés, ce qui rend ce problème tout aussi intéressant. Introduite par Watkins, cette famille de graphes possède un nombre de propriétés très intéressantes. D'ailleurs, plusieurs paramètres de graphes ont été étudiés sur cette classe de graphes de par sa structure qui est assez particulière. De plus, une étude de la $[1, 2]$-total domination sur cette classe de graphes est aussi menée dans cette thèse.

La deuxième et dernière variante étudiée, aussi une variante de la domination, appelée monitoring des arêtes, a été introduite par Dong *et al.* en 2008. Elle consiste à trouver un ensemble de sommets qui surveille (domine) l'ensemble des arêtes dans un graphe sachant qu'un sommet surveille une arête s'il forme un triangle avec les deux extrémités de l'arête. Une arête peut être monitorée par un ou plusieurs sommets. Dans ce contexte, plusieurs variantes du monitoring des arêtes sont considérées dans cette partie à savoir *monitoring des arêtes*, *monitoring uniforme des arêtes* et *monitoring pondéré des arêtes*. L'essence de ce problème réside dans sa nature combinatoire ainsi que son domaine d'application, plus particulièrement dans les réseaux de capteurs sans fil. De plus, il a été prouvé que trouver un ensemble minimum pour ce problème est NP-difficile. Vu la complexité de ce type de problème, nous nous sommes intéressés, en premier temps, par une étude théorique du problème : différentes variantes, les bornes, caractérisations, etc. Par la suite, nous avons étudié le probléme en profondeur dans différentes classes de graphes.

Mots clés : Décomposition des graphes, multidécomposition, $[1, 2]$-domination, $[1, 2]$-total domination, monitoring des arêtes, monitoring pondéré des arêtes, monitoring k-uniforme des arêtes.

"Imagination is more important than knowledge. Knowledge is limited. Imagination encircles the world.".
-Albert Einstein-

Contents

Chapter 1

Introduction

Graphs are considered as very powerful modeling tools in different areas of science such as physics, chemistry, sociology, game theory and many other areas. The concept of graphs was first introduced by Leonard Euler in 1735 with his work on the Seven Bridges of Königsberg [Eul41]. Since then, they have been considered as an important notion in discrete mathematics and used to model the problems of a wide variety of subjects. All these various subjects of practical interest can be considered as motivation to develop a large number of problems in graph theory such as graph coloring, domination sets, graph decomposition, independent sets, etc. Many other problems in graph theory can be added to this list, since each problem has a multitude of variants that can be explored.

Informally, a graph consists of some points called nodes and some lines between them called edges. Graph is used to model the connections between objects. As an example, a computer network can be modeled as a graph such that each server represented by a node and the connections between those servers represented by edges. Another example is to model a social network using graphs such that each individual (or organization) is represented by a node and the relation between them (friendship, kinship, common interest, financial exchange, dislike, relationships, etc.) by an edge. The theoretical study and the development of algorithms to manage graphs are therefore of major interest. Throughout this thesis, we try to discuss different graph problems in various classes of graphs.

The three major problems considered in this thesis are the *Decomposition* of complete multigraph into stars and cycles, the $[i, j]$-*Dominating Set* and the *Edge Monitoring Set* problems. The fact that these three problems are different nature allowed us to explore several proof techniques.

The *Decomposition* of graphs is one of the most famous and known problems in graph theory. It consists to break an input graph into subgraphs satisfying some constraints. Such problems fall broadly into two categories: the first called simple decomposition, consists to decompose the input graph into edge disjoint subgraphs of the same type; the second, called multiple decomposition, consists to decompose the input graph into two types of edge disjoint subgraphs or more. Determining if a graph G admits simple decomposition was proved to be NP-complete for all subgraph which have a connected component of size 3 or more. We will focus more closely on the specific multiple decomposition called decomposition of complete multigraph λK_n into stars S_k of k leaves and cycles C_k of k vertices (*a.k.a.* (S_k, C_k)-decomposition of λK_n). It consists in finding the partition of the edge set of the complete multigraph into edge disjoint isomorphic copies of S_k and C_k using at least one copy of each. Many questions can be asked but the most natural one is to find the conditions on λK_n for which (S_k, C_k)-decomposition exists. This allows us to find the require properties of λK_n in order to have a such decomposition.

The $[i, j]$-*Dominating Set* is an interesting variant of the dominating sets problem. It was introduced by Chellali *et al.* [CHHM13]. It is defined as follows. Let i and j be positive integers such that $i \leq j$. A subset $S \subseteq V$ in a graph $G = (V, E)$ is a $[i, j]$-dominating set if, for every vertex $v \in V \setminus S$, $i \leq |N(v) \cap S| \leq j$, that is, every vertex $v \in V \setminus S$ is adjacent to at least i but not more than j vertices in S. The minimum cardinality of a $[i, j]$-dominating set in a graph G is called the $[i, j]$-domination number, and is denoted $\gamma_{[i,j]}(G)$. In addition to its theoretical aspects, this problem has several practical applications. For example, in the case of servers in a computing network, or sets of monitoring devices in situations requiring surveillance, but with the need to establish such sets as efficiently or as cost effectively as possible, that is, without creating too much redundancy. For this reason, we give a particular interest to study the $[1, 2]$-dominating set in the particular graph, namely generalized Petersen graph by giving the exact value of the $[1, 2]$-domination number. This graph was introduced by Watkins and has a lot of interesting properties.

The *Edge Monitoring Set problem* is an effective mechanism for security of wireless sensors networks. It can also be considered as a variant of dominating sets

problem. The basic idea is that each communication link in the network is monitored (dominated) by nodes within the network and it is defined as follow. A node v can monitor (dominate) an edge e if the two extremities of e are neighbors of v (*i.e.* v and the two extremities of e form a triangle in the graph). Note that some edges can need more than one monitor. Finding the minimum set of monitor nodes for such problem is proved to be NP-complete by *Dong et al.* in 2008 [DLL08]. In the literature, the problem is studied from distributed systems and self stabilization point of view. In this thesis, we study this problem and two of its variants from the graph theoretical point of view. In the first one, namely k-uniform monitoring sets problem, all the edges of the considered graph need at least k monitors. The second variant, a more general version of the problem, namely weighted monitoring sets problem. It consists in assigning a weight for each node representing its cost.

This thesis is organized as follows: **Chapter 2** gives a short overview of some basic graph theory concepts. Moreover, this chapter presents preliminary definitions, that are needed for the understanding of the results exposed throughout this thesis. Then the rest of thesis is divided into three main parts.

In the first part, we study a problem related to the decomposition of graphs. This part is divided into two chapters. **Chapter 3** presents an overview of existing decomposition problems. In addition, several applications of decomposition are discussed to motivate the choice of this problem. In **Chapter 4**, we investigate the problem of the (C_k, S_k)-decomposition of the complete multigraph λK_n. We give the necessary and sufficient conditions for the existence of such decomposition.

The second part of this thesis is composed from two chapters. **Chapter 5** presents a literature review of the dominating set problem and its variants. We give a particular interest on the $[i, j]$-dominating set problem and $[i, j]$-total dominating set problem. In **Chapter 6**, we focus on the $[1, 2]$-dominating set problem and $[1, 2]$-total dominating set problem. We study two numerical invariants of graphs which concern the $[1, 2]$-dominating number and the $[1, 2]$-total dominating number. We give the exact value for generalized Petersen graphs $P(n, k)$ when $k = 2$.

The final part of the thesis is completely devoted to the edge monitoring problem. This part is split into three chapters. **Chapter 7** presents an overview of known results as well as new results about the edge monitoring problem. The motivation to study this problem is also discussed. We present the edge monitoring problem in general by presenting the problem and its variants. Some bounds on edge monitoring number and characterizations are also presented. Then, we focus in particular on 1-uniform monitoring problem by developing some results on general graphs and also in particular classes of graphs, e.g. path power, split graph, etc. An algorithm for finding the minimum 1-uniform edge monitoring set in the square of a tree is also presented. **Chapter 8** is devoted to the study of the edge monitoring problem from the perspective of parameterized complexity. Some preliminary notions are presented. We prove that the edge monitoring problem is $W[2]$-hard when parameterized by the size of the solution. Moreover, we present two algorithms that solve the problem in general graphs and in the particular case of apex-minor free graphs. Afterwards, we give in **Chapter 9** different study results of a more general problem, namely weighted edge monitoring on several graph classes: complete graphs, block graphs, interval graphs and cographs.

Finally, **Chapter 10** summarizes all results of this thesis and gives some suggestions for further research.

List of publications arising from this thesis

International journals

1. F. Beggas and B. Neggazi. A Note on Hamiltonian Decomposition of Bubble-Sort Graphs. International journal of Computer Mathematics 93(7): 1074-1077 (2016).
2. F. Beggas, M. Haddad and H. Kheddouci. Decomposition of complete multigraphs into stars and cycles. Discussiones Mathematicae Graph Theory 35(4): 629-639 (2015).
3. G. Bagan, F. Beggas, M. Haddad and H. Kheddouci. Edge Monitoring Problem on Interval Graphs. Electronic Notes in Discrete Mathematics 54: 331-336 (2016).
4. J. Baste, F. Beggas, H. Kheddouci and I. Sau. On the Parameterized Complexity of the Edge Monitoring Problem, Information Processing Letters 121: 39-44 (2017).

International conference

5. F. Beggas, M. Haddad and H. Kheddouci. Decomposition of complete multigraphs into stars and cycles. ICGT 2014, 9th International colloquium on graph theory and combinatorics, Grenoble, France, 2014.

Submitted papers

6. F. Beggas, V. Turau, M. Haddad and H. Kheddouci. $[1, 2]$-Domination in Generalized Petersen Graphs, submitted to Discrete Applied Mathematics (January 2017).
7. F. Beggas, M. Haddad and H. Kheddouci. Edge Monitoring in Graphs, submitted to Graphs and combinatorics (January 2017).

Chapter 2

Preliminaries

Contents

Graph theory is the study of the properties of graphs. This chapter presents basic notions of graph theory which are required throughout this thesis. We begin by introducing the concept of graph and the common terminology used around them. Most of this terminology is standard and can be found in any classical book on graph theory ([BM76, Ber62, W$^+$01]). After that, we define some classes of graphs and their properties useful for understanding the presented work. Then, we give some basic notions of a classical computational complexity, approximation Algorithms and Parameterized complexity. We will end this chapter by presenting two well-known graph problems relevant to this thesis.

2.1 Basic notations

In this section, we give a short overview of standard graph terminology used throughout this thesis. Some others will be given later when necessary.

Graph: a graph G is a pair of sets $(V(G), E(G))$, where $V(G)$ is the set of vertices (*a.k.a.* nodes) and $E(G) \subseteq V(G) \times V(G)$ is the set of edges, formed by pairs of vertices. If e is an edge that connects u and v. The vertices u and v are called the extremities of e. The cardinality of the vertex set $V(G)$, denoted by $|V(G)| = n$, is called the *order* of G. The cardinality of the edge set $E(G) = m$, we called the *size* of G and we denoted by $|E(G)|$.

Directed graph: a directed graph (*a.k.a.* digraph) is a graph where all the edges have a direction associated with them. In other words, its set of edges is represented by a set of ordered pairs of vertices, called directed edges or directed arcs.

Multigraph: is a graph which is permitted to have multiple edges (*a.k.a.* parallel edges) that have the same extremities. In other words, two vertices may be connected by more than one edge. A graph is *simple* if there is at most one edge between every two vertices.

Except for Part I, the graphs considered in this thesis, are an undirected finite graphs without loops or multiple edges.

Degree: is the number of edges incident to the vertex. It's also called the *local degree* or *valency*. The degree of a vertex v in the graph G is denoted by $\deg(v)$. If $\deg(v) = 0$, a vertex v is called an *isolated vertex*. A vertex of degree one is called a *leaf* or *pendant vertex*. The maximum degree of a graph G, denoted by $\Delta(G) = max\{deg(v) : v \in V(G)\}$, and the minimum degree of a graph, denoted by $\delta(G) = min\{deg(v) : v \in V(G)\}$.

Subgraph: a graph H is a subgraph of a graph G, denoted $H \subseteq G$, if the vertex set $V(H)$ of H is contained in the vertex set $V(G)$ of G and all edges of H are edges in G, *i.e*, $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. For any vertex subset $S \subseteq V(G)$, the *subgraph induced* by S denoted by $G[S] = (S, E_s)$ contains all the edges of $E(G)$ whose extremities belong to S. As particular subgraphs, we have clique and independent set defined below.

Clique: is a subset of vertices of a simple graph G such that every two distinct

vertices in the clique are adjacent. A maximal clique is a clique that cannot be extended by including one more adjacent vertex, that is, a clique which does not exist exclusively within the vertex set of a larger clique. A maximum clique of a graph G, is a clique, such that there is no clique with more vertices. The clique number of a graph G, denoted by $\omega(G)$, is the number of vertices in a maximum clique in G.

Independent set: independent set (*a.k.a. stable set*) is a set of vertices in a graph such that no two of which are adjacent. In other words, it is a set S of vertices such that for every two vertices in S, there is no edge connecting the two.

Neighborhood (vertex and edge): the (open) neighborhood of a vertex v in a graph G is the set of all vertices adjacent to v, denoted by $N(v) = \{u : uv \in E(G)\}$. The number of neighbors of v corresponds to the degree of v in G, then $|N(v)| = \deg(v)$. The closed neighborhood of v is $N[v] = N(v) \cup \{v\}$. For a set $S \subseteq V$, $N(S) = \bigcup_{v \in S} N(V)$ and $N[S] = \bigcup_{v \in S} N[V]$. The neighborhood of an edge e in a graph G is the set of all edges having at least a common extremity with the edge e, denoted by $N(e)$.

Connectivity: a graph G is connected if there exists a path between any two distinct vertices of G. Otherwise, the graph G is disconnected. The connectivity is minimum number of elements (vertices or edges) that need to be removed to disconnect the graph.

Distance and diameter: the distance between two vertices v and u in a graph G, denoted by $dist(u, v)$, is the number of edges of a shortest path connecting them. The diameter d of G is the maximum distance between any two vertices of G.

2.2 Some families of graphs

Graphs can be used as a modeling tool for many problems of practical importance. In this section, we present some wide-known family of graphs, that are considered throughout this thesis. Much more details and definitions on the graph classes can be found in [BS+99].

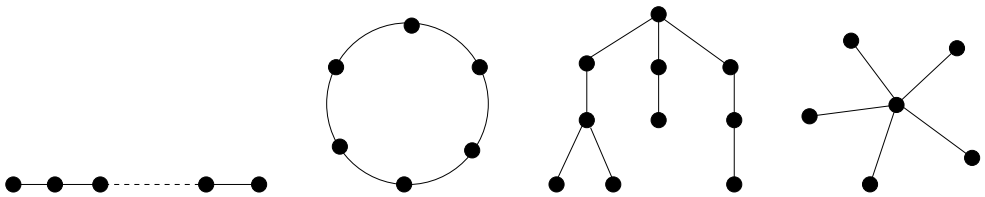
Path graph: a path P_n is a connected graph having n vertices (with length equal $n+1$). It's a sequence of vertices $(v_1, v_2, ..., v_n)$ such that each edge v_i, v_{i+1} exists in $E(P_n)$. The path graph P_n is also considered as a tree with two vertices having degree 1, and the other $n-2$ vertices with degree equal 2. A path is called simple if all its vertices are distinct (see Figure 2.1(a)). A path containing all the vertices of a graph G is called a *Hamiltonian path* of G.

Cycle graph: a cycle C_n with $n \geq 2$ is a connected graph having n vertices (n is also called the length of the cycle). It consists of a sequence of vertices $(v_1, v_2, ..., v_n)$ starting and ending at the same vertex, with each two consecutive vertices in the sequence adjacent to each other in the graph. In other words, for every $i \in \{1, ..., n-1\}$, the edge $v_i v_{i+1}$ exists in $E(C_n)$ and $v_n v_1$ also. A simple cycle is a cycle with no repetitions of vertices and edges (see Figure 2.1(b)). A cycle containing all the vertices of a graph G is called a *Hamiltonian cycle* of G.

Tree graph: a tree T_n is a connected graph with no cycles and having n vertices (see Figure 2.1(c)). Recall that a vertex with degree one is called a *leaf* and a vertex of degree at least two is called an *internal vertex*. A tree is called a *rooted tree* if one of its vertices has been designated the root, in which case the edges have a natural parent-child orientation, towards the root. A vertex v in a rooted tree is a descendant of a vertex u if u lies on the unique path from the root to v. The parent of a vertex v is the last vertex before v in a path from the root to v. The depth of a vertex v in a rooted tree is the length of the path from the root to v. Thus, the depth of the root is 0.

Star graph: the star graph S_n, is a tree with $n+1$ vertices such that one vertex, called the central node, has degree n and the other n vertices have degree 1 (see Figure 2.1(d)).

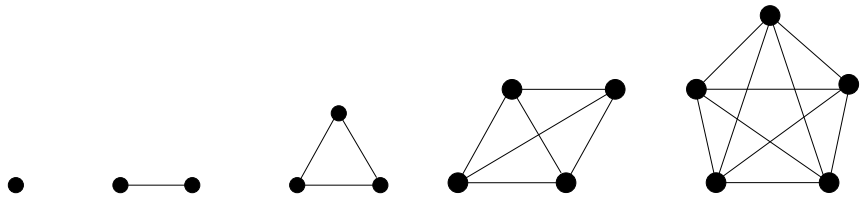
Complete graph: a complete graph (*a.k.a.* Clique) is a *simple undirected graph* in which every pair of distinct vertices is joined by exactly one edge. The complete graph with n vertices, denoted by K_n, is a regular graph with degree equal $n-1$ and it has $n(n-1)/2$ edges. In Figure 2.2, we give some examples of complete



(a) Path (b) Cycle (c) Tree (d) Star

Figure 2.1: Some basic simple graphs.

graphs. Furthermore, a *complete multigraph*, denoted by λK_n, is a complete graph K_n in which every edge is taken λ times.

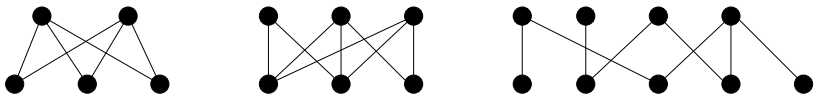


(a) K_1 (b) K_2 (c) K_3 (d) K_4 (e) K_5

Figure 2.2: Complete graphs K_n for $n = 1, 2, 3, 4, 5$.

Bipartite graph: we say a graph $G = (V, E)$ is *bipartite* if its vertex set $V(G)$ can be divided into two disjoint non-empty subsets A and B, such that every edge in $E(G)$ has one extremity in A and the other in B. Therefore, a bipartite graph is a graph that does not contain any odd-length cycle. The *complete bipartite graph* on n and m vertices, denoted by $K_{n,m}$ is the bipartite graph $G = (A, B, E)$, where A and B are disjoint subsets of size n and m, respectively, and E connects every vertex in A with every vertex in B. It follows that $K_{n,m}$ has $n * m$ edges. If $|A| = |B| = n$, that is, if the two subsets have equal cardinality, then the graph is called a *balanced* bipartite graph and we denoted by $K_{n,n}$. In Figure 2.3, we give three examples of bipartite graphs.

Multipartite graph: a graph $G = (V, E)$ is *multipartite* if whose set of vertices



(a) (b) (c)

Figure 2.3: Some examples of bipartite graphs.

$V(G)$ can be divided into non-empty disjoint subsets (called also parts) in which no two vertices in the same part have an edge connecting them. The complete multipartite graph is a multipartite graph such that any two vertices that are not in the same part have an edge connecting them. We will denote a complete multipartite graph with k parts by $K_{n_1,n_2,...,n_k}$ where k_i is the number of vertices in the i_{th} part of the graph.The bipartite graph is a multipartite graph having two parts. Figure 2.4 is an example of multipartite graph with three parts.

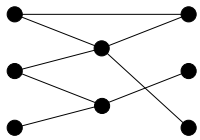


Figure 2.4: Example of multipartite graph.

Planar graph: planar graph is the graph that can be drawn on the plane in such a way that no edges cross each other. As an example, complete graphs are planar only for $n \leq 4$. The complete bipartite graph $K_{3,3}$ is nonplanar. More generally, a graph is planar if and only if it does not have K_5 or $K_{3,3}$ as a minor[1], as proved by Wagner [Wag37] .

Apex graph: an apex graph is a graph that becomes planar by the removal of a single vertex. The deleted vertex is called an apex of the graph. An apex graph may have more than one apex; for example, in the minimal nonplanar graphs K_5, every vertex is an apex. The apex graphs include graphs that are themselves planar, in which case again every vertex is an apex. The null graph is also counted as an apex

[1]A graph is a minor of another if the first can be obtained from the second by contracting some edges, deleting some edges, and deleting some isolated vertices.

graph even though it has no vertex to remove.

Regular graph: regular graph is a graph that each vertex has the same number of neighbors. In other words, every vertex of the graph has the same degree. A regular graph with vertices of degree k is called a $k$regular graph or regular graph of degree k. A 0-regular graph consists of disconnected vertices, a 1-regular graph consists of disconnected edges, and a 2-regular graph consists of disconnected cycles and infinite chains. A 3-regular graph is known as a cubic graph. The complete graph K_n is a $(n-1)$-regular graph.

Petersen graph: a Petersen graph is an undirected graph having 10 vertices and 15 edges as illustrated in Figure 2.5. It is a well known graph which is often used as an example or counterexample for graph problems. Many additional facts about the Petersen graph can be found in [HS93].

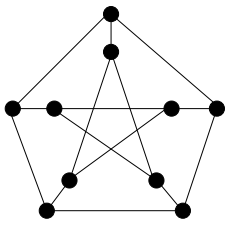


Figure 2.5: Petersen graph.

Generalized Petersen graph: Generalized Petersen graph is a family of cubic graphs introduced by Watkins [Wat69]. Generalized Petersen graph, denoted by $P(n,k)$ [Big93], such that $n \geq 3$ and $1 \leq k \leq \lfloor \frac{n-1}{2} \rfloor$, is a 3-regular graph with $2n$ vertices and $3n$ edges. It consists of a set of vertices defined as $\{u_0, u_1, ..., u_{n-1}, v_0, v_1, ..., v_{n-1}\}$ and a set of edges defined as $\{u_i u_{i+1}, u_i v_i, v_i v_{i+k} : 0 \leq i \leq n-1\}$ where all subscripts should be reduced to modulo n. With this notation, the (classical) Petersen graph, defined before, is $P(5,2)$. As a known result, $P(n,k)$ is bipartite if and only if n is even and k is odd.

Graph power: the graph power of a graph G is another graph denoted by G^k, that represents the k^{th} power of G It has the same set of vertices of G and an edge

exists between two vertices when their distance in G is at most k. G^2 is called the square of G and G^3 is called the cube of G. Figure 2.6 is an example of graph power.

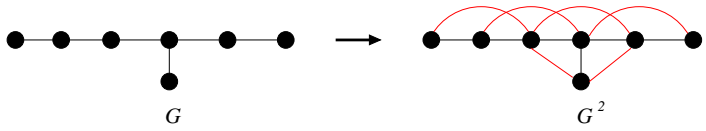


Figure 2.6: Example of graph power two G^2 of G.

Comparability graph: A graph $G = (V, E)$ is a comparability graph if there exists a poset $<$ over V such that $\{x, y\} \in E$ if and only if $x < y$ or $y < x$ for every $x, y \in V$.

Wheel graph: a wheel graph, denoted by W_n, is a graph with n vertices ($n \geq 4$), formed by connecting a single vertex (universal vertex) to the $n - 1$ other vertices that form a cycle of length $n - 1$.

Chordal graph: a graph G is said *Chordal* if every induced cycle in G should have at most three vertices. In other words, it is a graph in which all cycles of four vertices or more have a chord, which is an edge that is not part of the cycle but connects two vertices of the cycle.

Block graph: block graph (*a.k.a. clique tree*) is an undirected graph whose blocks are cliques. To find more characterization about block graphs in [BJT10]. Note that block graphs are chordal.

Split graph: a graph G is a split graph if its vertices can be partitioned to form a clique and an independent set [FH76]. Split graphs are a special class of Chordal graphs.

Interval graph: an interval graph is an undirected graph formed by a set of intervals. Each interval represents a vertex and each edge that connects two vertices corresponds to the intersection of two intervals. Figure 2.7 represents an example of interval graph.

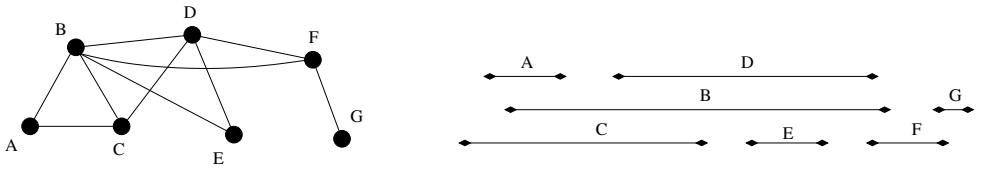


Figure 2.7: Example of interval graph.

Disc graph: we define a disk graph (*a.k.a. DG*) as the intersection graph of a set of disks in the Euclidean plane. DGs have a very simple geometric structure. The study of this class of graphs is motivated by its applications which can be found in radio networks, map labeling, and in sensor networks. Disc graph can be considered as an extension of the concept of the interval graphs family.

If all discs have unit diameter, we have a unit disc graph (*a.k.a. UDG*). It is a graph formed from a collection of equal radius circles in the plane such that the center of the circles represent the set of vertices, in which two vertices are connected by an edge when them corresponding circles intersect. To find more characterization about unit disc graphs, refer to [CCJ90a].

H-free graph: A graph G is called H-free for some graph H if G does not contain an induced subgraph isomorphic to H and G is called (H, F)-free for some graphs H and F if G does not contain any induced subgraph isomorphic to neither H or F. A graph G is *H-minor-free* if G does not contains H as a minor.

Cograph: cograph (*a.k.a. P_4-free graph*) is a graph that can be generated from the graph K_1 by complementation and disjoint union as follows: 1. any single vertex graph K_1 is a cograph; 2. if G is a cograph, so is its complement graph G; 3. if G and H are cographs, so is their disjoint union $G \cup H$.

As an example, the complete graphs and complete bipartite graphs are special cases of cographs. Most algorithmic problems can be solved on this class in polynomial time, and even linear, because of its structural properties.

2.3 Computational complexity

Throughout this section we give a brief introduction to the field of computational complexity. For a more complete introduction to computational complexity there are many good books such that a classic famous book by Garey and Johnson [GJ02], a book by Christos H. Papadimitriou [Pap03] or a more advanced book by S. Arora and B. Barak [AB09].

2.3.1 Classical complexity theory

An algorithm is a sequence of instructions that allow us to solve a computational problem. Informally, it takes an input data of the problem and transforms it to results. Hence, algorithm and problem are two concepts that compliment each other. As we explain in sections below, many practical problems can be represented by graphs. Then, the study and analysis of algorithms used to solve graph problems is therefore of practical importance.

Among the computational problems that usually exist in theory we find *decision* problems and *optimization* problems. A decision problem is to check if some property is true or false. Then, the two possible answers are yes or no. Optimization problem is to find a solution where the cost, quality, size, or some other measure is as large or small as possible. In fact, for every optimization problem, there is an associated decision problem.

The computational complexity helps us to classify these problems according to their difficulty. In other words, it permits to measure the complexity of the problem when computing the solution to see how difficult a problem is. In this context, complexity is measured by the amount of resources required to solve the problem. As the most common required resources, we distinguish two types: time complexity and space complexity.

Time complexity is generally expressed as functions of the input size of the problem, using $\mathcal{O}$ notation. In graph theory, for an input graph $G = (V, E)$, it is common to estimate complexity in terms of the number of vertices $|V|$ and/or in terms of the number of edges $|E|$. For example, a polynomial time algorithm is one for which the number of steps for a given input is upper bounded by a polynomial function of the size of the input.

The most popular question in computational complexity is what the difference between P and NP.

In computational complexity theory, all decision problems that have a *deterministic polynomial time* algorithm, belong to the complexity class P. In other words, the answer yes or no of the problem can be decided in polynomial time. All the decision problems in P are classified as nice or more technically speaking *tractable* or *efficient*.

There exist some problems that don't necessarily run in polynomial time but whose solutions can be verified in polynomial time. These problems belong to the class NP (*Nondeterministic polynomial time*). Observe that P $\subseteq$ NP.

The NP-complete complexity class represents the most difficult problems of the NP class such that it contains the set of all problems X in NP for which it is possible to reduce any other NP problem Y to X in polynomial time (polynomial reduction). All decision problems of this class are classified as bad, *intractable* or *inefficient*.

A problem X is NP-hard, if there is an NP-complete problem Y, such that Y is reducible to X in polynomial time. This means that we can solve Y quickly if we know how to solve X quickly. Formally, Y is reducible to X, if there exist a polynomial time algorithm A that transform instances y of Y to instances $x = A(y)$ of X in polynomial time, with the property that the answer to y is yes, if and only if the answer to $f(y)$ is yes. Intuitively, these are the problems that are at least as hard as the NP-complete problems. Then, if there exists a solution to one NP-hard problem in polynomial time, there exists a solution to all NP problems in polynomial time. Note that NP-hard problems do not have to be in NP.

Depending on the properties of the graph, the complexity can change. For example, the problem can be NP-hard in general classes of graphs and polynomial in some special classes.

Hence, the unfortunate fact that we cannot find the optimum solution in polynomial time doesn't mean that the problem cannot be studied or need to be ignored.

In practice, some solutions are possible. We have two possible ways in order to deal with NP-hard problems: approximation algorithms and parameterized complexity. Note that these two ways can collaborate together [Mar08] but in this thesis we use one of each separately for the same problem but in different graph classes

(see Chapters 8 and 9).

In the two following subsections we give more details about this two practical ways.

2.3.2 Approximation algorithms

Most of optimization problems are NP-hard and more especially the problems having important applications in real life. Assuming that P $\neq$ NP, it is unlikely that there can ever exist efficient polynomial time exact algorithms to solve NP-hard problems. In this perspective, the field of approximation algorithms allows us to find polynomial time algorithms with the fastest running time used to give approximate solutions of optimization problems. In other words, the aim is to relax the requirement that the given solution is the optimum and this help us to find an approximate solution in faster time.

Approximation needs to be close to the optimal solution, this guarantees the quality of the solution which is measured by the factor ρ. This means that, for a ρ-approximation algorithm A which give the approximate solution $A(x)$ for an instance x will not be less (for maximizing problem) or more (for minimizing problem) than a factor ρ times the value of an optimum solution.

Hence, A polynomial time approximation scheme (PTAS in short) is an algorithm which takes an instance of an optimization problem and a parameter $\varepsilon > 0$ to produce a solution that is within a factor $1 + \varepsilon$, in polynomial time. Other variants exist such as fully polynomial time approximation scheme (FPTAS in short) which is an approximation scheme whose time complexity is polynomial in the input size and also polynomial in $1/\varepsilon$. As typical examples for an approximation algorithms applied in graph theory, we have approximation algorithm for vertex cover problem (find minimum set S of vertex such that every edge in the graph is incident to at least one vertex in S) [Hoc82a], for traveling Salesman problem (find the shortest possible route, for a list of cities, that visits each city exactly once and returns to the starting city) [Lap92] and more others can be found in [ACG+12].

More details and some more advanced explanations on approximation complexity, can be found in [ACG+12, Vaz13].

2.3.3 The word of parameterized complexity

As another way to deal with the computational hard problems we have parameterized complexity, introduced in the 1990's. For a complete introduction to parameterized complexity there exist many good books for example, we have the two books by Downey and Fellows [DF12, DF13] and a more recent textbook by Cygan *et al.* [CFK+15].

Parameterized complexity can be seen as a refinement of classical complexity in which one takes into account not only the input size, but also a *parameter* k. As an example of parameters in graph theory, we have size of the solution, treewith, etc.

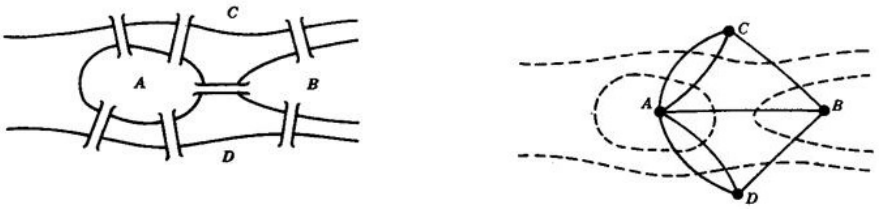
This *parameter* help us to get around the inevitable combinatorial explosion of classical computational complexity and separate the time complexity into two parts: first part that depends purely on the size of the input and represented with a polynomial function, and the second part which is an arbitrary function that depends on the *parameter* k. In this case, a problem is called fixed-parameter tractable (FPT in short) and the time complexity of the corresponding algorithm (fixed parameter tractable algorithm) is represented as $\mathcal{O}(p(n)f(k))$ Where $p(n)$ is some polynomial function of the input size n and $f(k)$ is an arbitrary function in k. Hence, this means that the complexity of the problem scales polynomially with the size of the input data which is nice as time complexity. However, it also scale arbitrarily (usually exponentially) with the parameter k. This separates out the central hardness of the problem such that the hard part (the bad part) of the problem is blamed on the parameter k, while the easy part (the nice part) of the problem is charged to the size of the input data. Thus, the choice of the right parameter k is very important since the problem can be considered as hard from parameterized viewpoint with some parameterization, but tractable (soluble) with another parametrization.

Since there exist many hard problem in literature, this approach has enormous practical implications for these problems. If you find a problem that's fixed parameter tractable and the parameter k has small values, it can be significantly more efficient to use the fixed parameter tractable algorithm than to use the classical brute force algorithm.

2.4 Two favorite graph problems

Graph modeling helps us to understand a problem because it determines a single formal vocabulary for different situations and allow us to find a method to solve the problem.

Graph theory was born in 1736 with the answer of Euler (1707-1783) to the famous problem of the bridges of Königsberg (Euler, 1736) [Eul41]. The problem is proposed as follows. Seven bridges of the city of Königsberg , the east Prussian city and now renamed Kaliningrad, relied four places as shown in Figure 2.8. The question is can all be traversed in one trip without doubling back, with the additional requirement that the trip ends in the same place it began. This problem is equivalent to asking if the multigraph with four nodes and seven edges (see Figure 2.8) has an Eulerian cycle [W+01].



(a) the bridges of Königsberg (b) Euler's graph representation

Figure 2.8: Königsberg Bridge Problem.

Thence, there exist a variety of graph problems in literature [BM76, W+01, Xu13]. Some problems are old such as color problems (1852) [JT11, Kub04], Traveling salesman problem (1832) [LLRKS85], and others more recent such as Roman domination problem (1999) [Ste99, CDHH04], Rainbow connection problem (2008) [CJMZ08].

An history of graph theory from 1736 to 1936 can be found in [BLW76], a more recent update can be found in [Bol13] and some open problems can be found in [Wes].

All problems considered in graph theory are typically motivated by them practical applications in a wide range of areas such as: networks [Deo94], biology [SS+73, PSM+11], chemistry [Bon91, HJ88] social science [WF94], etc.

Since this thesis is focusing on graph theory problems, let us in the following introduce the two central problems considered in this work, namely decomposition and domination, to acquaint the reader with. We hope this section helps the reader to understand our motivation to study these two problems.

2.4.1 Decomposition problems

Graph decomposition is one of the most famous and known problems in graph theory. It consists to break an input graph into smaller pieces (subgraphs) by satisfying some constraints.

Since it is a very vast research area, there exist various types of decomposition problems in literature. Nevertheless, there exist two major kinds of decompositions of graphs, depending on the way that we want to decompose, called edge-decompositions (*a.k.a.* edge-disjoint decomposition) and vertex-decompositions (*a.k.a.* vertex-disjoint decomposition) respectively.

The first kind consists to decompose the input graph into subgraphs such that each edge belongs to one and only one subgraphs. In other words, we decompose the edges of the input graph into groups and each group constitutes a subgraph. The second decomposition is based on vertices. It consists to decompose a graph into subsets of vertices such that each vertex must belong to one and only one subsets. Graph decomposition is usually associated to the edge-decompositions and in a lot of research vertex-decompositions are called colorings and not considered as decomposition. Then, in all this work, the only kind considered is the edge-decomposition and for the sake of simplicity we call it graph decomposition.

Graph decomposition is motivated by a lot of applications in practice such as in fault tolerance [MB98] spanning structures [Fre85], load balancing [Fox88, KNS09], graph similarity and matching [LH01], pattern recognition techniques [FP75], big graphs [SXZF07], etc.

More details on graph decomposition, motivation and our results can be found in Part I of this thesis.

2.4.2 Domination problems

The study of domination problems in graph theory has a long history. Mathematical study of domination in graphs began around 1960. A brief history of domination

in graphs are given in Chapter 5.

A dominating set of a graph is a subset D of the vertices such that every vertex is either in D or adjacent to a vertex in D. Over the years different variations of graph domination were introduced e.g. total dominating sets, connected dominating sets, k-tuples dominating sets, etc.

The first domination problems came from chess. In 1850, different chess players were interested in the minimum number of queens such that every square on the chess board either contains a queen or is attacked by a queen [HHS98]. Then, the number of required queens for such problem corresponds to the dominating number. Other than chess, domination in graphs has a lot of applications in several fields.

Concept of domination appears in facility location problems, where the number of facilities (e.g., fire stations, hospitals, shops, guards) is fixed and one attempts to minimize the distance that a person needs to travel to get to the closest facility. Domination can be also found in problems involving to find sets of representatives, in monitoring communication or electrical networks. As another application of domination we have cluster heads. In wireless sensor networks, it consists to group sensor nodes into clusters and electing cluster heads for all the clusters. Domination can help us here to select appropriate cluster heads.

The literature on this subject is rich and growing rapidly since we always have new variants of domination problems. In this thesis, we give a particular interest to study two variants of domination, called $[j, k]$-dominating set and edge monitoring problem. Our motivation about studying these two problems and more details can be found in Parts II and III respectively.

Part I

Graph decomposition

Chapter 3

Overview of graph decomposition

Contents

The first part of thesis is devoted to the problem of graph decomposition. Graph decomposition is incredibly well studied area and it is a very broad topic. The concept of graph decomposition proved to be useful in many ways and it is crucial in the study of lot of theoretical applications. This chapter is attended as an introduction to some graph decomposition problems. We give a brief history of the decomposition in graphs and the motivation of take up this area for the present research. We also present some well known results on decomposition of specific graphs which are closely related to the problem presented in the next chapter.

3.1 Motivation

Research in graph decomposition started with a result of Walecki in 1890 concerning the existence of a Hamiltonian decomposition of complete graph with an odd number of vertices and a lot of open problems have emerged after that.

Like most of research areas, graph decomposition has experienced an exponential growth and researchers have become increasingly specialized. Hence, for the same

type of decomposition we can find several results on different graphs and for the same graph we have different types of possible decomposition.

The study of graph decomposition has close ties to several areas including network theory, design and graphic theories, geometry, coding theory and obviously graph theory as well as other important areas. This motivated researchers to extensively develop this area. Results on graph decomposition are applied in a wide range of applications such as analysis of structures, fault tolerance in network architecture, detection of geometric patterns and textures in graphic design, graph decomposition into specific patterns, summarizing and compressing graphs, graph similarity and subgraph matching, the study of properties in big graphs... and more other applications. To have a general idea about how useful is the concept of graph decomposition in real life and to understand its importance, we focus on three different applications that use graph decomposition.

- Fault tolerance.

Fault tolerance is the property that allows a system to continue the different operations properly even if there is a failure in one or more of its components [LA12]. Fault tolerance is often related to Hamiltonian cycle decomposable architecture.
Hamiltonian decomposition aims to find all edge disjoint cycles where each one is a graph cycle through a graph G that visits every vertex exactly once [Ber78]. This type of decomposition have a direct application in networks field to have fault tolerance properties. Generally, Hamiltonian decomposition is used to obtain alternative communication routes in computer networks. Thus, if there is any communication failure in one circuit, then another circuit can be used. In other words, if the link between two different stations (or nodes) in the network is broken, we have the possibility to use another link.
However, other constructions are also used as a model in fault tolerant networks such as in [DH90] for trees architecture, [FD89a] for stars architecture and [FD89b] for complete multipartite architecture.

- Combinatorial designs.

Another research area that used a graph decomposition is the combinatorial construction in designs theory [Sti07]. This research area is very wide and discusses

a lot of fundamental questions based on arrangement of elements into subset with satisfying some properties. For example, in design theory, we associate the name handcuffed designs [HM77] for path decomposition and the name resolvable designs [HRCW72] for star decomposition. Both decompositions are the most used in this area but we also find a large number of combinatorial design problems that can be described in terms of another types of graph decomposition.
As a classical problem of combinatorial designs we find combinatorial index-file organization scheme problems [I$^+$78]. It consists to give a suitable file organization scheme in database systems. There exist different data file organizations used in a database environment. Using graph decomposition, the idea is to model the set of data files by graph and decompose it into specific subgraphs to give such an organization.

- Social network analysis.
The concept of social networks is widely growing since the importance of studying social interactions and behavioral science. Social network can be represented by a graph which consists on a set of vertices corresponding to the actors and a set of edges representing the relations between these actors. A small social network can be visualized directly by its corresponding graph but larger social networks can be more difficult to envision and analyze. In this perspective, there exist variant tools and techniques to study patterns of relationships that connect social actors in social networks, refer [Sco12, WF94]. As a particular technique, we have graph decomposition that helps us to study and analysis these social networks. For example, star decomposition is widely used to identify major actors and interactions in social network. We just need to decompose the network into stars and select all stars having maximum degree. Clique decomposition can be used for detecting communities in social networks. Furthermore, other graph decompositions can also be used to find a set of people or groups of people having some pattern of contacts or special interactions between them. There exist several important research papers and books in this sense, we can refer to [BZ03, BM04, FB07]

All the above facts, motivate us to take up this thrust area for the present research.

3.2 Two types of decomposition

In this section we have chosen to present two well-known graph decompositions namely simple decomposition and multidecomposition of graphs.

3.2.1 Simple decomposition of graphs

The H-decomposition of G aims to partitioning the edge set of G into edge disjoint copies of H. By other words, it consists of decomposing an input graph $G = (V, E)$ into a collection of smallest subgraphs $H_1, H_2, ..., H_k$, such that each edge of G belongs to exactly one subgraph $H_i : 1 \leq i \leq k$. Note that decomposing the graph G means that there is no remaining edges and all the edges need to belong to one subgraph. If G has an H-decomposition, then we say that G is H-decomposable.

Due to many applications in computer science, an intensive research about simple decomposition has been done on many special subgraphs. As an example of basics and well-known simple decomposition, we have star decomposition (denoted by S_k-decomposition) [Tar79], path decomposition (denoted by P_k-decomposition) [Tar83] and cycle decomposition (denoted by C_k-decomposition) [ABS90]. Other decompositions are studied in the literature, for more details, refer to Section 3.3. In all the decompositions, we consider the fact that all the subgraphs have the same type H and also the same size. Note that determining if a graph G admits an H-decomposition was conjectured to be NP-complete by Holyer in [Hol80] and proved in [CT91, DT92] for all subgraph H which have a connected component of size 3 or more.

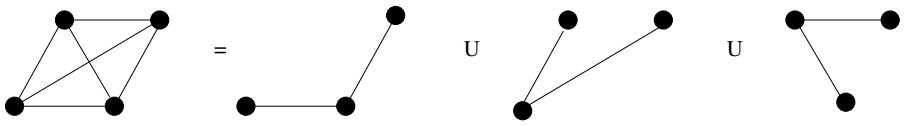


Figure 3.1: Example of P_3-decomposition of K_4.

As an example, Figure 3.1 presents the P_3-decomposition of K_4. The obvious necessary condition that K_4 need to have in order to admit a P_3-decomposition is that the size of K_4 have to be a multiple of the size of P_3. Thus, for example K_6 doesn't admit a P_3-decomposition since the necessary condition is not satisfied. Furthermore, depending on the type of decomposition for the same graph $G = K_4$,

additional conditions can be considered.

3.2.2 Multidecomposition of graphs

An (H, F)-decomposition of a graph G consists on finding a partition of the edge set of G into edge disjoint isomorphic copies of H and F using at least one copy of each.

Formally, it consists of decomposing an input graph $G = (V, E)$ into a collection of subgraphs of two types H and F (or more than two recently) as follow $H_1, H_2, ..., H_k, F_1, F_2, ..., F_l$, such that each edge of G belongs to only one subgraph $H_i : 1 \leq i \leq k$ or $F_i : 1 \leq i \leq l$.

If G has an (H, F)-decomposition, we say that G is (H, F)-decomposable (or (H, F)-multidecomposable).

As an example and to have a comparison point of view with simple decomposition, we decompose the same graph K_4 using in the above subsection into cycles and stars. This decomposition is called a (C_3, S_3)-multidecomposition and it is presented in Figure 3.2.

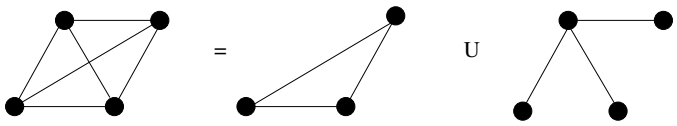


Figure 3.2: Example of (C_3,S_3)-decomposition of K_4.

3.3 Review of literature

Since the different applications relating to various decompositions on different graphs, this subject became an active research area. The literature on this subject has been surveyed and detailed in a lot of books and research papers such as the phd thesis of Priyadarshini [Pri13] which dealt with multidecomposition of multigraphs, the phd thesis of Sotteau [Sot80], the phd thesis of Gabel [Gab80], survey of Chung et al. [CG81], survey of Rodger [Rod91], survey of Lindner et al. on cycle decomposition [LR92], survey of Billington [Bil04], survey of Heinrich on path decomposition [Hei93], survey of Beineke [Bei96], survey of Bermond et al. [BS75], survey of Alspach et al. on cycle decomposition [ABS90], survey of Ushio [Ush93], survey of

Rodger [Rod91], the book written by Bosák [Bos90], the book of Foregger [For79], and the first results on graph decomposition can be found in the Lucas's book [Luc82].

In all the references cited above, the reader can find more detailed discussions about the graph decomposition. Now, we will focus more closely on a few specific decomposition problems that have more impact and are more intensively studied in the literature, and then present such a review of existing results on each type of decompositions. Note that we focus only on results that deal with necessary and sufficient conditions that need to have a graph to be decomposable into some special type of subgraphs.

When we consider the problem of H-decomposition of G, many questions can be asked but the most natural problem is to find the conditions on G for which H-decomposition exists. This allow us to find the require properties of the graph in order to have such decomposition.

Let consider the complete graph $G = K_n$. Complete graph is the most popular graph studied in decomposition problems. It can easily be seen that for any subgraph H of order at most n, the following two conditions are necessary in order to decompose K_n into H:

1. $\frac{n(n-1)}{2}$ is a multiple of the number of edges of H.
2. $n-1$ is a multiple of the greatest common divisor of the degrees of the vertices of H.

In other words, the size of K_n needs to be a multiple of the size of the subgraph H and the degree of K_n needs to be a multiple of the greatest common divisor of the degrees of the vertices of H. It's not difficult to find the necessary conditions. However, the real problem and the more difficult part is to establish the sufficient conditions.

In [Wil76], Wilson stated his fundamental theorem on the existence of an H-decomposition of the complete graph K_n for any fixed H as long as the number of edges of K_n is divisible by the number of edges of H and n is large enough. Since then, decomposition problems became an active research area. There have been several important research papers relating to various decompositions of different graphs. Let focus on three types of subgraphs: paths, cycles and stars. For the

complete graph K_n, the solution of the corresponding problem for $H = P_k$ was proved by Tarsi [Tar83]

Theorem 3.3.1 *Let k and n be positive integers. There exists a P_k-decomposition of K_n if and only if $k \leq n$ and $n(n1) \equiv 0[2k-2]$.*

for $H = C_k$ by Alspach et al. [AG01] and Sajna [Šaj02]

Theorem 3.3.2 *Let n and k be positive integers. K_n has a C_k-decomposition if and only if n is odd, $3 \leq k \leq n$, and $n(n-1) \equiv 0[2k]$.*

and for $H = S_k$ by Tarsi [Tar79] and Yamamoto et al. [YISE+75]

Theorem 3.3.3 *Let k and n be positive integers. There exists a S_k-decomposition of K_n if and only if $2k \leq n$ and $n(n1) \equiv 0[2k]$.*

Others results about H-decomposition and (H, F)-decomposition on some other specific graphs G have been investigated by many authors. We summarized some of them in the Table 3.1 for simple decomposition and Table 3.2 for multidecomposition. We have to note that some cited papers solve just some special cases. The hole problem is still open in some cases.

Graphs	**Paths**	**Cycles**	**Stars**
K_n	[Tar83]	[AG01, Šaj02]	[Tar79, YISE+75, LS96]
$K_{m,n}$	[Par98]	[Sot81]	[YISE+75]
λK_n	[Tar83]	[BHMS11]	[Tar79]
$\lambda K_{m,n}$	[Shy07, Tru85]	[Lee15]	[Lee15]
$K_{m,m,m}$	[LLL09, BCS10]	[CH77a, Cav98, Bil99, CB00]	[UTY+78, T+79]
λK_n^*	[MS06, MS12]	[Sch75, BF76, AGŠV03]	[CHR92]
$k-regular$	[FGK10, HJM99, BJ15]	[Mar12]	[LL05]
S_n^0	-	[LL00, MPS06]	[LLS+99]

Table 3.1: Table summarized some famous graph decomposition.

The problem of Hamiltonian decomposition of graphs has been extensively studied and it can be considered as the door of an infinite list of decomposition problems. In [Ber78], several problems related to Hamiltonian decompositions of graphs,

Graphs	P. & S.	P. & C.	S. & C.
K_n	[Shy10a]	[Shy12, Shy10b]	[AL14, BHK15]
$K_{m,n}$	[Shy13, LC13]	[JM14]	[Lee13]
λK_n	[LC15]	[PM09]	[BHK15]
$\lambda K_{m,n}$	[PM12, LC15]	[JM15]	[Lee15]
K_n^*	-	[Shy15]	-
S_n^0	-	-	[LL13]

Table 3.2: Table summarized some famous graph multidecomposition.

directed graphs and hypergraphs are treated. More results on Hamiltonian decomposition can be found in [ABS90, Hil84, BFM89, HR86, BN16].

Different types of decomposition of various types of graphs has been studied by many authors. One of the most studied is tree decomposition. For some results on tree decompositions; see [DO95, Pet96, BG12, Lon89]. Other types of decompositions are studied in various types of graphs such as crown decomposition of complete multigraphs [LG10], complete bipartite decomposition of complete graph [Tve82], multistars decomposition of complete bipartite multigraph [Lin10], isomorphic cubes decomposition of complete graph [Kot81], complete multipartite decomposition of complete graph [Hua91], and so on.

All the decompositions cited above are simple or multiple but recently, Lin and Jou investigate a new version of decomposition having three types of subgraphs. They consider the problems of the (C_k, P_k, S_k) -decomposition of the balanced complete bipartite graph $K_{n,n}$ (not published yet). In the same perspective, they also consider in [LJ16] the problem of the (C_k, P_k, S_k) -decomposition of the balanced complete bipartite multigraph λK_n, for $\lambda \geq 2$.

In the next chapter, we focus only on the *Multidecomposition of complete multigraph* into cycles and stars of same size.

Chapter 4

(S_k, C_k)-Multidecomposition of λK_n

Contents

In the previous chapter, we introduced the problem of *Decomposition of graphs into subgraphs* and we discussed some results that the research raised in this field. In this chapter, we focus on the *Multidecomposition of complete multigraph* into cycles and stars and we discuss the existence of such decomposition. Our decomposition result show the required conditions for the existence of (C_k, S_k)-multidecomposition. Note that the particularity is to have the same number of edges in all the subgraphs. A preliminary version of this work appeared in [BHK15].

This chapter is organized as follows: In Section 4.1, we present a couple of theorems related to this work and needed to be used in our discussion. After that we present some introductory results. More advanced results are discussed in Section 4.3 and Section 4.4 respectively. Furthermore, we discuss a new idea to deal with the remaining cases in Section 4.5. Section 4.6 concludes this first part.

4.1 Related work

Before presenting our results, we briefly revisit the essential theorems on C_k-decomposition and S_k-decomposition that are useful for our proofs.

Theorem 4.1.1 *(Tarsi [Tar79] and Yamamoto et al.[YISE+75]) A necessary and sufficient condition for the existence of an S_k-decomposition of λK_n is that:*

- $\lambda n(n-1) \equiv 0[2k]$,
- $n \geq 2k$ *for* $\lambda = 1$,
- $n \geq k+1$ *for even* λ,
- $n \geq k+1+k/\lambda$ *for odd* $\lambda \geq 3$.

Theorem 4.1.2 *(Bryant et al.[BHMS11]) Let λ, n and k be integers with $n,k \geq 3$ and $\lambda \geq 1$. There exists a decomposition of λK_n into cycles of k edges if and only if $k \leq n$, $\lambda\ (n-1)$ is even and k divides $\lambda n(n-1)/2$.*

There exists a decomposition of λK_n into cycles of k edges and a perfect matching if and only if $k \leq n$, $\lambda\ (n-1)$ is odd and k divides $\lambda n(n-1)/2 - (n/2)$.

Theorem 4.1.3 *(Alspach et al. [AG01] and Sajna [Šaj02]) Let n and k be positive integers. K_n has a C_k-decomposition if and only if n is odd, $3 \leq k \leq n$, and $n(n-1) \equiv 0[2k]$.*

Theorem 4.1.4 *(Yamamoto et al.[YISE+75]) Let $m \geq n \geq 1$ be integers. Then, $K_{m,n}$ is S_k-decomposable if and only if $m \geq k$ and $m \equiv 0[k]$ if $n < k$, $mn \equiv 0[k]$ if $n \geq k$.*

Theorem 4.1.5 *(Sotteau [Sot81]) For positive integers m, n, and k, the graph $K_{m,n}$ is C_k-decomposable if and only if m, n, and k are even, $k \geq 4$, $min\ \{m,n\} \geq k/2$, and $mn \equiv 0[k]$.*

4.2 Introductory results

In this section, we give some preliminary results

Let G be a graph. The order of G is the cardinality of its vertex set and the size of the graph G is the cardinality of its edge set. We begin with the following lemma to prove the necessary conditions when λK_n is (S_k, C_k)-decomposable:

Lemma 4.2.1 *Let $n \geq 3$ and $\lambda > 1$ be positive integers. If λK_n is (S_k, C_k)-decomposable, then $2 \leq k \leq n-1$ and $\lambda n(n-1) \equiv 0[2k]$.*

Proof. Since the minimum length of a cycle and the maximum size of a star in λK_n are respectively 2 and $n-1$, so $2 \leq k \leq n-1$ is necessary. Since λK_n has $\lambda n(n-1)/2$ edges and each subgraph in a (C_k, S_k)-decomposition has k edges, k has to divide $\lambda n(n-1)/2$. □

As an introduction result, we show in the following proposition that the necessary conditions in Lemma 4.2.1 of the (C_k, S_k)-decomposition of λK_n are also sufficient in the special case when $k = 4$.

Proposition 4.2.2 *Let $n > 4$ and $\lambda > 1$ be positive integers. There exists a (C_4, S_4)-decomposition if and only if $\lambda n(n-1)/2 \equiv 0[4]$.*

Proof. We distinguish two cases according to the parity of λ.

Case 1. λ is odd

Since $\lambda n(n-1)/2 \equiv 0[4]$ and λ is odd by assumption then $n(n-1) \equiv 0[8]$. We have two subcases:

Subcase 1.a. n is even

Since $n(n-1) \equiv 0[8]$ and n is even, this implies that $n \equiv 0[8]$. Let $n = 8\alpha$ with $\alpha \geq 1$, then λK_n can be decomposed into disjoint union of α copies of λK_8 and disjoint union of $\alpha(\alpha-1)/2$ copies of $\lambda K_{8,8}$. Every $\lambda K_{8,8}$ can be decomposed into S_4 using Theorem 4.1.4. We now decompose each λK_8 into C_4's and S_4's as follows: Note that $\lambda K_8 = K_8 \cup (\lambda - 1)K_8$. Since Theorem 4.1.1 implies that K_8 is S_4-decomposable and Theorem 4.1.2 guarantees that $(\lambda-1)K_8$ is C_4-decomposable, we have λK_8 is (C_4, S_4)-decomposable. Thus, λK_n is (C_4, S_4)-decomposable.

Subcase 1.b. n is odd

Since n is odd and $n(n-1) \equiv 0[8]$ by assumption then $n-1 \equiv 0[8]$. Let $n-1 = 8\alpha$. Since the degree of each vertex of λK_n equals to $\lambda(n-1)$ and is divisible by 4, we take one vertex and decompose all its incident edges into $2\lambda\alpha$ stars of 4 edges. The remaining graph is λK_{n-1} with $n - 1 = 8\alpha$. In this case, we use the same method

as the previous Subcase 1.a for λK_n with $n = 8\alpha$.

Case 2. λ is even

Recall that $n > 4$. We give the (C_4, S_4)-decomposition of λK_n as follows according to values of n:

- $n = 5$: Note that $\lambda K_5 = \lambda S_4 \cup \lambda K_4$. Since λ is even and using Theorem 4.1.2, we decompose λK_4 into C_4. Thus, λK_5 is (C_4, S_4)-decomposable.

- $n = 6$ or $n = 7$: We have $n(n-1) \equiv 0[2]$ and $\lambda n(n-1) \equiv 0[8]$ by assumption. Consequently, $\lambda \equiv 0[4]$ then we take incident edges of one vertex and decompose them into S_4's. The remaining graph is either λK_5 when $n = 6$ or λK_6 when $n = 7$. Both remaining graphs are C_4-decomposable using Theorem 4.1.2.

- $n = 8$: Since λ is even, λK_8 can be written as the disjoint union of $2K_8$'s. Now we give the (C_4, S_4)-decomposition of $2K_8$: each $2K_4$ is decomposed into C_4's by Theorem 4.1.2 and the $2K_{4,4}$ is decomposed into S_4's using Theorem 4.1.4. Since each $2K_8$ is (C_4, S_4)-decomposable then λK_8 is also (C_4, S_4)-decomposable.

- $n \geq 9$: Note that $\lambda K_n = \lambda K_4 \cup \lambda K_{n-4} \cup \lambda K_{4,n-4}$. Observe $|E(\lambda K_4)|$ and $|E(\lambda K_{4,n-4})|$ are divisible by 4. By assumption $|E(K_n)|$ is a multiple of 4, so $|E(\lambda K_{n-4})|$ is also a multiple of 4. We decompose λK_4 into cycles of 4 edges using Theorem 4.1.2 with λ even. $\lambda K_{4,n-4}$ is S_4-decomposable using Theorem 4.1.4. Since λ is even, we decompose λK_{n-4} into C_4 using Theorem 4.1.2. Thus, we conclude that λK_n is (S_4, C_4)-decomposable.

□

4.3 Multidecomposition of λK_n when $n \geq 4k$ or $n \geq 2k$ and λ even

In this section, we prove some lemmas and theorems each of them treating a special case of decomposition of λK_n into S_k's and C_k's.

The next proposition proves that λK_n is (S_k, C_k)-decomposable for all $n \geq 4k$ and $\lambda = 1$, so we complete the missing cases in [AL14] when $n \geq 4k$:

Proposition 4.3.1 *Let n and k be positive integers such that $n \geq 4k$ and $n(n-1)/2 \equiv 0[k]$, then the graph K_n is (S_k, C_k)-decomposable.*

Proof. Let $n = qk + r$ where q and r are integers with $0 \leq r < k$ and $q \geq 4$.
Note that: $K_n = K_{qk+r} = K_{2k} \cup K_{(q-2)k+r} \cup K_{2k,(q-2)k+r}$.

Clearly, $|E(K_{2k})|$ and $|E(K_{2k,(q-2)k+r})|$ are multiples of k. Thus $((q-2)k + r)((q-2)k+r-1)/2$ is also a multiple of k. We distinguish two cases according to the parity of k.

Case 1. k is odd
It follows that $K_{(q-2)k+r}$ is S_k-decomposable by Theorem 4.1.1 since $(q-2)k+r \geq 2k$, and $K_{2k,(q-2)k+r}$ is also S_k-decomposable by Theorem 4.1.4.

We write $K_{2k} = K_k \cup K_k \cup K_{k,k}$. Now, it is clear that each copy of K_k is C_k-decomposable when k is odd by Theorem 4.1.3 and $K_{k,k}$ is S_k-decomposable by Theorem 4.1.4.

Case 2. k is even
In this case, K_{2k} is S_k-decomposable by Theorem 4.1.1.
If n is even, then $(q-2)k + r$ is even. So, we can decompose $K_{2k,(q-2)k+r}$ into C_k using Theorem 4.1.5. Since $q \geq 4$, $(q-2)k + r \geq 2k$. Consequently, $K_{(q-2)k+r}$ is S_k-decomposable by Theorem 4.1.1. Conversely, if n is odd, then $(q-2)k + r$ is odd. Using Theorem 4.1.3, $K_{(q-2)k+r}$ can be decomposed into cycles of k edges and $K_{2k,(q-2)k+r}$ is S_k-decomposable by Theorem 4.1.4. Thus, we conclude that λK_n is (S_k, C_k)-decomposable when $\lambda = 1$. □

In the rest of this section, we will focus on complete multigraph λK_n where $\lambda > 1$. The following lemma gives sufficient conditions for decomposing λK_n into C_k's and S_k's where $\lambda > 1$ is odd and $n \geq 4k$.

Lemma 4.3.2 *Let n, k and $\lambda > 1$ be positive integers such that $n \geq 4k$ and λ is odd. If $\lambda n(n-1)/2 \equiv 0[k]$ then λK_n is (C_k, S_k)-decomposable.*

Proof. Let $n = qk + r$ where q and r are integers with $0 \leq r < k$ and $q \geq 4$.
Note that:

$$\begin{aligned}\lambda K_n = \lambda K_{qk+r} &= \lambda K_{2k} \cup \lambda K_{(q-2)k+r} \cup \lambda K_{2k,(q-2)k+r} \\ &= (\lambda - 1)K_{2k} \cup K_{2k} \cup \lambda K_{(q-2)k+r} \cup \lambda K_{2k,(q-2)k+r}\end{aligned}$$

$|E(\lambda K_{2k})|$ and $|E(\lambda K_{2k,(q-2)k+r})|$ are multiples of k. Using argument that $|E(\lambda K_n)|$ is a multiple of k i.e. $\lambda n(n-1)$ is divisible by k, then $\lambda((q-2)k+r)((q-$

$2)k + r - 1)/2 \equiv 0[k]$. Since $(\lambda - 1)(2k - 1)$ is even and $2k \geq k$ then $(\lambda - 1)K_{2k}$ is C_k-decomposable by Theorem 4.1.2. For K_{2k} and using Theorem 4.1.1 with $\lambda = 1$ implies that K_{2k} is S_k-decomposable. We now decompose $\lambda K_{(q-2)k+r}$:
We have $q \geq 4$, then $(q-2)k+r \geq 2k+r$ implies that $(q-2)k+r \geq 2k \geq 3k/2+1$ for any $k \geq 2$. Given that $\lambda \geq 2$ then $3k/2+1 \geq k+1+k/\lambda$ so $(q-2)k+r \geq k+1+k/\lambda$. Using Theorem 4.1.1 when λ is odd, since $(q-2)k + r \geq k + 1 + k/\lambda$, we have $\lambda K_{(q-2)k+r}$ is S_k-decomposable. Note that $\lambda K_{2k,(q-2)k+r}$ can be decomposed into λ copies of $K_{2k,(q-2)k+r}$. Since $K_{2k,(q-2)k+r}$ is S_k-decomposable by Theorem 4.1.4, so is $\lambda K_{2k,(q-2)k+r}$. Thus λK_n is (C_k, S_k)-decomposable. □

In the following lemmas, we will give sufficient conditions of the decomposition of λK_n into C_k's and S_k's where $n \geq 2k$ and λ is even or $gcd(\lambda, k) = 1$.

Lemma 4.3.3 *Let n, k and λ be positive integers such that λ is even. For all $n \geq 2k$, if $\lambda n(n-1)/2 \equiv 0[k]$ then λK_n is (C_k, S_k)-decomposable.*

Proof. Let $n = qk + r$ where q and r are integers with $0 \leq r < k$ and $q \geq 2$.
Note that: $\lambda K_n = \lambda K_{qk+r} = \lambda K_{(q-1)k} \cup \lambda K_{k+r} \cup \lambda K_{(q-1)k,k+r}$.

Obviously, $|E(\lambda K_{(q-1)k})|$ and $|E(\lambda K_{(q-1)k,k+r})|$ are multiples of k. Thus, $\lambda(k + r)(k + r - 1)/2 \equiv 0[k]$ from the assumption that $\lambda n(n-1)/2$ is divisible by k. $\lambda K_{(q-1)k}$ and λK_{k+r} are C_k-decomposable by Theorem 4.1.2 because λ is even, $(q-1)k \geq k$ and $k + r \geq k$ by assumption. Note that $\lambda K_{(q-1)k,k+r}$ can be decomposed into λ copies of $K_{(q-1)k,k+r}$. Since $K_{(q-1)k,k+r}$ is S_k-decomposable by Theorem 4.1.4, so is $\lambda K_{(q-1)k,k+r}$. Thus, λK_n is (C_k, S_k)-decomposable. □

Lemma 4.3.4 *Let n, k and $\lambda > 1$ be positive integers such that $gcd(\lambda, k) = 1$. For all $n \geq 2k$, if $\lambda n(n-1)/2 \equiv 0[k]$ then λK_n is (C_k, S_k)-decomposable.*

Proof. From the previous lemma, we only have to examine the case when λ is odd. We can decompose λK_n as an edge disjoint union of $(\lambda - 1)K_n$ and K_n. Since $gcd(\lambda, k) = 1$, then $|E(K_n)| \equiv 0[k]$. It is clear that $(\lambda - 1)K_n$ has a (C_k, S_k)-decomposition by Lemma 4.3.3. Now we decompose K_n into stars of k size by Theorem 4.1.1 since $n \geq 2k$. Thus λK_n is (C_k, S_k)-decomposable. □

Using Proposition 4.3.1 and Lemmas 4.3.2, 4.3.3 and 4.3.4, we obtain the following Theorem:

Theorem 4.3.5 *Let n, k and λ be positive integers. If $\lambda n(n-1)/2 \equiv 0[k]$ and*

- *$n \geq 4k$, or*
- *$n \geq 2k$ and $\lambda > 1$ is even or $gcd(\lambda, k) = 1$,*

then λK_n is (C_k, S_k)-decomposable.

4.4 Multidecomposition of λK_n when k is prime or divides either $n-1$,n or λ

One can easily check that λK_n is (C_2, S_2)-decomposable if and only if $n > 2$, $\lambda > 1$ and $\lambda n(n-1) \equiv 0[4]$. Thus, we admit the following lemma without proof.

Lemma 4.4.1 *Let $n > 2$ and $\lambda > 1$ be a positive integers. There exists a decomposition of λK_n into copies of S_2 and copies of C_2 if and only if $\lambda n(n-1)/2$ is even.*

In Lemmas 4.4.2-4.4.4, we will show the sufficient conditions of the decomposition of λK_n into C_k's and S_k's when $n = k+1$, $n = 2k+1$ and $n = 3k+1$, respectively with $k \geq 3$.

Lemma 4.4.2 *Let $n = k+1$, $\lambda > 1$ and $k \geq 3$ be positive integers. There exists a decomposition of λK_n into copies of S_k and C_k if and only if $\lambda k(k-1)/2 \equiv 0[k]$.*

Proof. We split the proof into two cases as follows:
Case 1. k is odd or λ is even
By assumption, $n = k+1$ and the degree of each vertex of λK_n is λk. We use one vertex in order to construct λ stars of k edges. The remaining graph is λK_{n-1}. Since k is odd or λ is even and we have $n-1 = k$, then $\lambda(n-2) = \lambda(k-1)$ is always even and $\lambda k(k-1)/2 \equiv 0[k]$, so by Theorem 4.1.2 λK_{n-1} is decomposable into C_k-decomposable. Thus, λK_n is (S_k, C_k)-decomposable.
Case 2. k is even and λ is odd
This subcase does not exist because by assumption $\lambda k(k-1)/2 \equiv 0[k]$, implies $\lambda(k-1)$ to be even, a contradiction. The opposite implication is clear to proof. □

In the two following lemmas we will show that when $n = 2k+1$ or $n = 3k+1$, the complete multigraph λK_n can be decomposed into some k-cycles and k-stars.

Lemma 4.4.3 *Let $n = 2k+1$ and $\lambda > 1$ be positive integers and let k be a positive integer, $k \geq 3$. There exists a decomposition of λK_n into copies of S_k and C_k for any k.*

Proof. The number of edges in λK_{2k+1}, $\lambda k(2k+1)$, is a multiple of k. We decompose λK_{2k+1} as follows : $\lambda K_{2k+1} = (\lambda-1)K_{2k+1} \cup K_{2k+1}$. Clearly, $|E((\lambda-1)K_{2k+1})|$ and $|E(K_{2k+1})|$ are multiples of k. We decompose $(\lambda-1)K_{2k+1}$ into C_k's and K_{2k+1} into S_k's. Hence λK_{2k+1} is (S_k, C_k)-decomposable. □

Lemma 4.4.4 *Let $n = 3k+1$, $\lambda > 1$ and $k \geq 3$ be positive integers. There exists a decomposition of λK_n into copies of S_k and C_k if and only if $3\lambda k(3k-1)/2 \equiv 0[k]$.*

Proof. We split the proof into two cases as follows:

Case 1. λ is even

This case is solved by Lemma 4.3.3.

Case 2. λ is odd

If k is odd, note that: $\lambda K_{3k+1} = \lambda K_{2k+1} \cup \lambda K_k \cup \lambda K_{2k+1,k}$. By Lemma 4.4.3, λK_{2k+1} is (S_k, C_k)-decomposable. λK_k can be decomposed into C_k's and $\lambda K_{2k+1,k}$ is S_k-decomposable.

If k is even, $3\lambda(3k+1)$ is not even so this case can't exist. The opposite implication is clear to proof. □

In the following proposition, we prove that for any k that divides n or $n-1$, λK_n is (S_k, C_k)-decomposable.

Proposition 4.4.5 *For integers k and n and λ with $\lambda > 1$ and $2 \leq k \leq n+1$, if $n \equiv 0, 1[k]$ and $\lambda n(n-1)/2 \equiv 0[k]$, then λK_n is (S_k, C_k)-decomposable.*

Proof. For the case when $n = k+1$, $n = 2k+1$ and $n = 3k+1$: see Lemmas 4.4.1, 4.4.2, 4.4.3 and 4.4.4, respectively.

By Theorem 4.3.5, if $n = \alpha k + 1$ or $n = \alpha k$ with $\alpha \geq 4$, then λK_n is (S_k, C_k)-decomposable.

To complete the proof, we study the cases when $n = 2k$ and $n = 3k$.

For $n = 2k$: When λ is even, λK_n is (S_k, C_k)-decomposable by Lemma 4.3.3. When λ is odd, observe that: $\lambda K_{2k} = (\lambda - 1)K_{2k} \cup K_{2k}$. $(\lambda - 1)K_{2k}$ is C_k-decomposable by Theorem 4.1.2 and K_{2k} is S_k-decomposable by Theorem 4.1.1.

For $n = 3k$: If λ is even, we have λK_n is (S_k, C_k)-decomposable by Lemma 4.3.3. If λ is odd and k is odd, then $\lambda K_{3k} = (\lambda-1)K_{3k} \cup K_{3k}$ since $|E(K_{3k})|$ and $|E((\lambda-1)K_{3k})|$ are multiples of k. Thus, $(\lambda-1)K_{3k}$ is C_k-decomposable by Theorem 4.1.2 and K_{3k} is S_k-decomposable by Theorem 4.1.1. On the other hand, if λ is odd and k is even, then it is sufficient to show that $\lambda 3k(3k-1) \equiv 0[2k]$ is not true in this case. So, when λ is odd, k must be also odd. □

In the following proposition, we will show the decomposition of λK_n into S_k's and C_k's when λ is a multiple of k:

Proposition 4.4.6 *For integers k, n with $2 \leq k \leq n-1$, if $\lambda \equiv 0[k]$, then λK_n is (S_k, C_k)-decomposable.*

Proof. Since $n \geq k+1$ we distinguish two cases:

Case 1. $n \geq k+2$

$\lambda \equiv 0[k]$ implies that the degree of each vertex of λK_n is multiple of k. Thus, we can construct stars S_k using each vertex of the multigraph. We first decompose incident edges of some vertex into S_k's in a circular manner as illustrated by Example 4.4.7. This process is repeated until the remaining graph is a λK_m where $m = k+1$ if k is even and $m = k$ if k is odd.

If k is odd, the remaining graph is λK_k and $\lambda k(k-1)/2 \equiv 0[k]$, this implies that λK_k can be decomposed into cycles of size k by Theorem 4.1.2. If k is even, the remaining graph is λK_{k+1} which has $\lambda k(k+1)/2$ edges, thus number of edges is divisible by k. Since λk is even, the graph λK_{k+1} can be decomposed into cycles of k size by Theorem 4.1.2. Hence, λK_n is (S_k, C_k)-decomposable.

Case 2. $n = k+1$

Since the degree of each vertex is λk, we decompose the incident edges of one vertex into λ copies of S_k. The remaining graph is λK_k. By assumption, λK_k has number of edges divisible by k, this implies that $\lambda k(k-1)/2 \equiv 0[k]$. Since $\lambda(k-1)$ is even, we decompose λK_k into copies of C_k using Theorem 4.1.2. □

Example 4.4.7, illustrated by Figure 4.1, shows how the proposition 4.4.6 is applied for a graph $3K_5$:

Example 4.4.7 *(S_3, C_3)-decomposition of a graph λK_n with $n = 5$ and $\lambda = 3$ is as follows:*

- *Considering the graph* $3K_5$*, since* $\lambda \equiv 0[3]$ *then* $|E(3K_5)| \equiv 0[k]$.
- *Taking on a vertex of* $3K_5$ *called* v*, we decompose the graph into* $\lambda(n-1)/k = 4$ *stars by rotation. This rotation is applied on all the incident edges of the considered node* v *(Figure 4.1 illustrate rotation construction).*
- *The remaining graph is* $3K_4$*. The same rotation construction is applied for finding* $\lambda(n-2)/k = 3$ *stars. This rotation construction is applied until the remaining graph is* $3K_k(k = 3)$.
- *The remaining graph* $3K_3$ *can be decomposed into 3 copies of* C_3.

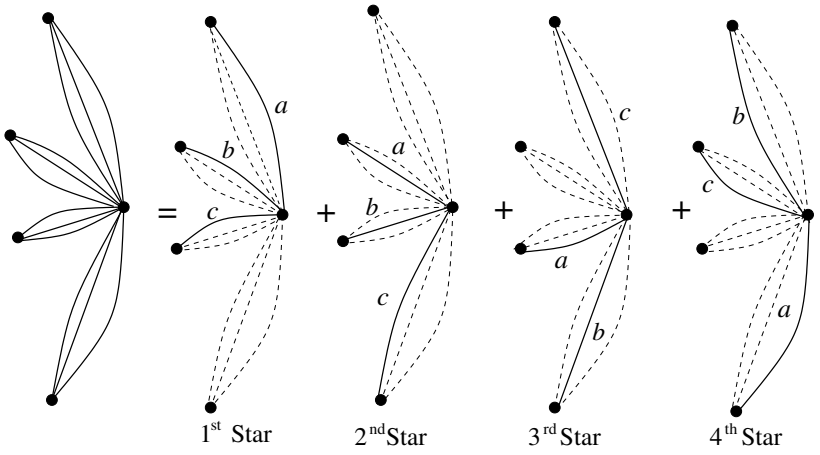


Figure 4.1: Rotation construction of stars. The edges of a star are labeled by a, b and c.

In the following corollary, we will investigate the problem of decomposing λK_n into S_k's and C_k's for each prime number k.

Corollary 4.4.8 *Let* n *and* $\lambda > 1$ *be positive integers and let* k *be a positive prime number. There exists a* (C_k, S_k)*-decomposition of* λK_n *if and only if* $n \geq k+1$ *and* $\lambda n(n-1)/2 \equiv 0[k]$.

Proof. We show that the necessary conditions given by Lemma 4.2.1 are also sufficient. $\lambda n(n-1)/2$ is a multiple of k and k is a prime number, so we distinguish three cases according to the multiplicity of n, $n-1$ and λ.
When k divides n or k divides $n-1$, this case is proved in Proposition 4.4.5. When k divides λ, this case is proved in Proposition 4.4.6. □

The following theorem is a direct consequence of Propositions 4.4.5 and 4.4.6 and Corollary 4.4.8:

Theorem 4.4.9 *Let n, k and $\lambda > 1$ be positive integers. λK_n is (S_k, C_k)-decomposable if $\lambda n(n-1)/2 \equiv 0[k]$ and :*

- *k is prime, or*
- *k divides either $n-1$, n or λ.*

4.5 Discussion on multidecomposition of λK_n when $n < 2k$

In this section, we focus on the multidecomposition of λK_n when the number of vertices $n < 2k$. We give a possible idea to deal with this case.

In Theorem 4.4.9, we treated the cases when k is prime and when k divides either $n-1$, n or λ. Now, we focus on the remaining cases. In the following example we will show that if we fix k, the number of possibilities, not treated below, of n is bounded.

(1) $k = 4$, this means that $\frac{\lambda n(n-1)}{2}$ needs to be a multiple of 4. Thus, there is no remaining cases since all the possible cases are treated above.

With the same reasoning, we obtain the remaining values of n as follow:

(2) $k = 6$, the possible values of n are $8, 9, 10$.

(3) $k = 8$, the possible values of n are $12, 13$.

(4) $k = 9$, the possible values of n are $12, 13, 15, 16$.

(5) $k = 10$, the possible values of n are $12, 13, 15, 16, 17$.

(6) $k = 11$, there are no remaining cases.

And so on.

As the number of vertices n of the complete multigraph λK_n is pretty close to k, we prove in the following that the degree of the graph λK_n have an impact on how we decompose.

1. If k is even and $\lambda(n-1)$ is even, then if such a decomposition exists, the number of stars S_k needs to be even.

2. If k is even and $\lambda(n-1)$ is odd, then if such a decomposition exists, the number of stars S_k needs to be odd.

3. If k is odd and $\lambda(n-1)$ is even, then if such a decomposition exists, the number of stars S_k needs to be even.

4. If k is odd and $\lambda(n-1)$ is odd, then if such a decomposition exists, $n-k$ needs to be odd.

These four cases are true for $n < 2k$. For example, in the first case, each star has an even number of edges k and the degree of its central node is even. Since to decompose a graph into cycles without any remaining edges, the degree of each vertex of the graph needs to be even. Thus, if we begin by decompose the graph into stars, we must verify that in the remaining graph all vertices have an even degree. Since $n < 2k$, the number of stars must be even.

This means that if we give a specific placement for each star in the complete multigraph, this helps us to define a method to decompose the remaining edges into cycles.

4.6 Conclusion

The contribution of the first part is the study of the *Multidecomposition of complete multigraph* into cycles and stars. Chapter 3 presented an introduction to particular graph decomposition problems and gave a brief survey of the famous decomposition problems studied in the literature.

Abueida and Lian [AL14] gave necessary and sufficient conditions for decomposing K_n into cycles and stars of k edges, for $n \geq 4k$ and k even or n odd. Chapter 4 improved results on this decomposition and extended it for the complete multigraph λK_n. Thus, we presented necessary and sufficient conditions for different cases as follows:

- k is prime,
- k divides either $n-1$, n or λ,
- $n \geq 2k$ and λ is even or $gcd(\lambda, k) = 1$,

- $n \geq 4k$, independently of the parity of n or k, thus improving result of Abueida and Lian [AL14].

Part II

Domination in Graphs

Chapter 5

Overview of some domination sets problems

Contents

This part of thesis is devoted to the domination problems in graphs. The literature contains extensive studies of many different types of domination in graphs. In our thesis, we consider a variant of the domination set problem, called $[i, j]$-domination set problem. The motivation for studying this variant of domination problem is rich and varied from an application perspective. In this chapter, a review of some well known results about variants of domination sets problems is given. We also provide a brief history and some motivations to investigate the domination graph problems. We next focus on $[i, j]$-domination and $[i, j]$-total domination problems. These two problems are discussed in details. Afterwards, we give a particular interest on the $[1, 2]$-domination problem and we discuss some results related to this variant. Since, in the next chapter, we consider the $[1, 2]$-dominating set problem

restricted to a particular class of graphs, namely generalized Petersen graphs, then we provide some basic definitions and properties about these graphs, followed by a discussion of different types of domination problems studied on it.

5.1 History and motivation

The Queens problem can be considered as the origin of the study of dominating set in graphs. In 1850, the chess fans in Europe considered the problem of determining the minimum number of queens that can be placed on a chessboard such that all squares are either occupied by queens or attacked (or dominated) by at least one queen. Then, the number of required queens for such problem corresponds to the dominating number. Therefore, the problem of chess queen placement can be identified more generally as a problem of dominating the vertices of a graph [HHS98]. Formally, a dominating set for a graph $G = (V, E)$ is a subset S of V such that every vertex of V is either in S or has a neighbor in S. A dominating set S is a minimal dominating set if no proper subset $S' \subset S$ is a dominating set. The domination number $\gamma(G)$ of a graph G is the minimum cardinality of a dominating set of G. We call such a set a γ-set of G. Determining the minimum domination number is proved to be NP-hard [HHS98]. Therefore, some theorems about the minimal dominating sets in graphs where given by Ore [OO62] in 1962.

The Dominating set is very important class of problem with several theoretical and practical applications. This problem has attracted many theoretical researches, therefore many results have been proposed and different variants have been identified by the graph community.

Excellent surveys on graph domination are provided by Haynes *and al.* [HHS98, HHS97], Cockayne [Coc78] and recently by Henning [Hen09] for total domination.

In practical side, the dominating sets gave a special interest in computer system field due to their importance for several applications. The structure of dominating set can be useful as overlays in computer networks. These structures are usually used for designing efficient protocols in wireless sensor and ad-hoc networks [GHJ+08, YKR06, BDTC05, KMW04]. For example, the Dominating set are used for clustering approaches in wireless sensor networks (WSNs) for load balancing and extending the network lifetime [YKR06]. It is used also for optimization and

designing protocols in social networks and many other fields. Readers can refer to Haynes *et al.*'s book [HHS98] for more descriptions on some variants of domination in graphs and their applications in several areas.

For all these reasons, different variants of dominating set have been identified by the graph community, such as total dominating set [CDH80], double dominating set [HH00], restrained domination [DHHM00] and so on.

In this thesis, we give a particular interest on the $[i, j]$-dominating set. The motivation of studying this type of domination is the fact that this type of domination has relation with a lot of other domination problems e.g. perfect domination.

5.2 Some variants of dominating sets problems

In this section, we present some variants of dominating sets problems before presenting our problem.

5.2.1 Total dominating set

The total domination is an important parameter as it ensures connectivity in the network even after failure of few of the communication points. Formally, a set $S \subseteq V(G)$ is a total dominating set of a graph $G = (V, E)$ if $N(S) = V$. It means that each vertex in V is adjacent to at least one vertex in S. The total domination number $\gamma_t(G)$ is the minimum cardinality of a total dominating set. Note that a dominating set S is a total dominating set if $G[S]$, the subgraph induced by S has no isolated vertices. Clearly, $\gamma(G) \leq \gamma_t(G)$.

5.2.2 Perfect dominating set

A dominating set S of a graph G is perfect if each vertex of $V(G) \setminus S$ is dominated by exactly one vertex in S. More advanced details can be found in [LS90].

5.2.3 Connected dominating set

A connected dominating set is a dominating set that induces a connected subgraph of the graph G. The connected domination number, denoted by $\gamma_c(G)$, is the minimum cardinality of a dominating set S such that $G[S]$ is connected.

In Figure 5.1, the red vertices present an example of a minimum connected dominating set in G which is also a total dominating set and perfect dominating set.

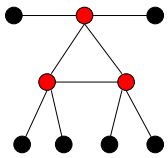


Figure 5.1: An example of equality in domination, total domination, connected domination and perfect domination.

5.2.4 Efficient dominating set

A dominating set S in G is an efficient dominating set for G if for every vertex v in V, there is exactly one u in S dominating v.

5.2.5 Independent dominating set

A dominating set S that is independent (forms an independent set) is an independent dominating set, and the minimum cardinality of an independent dominating set of G is the independent domination number $i(G)$.

5.2.6 k-tuple dominating set

Introduced by Harary and Haynes in [HH00]. For a fixed positive integer k, a k-tuple dominating set (resp. k-tuple total dominating set) of a graph $G = (V, E)$ is a subset S of V such that $|N[v] \cap S| \geq k$ (resp. $|N(v) \cap S| \geq k$) for every vertex $v \in V$. The k-tuple domination number $\gamma_{\times k}(G)$ (resp. k-tuple total domination number $\gamma_{\times k,t}(G)$) is the minimum cardinality of a k-tuple dominating set (resp. k-tuple total dominating set) of G. A dominating set of G is a 1-tuple dominating set of G. A total dominating set of G is a 1-tuple total dominating set of G. A double dominating set of G is a 2-tuple dominating set of G.

5.3 $[i, j]$-domination sets in graphs

In this section we provide definitions, some of the fundamental theorems and results founded about the $[i, j]$-domination sets.

Definition 1 *($[i, j]$-domination set [CHHM13]) A subset $S \subseteq V$ in a graph $G = (V, E)$ is a $[i, j]$-dominating set if, for every vertex $v \in V \setminus S$, $i \leq |N(v) \cap S| \leq j$, that is, every vertex in $V \setminus S$ is adjacent to at least i vertices, but not more than j vertices in S. The $[i, j]$-domination number in G, denoted $\gamma_{[i,j]}(G)$, equals the minimum cardinality of a $[i, j]$-dominating set in G. A $[i, j]$-dominating set with cardinality $\gamma_{[i,j]}(G)$ is called a $\gamma_{[i,j]}(G)$-set.*

Note that, for $i \geq 1$, a $[i, j]$-dominating set S is a dominating set, since every vertex in $V\S$ has at least one neighbor in S.

Definition 2 *($[i, j]$-total domination set [CHHM13]) A subset $S \subseteq V$ in a graph $G = (V, E)$ is a $[i, j]$-total dominating set if, for every vertex $v \in V$, $i \leq |N(v) \cap S| \leq j$, that is, every vertex in V is adjacent to at least i vertices, but not more than j vertices in S. The $[i, j]$-total domination number in G, denoted $\gamma_{t[i,j]}(G)$, equals the minimum cardinality of a $[i, j]$-total dominating set in G. A $[i, j]$-total dominating set with cardinality $\gamma_{t[i,j]}(G)$ is called a $\gamma_{t[i,j]}(G)$-set.*

It was mentioned in [CHHM13] that $[i, j]$-dominating sets are related to different types of domination. For example, $[1, 1]$-dominating set represents the perfect dominating set and the $[1, 1]$-total dominating set represents the efficient dominating set.

Some papers gave a particular interest to study the $[1, 2]$-dominating set problem. In [CHHM13], the authors proved that $\gamma(G) = \gamma_{[1,2]}(G)$ for some classes of graphs, as in:

Theorem 5.3.1 *[CHHM13] If G is a P_4-free graph or claw-free graphs, then $\gamma(G) = \gamma_{[1,2]}(G)$.*

The authors also showed that there are some graph classes for which $\gamma_{[1,2]}(G)$ is strictly less than n. Then, they proved the following theorem:

Theorem 5.3.2 *[CHHM13] If G is a non-trivial k-regular graph for any $k \leq 4$, then $\gamma_{[1,2]}(G) < n$.*

They have also studied the $[1,3]$-dominating set for grid graphs and proved that $\gamma(G) = \gamma_{[1,3]}(G)$.

From complexity point of view, they proved that $[1,2]$-dominating set is NP-complete for bipartite graphs.

They also observed that for a tree of order n and k leaves, $\gamma_{[1,2]}(G) \leq n-k$, and asked for a characterization of the trees for which the equality holds. In [YW14], the authors answered partially this question in the following theorem:

Theorem 5.3.3 *[YW14] Let T be a tree of order n with k leaves such that every non-leaf vertex has degree at least 4. Then $\gamma_{[1,2]}(T) = n - k$.*

They also give some characterization to solve some open questions posed in [CHHM13].

In the same perspective, in [GHM16], the authors gave a particular interest to study the $[1,2]$-dominating set and $[1,2]$-total dominating set problems in trees by giving a linear algorithm for finding $\gamma_{[1,2]}(T)$ and $\gamma_{t[1,2]}(T)$.

In [BDGP16], the authors studied a more general problem: $[1,j]$-dominating sets in graphs. They proved that the associated decision problem is NP-complete for chordal graphs and they proposed a linear time algorithm for finding $\gamma_{[1,j]}(G)$ for a tree and a polynomial time algorithm for finding $\gamma_{[1,j]}(G)$ for a fixed j in a split graph.

5.4 Domination in generalized Petersen graph

In this section, we review some dominating sets problems studied on generalized Petersen graphs. For many classes of graphs the exact values of $\gamma(G)$ are known, e.g., $\gamma(P_n) = \gamma(C_n) = \lceil \frac{n}{3} \rceil$. For the class of generalized Petersen graphs $P(n,2)$ introduced by Watkins [Wat69] it was conjectured by Behzad *et al.* that $\gamma(P(n,2)) = \lceil \frac{3n}{5} \rceil$ holds [BBP08]. This conjecture was later independently verified by several researchers [EJM09, FYJ09, YKX09, LZ14]. In particular, Behzad et al. (2008) and Yan et al. (2009) determined the domination number of the generalized Petersen graph $P(n,k)$ with $n = 2k+1$, the exact domination number is $\lceil \frac{3n}{5} \rceil$. Then, Liu et al. (2014) determined the exact domination number of $P(n,k)$ with $n = 2k$ and $n = 2k+2$. Fu et al. (2009) proved that the domination number of $P(n,k)$ with $k = 2$ is $n - \lfloor \frac{n}{5} \rfloor - \lfloor \frac{n+2}{5} \rfloor$.

Others variants of dominations are studied in literature and then some of these domination numbers are also known for generalized Petersen graphs. Cao *et al.* computed the total domination number of $P(n,2)$ as $\gamma_t(P(n,2)) = 2\lceil \frac{n}{3} \rceil$ [JWM09]. Zelinka studied three numerical invariants of graph domination namely the domatic number[2] [CH77b], total domatic number [CDH80] and k-ply domatic number [Zel84] with $k = 2$ and $k = 3$ in generalized Petersen graphs. He gave exact values for some specific cases [Zel02]. Fu et al. studied Roman domination in generalized Petersen graphs [XYB09]. In [BŠ07], Bresar and Sumenjak studied the 2-rainbow domination number[3], denoted by γ_{r2}, in generalized Petersen graph and showed that $\lceil \frac{4n}{5} \rceil \leq \gamma_{r2}(P(n,k)) \leq n$ for any $P(n,k)$, where n and k are relatively prime numbers. Shortly after, Tong et al. investigated the problem in $P(n,k)$ with $k = 2$ [TLYL09] and in the same period Xu studied the problem in $P(n,k)$ with $k = 3$ [Xu09]. In [LXS13], Li et al. considered the problem of signed total domination in $P(n,2)$. Further results about other types of domination in this type of graphs can be found in [BF11, XK11, SLY+14, K+10].

[2] A domatic partition of a graph $G = (V,E)$ is a partition of V into disjoint sets $V_1, V_2, ..., V_K$ such that each V_i is a dominating set for G. The domatic number is the maximum size of a domatic partition.

[3] A k-rainbow dominating function of a graph G is a function f from $V(G)$ to the set of all subsets of $\{1,2,\ldots,k\}$ such that for any vertex v with $f(v) = \emptyset$ we have $\cup_{u \in N_G(v)} f(u) = \{1,2,\ldots,k\}$

Chapter 6

$[1,2]$-domination in general Petersen graphs

Contents

This chapter considers $[1,2]$-domination and $[1,2]$-total domination, a concept introduced by Chellali et al. [CHHM13]. In this sense, we study two numerical invariants of graph domination, namely the $[1,2]$-domination number $\gamma_{[1,2]}(G)$ and $[1,2]$-total domination number $\gamma_{t[1,2]}(G)$ of a graph G. We investigate them for generalized Petersen graphs $P(n,k)$ for $k=2$. Obviously $\gamma_{[1,2]}(P(n,1)) = \gamma(P(n,1))$.

6.1 Notations

In this section we introduce some definitions and notions that we use throughout this chapter. Let recall the formal definition of generalized Petersen graphs.

Definition 3 *Let $n,k \in \mathbb{N}$ with $k < n/2$. The* generalized Petersen graph $P(n,k)$ *is the undirected graph with vertices $\{u_0,\dots,u_{n-1}\} \cup \{v_0,\dots,v_{n-1}\}$ and edges $\{\{u_i,u_{i+1}\},\{u_i,v_i\},\{v_i,v_{i+k}\} \mid 0 \le i < n\}$.*

In this work indices are always interpreted modulo n, e.g. $v_{n+i} = v_i$. Fig. 1 shows the graphs $P(5,2)$ and $P(6,2)$, vertices depicted in black form a [1,2]-dominating

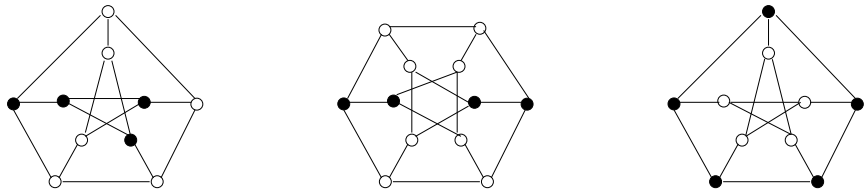


Figure 6.1: The minimum $[1,2]$-domination sets of the generalized Petersen graphs $P(5,2)$ and $P(6,2)$ and the minimum $[1,2]$-total dominating set for $P(5,2)$.

set of minimum size, i.e., $\gamma_{[1,2]}(P(5,2)) = \gamma_{[1,2]}(P(6,2)) = 4$ and also for the graph $P(5,2)$, vertices depicted in black form a [1,2]-total dominating set of minimum size $\gamma_{t[1,2]}(P(5,2)) = 5$.

The proofs of this chapter use the following notion of a *block*.

Definition 4 *A* block *b of $P(n,2)$ is the subgraph induced by the six vertices $\{v_{i-1}, v_i, v_{i+1}, u_{i-1}, u_i, u_{i-1}\}$ for any $i \in \{0, \dots n-1\}$. A block is called* positive *if two of the indices of $\{v_{i-1}, v_i, v_{i+1}\}$ are odd, otherwise it is called* negative.

Fig. 6.1 shows three disjoint blocks of $P(n,2)$. The second block is *positive* while the other two are *negative*. Note that blocks can overlap. If b is a block, the block to the left is denoted by b^- and that to the right by b^+.

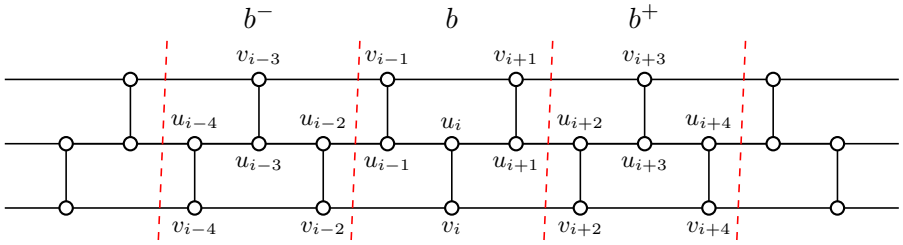


Figure 6.2: Partition of $P(n,2)$ into blocks.

Let S be a $[1,2]$-dominating set. For a subset $U \subseteq V$ denote by $\gamma_S(U)$ the number of vertices of S that are in U, i.e., $\gamma_S(U) = |U \cap S|$. For $i \geq 0$ let $\mathcal{B}_i(S)$ be the set of all blocks b with $\gamma_S(b) = i$.

Note that $\mathcal{B}_0(S) = \emptyset$ for any dominating set S of $P(n,2)$.

6.2 $[1,2]$-dominating set of $P(n,2)$

In this section we analyze the $[1,2]$-domination number of the generalized Petersen graphs $P(n,2)$ and prove the following theorem.

Theorem 6.2.1 $\gamma_{[1,2]}(P(n,2)) = \begin{cases} 2n/3 & \text{if } n \equiv 0,3[6] \\ 2\lfloor n/3 \rfloor + 1 & \text{if } n \equiv 1[6] \\ 2\lfloor n/3 \rfloor + 2 & \text{otherwise.} \end{cases}$ *for* $n \geq 5$.

Denote by $f(n)$ the value of the right side of the equation in Theorem 6.2.1.

$$f(n) = \begin{cases} 2n/3 & \text{if } n \equiv 0,3[6] \\ 2\lfloor n/3 \rfloor + 1 & \text{if } n \equiv 1[6] \\ 2\lfloor n/3 \rfloor + 2 & \text{otherwise.} \end{cases} \quad \text{for } n \geq 5.$$

The correctness of Theorem 6.2.1 for $n < 12$ can be verified manually.

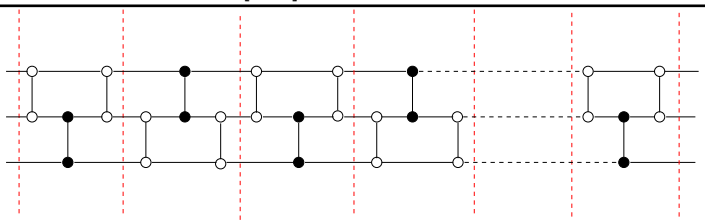
Lemma 6.2.2 $\gamma_{[1,2]}(P(n,2)) = f(n)$ *for* $5 \leq n < 12$.

Proof. By inspection, we can easily see that the following sets S_n are minimum $[1,2]$-dominating sets of $P(n,2)$. $S_5 = \{u_1, v_1, v_3, v_4\}$, $S_6 = \{u_1, v_1, u_4, v_4\}$, $S_7 = \{u_0, v_1, v_2, v_3, u_4\}$, $S_8 = \{u_1, v_1, u_4, v_4, v_6, v_7\}$, $S_9 = \{u_1, v_1, u_4, v_4, u_7, v_7\}$, $S_{10} = \{u_1, v_1, u_4, v_4, u_7, v_7, u_8, v_8\}$ and $S_{11} = \{u_1, v_1, u_4, v_4, u_7, v_7, v_9, v_{10}\}$. □

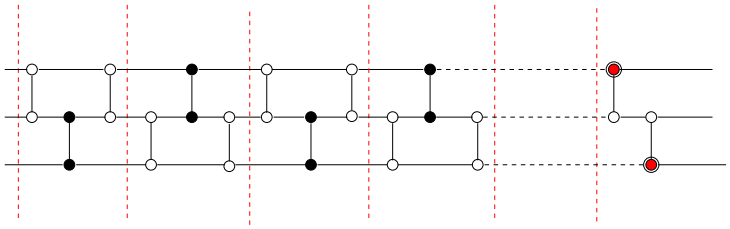
Lemma 6.2.3 $\gamma_{[1,2]}(P(n,2)) \leq f(n)$ *for* $n \geq 5$.

Proof. To prove that $f(n)$ is an upper bound of $\gamma_{[1,2]}(P(n,2))$, we give in Fig. 6.3 the corresponding construction for each case. For $n \equiv 0,3[6]$, we choose the middle pair of nodes of each block. For the cases $n \equiv 2,4,5[6]$, we do the same as the previous case by choosing the middle pair of nodes of each block. Then, we add two dominating nodes as depicted in red in Fig. 6.3. For the case $n \equiv 1[6]$, we choose two nodes from each block as shown in Fig. 6.3 except in the two successive blocks preceding the block with only two nodes. In these two blocks we choose five nodes as depicted in Fig. 6.3. This means that we have $2n/3$ nodes plus one additional dominating node. □

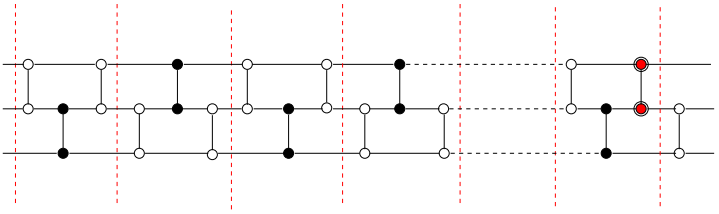
Thus, it suffices to prove that $f(n)$ is a lower bound. Assume that there exists a minimal $[1,2]$-dominating set S of $P(n,2)$ with $|S| < f(n)$. Lemma 6.2.2 yields $n \geq 12$. The remaining proof is split into two parts depending on whether $\mathcal{B}_1(S)$ is empty or not. The case when $\mathcal{B}_1(S)$ is empty is proved in Section 6.2.1 and the case when $\mathcal{B}_1(S)$ is not empty is proved in Section 6.2.2.



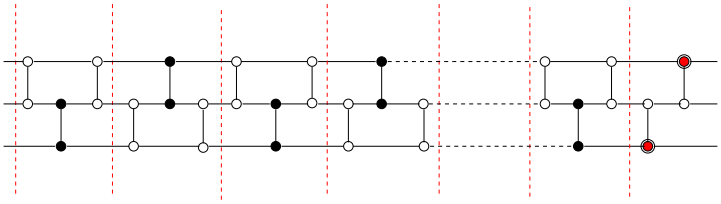
(a) Case $n \equiv 0, 3[6]$



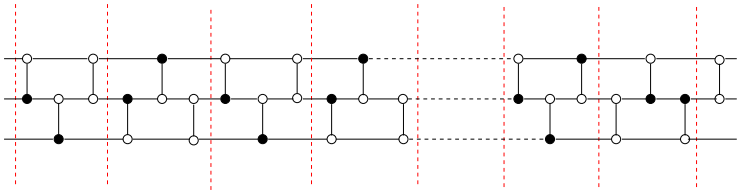
(b) Case $n \equiv 2[6]$



(c) Case $n \equiv 4[6]$



(d) Case $n \equiv 5[6]$



(e) Case $n \equiv 1[6]$

Figure 6.3: $f(n)$ is an upper bound of $\gamma_{[1,2]}(P(n,2))$ for all $n > 12$

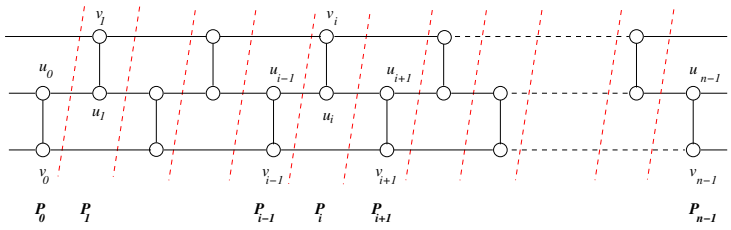


Figure 6.4: The partition of $P(n,2)$ into n pairs.

6.2.1 Case when $\mathcal{B}_1(S)$ is empty

The vertices of $P(n,2)$ are grouped into n pairs of vertices $p_i = \{v_i, u_i\}$ as depicted in Fig. 6.4. Since $\mathcal{B}_1(S) = \emptyset$ this means that for $i = 1, ..., n$

$$\gamma_S(p_i) + \gamma_S(p_{i+1}) + \gamma_S(p_{i+2}) \geq 2$$

(subscripts are always taken modulo n). Note that $\gamma_S(p_i) \leq 2$ for all i. Consider the following system of inequalities for integer valued variables $x_0, \ldots, x_{n-1}$.

$$\begin{aligned} x_i &\leq 2 \\ x_i + x_{i+1} + x_{i+2} &\geq 2 \\ \textstyle\sum_{i=0}^{n-1} x_i &< f(n) \end{aligned} \tag{6.1}$$

Note that $x_i = \gamma_S(p_i)$ is a solution for these equations. We will show that no solution of Eq. (6.1) is induced by a $[1,2]$-dominating set.

Lemma 6.2.4 *Let x be a solution of Eq. (6.1) with $x_i = 2$ for some i. Let $\hat{x} = x$ except $\hat{x}_{i+1} = \hat{x}_{i+2} = 0$ and $\hat{x}_{i+3} = 2$. Then $\hat{x}$ is a solution of Eq. (6.1) with $\sum_{i=0}^{n-1} \hat{x}_i \leq \sum_{i=0}^{n-1} x_i$.*

Proof. Obviously $\hat{x}$ satisfies the first two sets of inequalities. Note that $x_{i+1} + x_{i+2} + x_{i+3} \geq 2$ since x is a solution of Eq. (6.1). Thus, $\hat{x}_{i+1} + \hat{x}_{i+2} + \hat{x}_{i+3} \leq x_{i+1} + x_{i+2} + x_{i+3}$. □

Lemma 6.2.5 *Let x be any solution of Eq. (6.1). Then $x_i \leq 1$ for $i = 0, \ldots, n-1$.*

Proof. Let x be any solution of Eq. (6.1) such that $x_i = 2$ for some i. Without loss of generality $i = 0$. By Lemma 6.2.4 there exist a solution which coincides with x except $x_1 = x_2 = 0$ and $x_3 = 2$. Repeatedly applying Lemma 6.2.4 proves that there exits a solution $\hat{x}$ of Eq. (6.1) with $\hat{x}_k = 2$ and $\hat{x}_{k+1} = \hat{x}_{k+2} = 0$ for

$k = 0, 1, \ldots, \lfloor n/3 \rfloor$. If $n \equiv 0\,[3]$ then $\sum_{i=0}^{n-1} \hat{x}_i = 2n/3 = f(n)$, which is impossible. Suppose $n \equiv 1\,[3]$. Then $\hat{x}_{n-1} = 2$ otherwise the second constraint for $i = n-2$ would be violated. This leads to the contradiction $\sum_{i=0}^{n-1} \hat{x}_i = 2\lfloor n/3 \rfloor + 2 \geq f(n)$. Hence, $n \equiv 2\,[3]$. Then $\hat{x}_{n-2} = 2$ otherwise the second constraint for $i = n-2$ is not satisfied. Again this leads to the contradiction $\sum_{i=0}^{n-1} \hat{x}_i = 2\lfloor n/3 \rfloor + 2 = f(n)$. This proves $x_i \leq 1$ for all i. □

Lemma 6.2.6 *If $n \not\equiv 4\,[6]$ then Eq. (6.1) has no solution. If $n \equiv 4\,[6]$ then any solution of Eq. (6.1) is a rotation of the solution $(1,1,0,1,1,0,\ldots,1,1,0,1)$.*

Proof. Let x be any solution of Eq. (6.1). By Lemma 6.2.5 $x_i \leq 1$ for $i = 0, \ldots, n-1$. Denote by n_0 the number of variables with $x_i = 0$. Thus $\sum_{i=0}^{n-1} x_i = n - n_0$. Note that if $x_i = 0$ then either $x_{i+1} = 1$ or $x_{i-1} = 1$, thus no adjacent variables have both value 0. Denote by $l_1, \ldots, l_{n_0}$ the lengths of maximal sequences of consecutive x_i with $x_i = 1$. Note that $l_j \geq 2$ for all j. Then

$$\sum_{i=0}^{n-1} x_i = \sum_{j=1}^{n_0} l_j = 2n_0 + \sum_{j=1}^{n_0} (l_j - 2).$$

This implies

$$3\sum_{i=0}^{n-1} x_i = 2n + \sum_{j=1}^{n_0} (l_j - 2).$$

If $n \equiv 0\,[3]$ then $\sum_{i=0}^{n-1} x_i \geq 2n/3 = f(n)$. A contradiction. If $n \equiv 2\,[3]$ then again this leads to the contradiction $\sum_{i=0}^{n-1} x_i = 2\lfloor n/3 \rfloor + (4 + \sum_{j=1}^{n_0}(l_j - 2))/3 \geq 2\lfloor n/3 \rfloor + 2 \geq f(n)$. Finally if $n \equiv 1\,[6]$ then $\sum_{i=0}^{n-1} x_i = 2\lfloor n/3 \rfloor + (2 + \sum_{j=1}^{n_0}(l_j - 2))/3 \geq 2\lfloor n/3 \rfloor + 1 = f(n)$. This contradiction proves that for $n \not\equiv 4\,[6]$ Eq. (6.1) has no solution.

Let $n \equiv 4\,[6]$. Then $\sum_{i=0}^{n-1} x_i = 2\lfloor n/3 \rfloor + (2 + \sum_{j=1}^{n_0}(l_j - 2))/3 < f(n) = 2\lfloor n/3 \rfloor + 1$ implies $3 = 2 + \sum_{j=1}^{n_0}(l_j - 2)$. This yields that there exists i such that $l_i = 3$ and $l_j = 2$ for all $j \neq i$. Thus, x is a rotation of the solution $(1,1,0,1,1,0,\ldots,1,1,0,1)$. □

Lemma 6.2.7 *The solution $x = (1,1,0,1,1,0,\ldots,1,1,0,1)$ is not induced by a $[1,2]$-dominating set of $P(n,2)$.*

Proof. Assume there exists a $[1,2]$-dominating set S such that $x_i = \gamma_S(b_i)$. Two vertices of the first two pairs must be in S. All four possibilities lead to a contradiction as shown in the following.

Case 1. $v_0, u_1 \in S$ (see Fig. 6.5). Since S is $[1,2]$-dominating the lower vertex of the last pair p_{n-1} must be in S. Now the same argument implies that the middle vertex of pair p_3 must be in S. This yields that the lower vertex of pair p_7 must be in S, otherwise the lower vertex of pair p_5 is not dominated. Repeating this argument shows that the lower vertex of pair p_{n-3} must be in S (note that $n \equiv 4\,[6]$). Thus, S does not dominate the middle vertex of pair p_{n-2}. Contradiction.

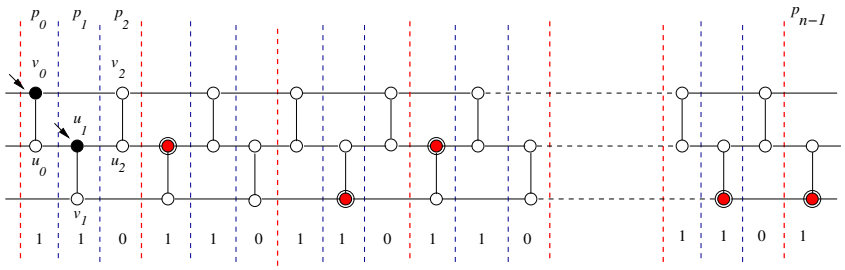


Figure 6.5: If $v_0, u_1 \in S$ then vertices depicted in red must also be in S.

Case 2. $u_0, v_1 \in S$ (see Fig. 6.6). In order to dominate the middle vertex of pair p_2 the middle vertex of p_3 must be in S. Similarly the lower vertex of pair p_7 must be in S to dominate the lower vertex of p_5. This results in the pattern shown in Fig. 6.6. This is impossible because all three neighbors of the lower vertex of p_{n-1} are in S.

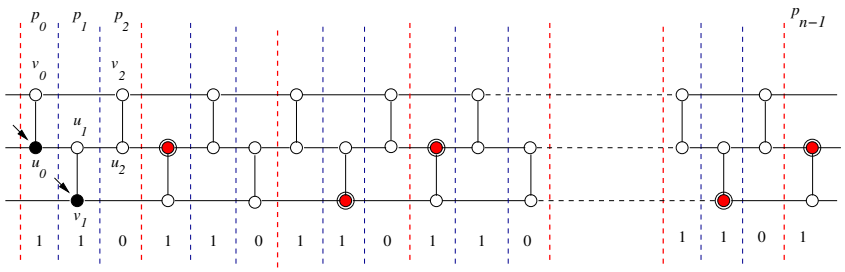


Figure 6.6: If $u_0, v_1 \in S$ then vertices depicted in red must also be in S.

Case 3. $u_0, u_1 \in S$. The same reasoning as above leads to the situation depicted in Fig. 6.7. This gives also rise to a contradiction since the upper vertex of pair p_{n-2} is not dominated.

Case 4. $v_0, v_1 \in S$. The same reasoning as above leads to the situation depicted in Fig. 6.8. This is impossible because all three neighbors of the lower vertex of p_{n-1} are in S.

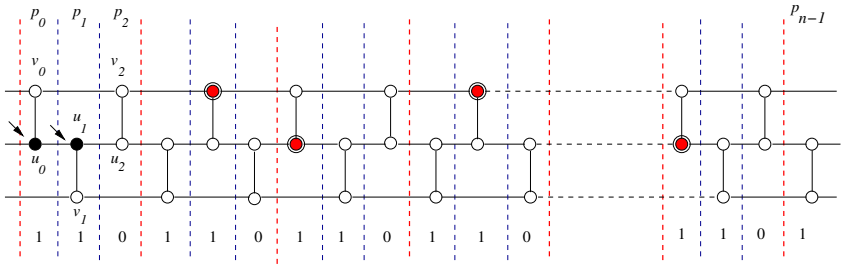


Figure 6.7: If $u_0, u_1 \in S$ then vertices depicted in red must also be in S.

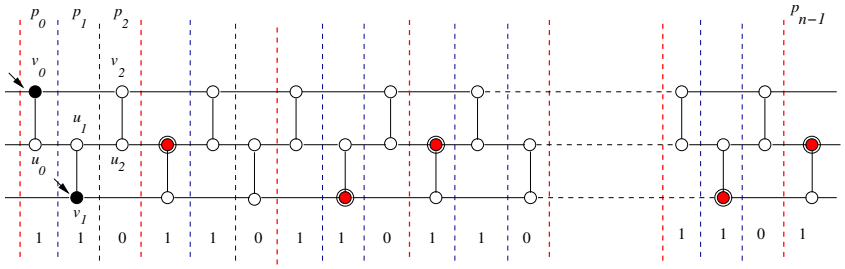


Figure 6.8: If $v_0, v_1 \in S$ then vertices depicted in red must also be in S.

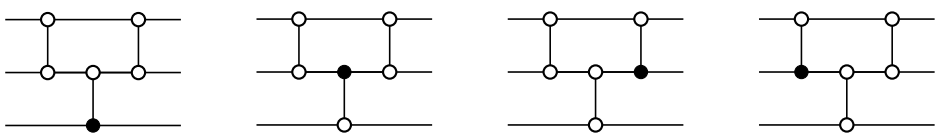
□

This concludes the proof of Theorem 6.2.1 for the case $\mathcal{B}_1(S) = \emptyset$.

6.2.2 Case when $\mathcal{B}_1(S)$ is not empty

The following simple observation is based on the fact that the central vertex of a block b can only be dominated by a vertex within b.

Lemma 6.2.8 *Any positive block $b \in \mathcal{B}_1(S)$ corresponds to one of the four blocks shown in Fig. 6.9. A similar result holds for negative blocks.*



(a) Type A (b) Type B (c) Type C (d) Type D

Figure 6.9: The four types of positive blocks with $\gamma_S(b) = 1$.

In the following the four different types of blocks are considered individually.

Lemma 6.2.9 *Let S be a $[1,2]$-dominating set of $P(n,2)$ containing a block b of type B and $n \geq 12$. Then there exists a $[1,2]$-dominating set S' of $P(n,2)$ not containing a block of type B such that $|S'| = |S|$.*

Proof. In order to dominate v_{i-1} and v_{i+1} from block b, vertices v_{i-3} from block b^- and v_{i+3} from b^+ need to be in S. The idea is to move some dominating nodes such that block b is not no longer of type B and no new block of type B emerges while S is still $[1,2]$-dominating and the cardinality of S remains. The proof is divided into four cases, considering whether u_{i-2} from block b^- and u_{i+2} from b^+ are in S or not. The notation of the nodes is taken from Fig. 6.1.

Case 1. $u_{i-2}, u_{i+2} \in S$. If v_{i-4} and v_{i+4} are not in S then $S' = S/\{u_i\} \cup \{v_i\}$. If v_{i-4} or v_{i+4} are in S then $S' = S/\{u_{i-2}\} \cup \{u_{i-1}\}$ or $S' = S/\{u_{i+2}\} \cup \{u_{i+1}\}$.

Case 2. $u_{i-2} \notin S$, $u_{i+2} \in S$. To dominate u_{i-2} and v_{i-2} we consider two subcases.

Subcase 2.1. $v_{i-2} \in S$. If u_{i-3} is not in S then $S' = S/\{u_i\} \cup \{u_{i-1}\}$. If $u_{i-3} \in S$ then there are three possibilities depending on which vertex dominates u_{i+4}. Hence, if $u_{i+3} \in S$ then $S' = S/\{u_{i+2}\} \cup \{v_i\}$. If $v_{i+4} \in S$ then $S' = S/\{u_{i+2}\} \cup \{u_{i+1}\}$. Otherwise, the vertex u_{i+4} is dominated by node u_{i+5} of block b^{++} then $S' = S/\{u_i\} \cup \{v_{i-1}\}$.

Subcase 2.2. $v_{i-2} \notin S$. This implies that u_{i-3} and v_{i-4} from are both in S. Then $S' = S/\{u_{i-3}\} \cup \{u_{i-1}\}$.

Case 3. $u_{i+2} \notin S$, $u_{i-2} \in S$. This case is symmetric to case 2.

Case.4. $u_{i+2}, u_{i-2} \notin S$. In order to dominate u_{i-2} and v_{i-2} two situations must be considered.

Subcase 4.1. $v_{i-2} \in S$. Since v_{i-2} is in S and v_i is not in S then v_{i+2} cannot be a dominating node. This yields that u_{i+3} and v_{i+4} are in S. Then $S' = S/\{u_{i+3}\} \cup \{u_{i+1}\}$.

Subcase 4.2. $v_{i-2} \notin S$. This implies $u_{i-3}, v_{i-4} \in S$. Therefore, $S' = S/\{u_{i-3}\} \cup \{u_{i-1}\}$. □

The next Lemma finally completes the proof of Theorem 6.2.1.

Lemma 6.2.10 *If $\mathcal{B}_1(S) \neq \emptyset$ and $n \geq 6$ then $|S| \geq f(n)$.*

Proof. Let n be minimal such that the lemma is false. Then $n \geq 12$ by Lemma 6.2.2. Let $\mathcal{S}_B$ the set of all $[1,2]$-dominating sets S of $P(n,2)$ not containing a block of type B and $|S| < f(n)$. Then $\mathcal{B}_1(S) \neq \emptyset$ for all $S \in \mathcal{S}_B$ by the first part of the proof. Let p be the largest number such that $|\mathcal{B}_1(S)| \geq p$ for each $S \in \mathcal{S}_B$. Then $p \geq 1$. Let $\mathcal{M}_p$ be the set of all $S \in \mathcal{S}_B$ with $|\mathcal{B}_1(S)| = p$.

Claim 1: $P(n,2)$ does not contain a block of type A for any $S \in \mathcal{M}_p$.
Assume false. Let $S \in \mathcal{M}_p$ and b a positive block of type A. Then the nodes v_{i+3} and u_{i+2} of b^+ must be dominating. Assume $\gamma_{b^+}(S) \geq 3$. Then $S' = S \setminus \{u_{i+2}\} \cup \{u_i\}$ is also a $[1,2]$-dominating set. Thus, $\gamma_b(S') = 2$. Then $|\mathcal{B}_1(S')| = |\mathcal{B}_1(S)| - 1 < p$ since $\gamma_b(S) = 1$. This yields $S' \notin \mathcal{S}_B$ and therefore $\mathcal{B}_1(S) = \emptyset$. Thus, $\gamma_{b^+}(S) = 2$.

Let b^{++} be the positive block to the right of b^+. Then the nodes u_{i+5} and v_{i+6} of b^{++} must be dominating. Next we remove the nodes of the blocks b and b^+ and connect the corresponding nodes of blocks b^- and b^{++}. The resulting graph is isomorphic to $P(n-6,2)$. Furthermore, $S' = S \setminus \{v_i, u_{i+2}, v_{i+3}, u_{i+5}\}$ is a $[1,2]$-dominating set of this graph. Thus, $|S'| = |S| - 4 \geq f(n-6)$ by the choice of n. Therefore $|S| \geq f(n-6) + 4 = f(n)$. This implies $|S| \geq f(n)$. This contradiction proves claim 1 for positive blocks of type A. The same argument shows that there are no negative blocks of type A.

Claim 2: $P(n,2)$ does not contain a block of type D for any $S \in \mathcal{M}_p$
Assume false. As above we only need to consider the positive case. Let $S \in \mathcal{M}_p$ and b a positive block of type D. Then nodes v_{i+3} and u_{i+2} of b^+ must be dominating. Assume $\gamma_{b^+}(S) = 2$. Then again the nodes u_{i+5} and v_{i+6} of block b^{++} must be dominating. We distinguish two cases. If v_{i-2} is not a dominating node then $S' = S \setminus \{v_{i+3}\} \cup \{v_{i+1}\}$ else (v_{i-2} is a dominating node) then we have again two subcases depending on $\gamma_{b^{++}}(S)$. If $\gamma_{b^{++}}(S) = 2$ then the nodes u_{i+8} and v_{i+9} of the block to the right of b^{++} must be dominating nodes. We remove the nodes of the blocks b and b^+ and connect the corresponding nodes of blocks b^- and b^{++} with $S' = S \setminus \{u_{i-1}, u_{i+2}, v_{i+3}, v_{i+6}\}$. Similar to the proof of claim 1 this leads to a contradiction. If $\gamma_c(S) \geq 3$ then at least one of the nodes v_{i+5} and u_{i+6} is a dominating node. Then we again remove the nodes of the blocks b and b^+ and connect the corresponding nodes of blocks b^- and b^{++} with $S' = S \setminus \{u_{i-1}, u_{i+2}, v_{i+3}, u_{i+5}\}$. Similar to the proof of claim 1 this leads to a contradiction.

Hence, $\gamma_{b^+}(S) \geq 3$. In the following we will construct a new $[1,2]$-dominating

set S' with $|\mathcal{B}_1(S')| < p$. This is a contradiction.

Case 1. $v_{i+2}, u_{i+3} \in S$. There are three subcases. If $v_{i-3} \notin S$ then $S' = S \setminus \{u_{i+2}\} \cup \{v_{i+1}\}$ and if $v_{i-2} \notin S$ then $S' = S \setminus \{u_{i+2}\} \cup \{u_i\}$. If $v_{i-3}, v_{i-2} \in S$ then $S' = S \setminus \{u_{i-1}, u_{i+2}\} \cup \{u_i, v_i\}$.

Case 2. Neither v_{i+2} nor u_{i+3} are in S. Since $\gamma_{b^+}(S) \geq 3$ this implies that v_{i+4} is a dominating node and $S' = S \setminus \{u_{i+2}\} \cup \{u_{i+1}\}$.

Case 3. If $v_{i+2} \in S$ and $u_{i+3} \notin S$ then $S' = S \setminus \{u_{i+2}\} \cup \{u_{i+1}\}$.

Case 4. If $v_{i+2} \notin S$ and $u_{i+3} \in S$ we distinguish two cases: If $v_{i+4} \in S$ then $S' = S \setminus \{u_{i+2}\} \cup \{u_{i+1}\}$ else we have four subcases depending on which node dominates u_{i+5}:

1. If $v_{i+5} \in S$ then $S' = S \setminus \{v_{i+3}\} \cup \{u_{i+1}\}$.

2. If $u_{i+5} \in S$ then $S' = S \setminus \{u_{i+3}\} \cup \{u_{i+1}\}$.

3. If $u_{i+6} \in S$ then we distinguish three cases depending on which node dominates v_{i+4}. If $u_{i+4} \in S$ then $S' = S \setminus \{u_{i+2}, u_{i+4}\} \cup \{v_{i+2}, u_i\}$. If $v_{i+4} \in S$ then $S' = S \setminus \{u_{i+2}, v_{i+4}\} \cup \{v_{i+2}, u_i\}$. Finally if $v_{i+6} \in S$ then we remove the nodes of the blocks b and b^+ and connect the corresponding nodes of blocks b^- and c.

4. If $u_{i+4} \in S$ then $S' = S \setminus \{u_{i+2}, u_{i+3}\} \cup \{u_{i+1}, v_{i+4}\}$.

This proves claim 2.

Claim 3: $P(n,2)$ does not contain a block of type C for any $S \in \mathcal{M}_p$

This case is symmetric to the second claim.

Claim 4: $P(n,2)$ does not contain a block of type B for any $S \in \mathcal{M}_p$

If S contains a block of type B then by Lemma 6.2.9 there exists $S' \in \mathcal{S}_B$ which does not contain a block of type B. The above claims yield $\mathcal{B}_1(S') = \emptyset$. This contradiction concludes the proof of the lemma. □

6.3 $[1,2]$-total dominating set of $P(n,2)$

In this section, we investigate the problem of $[1,2]$-total domination and prove the following result.

Theorem 6.3.1 $\gamma_{t[1,2]}(P(n,2)) = \begin{cases} 5 & \text{if } n = 5 \\ 2n/3 & \text{if } n \equiv 0,3[6] \quad \text{for } n \geq 6. \\ 2\lfloor n/3 \rfloor + 2 & \text{otherwise.} \end{cases}$

Note that $\gamma_{t[1,2]}(P(n,2)) = \gamma_{[1,2]}(P(n,2))$ except for the case $n = 5$ and $n \equiv 1[6]$.

Denote by $g(n)$ the value of the right side of the equation in Theorem 6.3.1. We begin by proving an upper bound.

Lemma 6.3.2 $\gamma_{t[1,2]}(P(n,2)) \leq g(n)$ *for* $n \geq 5$.

Proof. For $n = 5$, the construction is already given in Fig. 6.1. In Fig. 6.12, we give the construction of the minimum $[1,2]$-total dominating set in $P(n,2)$ for $n \equiv 1[6]$. The proposed construction is based on the selection of one pair of nodes of the middle in each block which corresponds to $2n/3$ nodes. Then, we add two additional dominating nodes as depicted in color red in Fig. 6.12. For the cases $n \not\equiv 1[6]$, we refer to Fig. 6.3 since the provided sets are already total dominating sets. □

Now, it remains to prove that $g(n)$ is a lower bound. Let S be a total [1,2]-dominating set of minimum size of $P(2,n)$. Let $G[S]$ be subgraph induced by S. By definition of a $[1,2]$-total dominating set, each connected component of $G[S]$ has at least two vertices and every vertex of $G[S]$ has degree 1 or 2. Hence every connected component is either a path or a cycle. Let x_l and y_l be the numbers of connected components that are paths and cycles of order l, respectively. Observe that $x_1 = 0$ and $y_1 = \cdots = y_4 = 0$. Moreover, each path of order l dominates at most $2l+2$ vertices and each cycle of l vertices dominates at most $2l$ vertices. Thus,

$$\sum_{l \geq 2} (2l+2)x_l + 2ly_l \geq 2n \qquad (1)$$

$$\sum_{l \geq 2} l(x_l + y_l) = |S| \qquad (2)$$

From (1) and (2) we can deduce

$$|S| + \sum_{l \geq 2} x_l \geq n \qquad (3)$$

Also one may observe that

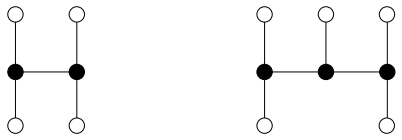
$$\sum_{l \geq 2} lx_l \geq 2 \sum_{l \geq 2} x_l \qquad (4)$$

In the following we study $\gamma_{t[1,2]}(P(n,2))$ according to residue of $n[6]$. Suppose $|S| < \gamma_{t[1,2]}(P(n,2))$.

(a) $n = 5$: This case can be checked by inspection.

(b) $n \equiv 0[3]$: If $n = 3k$ then $\gamma_{t[1,2]}(P(n,2)) = 2k$ and $|S| < 2k$. Inequality (3) becomes $|S| + \sum_{l\geq 2} x_l \geq 3k$, thus $\sum_{l\geq 2} x_l \geq k+1$. From (2), we deduce $\sum_{l\geq 2} l(x_l + y_l) < 2k$ thus $\sum_{l\geq 2} lx_l + \sum_{l\geq 2} ly_l < 2k$. Using (4), we obtain $2k + 2 + \sum_{l\geq 2} ly_l < 2k$, a contradiction.

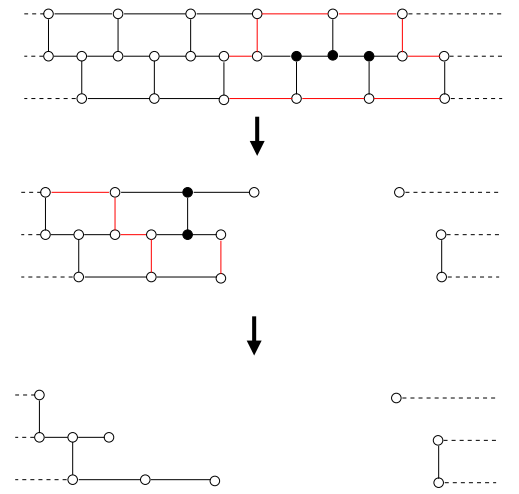
(c) $n \equiv 2,4,5[6]$: If $n = 6k + i$ with $i \in \{2,4,5\}$ then $\gamma_{t[1,2]}(P(n,2)) = 4k+2$ and $|S| < 4k+2$. Inequality (3) becomes $|S| + \sum_{l\geq 2} x_l \geq 6k + i - 4k - 2$ for $i \in \{2,4,5\}$, thus $\sum_{l\geq 2} x_l \geq 2k + j$ with $j \in \{1,3,4\}$. From (2), we deduce $\sum_{l\geq 2} l(x_l + y_l) < 4k+2$ thus $\sum_{l\geq 2} lx_l + \sum_{l\geq 2} ly_l < 4k+2$. Using (4), we obtain $4k + 2j + \sum_{l\geq 2} ly_l < 4k+2$ with $j \in \{1,3,4\}$, a contradiction.



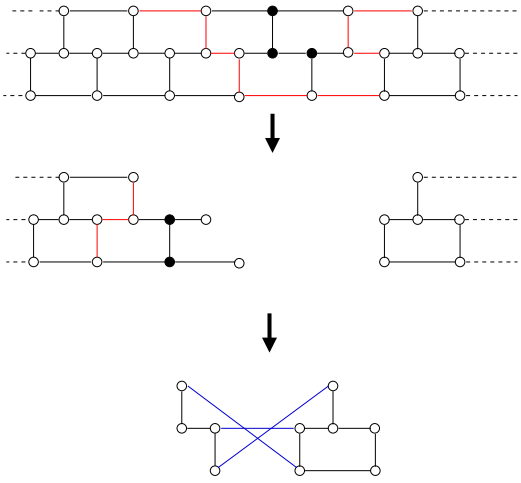
(a) (b)

Figure 6.10: Maximal components induced by P_2 and P_3 and their neighbors.

(d) $n \equiv 1[6]$: If $n = 6k+1$ then $\gamma_{t[1,2]}(P(n,2)) = 4k+2$ and $|S| < 4k+2$. Inequality (3) becomes $|S| + \sum_{l\geq 2} x_l \geq 6k+1$, thus $\sum_{l\geq 2} x_l \geq 2k$. From (2) and using (4), we obtain $4k \leq \sum_{l\geq 2} lx_l + \sum_{l\geq 2} ly_l \leq 4k+1$. This implies $\sum_{l\geq 2} ly_l = 0$, thus $4k \leq |S| = \sum_{l\geq 2} lx_l \leq 4k+1$. Since $\sum_{l\geq 2} lx_l \leq 4k+1$ and $\sum_{l\geq 2} x_l \geq 2k$, we have $\sum_{l\geq 2} lx_l \leq 2\sum_{l\geq 2} x_l + 1$. This is only possible if $x_3 = 1$ and $x_j = 0, \forall j > 3$. Thus, $G[S]$ is the union one path P_3 and x_2 paths P_2. Since every P_2 component can dominate at most 6 vertices and the P_3 component can dominate at most 8 vertices, we deduce $6x_2 + 8 \geq 12k + 2 = 2n$. On the other hand, recall $|S| = 2x_2 + 3 \leq 4k+1$ thus $6x_2 + 8 \leq 12k+2$. Hence, $6x_2 + 8 = 12k+2$. This implies that $P(n,2)$ can be partitioned into x_2 components of type shown in Fig. 6.10(a) and one component shown in Fig. 6.10(b) Suppose



(a)



(b)

Figure 6.11: Impossible partitionings.

such partitioning exists. In the following we study the partitioning by making consecutive extractions of components. Extracting a component means deleting all its vertices from the graph. Moreover, an extraction is said to be *forced* if there is no other option. Recall that the set of vertices of $P(n, 2)$ is the union of the two sets $U = \{u_0, \ldots, u_{n-1}\}$ and $V = \{v_0, \ldots, v_{n-1}\}$. Vertices of U and V form the two main cycles of $P(n, 2)$ respectively. Two cases are possible.

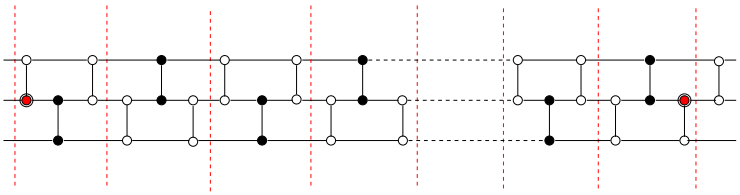


Figure 6.12: The construction of $\gamma_{t[1,2]}(P(n,2))$ for $n \equiv 1[6]$.

Either all three vertices of the P_3 component are on the same main cycle or two of them are on one cycle and the third on the other. In the first case, once the P_3 dominated component is extracted, the next forced extraction of a P_2 dominated component would imply the appearance of a vertex with a degree 2 (as shown by Fig. 6.11(a)). In the second case, after extracting the P_3 dominated component and after several forced extractions of P_2 dominated components(as shown by Fig. 6.11(b)), one may easily confirm that such a partitioning is impossible. We conclude that $x_3 = 0$, a contradiction.

This concludes the proof of Theorem 6.3.1.

6.4 Conclusion

Generalized Petersen graphs are very important structures in computer science and communication techniques since their particular structures and interesting properties. In this part, we considered a variant of the dominating set problem, called the $[1,2]$-dominating set problem. We studied this problem in generalized Petersen graphs $P(n,k)$ for $k = 2$. We gave the exact values of the $[1,2]$-domination numbers and the $[1,2]$-total domination numbers of $P(n,2)$.

Part III

Edge monitoring problem

Chapter 7

Introduction and first results

Contents

In this part of thesis, we are interested in a recent problem that deals with security networks, namely edge monitoring problem. This part is divided into three chapters and it is organized as follows: **Chapter 7** presents the motivation to study this problem and defines the edge monitoring problem from the graph theory point of view. Different variants of the problem are also presented. Then, we focus on 1-uniform monitoring problem by presenting some bounds on edge monitoring number in general graphs. We propose some characterizations related to those bounds. We also discuss a relation between this problem and two famous problems in graph theory: triangle packing problem [HR06] and double total dominating sets problem [HK10]. Some results on specific classes of graphs are also discussed such as path

power, cycle power, split graph, etc. A linear time algorithm for the square of a tree is also presented. In **Chapter 8**, we study the Edge Monitoring problem from the perspective of parameterized complexity. Afterwards, we focus in **Chapter 9** on the weighted version of the edge monitoring problem, called Weighted Edge Monitoring.

7.1 Motivation

Wireless Sensor Networks (WSNs in short) are increasingly used in the environment and industry thanks notably to the latest developments in the field of networks in the last few years [ASSC02]. The need to observe, analyze and control such type of area is essential to many environmental and scientific applications (e.g. measuring pollution levels, detecting earthquake activity, military surveillance, home health care, assisted living...). Anticipating security problems allows to protect the network from a variety of attacks. Many approaches have been proposed to protect sensor networks [LGBK12, RIBJ15, HWK+05].

In this thesis, we are interested in the Edge Monitoring mechanism for the security of wireless sensor networks. The basic idea of the Edge Monitoring problem [NHTK14a, WZMX10, DLL+11] is to select some nodes as monitors in a given sensor network. These monitors are employed for carrying out monitoring operations by listening promiscuously to the transmission of two nodes. They can also perform basic operations of communication and sensing in the network.

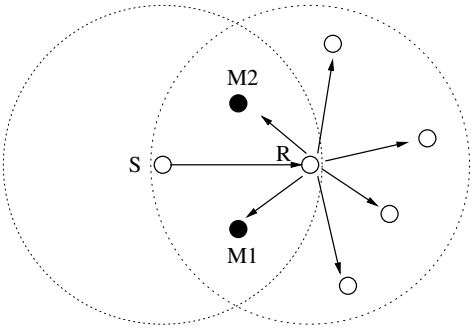


Figure 7.1: An example to illustrate the Edge monitoring problem.

The idea is illustrated in Figure 7.1. Each node in the network has a transmission range. The monitors (or watchdogs) are placed in the intersection of the

transmission ranges of the sending (S) and the receiving (R) nodes. They monitor nodes by listening promiscuously to the transmissions of both nodes. When node S forwards a message to R, the watchdog of this link verifies that node R also forwards the message. If R does not forward the message, then it is misbehaving. Similar to this, monitoring nodes are able to detect any malicious actions such as delaying, dropping, modifying, or even fabricated packets.

The Edge Monitoring problem was introduced in sensor networks [DLL+11, DLL08] as self-monitoring. Self-monitoring is an effective mechanism for the security of wireless sensor networks. Dong *et al.* studied the problem by modeling the communication network as a unit disk graph (UDG). They propose a polynomial-time approximation scheme for the problem in UDG graphs with a geometric representation [DLL08]. They also propose two distributed polynomial algorithms with provable approximation ratios.

In [HL06, WZL07, WLZ08], the authors concentrated on the system-level fault diagnosis of the network, especially detecting node failures as self-protection. The authors of [DLL+11, DLL08] focused on the fundamental issue of designing an edge self-monitoring topology, where every transmission link can be monitored by nodes within the network. In [NHTK14b],the authors studied the problem of edge monitoring from the perspective of self-stabilizing systems. They propose a polynomial self-stabilizing algorithm which operates under distributed daemon for computing a minimal edge monitoring set.

7.2 Definitions and variants of problem

All the graphs we consider in this part are undirected and contain neither loops nor multiple edges.

Let $G = (V, E)$ be a graph. Let v be a vertex of G. $N(v)$ denotes the set of vertices adjacent to v and $N_e(v)$ denotes the set of edges having v as an extremity. Let $c = \{0, 1, \ldots, k-1\}$ be a k-coloring of edges of G. Consider the following definitions:

Definition 5 *(Monitor) A vertex $v \in V$ is said to monitor (or to be a monitor of) an edge $e = \{u, w\}$ iff. v is adjacent to both extremities of e ($\{v, u\} \in E$ and $\{v, w\} \in E$). The set of monitors of an edge e is denoted by $M(e)$.*

Definition 6 *(Monitorable edge) An edge $e = \{u, v\}$ colored with color i is said to be monitorable if $|M(e)| \geq i$. The coloring c is said to be monitorable if all edges of G are monitorable.*

For the sake of simplicity, all colorings considered are monitorable.

Definition 7 *(Monitoring Set) Let $S \subseteq V$ be a subset of vertices. S is said to be a Monitoring set of G with regards to a coloring c iff. $\forall e \in E : |M(e) \cap S| \geq c(e)$. The edge monitoring number of G, denoted by $\gamma_m(G, c)$ is the minimum cardinality of a monitoring set according to the coloring c. If the vertices of G are weighted by a function $w : V \to \mathbb{Q}^+$, the weighted edge monitoring number of G, denoted by $\gamma_m(G, w, c)$, is equal to the minimum sum of weights of a monitoring set.*

Definition 8 *(Uniform edge monitoring) A monitoring whose coloring uses only one color k is called k-uniform edge monitoring (k-uniform monitoring for short).*

Observation 7.2.1 *The largest possible color in any monitoring is the color $n-2$.*

Observation 7.2.2 *Any i-uniform monitoring set is also a j-uniform monitoring set for all values $j < i$.*

Now, let formally define the different variants of the problem:

Edge Monitoring
Input: An edge colored graph $(G = (V, E), c)$, an integer $k \geq 0$.
Question: Is there a monitoring set S of G such that $|S| \leq k$?

---

k-Uniform Edge Monitoring
Input: A graph $G = (V, E)$, a color c, $\forall e \in E : c(e) = k$, an integer $t \geq 0$.
Question: Is there a monitoring set S of G such that $|S| \leq t$?

Let $G = (V, E), w, c$ such that G is a graph, $w : V \to \mathbb{Q}^+$ and $c : E \to \mathbb{N}$.

Weighted Edge Monitoring
Input: A weighted graph $(G = (V, E), w, c)$, a rational $k \geq 0$.

Question: Is there a monitoring set S of G, c such that $w(S) \leq k$?

The k-uniform weighted monitoring problem is a Weighted Edge Monitoring with the particularity that all the edges have the same color $c = k$.

7.3 Introductory results

In order to illustrate the behavior of the edge monitoring, we give some trivial characterizations of some special cases of the complete graph K_n. We first begin with $\{0, 1\}$-colorings.

Let K_n be a complete graph of at least 3 vertices.

Since any subset of three vertices monitors all edges of K_n, we deduce that for any $\{0, 1\}$-coloring c of G we have $\gamma_m(K_n, c) \leq 3$.

This observation is supported with the following characterizations of special cases:

- $\gamma_m(K_n, c) = 1$ *iff.* $\exists v \in V$ *s.t.* $\forall e \in N_e(v) : c(e) = 0$.
- $\gamma_m(K_n, c) \leq 2$ *iff.* $\exists uv \in E : c(uv) = 0$.

Now, we consider monitoring K_n with $\{i, \ldots, n-2\}$-colorings. That is we assume at least one edge having a maximum color $c = n - 2$.

- $\gamma_m(K_n, c) = n$ *iff.* $\exists e_1, e_2 \in E$ *s.t.* e_1 is not adjacent to e_2 and $c(e_1) = c(e_2) = n - 2$.
- $\gamma_m(K_n, c) = n - 1$ *iff.* for more than one edge with color $n - 2$, the subgraph induced by E_{n-2}, the edges colored with $n - 2$, must be a star or for one edge with color $n - 2$, we must have at least one edge nonadjacent to the edge of color $n - 2$, having a color $n - 3$.
- $\gamma_m(K_n, c) = n - 2$ *iff.* there are one and only one edge having the color $n - 2$ and all the edges adjacent to this edge must have a color $c < n - 2$ and all the edges nonadjacent having a color $c < n - 4$.

Now, we consider monitoring K_n in the case color $c = 1$.

Proposition 7.3.1 *Let K_n and $K_n - I$ denote the complete graph with n vertices and complete graph with n vertices from which a 1-factor (a perfect matching) I has been removed, respectively. The edge monitoring number for K_n and $K_n - I$ for any $n \geq 3$ is 3.*

Proof. For K_n: Let m_1 be the first monitor chosen among the vertices of K_n. This monitor will monitor all the edges of $K_{n-1} = K_n - \{m_1\}$ because both ends of each edge in K_{n-1} have a link with m_1. The remaining edges is all the edges having the monitor m_1 as end. To monitor these edges, we choose a vertex belonging to K_{n-1} as a second monitor m_2. All edges will be monitored except m_1m_2, so we choose a third monitor m_3 in order to monitor this edge. Finally, we needed three monitors to monitor all edges.
For $K_n - I$: Let m_1 be the first monitor. As shown in Figure 7.2, this monitor have a connection with all the vertices except s_1. So m_1 will monitor all edges except the edges having m_1 or s_1 as end. We choose the second monitor m_2, we are sure that m_2 has connection with m_1 and s_1 but not with s_2. m_2 will monitor the remaining edges except the edges m_1m_2, m_2s_1 and s_1s_2. To monitor the remaining edges, we choose the third monitor m_3 from the rest. m_3 has necessarily connection with m_1, m_2, s_1 and s_2.

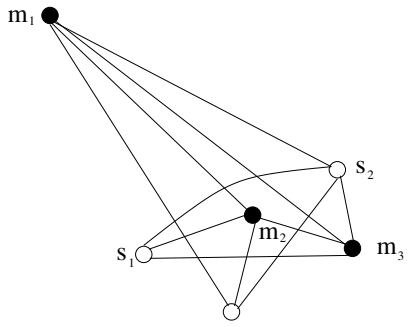


Figure 7.2: Edge Monitoring for $K_6 - I$.

□

It follows that the edge monitoring number of the complete graph, the complete graph minus 1-factor and the complete multipartite graph is equal to 3.

In the rest of the chapter we focus on monitoring graphs with a special case of coloring with only one color $c = \{1\}$.

Note that we restrict Edge Monitoring to apply only on graphs where every edge is part of at least one triangle. Indeed if it is not the case, then we can directly answer that the problem has no solution in polynomial time. This restriction is no big deal because in practice either we add sensors that cover the edges that are not in a triangle or remove the edges by forbidding communications on these edges.

7.4 1-uniform monitoring of general graphs

The 1-uniform monitoring problem (1-UMP) assumes the uniform coloring where all the edges have the same color $c = 1$. A graph admitting a 1-uniform monitoring set is called a 1-monitorable graph. In the following, we give a trivial characterizations of 1-monitorable graphs. An example of such graphs are maximal planar graphs since every edge belongs to a triangle. However, the graph illustrated in Figure 7.3 admits a 1-uniform monitoring and it's not a maximal planar graph.

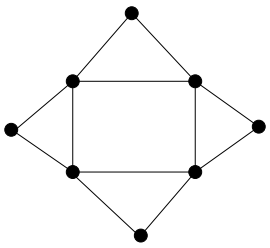


Figure 7.3: A 1-uniform monitorable non maximal planar graph.

Recall from Definition 6, a graph $G = (V, E)$ admits 1-uniform monitoring *iff.* $|M(e)| \geq 1$ for all $e \in E$. Thus, recognizing 1-monitorable graphs could be performed in polynomial time. From now on we consider only 1-monitorable graphs and we use the term graph or 1-monitorable graph equivalently and interchangeably.

We first give an introductory example in a simple graph class, namely, wheel graph.

Proposition 7.4.1 *The wheel graph W_n has*

$$\gamma_m(W_n, 1) = \begin{cases} 2\alpha + 1 & \text{if } n - 1 \equiv 0[4] \\ 2\alpha + 2 & \text{if } n - 1 \equiv 1[4] \\ 2\alpha + 3 & \text{otherwise.} \end{cases}$$

with $\alpha = \lfloor \frac{n-1}{4} \rfloor$.

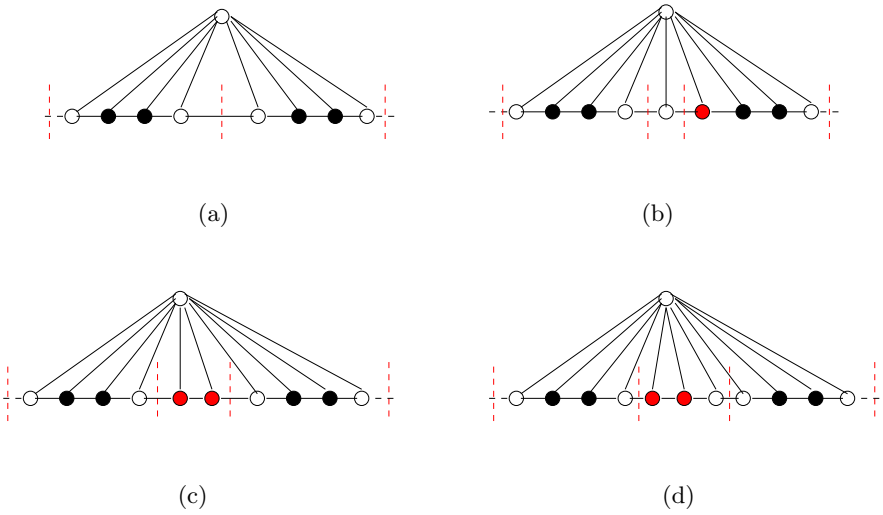


Figure 7.4: Edge monitoring of the wheel graph. The red nodes are the monitors that must be added.

Proof. Clearly, the universal vertex need to be monitor to monitor all lateral edges (edges of the cycle). For $n = 3, 4, 5$, $\gamma_m(W_n, 1) = 3$. We can easily check it by inspection. For $n > 5$ and using Observation 7.5.2, it implies that if we choose a vertex v as monitor from the vertices of the cycle, we must choose at least another vertex as monitor adjacent to v from the cycle. Then, all lateral vertices (vertices of the cycle) choose as monitors need to form paths of at least two monitors. Let decompose the lateral vertices of the graph into blocks of four vertices. We distinguish three cases according to the value of $n - 1$:

(a) For $n - 1 \equiv 0[4]$: As shown in Figure 7.4(a), each two monitors can only monitor four edges. This implies that we need two monitors for each block. Since $n - 1 \equiv 0[4]$ and in order to monitor all the edges, we must choose $2(\frac{n-1}{4})$ vertices to monitor the remaining edges plus one (the universal vertex).

(b) For $n - 1 \equiv 1[4]$: In this case, we use the same method using in the precedent case to monitor the edges. We have one edge more and in order to monitor it, we must choose one monitor more as shown in Figure 7.4(b).

(c) For $n - 1 \equiv 2[4]$: We need to prove that we need two more monitors in order to monitor the two added edges. To monitor four edges and using only two monitors, the monitors must be positioned in the center of each block. Since each monitor need to be adjacent to two other monitors (Observation 7.5.2), it implies that if we

add one monitor to monitor the two added edges, this monitor must be adjacent to the two monitors of the adjacent block then only one edge can be monitored. Thus, we need two more monitors as shown in Figure 7.4(c).
(d) For $n-1 \equiv 3[4]$: In this case, we will have three more edges to be monitored. Since one monitor can only monitor two edges then we need at least two other monitors as depicted in Figure 7.4(d). This complete the proof. □

7.5 Bounds on edge monitoring number

The decision problem to determine the edge monitoring number of a graph is NP-complete (see Chapter 8). Hence it is of interest to determine some bounds on the edge monitoring number of a graph.

The first proposition of this section gives a trivial lower bounds of the edge monitoring number $\gamma_m(G,1)$ for any 1-monitorable graph G.

Proposition 7.5.1 *Let G be a 1-monitorable graph. The edge monitoring number $\gamma_m(G,1) \geq 3$.*

Proof. By contradiction, we prove there is no possible monitoring with 2 or less monitors. Suppose a monitoring set with 2 monitors u and v. If u and v are neighbors then the edge uv is not monitored and needs at least one new monitor. Otherwise, u and v belong to two triangles. This implies that at least two other monitors will be needed. Thus, any monitoring set of G will have three monitors at least. □

Let G be a connected graph of order $n \geq 3$. There exist graphs where all vertices should be monitors. Thus, $3 \leq \gamma_m(G,1) \leq n$.

Observation 7.5.2 *Let $G=(V,E)$ be a 1-monitorable graph and let S be a monitoring set of G. Each vertex of G is adjacent to at least two vertices of S.*

To prove the previous observation, one may rely on the fact that a 1-monitorable graph G has minimum degree at least 2. We now present a characterization of graphs reaching the upper bound:

Observation 7.5.3 *Let $G=(V,E)$ be a 1-monitorable graph. $\gamma_m(G,1)=n$ iff. for each vertex v of V there exists an edge e of E such that $M(e)=\{v\}$.*

Thus, recognizing graphs G having $\gamma_m(G,1) = n$ could be performed in polynomial time.

The following proposition gives more precise description of such graphs having a universal vertex *i.e.* a vertex adjacent to all others.

Proposition 7.5.4 *Let G be a 1-monitorable graph.*
(1) If $\Delta(G) = n-1$ and $|E(G)| > \frac{3n-3}{2}$ then $\gamma_m(G,1) < n$.
(2) If $\Delta(G) = n-1$ and $|E(G)| = \frac{3n-3}{2}$ then $\gamma_m(G,1) = n$.

Proof. Let G be a monitorable graph with order $n \geq 4$. Let $\Delta = n-1$ then we have at least one vertex v with degree $n-1$. If we select this vertex v as monitor, this permit to monitor all the edges of the graph G except $n-1$ edges that have v as one end. In the case (2), the number of edges of G is $\frac{3n-3}{2}$ and according to the definition of the monitorable graph this means that there are one vertex with degree $n-1$ and the remaining vertices have degree 2 and n is odd. Otherwise, if n is even $|E(G)|$ is not natural number. It implies that the graph is a set of triangles sharing the same vertex v. Then, in this case, all the vertices of this graph must be monitors in order to monitor all the edges. In the case (1), the number of edges is bigger than $\frac{3n-3}{2}$. This means that for the $n-1$ vertices, we have more than $\frac{n-1}{2}$ edges. Thus, to monitor the $n-1$ remaining edges, we need less than $n-1$ vertices so $\gamma_m(G,1) < n$. □

Now, we mention a basic lower bound for edge monitoring number.

Theorem 7.5.5 *Let G be a monitorable graph of order n with no isolated vertices. Then $\gamma_m(G,1) \geq \frac{2n}{\Delta}$.*

Proof. Let S be a set of monitors of G. Then, using Observation 7.5.2, every vertex of G is adjacent to two vertices of S. That is, all neighbors of the set S $N(S) = 2n$. Since every $v \in S$ can have at most Δ neighbors, it follows that $\Delta\gamma_m(G,1) \geq 2n$. The theorem follows by dividing this inequality by Δ. □

Corollary 7.5.6 *Let G be a monitorable graph of order n. If $\Delta \leq \frac{n}{k}$ for some positive integer $k > 1$, then $\gamma_m(G,1) \geq 2k$. If $\Delta < \frac{n}{k}$, then $\gamma_m(G,1) \geq 2k+1$.*

Proof. By Theorem 7.5.5, $\gamma_m(G,1) \geq \frac{2n}{\Delta}$. If $\Delta \leq \frac{n}{k}$, then substitution yields $\gamma_m(G,1) \geq 2k$. Moreover, if $\Delta < \frac{n}{k}$, then by substitution again, we have $\gamma_m(G,1) > 2k$. Hence $\gamma_m(G,1) \geq 2k+1$. □

Recall the following definitions:

Definition 9 *(k-tuple dominating set) For a fixed positive integer k, a k-tuple dominating set (resp. k-tuple total dominating set) of a graph $G = (V, E)$ is a subset S of V such that $|N[v] \cap S| \geq k$ (resp. $|N(v) \cap S| \geq k$) for every vertex $v \in V$. The k-tuple domination number $\gamma_{\times k}(G)$ (resp. k-tuple total domination number $\gamma_{\times k,t}(G)$) is the minimum cardinality of a k-tuple dominating set (resp. k-tuple total dominating set) of G. A dominating set of G is a 1-tuple dominating set of G. A total dominating set of G is a 1-tuple total dominating set of G. A double dominating set of G is a 2-tuple dominating set of G.*

Definition 10 *(Edge disjoint triangle packing) A graph $G = (V, E)$ is said to have a k packing of triangles K_3 if there exist k disjoint copies $(K_3)^1, ..., (K_3)^k$ of K_3 in the vertex set of G. The packing is called edge-disjoint if we allow $(K_3)^1, ..., (K_3)^k$ to have some vertices in common but no edges exist in $(K_3)^i \cap (K_3)^j$ when $i \neq j$.*

In the following, we give better lower and upper bounds:

Theorem 7.5.7 *Let $G = (V, E)$ be a monitorable graph and let F be a subgraph such that F is the maximum edge disjoint triangle packing of G. Then, $\gamma_{\times 2,t}(G) \leq \gamma_m(G, 1) \leq |V(F)|$.*

Proof. Let $D \subseteq V$ be $\gamma_m(G, 1)$-set of G. From Observation 7.5.2, we deduce that D is also a double total dominating set and we hence obtain $\gamma_{\times 2,t}(G) \leq \gamma_m(G, 1)$. For the upper bound, let F be a subgraph which represent the maximum edge disjoint triangle packing of G. We prove that any $V(F)$ is edge monitoring set for G. Assume, to the contrary, that $V(F)$ is not an edge monitoring set. This implies that $\exists e \in G$ such that $e = uv$ is not monitored by any vertex of $V(F)$. By assumption, the graph G is monitorable, then e belongs to at least one triangle $< v, u, w >$. The triangle $< v, u, w >$ has to belong to F for otherwise F is not a maximum edge disjoint triangle packing of G. Contradiction. Thus, $V(F)$ is an edge monitoring set and then $\gamma_m(G, 1) \leq |V(F)|$. □

To finish this section, we give a brief study on how the monitoring set evolves with edge deletion. In particular, we consider the example of complete graphs and deleting edges by blocks of perfect matchings I.

Proposition 7.5.8 *Let K_n be a complete graph with n even. $K_n - \alpha I$ remains 1-monitorable only if $\alpha < \frac{n-2}{2}$.*

Proof. We begin by prove that if $\alpha = \frac{n-2}{2}$ then we have edges that don't form a triangle. Note that $K_n = K_{\frac{n}{2}} \cup K_{\frac{n}{2}} \cup K_{\frac{n}{2},\frac{n}{2}}$. Note that the degree of each vertex of $K_{\frac{n}{2},\frac{n}{2}}$ is $\frac{n}{2}$. When we remove $\frac{n-2}{2}$ perfect matchings from $K_{\frac{n}{2},\frac{n}{2}}$, we are sure that the remaining edges in $K_{\frac{n}{2},\frac{n}{2}}$ form a perfect matching and we denote it by PM. Then, $K_n - (\frac{n-2}{2})I = K_{\frac{n}{2}} \cup K_{\frac{n}{2}} \cup PM$. We prove easily that every edge e in PM don't belong to triangle because we can't have a common adjacent vertex between any two vertices in PM. Now, for any $\alpha > \frac{n-2}{2}$, we proceed by the same way as below for the first $\frac{n-2}{2}$ perfect matchings then we remove $\frac{n-2}{2}$ perfect matching from $K_{\frac{n}{2},\frac{n}{2}}$ After that, for the remaining perfect matchings, we remove them from each $K_{\frac{n}{2}}$ if $\frac{n}{2}$ is even. If $\frac{n}{2}$ is odd, we remove an additional edge for each perfect matching from the edges of PM. Using this method to remove α perfect matchings, we are sure that all remaining edges of PM doesn't belong to a triangle.

Now, we prove that even if $\alpha = \frac{n-2}{2} - 1$ or less, every edge in $K_n - \alpha I$ belongs to triangle. In the case when $\alpha = \frac{n-2}{2} - 1 = \frac{n-4}{2}$, the two extremities of every edge is not connected to $\frac{n-4}{2}$ vertices. Suppose that the two extremities don't have the same disconnected vertices set, then the two extremities have $n-4$ disconnected vertices. This implies that we are sure that, in the worst case, every two extremities of any edge in $K_n - (\frac{n-4}{2})I$ have two vertices (neighbors) in common. □

7.6 1-uniform monitoring for some graph classes

7.6.1 Planar unit disc graphs

A graph $G = (V, E)$ is a unit disk graph if it there exists a map $f : V \to \mathbb{R}^2$ satisfying

$$\{u, v\} \in E \Leftrightarrow \|f(u) - f(v)\| \leq 2$$

f is called a geometric representation of G.

Recognizing whether a graph G is an unit disk graph is NP-hard [BK98]. Thus, computing a geometric representation of an unit disk graph is also NP-hard. Consequently, we suppose that an unit disk graph G is given with a geometric representation f.

Dong et al [DLL+11] prove that k-uniform Edge Monitoring is NP-complete on unit disk graphs for every $k \geq 2$. We prove a stronger result for 1-uniform Edge Monitoring.

Theorem 7.6.1 *1-uniform* Edge Monitoring *is* NP*-complete on planar unit disk graphs given with a geometric representation.*

The proof is inspired by Theorem 4.1 in [CCJ90b]. As in [CCJ90b] we use the following lemma:

Lemma 7.6.2 *[Val81] A planar graph G with maximum degree 4 can be embedded in the plane using $O(|V|)$ area in such a way that its vertices are at integer coordinates and its edges are drawn so that they are made up of horizontal or vertical segments.*

Proof. (of Theorem 7.6.1) We show a reduction from PlanarVertexCover with maximum degree 3 which is NP-complete [GJ77]. Let $G = (V, E)$ be a planar graph with maximal degree 3. Let $\{e_1, \ldots, e_{|E|}\}$ be the edges in G. Let $N > 0$ be an integer. We draw G in the plane using Lemma 7.6.2 (see Figure 7.5) and we adjust the scale such that each vertex is at coordinate (xN, yN) for some integers x and y. We build $G' = (V', E')$ from G by replacing each edge $e_i = \{u, v\}$, $i \in [1, \ldots, |E|]$, by a subgraph G_{e_i} of vertices $\{a_{i,0} = u, b_{i,0}, b'_{i,0}, a_{i,1}, b_{i,1}, b'_{i,1}, \ldots, a_{i,2n_i}, b_{i,2n_i}, b'_{i,2n_i}, a_{i,2n_i+1} = v\}$ where n_i is an integer that depends on the length of the embedding of e_i. For each $j \in [0, 2n_i]$, we connect $b_{i,j}$ and $b'_{i,j}$ to $a_{i,j}$ and $a_{i,j+1}$ and we connect $b_{i,j}$ to $b'_{i,j}$ (see Figure 7.6).

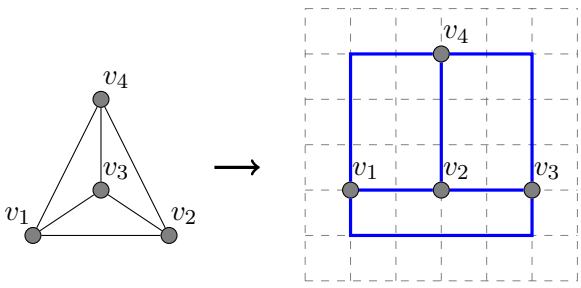


Figure 7.5: A representation of K_4 in the grid

It is easily seen that the obtained graph G' is planar and that there exists an unit disk representation of G' for suitable N and $(n_i)_{i \in [1,|E|]}$. Now, we prove that

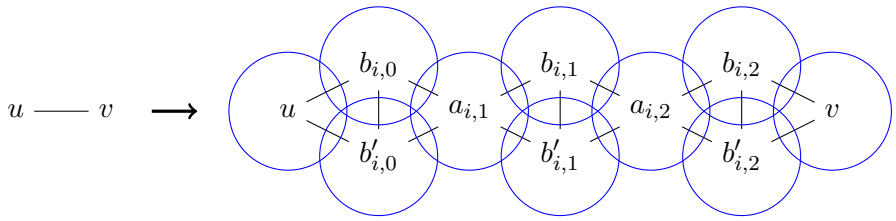


Figure 7.6: An edge $e_i = \{u, v\}$ and its associate graph G_{e_i} for $n_i = 1$

G admits a vertex-cover S such that $|S| \leq k$ if and only if G' has a monitoring set S' such that $|S'| \leq k' = k + \sum_{i \in [1,|E|]}(5n_i + 2)$. Let A be the set of vertices $a_{i,j}$ for $i \in [1, |E|]$ and $j \in [1, 2n_i]$. Let B be the set of vertices $b_{i,j}$ and $b'_{i,j}$ for $i \in [1, |E|]$ and $j \in [0, 2n_i]$. Clearly, V' is the disjoint union of V, A and B. Moreover, $|A| = \sum_{e_i \in E}(2n_i)$ and $|B| = \sum_{e_i \in E}(4n_i + 2)$. The proof is an immediate consequence of these three facts.

(1) If a set $S' \subseteq V'$ monitors G' then $B \subseteq S'$: otherwise, there exists a vertex $b_{i,j}$ or $b'_{i,j}$ that is not in S'. Then $\{a_{i,j}, b_{i,j}\}$ or $\{a_{i,j}, b'_{i,j}\}$ is not monitored by S'.

(2) Let S be a vertex-cover of G. Then there is a set $A' \subseteq A$ such that $|V(G_{e_i}) \cap A'| = n_i$ for every $i \in [1, |E|]$ and such that $S \cup A' \cup B$ is a monitoring set of G': let $e_i = \{u, v\}$ be an edge in G. If $u \in S$, then we choose all $a_{i,2j}$ for $j \in [1, n_i]$ as elements of A'. Otherwise ($v \in S$), we choose $a_{i,2j+1}$ for $j \in [0, n_i - 1]$. It is easily seen that $S \cup A' \cup B$ is a monitoring set of G'.

(3) There exists a minimum monitoring set S' of G' such that $V \cap S'$ is a vertex-cover of G and $|V(G_{e_i}) \cap A \cap S'| = n_i$ for every $i \in [1, |E|]$: assume that $V \cap S'$ is not a vertex cover of G. Let $e_i = \{u, v\}$ be an edge in G not covered by $V \cap S'$. Then, it is easily seen that $|V(G_{e_i}) \cap A \cap S'| > n_i$. Otherwise, an edge $\{b_{i,j}, b'_{i,j}\}$ for some j is not covered by S'. Thus, we can replace these vertices by u and n_i vertices in $V(G_{e_i}) \cap A$ which monitors every edge $\{b_{i,j}, b'_{i,j}\}$. By iterating this process on every edge in G, we obtain a set with the desired properties. Now, assume that $V \cap S'$ is a vertex cover of G but there is some i such that $|V(G_{e_i}) \cap A \cap S'| \neq n_i$. It is easily seen that $|V(G_{e_i}) \cap A \cap S'| < n_i$ implies that an edge $\{b_{i,j}, b'_{i,j}\}$ for some j is not covered by S' and if $|V(G_{e_i}) \cap A \cap S'| > n_i$ then S' is not minimum. □

7.6.2 Split graphs

We prove that the problem is NP-complete on split graphs.

Lemma 7.6.3 *Let $G = (V = C \cup I, E)$ be a split graph with minimum degree $\delta(G) \geq 2$ and such that $|C| \geq 3$ then, there exists a minimum 2-tuple dominating set (resp. monitoring set) $S \subseteq C$.*

Proof. Let S be a set that minimizes $|S \cap I|$ among all minimum 2-tuple dominating sets of G. For the sake of contradiction, suppose $S \cap I$ non empty and let v be a vertex in $S \cap I$. If $N(v) \subseteq S$, then $S - v$ is also a 2-tuple dominating set of G. Thus, G is not minimum. Now, suppose that $N(v) \nsubseteq S$. Then, there exists $u \in N(v) \setminus S$. Then $S' = S \cup \{u\} - v$ is a minimum 2-tuple dominating set of G with $|S' \cap I| < |S \cap I|$. Thus S does not minimize $|S \cap I|$.

The proof for monitoring sets is quite similar to the proof for 2-tuple dominating sets. Let S be a set that minimizes $|S \cap I|$ among all minimum monitoring sets of G. For the sake of contradiction, suppose $S \cap I$ non empty and let v be a vertex in $S \cap I$. S contains at least 3 vertices and $|S \cap C| \geq 2$. Otherwise, S does not monitor all edges between C and I. If $N(v) \subseteq S$ and $|S \cap C| \geq 3$, then $S - v$ is also a monitoring set of G. Thus S is not minimum. If $N(v) \subseteq S$ and $|S \cap C| = 2$ then choose a vertex $u \in C \setminus S$. Thus, $S' = S \cup \{u\} - v$ is a minimum monitoring set with $|S' \cap I| < |S \cap I|$. Now, suppose that $N(v) \nsubseteq S$ and let $u \in N(V) \setminus S$. Then, $S' = S \cup \{u\} - v$ is a minimum monitoring set with $|S' \cap I| < |S \cap I|$. That contradicts our assumption. □

Lemma 7.6.4 *Let $G = (V = C \cup I, E)$ be a split graph with minimum degree $\delta(G) \geq 2$ and such that $|C| \geq 3$ and $\gamma_{\times 2}(G) \geq 3$. Then, $\gamma_m(G, 1) = \gamma_{\times 2}(G)$.*

Proof. By Theorem 7.5.7, we have $\gamma_{\times 2}(G) \leq \gamma_m(G, 1)$. We will prove that $\gamma_{\times 2}(G) \geq \gamma_m(G, 1)$. Let S be a minimum 2-tuple dominating set of G. Thanks to Lemma 7.6.3, we can assume without loss of generality that $S \subseteq C$. Since $|S| \geq 3$, S monitors all edges in $G[C]$. Let $\{u, v\}$ be an edge in G such that $u \in C$ and $v \in I$. Since S dominates twice the vertex v, there is a vertex $u' \in S \cap N(v)$ distinct to u. Thus $\{u, v\}$ is monitored by u'. Consequently, S is a monitoring set of G. □

Since 2-tuple domination is NP-complete on split graphs [LC03] and by Lemma 7.6.4, we obtain the following result.

Theorem 7.6.5 *1-uniform* Edge Monitoring *is NP-complete on split graphs.*

7.6.3 Comparability graphs

We also prove that the problem is NP-complete on comparability graphs.

Theorem 7.6.6 *1-uniform* Edge Monitoring *is NP-complete on comparability graphs.*

Proof. We do a reduction from Total Domination on bipartite graphs which has been proved NP-complete [PLH82]. Let $G = (V, E)$ be a bipartite graph. Without loss of generality, assume that G has no isolated vertices. Let G' be the graph obtained from G by adding a universal vertex u. It is clear that G' is a comparability graph. We will prove that $\gamma_m(G', 1) = \gamma_t(G) + 1$. Let S be a total dominating set of G. Then, $S \cup \{u\}$ is a monitoring set of G. Indeed, every edge in E is covered by u and for every edge $\{u, v\}$ with $v \in V$, there is a vertex $v' \in N(v) \cap S$. Thus, $\{u, v\}$ is monitored by v'. Now, let S be a monitoring set of G'. Then, $u \in S$ because u is the only vertex that monitors edges in E. $S - u$ is a total dominating set of G. Indeed, let v be a vertex in V. $\{u, v\}$ is an edge of G' monitored by a vertex $v' \in S - u$ distinct from v. Thus, v is dominated v'. □

7.6.4 General results on graph power

Let $G = (V, E)$ be a graph. Let $d(G)$ be the diameter of G. Throughout this section we assume that G is connected. If G is disconnected then, $\gamma_m(G, 1)$ is the sum of the edge monitoring numbers of its components. We define G^k as the k^{th} power of G that has the same set of vertices as G, but in which two vertices are adjacent when their distance in G is at most k.

Since the graph G does not necessarily have the properties of a monitorable graph, we discuss the edge monitoring number of some graph power.

Theorem 7.6.7 *The graph power G^k of any connected graph G with at least $k+1$ vertices is $(k-1)$-monitorable.*

Proof. Since G is a connected graph, then each vertex $v \in G$ has degree at least 1. Note that G has at least $k+1$ vertices and no isolated vertices this implies that each vertex in G^k will be connected to at least k vertices. Now, in order to prove that each edge in G^k is $k-1$-monitorable, it suffices to prove that each edge in G^k belongs to $k-1$ triangles. By definition of graph power G^k, two vertices are adjacent when their distance in G is at most k. Let $e = uv$ be an edge in G^k. For

each edge in G^k, each extremity is connected to at least k vertices. Consider one extremity v, v has at least $k-1$ connections with other vertices and one connection with u. Furthermore, the other extremity u is also connected to at least k vertices. Since v is adjacent to u and G has at least $k+1$ vertices, then u has at least $k-1$ common vertices with v. This means that any edge e in G^k belongs to at least $k-1$ triangles. Hence, G^k is $k-1$-monitorable. □

Proposition 7.6.8 *Let G be a connected monitorable graph. If G is monitorable, then $\gamma_m(G^k,1) = 3 \leq \gamma_m(G^{k-1},1) \leq \cdots \leq \gamma_m(G^2,1) \leq \gamma_m(G,1)$ with $k \geq d-1$ and d the diameter of the graph G.*

Proof. Let $\gamma_m(G,1) = t$. We can easily prove that the same number t can monitor all edges of the graph G^i for any $i > 1$.
First, we prove that is true for the first graph power G^2. G^2 has the same set of vertices as G, but we add edges between two vertices when their distance in G is 2. We simply prove that all the additional edges can be monitoring by the same monitors of G. We have two subcases A and B as showing in Figure 7.7 and the details of each subcases are as follow:

- If we connect any two vertices, of distance two, having one monitor in common, denoted by v. Clearly this monitor v can monitor the new edge e (see Figure 7.7(A)).
- In the contrary, there is no monitor in common between the two vertices in the graph G, as shown in Figure 7.7(B). Since all edges of the graph G are monitored, the edges e_1 and e_2 have one monitor (or more). Since the graph G^2 allows to connect each two vertices of distance 2, then the edges e_3 and e_4 connect each extremity of the edge e with the monitors of e_1 and e_2, respectively. This implies that the edge e can be monitored by any monitors of edges e_1 or e_2.

Hence, $\gamma_m(G^2,1) \leq \gamma_m(G,1)$. By the same way, we can easily prove that $\gamma_m(G^3,1) \leq \gamma_m(G^2,1)$ and so on. Besides, the graph G^k, for $k \geq d-1$ is a complete graph and for $k = d-1$ is a complete graph minus one factor, then $\gamma_m(G^k,1) = 3$ (For more details see the proof of Proposition 7.3.1). □

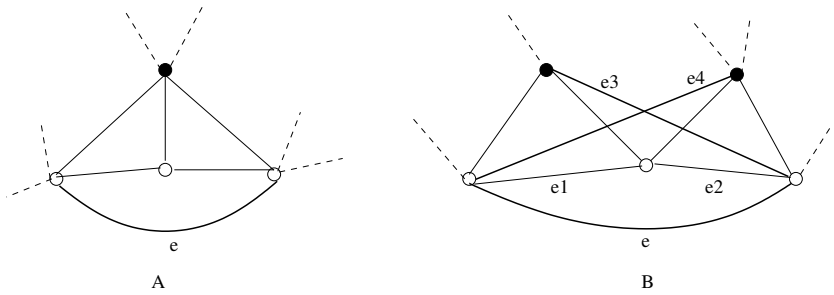


Figure 7.7: Edge monitoring in graph power.

7.6.5 Linear algorithm for the square of tree

One of the important classes of graphs is the trees. The importance of trees is evident from their applications in various areas. Besides, many hard problems are efficiently solvable on trees, and this is the second reason for studying the edge monitoring problem in this classe of graphs.

Let T be a tree of order $n \geq 3$. In the following, we give an algorithm for the edge monitoring of the square of any tree T, denoted by T^2. Please observe that trees are not monitorable graphs.

First we present some definitions. A leaf is a vertex of degree 1. If a vertex is not a leaf this means that is an inner vertex of degree at least 2. A forest is a disjoint union of trees.

We start with the following propositions that can be easily obtained, thus proofs are omitted.

Proposition 7.6.9 *Let T be a tree of order $n \geq 3$. If $d(T) \leq 4$, then $\gamma_m(T^2, 1) = 3$.*

Proposition 7.6.10 *For any tree T with diameter at least 5, there exist $\gamma_m(T^2, 1)$-set that contains no leaves of T.*

In the rest of this section, we only consider $\gamma_m(T^2, 1)$-sets for which the Proposition 7.6.10 holds.

Lemma 7.6.11 *Let T be a tree of order n. T^2 is the square of the tree T. Let S be a set of monitors of T^2. Let $V(T)$ be the set of vertices of T. Let L be a set of leaves of T. If $\exists v \in V(T)$ such that one of the following conditions hold in T:*
(a) *$deg(v) = 2$ or*

(b) $|N(v) \cap L| = deg(v) - 2$ *or*

(c) $\exists u \in L : \ dist(u, v) = 2$ *or*

(d) $\exists u \in L : \ dist(u, v) = 3$ *and let* v, x, y, u *be the path between* v *and* u *then* $|N(x) \cap L| = deg(x) - 2$

then $v \in S$.

Proof. Let v be a vertex of T and S the set of monitors of T^2. To prove that v must be in S, we just need to verify that for each case there exists at least one edge that can only be monitored by v. These four possible cases are illustrated in Figure 7.8. The red edges are the edges that candidate be monitored. The Figure 7.9 represents a counterexample to explain why we need to add the condition $|N(x) \cap L| = deg(x) - 2$ in **(d)**.

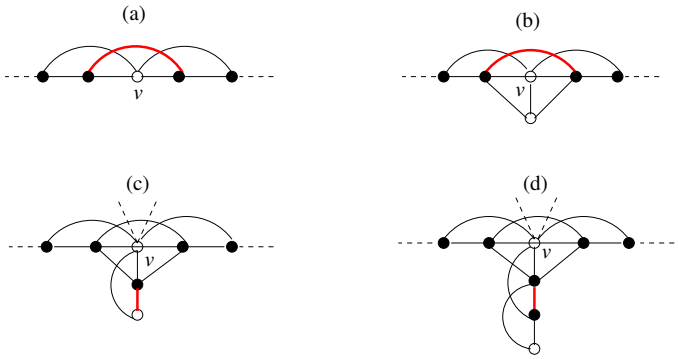


Figure 7.8: The four cases of Lemma 7.6.11 for which v must be monitor. A non monitor vertex is represented by white vertex. The red edges are the edges that can only be monitored by v.

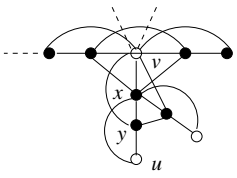


Figure 7.9: $\exists u \in L : \ dist(u, v) = 3$ and let v, x, y, u be the path between v and u then $|N(x) \cap L| \neq deg(x) - 2$.

This completes the proof. □

Let decompose the set of vertices of T into three subsets. Let V_1 be a set of vertices that are leaves. Let V_2 be a set of vertices that satisfy the conditions of

Lemma 7.6.11. The set V_3 is the remaining vertices that belong to neither V_1 nor V_2. This means that there is no monitors that belong to the set V_1 and all vertices of V_2 are monitors. In the following lemma, we will show the conditions that must satisfy the set of vertices that are non monitors.

Lemma 7.6.12 *Let T be a tree of order n. T^2 is the square of this tree. Let S be a monitoring set of T^2. Let $\{u, v\} \in V(T)\backslash S$ and $\{x, y\} \in V(T) \cap S$ be four vertices in T. **(1)** If $deg(v) = 3$ in T and $v \notin S$, then v cannot be adjacent to a non monitor in T. **(2)** If $deg(v) \geq deg(u) > 3$, then u may be adjacent to v in T but not to a third vertex $w \in V(T)\backslash S$. **(3)** If $deg(x) = 2$ in T or $deg(x) > 2$ in T and x is adjacent to at least $deg(x) - 2$ leaves (same for y), then u and v cannot form the path with x and y in T as follow u, x, y, v.*

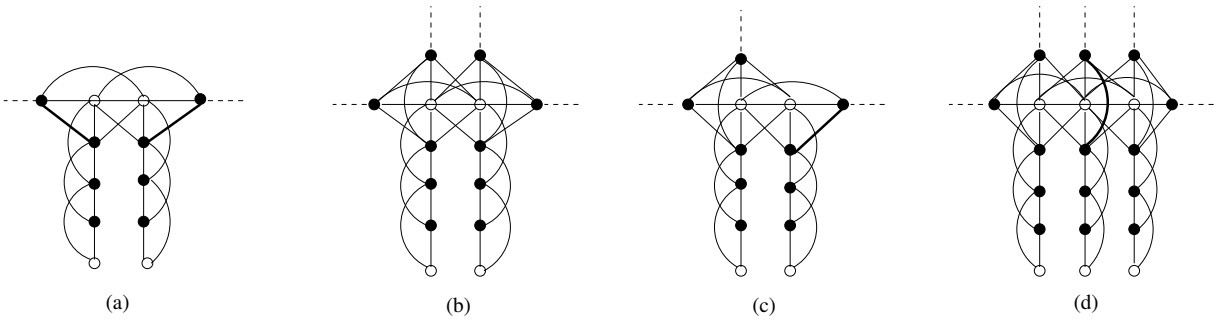


Figure 7.10: The $V\backslash M$-vertex induced subgraph from T does not contain P_3 (and can contain P_2 in some conditions). A non monitor vertex is represented by white vertex. The red edges are the edges that can only be dominated by a white vertex.

Proof. Let $\{u, v\} \in V(T)$ two vertices such that $\{u, v\} \notin M$. We deduce three cases depending on the degree of v and u:
(1) If v has degree 3, then v cannot be adjacent to another non monitor in T. Assume the contrary. It is easy to observe that if at least one vertex $v \notin M$ has degree equal to 3 and it is adjacent to another vertex $u \notin M$ in T, then there exist some edges not monitored (see Figure 7.10(a) and 7.10(b)).
(2) If $deg(v) \geq deg(u) > 3$, then u may be adjacent to v in T but not to a third vertex $w \in V(T)\backslash M$. Figure 7.10(c) and 7.10(d) illustrate this case. We can easily see that we can admit two adjacent non monitors in T but if we have three non monitors, we have an edge that cannot be monitored.

(3) If $deg(x) = 2$ in T or $deg(x) > 2$ in T and x is adjacent to at least $deg(x) - 2$ leaves (same for y), then u and v cannot form the path with x and y in T as follow u, x, y, v. Assume the contrary. This implies that there exist four vertices $\{u, v, x, y\}$ forming a path in T such that $\{u, v\} \in V(T) \backslash M$ and $\{x, y\} \in M$ and $deg(x) = deg(y) = 2$ in T or the vertices x (or y) have at least $deg(x) - 2$ (or $deg(y) - 2$) leaves. The Figure 7.11 shows that in these two subcases, there exists an edge that cannot be monitored. This contradicts our assumption and complete the proof.

Figure 7.11: A counterexample to prove the condition (3) of Lemma 7.6.12. A non monitor vertex is represented by white vertex and monitor by black vertex. The red edges are the edges that can only be dominated by a white vertex.

□

From Proposition 7.6.10, we deduce that there exist $\gamma_m(T^2, 1)$-sets without any vertices in V_1. From Lemma 7.6.11, we can observe that all vertices in V_2 must be monitors. The vertices in V_3 need to satisfy some conditions, see Lemma 7.6.12. The idea is to construct the graph $G = (V, E)$ such that $V = V_3(T)$ be the set of vertices and E be the set of edges consisting on all the edges in $E(T)$ having both endpoints in $V_3(T)$ and we add an edge between two vertices $\{u, v\} \in V_3$ if these vertices form a path as follow: u, x, y, v such that $\{x, y\} \in V_2$ and $deg(x) = 2$ in T or $deg(x) > 2$ in T and x is adjacent to at least $deg(x) - 2$ leaves (same for y). Note that if both vertices u and v have degree bigger than 3 then we give $weight = 2$ to the added edge. All the other edges of the graph have $weight = 1$. The idea of adding some edges in the graph help us to keep the condition (3) of Lemma 7.6.12. It is easy to see that the graph G constitutes a forest of trees.

We first discuss how we can find the minimum number of monitors from this forest. By Lemma 7.6.12, we deduce that we cannot have two adjacent non monitors except in some conditions and we cannot have more then two non monitors constituted a path in T. The conditions of Lemma 7.6.12 will be illustrated as a

new problem that we call Mixed $\{1,2\}$-Independent Set in a coloring forest.

We begin by proposing a coloring scheme for the vertices of the forest. The vertices are colored as follows:

1. Red vertices V_r are the set of vertices having degree equals to 3 in T.
2. Blue vertices V_b are the set of vertices having degree higher than 3 in T.

The definition of the Mixed $\{1,2\}$-Independent Set problem is based on an existing definition of the k-Independent set given by Fink et al. in [FJ85]:

Definition 11 *A subset S of V is a k-independent if the maximum degree of the subgraph induced by the vertices of S is less or equal to $k-1$.*

In the following, the degree of a vertex is represented by the sum of weights of its incident edges.

Now, let define the Mixed $\{1,2\}$-Independent Set problem:

MIXED $\{1,2\}$-INDEPENDENT SET

Input: A tree graph $T' = (V, E)$, a partition (V_r, V_b) of V.

Output: A subset $S = S_r \cup S_b$ of $V = V_r \cup V_b$ is a mixed $\{1,2\}$-independent set if it is a 2-independent set such that S_r is a 1-independent set.

Maximum Mixed $\{1,2\}$-Independent Set is to find the largest mixed $\{1,2\}$-independent set for a given graph T'. We first give an algorithm that finds a minimum set of monitors that satisfy conditions of Lemma 7.6.12. The proposed algorithm looks for a variant of the Maximum Mixed $\{1,2\}$-Independent Set in a colored tree. It will be applied to each tree of the forest. In other words, the following algorithm returns for each tree of the forest a maximum mixed $\{1,2\}$-independent set S that represents vertices that are not monitors. Hence, finding S induces finding the minimum set of monitors by complementarity.

Before presenting the algorithm, let's present a well-known algorithm for the maximum independent set problem. We will show that for a tree $T = (V, E)$, using dynamic programming, the maximum 1-independent set (*a.k.a.* independent set) problem can be solved in linear time:
Root the tree at an arbitrary vertex. Then, each vertex defines a subtree (the one hanging from it). Dynamic programming proceeds from smaller to larger subprob-

lems then bottom up in the rooted tree (induced by depth first search (DFS in short)).

Suppose that we know the size of the largest independent set, denoted by MIS, of all subtrees below a node v. What is the maximum independent set in the subtree hanging from v? Two cases are possible: v is in the maximum independent set or is not.

Let $O(v)$ be the size of the maximum independent set in the subtree rooted at vertex v such that $v \notin MIS$, then the maximum independent set is the union of the maximum independent sets of the subtrees of the children of v. Let $I(v)$ be the size of the maximum independent set in the subtree rooted at vertex v such that $v \in MIS$, then the maximum independent set consists of v added to the union of the maximum independent sets of the subtrees of the children of v such that they do not belong to the maximum independent set. It follows that the time complexity of such algorithm is linear $\mathcal{O}(|V|)$.

Let's define an algorithm for the mixed $[1,2]$-independent set based on the same idea of the above algorithm. Let $T' = (V, E)$ be a tree with $V = V_r \cup V_b$ meaning the vertices are either colored red or blue. Suppose that we know the size of the largest mixed $[1,2]$-independent set, denoted by $MMIS$, of all subtrees below a node v. We have two possible cases:
1. Let $O(v)$ be the size of the maximum independent set in the subtree rooted at vertex v such that $v \notin MMIS$, then the maximum independent set is the union of the maximum independent sets of the subtrees of the children of v.
2. Let $I(v)$ be the size of the maximum independent set in the subtree rooted at vertex v such that $v \in MMIS$. Let $c(v)$ be the set of children of v. Then $c(v) = c_r(v) \cup c_{b1}(v) \cup c_{b2}(v)$ where $c_r(v)$ is the set of children of v having color red. $c_{b1}(v)$ is its children of color blue and connected to v with an edge of weight 1. $c_{b2}(v)$ is its children of color blue and connected to v with an edge of weight 2. The algorithm is obtained as follows:

It is easy to see that this algorithm is of linear complexity $\mathcal{O}(|V|)$.

Using the structure of an optimum solution described by lemmas and propositions of this section as well as the previous algorithm, we present a linear-time algorithm for solving the Edge monitoring problem on square tree graphs.

Algorithm 7.1: Compute the mixed $\{1,2\}$-Independent Set in trees

Input: A tree $T' = (V, E)$ rooted at r, a partition (V_r, V_b) of V.
Output: The mixed $\{1,2\}$-Independent Set in T'.
Begin
  For each vertex $v \in V$ in DFS order **do**
    $$O(v) = \sum_{u \in c(v)} max(I(u), O(u));$$
    If $v \in V_r$ **then**
      $$I(v) = 1 + \sum_{u \in c(v)} O(u);$$
    Else then
      $$I(v) = 1 + \sum_{u \in c_r(v) \cup c_{b2}(v)} O(u) + \sum_{u \in c_{b1}(v)} max(I(u), O(u));$$
    End If
  End For each
  Return $max(I(r), O(r))$;
End

As described bellow, the vertices set of the tree T can be decomposed into three subsets V_1, V_2 and V_3. The algorithm 7.2 is divided into two steps. First, all vertices V_1 represents the leaves of the tree. We mark them as non monitors. All the vertices of V_2 (as described previously) are marked as monitors. Note that the $deg_T(v)$ is the corresponding degree of the vertex $v \in V(T)$ and not in T^2. The step 2 of the algorithm considers only the vertices V_3 (*i.e.* the remaining vertices). First, it applies the proposed coloring scheme on the vertices V_3 of the tree T. Some edges are added to coincide with the properties of Lemma 7.6.12. This step will require to call Algorithm 7.1 in order to find the maximum $[1,2]$-independent set and then find the minimum monitoring set on the forest induced by vertices of V_3.

We conclude that Algorithm 7.2 computes a minimum set of monitors for any square of a tree of order $n > 4$ in linear time.

Thus, we have the following theorem.

Algorithm 7.2: Compute $\gamma_m(T^2, 1)$-set for a tree square T^2

Input: A tree $T = (V, E)$.

Output: $\gamma_m(T^2, 1)$-set.

Step 1:

Compute the set of leaves L;

For each $v \in L$ **do** $color(v) = white$;

For each vertex $v \in V(T)$ **do**

If $deg_T(v) = 2$ or $|N(v) \cap L| = deg_T(v) - 2$ or $\exists u \in L :\ dist(u, v) = 2$ or $\exists u \in L :\ dist(u, v) = 3$ and a path v, x, y, u between v and u with $|N(x) \cap L| = deg_T(x) - 2$ (Cf. Lemma 7.6.11) **then**

$color(v) = black$;

Put v in $\gamma_m(T^2, 1)$-set;

End If

End For each

Step 2:

The induced subgraph by the remaining uncolored vertices forms a forest F;

For each $v \in F$

If $deg_T(v) = 3$ **then** $color(v) = Red$;

Else $color(v) = Blue$;

End If

For each u in F **do**

If there exist x and y with $deg_T(x) = 2$ or $deg_T(x) > 2$ and x adjacent to at least $deg_T(x) - 2$ leaves (same for y) and u, x, y, v form a path in F (Cf. Lemma 7.6.12**(3)**) **then**

Add an edge with weight 2 between the two vertices v and u in F;

End If

End For each

End For each

Apply Algorithm 7.1 on each tree of the forest;

Let $S(F)$ be the union of the maximal mixed $\{1, 2\}$-independent set of each tree of the forest;

For each $v \in F$ **do**

If $v \in S(F)$ **then**

$color(v) = white$;

Else

$color(v) = black$;

Put v in $\gamma_m(T^2, 1)$-set;

End If

End For each

Return $\gamma_m(T^2, 1)$-set

End

Theorem 7.6.13 *There exists an algorithm which computes the minimum number of monitors for square tree in linear time.*

7.6.6 Power of cycles

Now, before presenting some results about the edge monitoring number of the cycle power graph, we detail two lemmas that can be useful in our proofs.

Let u and v be two vertices in C_n. Let $d_e(uv)$ be the distance between two vertices which is measured by the number of vertices in the shortest path connecting these two vertices in C_n (not in C_n^m). Let call the edge that connects u and v in C_n^m within distance $d_e(uv) = m$ in C_n a *long edge* and call the portion of vertices and edges between u and v in C_n^m, the path P_{uv} of e. It is clear that the long edge uv can be monitored by any vertex of the path P_{uv}. By the following lemma, we prove that for each portion of C_n having $2m-1$ vertices, we need at least three monitors.

Lemma 7.6.14 *Let C_n^m be the m^{th} power of a cycle graph of order n. Let P_{2m-1} be a sequence of $2m-1$ vertices of C_n. Let S be a $\gamma_m(C_n^m, 1)$-set of C_n^m. We have $|V(P_{2m-1}) \cap S| \geq 3$.*

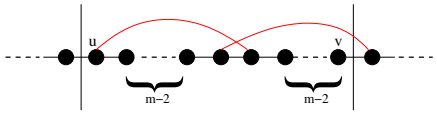


Figure 7.12: A sequence of $2m-1$ vertices of C_n. The red edges represent the two long edges.

Proof. Let $P_{2m-1} = uv$ be a sequence of $2m-1$ vertices of C_n. This sequence has two long edges that need to be monitored only by nodes in P_{2m-1} as depicted in Figure 7.12. Consequently, we have two monitors. Let call them m_1 and m_2 from left to right. As mentioned above, these two long edges can be monitored by any vertex of their paths.

Let $x = d_e(m_1m_2)$ be the distance between the two monitors in C_n. Let $x_1 = d_e(um_1)$ be the distance between u and m_1 and $x_2 = d_e(m_2v)$ the distance between m_2 and v. Now, let us prove that whatever the placement of the two monitors in P_{2m-1}, we will need at least a third monitor in P_{2m-1}. We have four cases:

(1) $x = 1$: this implies that the two monitors are adjacent in C_n then we have only one possibility as depicted in Figure 7.13(a). The red edge is an edge monitored only by a vertex in P_{2m-1}. Thus we need a third monitor.
(2) $x = m$: the long edge, depicted in red in Figure 7.13(b), can be monitored only by a monitor in P_{2m-1}.
(3) $x > m$: this means that we have at least the two long edges, as shown in Figure 7.13(c), are monitored only by a vertex in P_{2m-1}.
(4) $1 < x < m$: we have two possible configurations depending on the distances x_1 and x_2.

- If $x + x_2 \geq m$ and $x + x_1 \geq m$ then the edge having extremities the two monitors need to be monitored by a vertex in P_{2m-1} as depicted in Figure 7.13(d).

- If $x + x_2 < m$ then $x_1 > m$. It implies that the long edge depicted in red in Figure 7.13(e) need to be monitored by a vertex in P_{2m-1} (the case when $x + x_1 < m$ is symmetric to this case).

□

Lemma 7.6.15 *Let C_n^m be a power cycle graph of order n. If $n = 2m + 2$ then $C_n^m = K_n - I$ and for any $n \leq 2m + 1$, then $C_n^m = K_n$.*

Proof. Let C_n^m be a cycle graph power m of order n. For $n \leq 2m + 1$: if we prove that $C_{2m+1}^m = K_{2m+1}$, this implies that for any $n \leq 2m + 1$, $C_n^m = K_n$. Assume, to the contrary, that $C_{2m+1}^m \neq K_{2m+1}$. This means that there exist at least two nonadjacent vertices in C_{2m+1}^m. From the definition of graph power, each vertex of C_{2m+1}^m will be connected to all vertices having distance less or equal to m. Since we have $2m + 1$ vertices, each vertex will be connected to $2m$ vertices (m vertices on its left and m on its right). Thus, this contradicts the supposition that $C_{2m+1}^m \neq K_{2m+1}$.
For $n = 2m + 2$: similarly to the previous case, we can easily prove that C_{2m+2}^m is a complete graph minus one factor. □

Theorem 7.6.16 *The cycle power C_n^m has*

$$\gamma_m(C_n^m, 1) = \begin{cases} 3 & \text{if } n \leq 3m \\ n & \text{if } m = 2 \text{ and } n > 3m \\ 3\lfloor \frac{n}{2m-1} \rfloor + c & \text{otherwise.} \end{cases}$$

with

$$c = \begin{cases} 0 & \text{if } n \equiv 0[2m-1] \\ 1 & \text{if } n \equiv 1[2m-1] \\ 2 & \text{if } n \equiv k[2m-1] \text{ with } 2 \leq k \leq m \\ 3 & \text{otherwise.} \end{cases}$$

.

Proof. We distinguish four cases as follows:

Case 1. $n \leq 2m + 1$:

By Lemma 7.6.15 we have $C_n^m = K_n$ and using Proposition 7.3.1 we have $\gamma_m(C_n^m, 1) = 3$.

Case 2. $n = 2m + 2$:

Using Lemma 7.6.15 and Proposition 7.3.1, we have $C_n^m = K_n - I$ then $\gamma_m(C_n^m, 1) = 3$.

Case 3. $2m + 2 < n \leq 3m$:

Let decompose C_n^m into three components, as illustrated in Figure 7.14, called $Comp_1$, $Comp_2$ and $Comp_3$ as follow: (a) The first two components $Comp_1$ and $Comp_2$ have the same number of vertices $m - 1$ and a common vertex, denoted by v_i. (b) The third component $Comp_3$ is composed from the remaining vertices of the cycle and has two common vertices v_j and v_k with $Comp_1$ and $Comp_2$ respectively. Note that the vertices v_i, v_j and v_k form a triangle. We prove that these three vertices can monitor all the edges of C_n^m. We show that each vertex can monitor a set of specific edges. Consider the first monitor v_i and the proof is the same for the other monitors v_j and v_k.

The monitor v_i monitors the following edges:

1. All edges having the two extremities in the adjacent components of v_i: one extremity in $Comp_1$ and the other one in $comp_2$ such that v_i is not an extremity of these edges.

2. All edges having the both extremities in the same adjacent component $Comp_1$ or $Comp_2$ such that v_i is not an extremity of these edges.

3. The edge having as extremities the two other monitors v_j and v_k.

Case 4. $n > 3m$:

Subcase 4.a. $m > 2$:

Since each monitor must be adjacent to two monitors (from Observation 7.5.2), this means that the set of monitors need to form a set of triangles. Thus, the aim is to find the minimum triangles set that covers all the edges in order to have the smallest number of monitors. In this perspective, we define a method of monitors selection as follow: first, we choose randomly one vertex as monitor, this vertex can monitor exactly $m-1$ long edges from its right and its left(C_n^m is symmetric) and all edges having each extremity in distance $d_e = m$ with this monitor except edges having this monitor as extremity (see example of Figure 7.15.step 1). Then, in order to monitor the edges having the first monitor as extremity and more precisely the two long edges e_1 and e_2 incident from this monitor (see Figure 7.15.step 2), we need to choose one monitor which belongs to the path P_{e_1} (resp. to e_2). Consequently, we choose as monitor the vertex having distance $m-1$ from the first monitor chosen because it's the farthest vertex which can monitor the edge e_1 (the same for e_2 because of the symmetry). Since the distance between the first monitor and the second one is $m-1$, then the second monitor can monitor all the edges having the first monitor as end (same for the other part because of the symmetry) except the edge having two monitors as end. Consider e_3 and e_4 the edges having both ends as monitors. Then, in order to monitor these edges, we choose the farthest vertices that can monitor them. Thus, we choose the adjacent vertex to the second monitor as monitor (same for the other part) (as shown in Figure 7.15.step 3). Consequently, using this method, we are sure to have the minimum number of monitors for this part of the cycle then we just need to do the same for the rest. It means that to monitor all edges, one time we choose one vertex as monitor and after $m-2$ vertices, we need to choose two monitors and so on. Using Lemma 7.6.14, we deduce that this construction gives an optimal solution.

Subcase 4.b. $m = 2$:

Using Observation 7.5.3, we can easily see that all vertices must be monitors to monitor all the edges. □

7.6.7 Power of paths

The idea of proof for powers of a path is the same as the proof of powers of a cycle.

Theorem 7.6.17 *The path power P_n^m has*

$$\gamma_m(P_n^m, 1) = \begin{cases} 3 & \text{if } n \leq 2m+1 \\ 4 & \text{if } 2m+1 < n \leq 3m \\ n-2 & \text{if } n > 3m \text{ and } m = 2 \\ 3\lfloor \frac{n-2m-2}{2m-1} \rfloor + 4 + c & \text{otherwise.} \end{cases}$$

with

$$c = \begin{cases} 0 & \text{if } n-2m-2 \equiv k[2m-1] \text{ with } 0 \leq k \leq m-2 \\ 1 & \text{if } n-2m-2 \equiv k[2m-1] \text{ with } m-1 \leq k \leq 2m-3 \\ 2 & \text{otherwise.} \end{cases}$$

.

Proof. We distinguish three cases as follows:

Case 1. $n \leq 2m+1$:

In this case, we prove that three monitors are sufficient to monitor all the edges of the power of path, denoted by P_n^m. The idea is to find the triangle whose vertices can monitor all edges of the power of path. For example, if we choose three successive monitors from the center of the path, then each monitor monitors all edges in $m-1$ distance from it left and it right, except the edges having this monitor as end. Moreover, each monitor monitors the edge connected by the two other monitors. Thus, three monitors are enough.

Case 2. $2m+1 < n \leq 3m$:

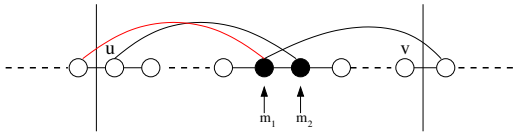
We can easily verify that three monitors are not enough to monitor all edges. Then, we prove that four monitors are sufficient. We divide the path into three components. The two components of extremity (right and left) contain $m+1$ vertices each and the component of the middle contains the rest of vertices. We choose two adjacent monitors, from each component of extremity, in order to monitor all edges in distance $m-1$ from the end of the path. Since the number of vertices that has the component of the middle is at most $m-2$, we are sure that two two vertices of each component form a triangle with at least one vertex from the other component and all the edges are monitored.

Case 3. $n > 3m$:

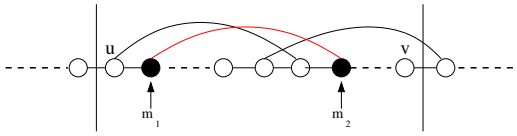
If $m = 2$, then it is easy to see that we need to choose all vertices as monitors except the two vertices of extremities.

If $m > 2$, we divide the path into three components as in Case 2. The two components of extremity contain $m-1$ vertices and the third component contain the rest of vertices. In order to monitor all the edges of the components of extremity, we must choose two adjacent monitors from each extremity of the middle components as shown in Figure 7.16(step 1). To monitor the edges of the middle component, we define the following method to select monitors: the idea is to find the minimum number of triangles that cover all the remaining edges to have the minimum number of monitors. We know that the largest triangle in the path power P_n^m is extended over a distance of $m+1$. This means that the biggest distance between two monitors in the same triangle is $m-2$. This implies that after each $m-2$ vertices we must have at least one monitor. In order to have the minimum number of monitors, after each $m-2$ vertices, we choose one monitor and the second time two adjacent monitors and so on (See Figure 7.16) in order to constitute our triangles. It permits to have succession of triangles connected along the path (same idea as cycle power C_n^m for $n > 3m$ and $m > 2$). Using Lemma 7.6.14, we can easily deduce that this construction gives an optimal solution.

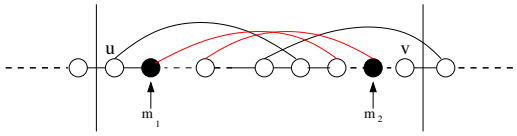
This complete the proof. □



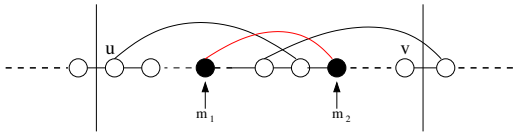
(a)



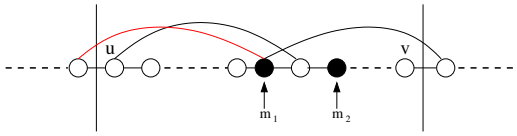
(b)



(c)



(d)



(e)

Figure 7.13: The different cases depending on the placement of the two monitors m_1 and m_2 in P_{2m-1}.

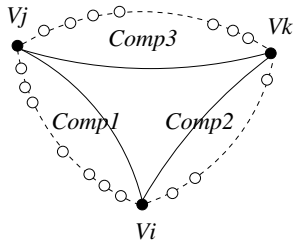


Figure 7.14: Cycle power C_n^m for $2m + 2 < n \leq 3m$.

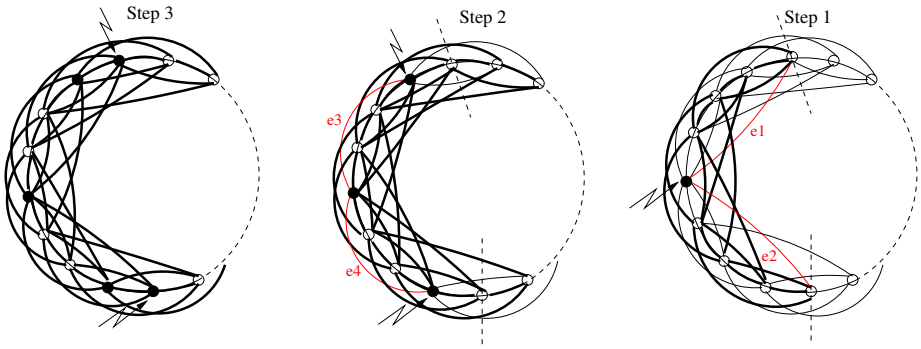


Figure 7.15: An example of monitors selection in cycle power C_n^m for $n > 3m$ and $m = 4$.

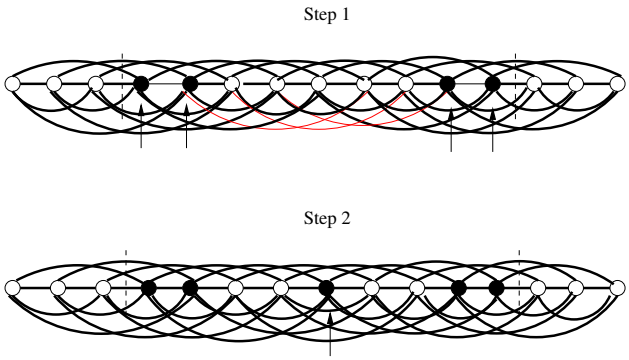


Figure 7.16: The minimum set of monitors on the path P_{15}^4.

Chapter 8

Parameterized algorithms and complexity

Contents

In this chapter we review some basic notions related to parameterized complexity. Then, we focus in particular on 1-uniform monitoring problem (1-UMP) by proving that the problem in NP-complete even when restricted to (P_5, C_4)-free 1-monitorable graphs. We also prove that 1-uniform Edge Monitoring cannot be approximated within $(1-\varepsilon)\ln|V|$ for any $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{O(\log\log n)})$. Afterthat, we prove that the 1-uniform edge monitoring problem is $W[2]$-hard when parameterized by the size of the solution. Then, we present two algorithms that solve a more general problem, namely Edge Monitoring. The first one solves the version of the problem parameterized by the treewidth in time $2^{\mathcal{O}(\mathbf{tw}^2\cdot\log(\max_{e\in E} c(e)))}\cdot n$ where $\mathbf{tw}$ is the treewidth of the input graph and $c : E \to \mathbb{N}$ is a color function such that each edge e should be monitored $c(e)$ times. The second one solves the version of the problem parameterized by k, the size of the solution, in time $2^{\mathcal{O}(\sqrt{k}\cdot\log(\max_{e\in E} c(e)))}\cdot n$ when the input graph is apex-minor-free, in particular, when it is planar, by using Bidimensionality Theory [DH08, DFHT04, DHT04]. We also

prove that the edge monitoring problem is $W[1]$-complete on complete graphs when the parameter is the size of the solution.

8.1 Preliminary notions

In this section we introduce some basic definitions.

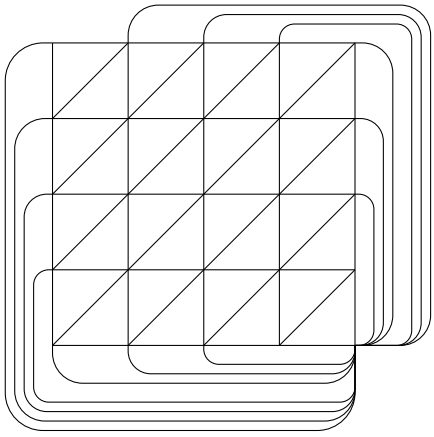


Figure 8.1: The triangulated grid Γ_5.

Let k be an integer. The *triangulated grid* of size k is the graph $\Gamma_k = (V_k, E_k)$ such that $V_k = \{\ell_{i,j} | 1 \leq i, j \leq k\}$ and $E_k = \{\{\ell_{i,j}, \ell_{i+1,j}\} \mid 1 \leq i \leq k-1, 1 \leq j \leq k\} \cup \{\{\ell_{i,j}, \ell_{i,j+1}\} \mid 1 \leq i \leq k, 1 \leq j \leq k-1\} \cup \{\{\ell_{i,j+1}, \ell_{i+1,j}\} \mid 1 \leq i \leq k-1, 1 \leq j \leq k-1\} \cup \{\{\ell_{1,j}, \ell_{k,k}\}, \{\ell_{k,j}, \ell_{k,k}\} \mid 1 \leq j \leq k\} \cup \{\{\ell_{i,1}, \ell_{k,k}\}, \{\ell_{i,k}, \ell_{k,k}\} \mid 1 \leq i \leq k\}$. Note that Γ_k is triangulated. For an illustration, the graph Γ_5 is depicted in Figure 8.1. If $i_0, j_0 \in \{1, \ldots, k-1\}$, we call *the square* (i_0, j_0) of Γ_k the set $\{\ell_{i_0,j_0}, \ell_{i_0+1,j_0}, \ell_{i_0,j_0+1}, \ell_{i_0+1,j_0+1}\}$ and the *diagonal* (i_0, j_0) the edge $\{\ell_{i_0+1,j_0}, \ell_{i_0,j_0+1}\}$.

Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two graphs. We say that H is a *contraction* of G if we can partition V_G into $|V_H|$ sets $(R_u)_{u \in V_H}$ such that for all $u \in V_H$, R_u is not empty and $G[R_u]$ is connected, and such that $\{u_1, u_2\} \in E_H$ if and only if there exist $v_1 \in R_{u_1}$ and $v_2 \in R_{u_2}$ such that $\{v_1, v_2\} \in E_G$.

Treewidth A *tree-decomposition* of width w of a graph $G = (V, E)$ is a pair $(\mathcal{T}, \sigma)$, where $\mathcal{T}$ is a tree and $\sigma = \{B_t | B_t \subseteq V, t \in V(\mathcal{T})\}$ such that:

- $\bigcup_{t \in V(\mathcal{T})} B_t = V$,

- For every edge $\{u, v\} \in E$ there is a $t \in V(\mathcal{T})$ such that $\{u, v\} \subseteq B_t$,
- $B_i \cap B_k \subseteq B_j$ for all $\{i, j, k\} \subseteq V(\mathcal{T})$ such that j lies on the path between i and k in $\mathcal{T}$, and
- $\max_{i \in V(\mathcal{T})} |B_t| = w + 1$.

A tree-decomposition rooted at a node t_r is *nice* if the following conditions are fulfilled:

- $B_{t_r} = \emptyset$,
- each node has at most two children,
- for each leaf $t \in V(\mathcal{T})$, $B_t = \emptyset$,
- if $t \in V(\mathcal{T})$ has exactly one child t', then either
  - $B_t = B_{t'} \cup \{v\}$ for some $v \notin B_{t'}$ and this node is called an *introduce vertex*, or
  - $B_t = B_{t'} \setminus \{v\}$ for some $v \in B_{t'}$ and this node is called a *forget vertex*, and
- if $t \in V(\mathcal{T})$ has exactly two children t' and t'', then $B_t = B_{t'} = B_{t''}$. This node is called a *join vertex*.

The sets B_t are called *bags*. The *treewidth* of G, denoted by $\mathbf{tw}(G)$, is the smallest integer w such that there is a tree-decomposition of G of width w. When context is clear we will use the notation $\mathbf{tw}$ instead of $\mathbf{tw}(G)$. An *optimal tree-decomposition* is a tree-decomposition of width $\mathbf{tw}(G)$. Moreover, if we have a tree-decomposition, then we can build a nice tree-decomposition of G with the same width in polynomial time [Klo94a].

8.2 Complexity of 1-uniform monitoring problem

8.2.1 Algorithmic complexity and approximability

We first study 1-UMP for P_4-free graphs, also called cographs. We recall the following theorem from Pim, *et al.*

Theorem 8.2.1 *[PvD10] A graph G is P_4-free iff. each connected induced subgraph of G contains a dominating induced C_4 or a dominating vertex.*

The previous theorem is a better reformulation of Wolk's result.

Theorem 8.2.2 *[Wol62] A graph G is (P_4, C_4)-free iff. each connected induced subgraph of G contains a dominating vertex.*

We deduce an algorithm that computes $\gamma_m(G, 1)$ of (P_4, C_4)-free 1-monitorable graphs as follows:

Corollary 8.2.3 *Let G be (P_4, C_4)-free 1-monitorable graph. There exists an algorithm that computes $\gamma_m(G, 1)$ in polynomial time.*

Proof. Let S be the monitoring set to be computed. We begin with $S = \emptyset$. Let u be a vertex dominating all other vertices of G. Determining such vertex is easy. Add u to S. Now, consider the graph $G - u = G_1 \cup \cdots \cup G_k$ where the G_i's are connected components. Then for all G_i's do the following. Let v_i be a dominating vertex of G_i. Add v_i and one of its neighbors to S. □

In the following, we will prove that 1-UMP is NP-complete on (P_5, C_4)-free graphs.

Theorem 8.2.4 *1-UMP is NP-complete even when restricted to (P_5, C_4)-free 1-monitorable graphs.*

Proof. Since it is possible to check a candidate solution of 1-UMP in polynomial time, 1-UMP belongs to NP. To prove its NP-hardness, we give a reduction from the total dominating set problem that was proven to be NP-complete in split graphs by Bartossi in 1984 [Ber84].

Let $G = (K_n, I_m)$ be a split graph where K_n is a clique of n vertices and I_m is an independent set of m vertices. Without loss of generality we consider only split graphs with minimum degree $\delta(G) \geq 2$, for otherwise G wouldn't be 1-monitorable.

We construct a graph G' from the split graph G by replacing each vertex of the independent set I_m with a K_2, and call the set of obtained vertices I'. G' is a (P_5, C_4)-free 1-monitorable graph. We prove that G admits a total dominating set of size at most k *iff.* G' admits a monitoring set of size at most $2k$.

Assume G admits a total dominating set S of size $k \geq 3$. If S contains a vertex of I_m then replace it by any of its neighbors in K_n. The obtained set is still a total dominating set of G. Then we construct S' a monitoring set of G' from S as follows. For every K_2 of I', if it is adjacent to only one vertex of S then add in S' another of its neighbors. This is always possible since $\delta(G) \geq 2$. Observe that S' is a monitoring set of G' such that $|S'| \leq 2k$.

Now, assume G' admits a monitoring set S' of $2k$ vertices. Let $S' = S_1 \cup S_2$ such that $S_1 \cap S_2 = \emptyset$ and $S_1 \subset K_n$ is the smallest set of vertices sufficient to monitor I'. We prove $S_1 \leq S_2$. Suppose $S_1 > S_2$. Construct S_2' by replacing every vertex of S_2 that belongs to I' by one of its neighbors in K_n. Observe S_2' is sufficient to monitor I'. A contradiction. Thus S_1 is a total dominating set of G such that $S_1 \leq k$. □

In the following, we study relationship between 1-UMP and other problems such as Hitting Set Problem (HSP), Set Cover Problem (SCP) and Vertex Cover Problem (VCP) in hypergraphs [Kar72].

The Hitting Set Problem (HSP) is defined as follows: Let S be a finite set, C a collection of subsets $C_1, C_2, \ldots, C_m$ of S and k a positive integer. The couple (S, C) has a hitting set of size at most k, *iff.* $\exists X \subseteq S : |X| \leq k$ and $\forall C_i \in C, C_i \cap X \neq \emptyset$. The decision problem associated with HSP has been proven to be NP-complete [Kar72].

Theorem 8.2.5 *1-UMP is turing reducible to HSP, 1-UMP $\leq_T$ HSP.*

Proof. We give a reduction that solves 1-UMP, assuming the algorithm solving HSP to be already known. Let graph $G = (V, E)$ a 1-monitorable and k a positive integer be an instance of 1-UMP. Create an instance (S, C, k) of HSP as follows: $S = V$ and the collection $C = \{M(e_1), \ldots, M(e_m)\}$ such that $E = \{e_1, \ldots, e_m\}$. □

Nevertheless, from Theorem 8.2.5 all results and approximation algorithms that deal with HSP [SC10, AvG09, LY02] can be used to have similar results for 1-UMP. Moreover, HSP, SCP and VCP on hypergraphs are equivalent problems. This implies that 1-UMP is a special case of all the three problems. Hence all the result for SCP [Hoc82b, Sla96] and VCP [Hal02, Kar05] could be considered and refined for 1-UMP.

Thus, we deduce:

Corollary 8.2.6 *1-UMP $\leq_T$ SCP and 1-UMP $\leq_T$ VCP.*

Theorem 8.2.7 *1-uniform* Edge Monitoring *cannot be approximated within* $(1-\varepsilon)\ln|V|$ *for any* $\varepsilon > 0$*, unless* NP $\subseteq$ DTIME$(n^{O(\log\log n)})$.

Proof. It has been proved in [CC08] that Total Dominating Set cannot be approximated within $(1-\varepsilon)\ln|V|$ for any $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{O(\log\log n)})$. We will define an approximation preserving reduction from Total Dominating Set to 1-uniform Edge Monitoring. Let $G = (V, E)$ be a graph without isolated vertex. We construct G' from G by adding three vertices u, v, w which form a clique and connecting u to every vertex in V. We will prove that $\gamma_m(G', 1) = \gamma_t(G) + 3$.

Let S be a total dominating set of G and $S' = S \cup \{u, v, w\}$. Then S' is a monitoring set of G'. Indeed, the edges uv, uw and vw are monitored by w, v and u respectively. The edges in E are monitored by u. Let x be a vertex in V then x has a neighbor y in S. Thus, ux is monitored by y.

Now, let S be a monitoring set of G'. $\{u, v, w\} \subseteq S$. Otherwise, uv, vw or uw is not monitored by S. Let $S' = S \setminus \{u, v, w\}$. We will prove that S' is a total dominating set of G. Let x be a vertex of G. The edge xu is monitored by a vertex y in S'. Since $\{x, y, u\}$ forms a triangle, x is adjacent to a vertex in S'. Hence, $\gamma_m(G', 1) = \gamma_t(G) + 3$.

Using the same method as in Theorem 1 of [KL04] we obtain the desired result. □

8.2.2 $W[2]$-hardness of 1-uniform edge monitoring problem

Now, we show that the problem is $W[2]$-hard when parameterized by the size of the solution. In order to prove that, we reduce from Red-Blue Dominating Set, which is known to be $W[2]$-hard [DF13].

Red-Blue Dominating Set

Input: A graph $G = (V, E)$, a partition (V_r, V_b) of V, and an integer k.

Output: A set $S \subseteq V_b$ of size at most k such that $\forall v \in V_r$, $S \cap N(v) \neq \emptyset$.

Theorem 8.2.8 1-UNIFORM EDGE MONITORING *is* $W[2]$*-hard parameterized by the size of the solution.*

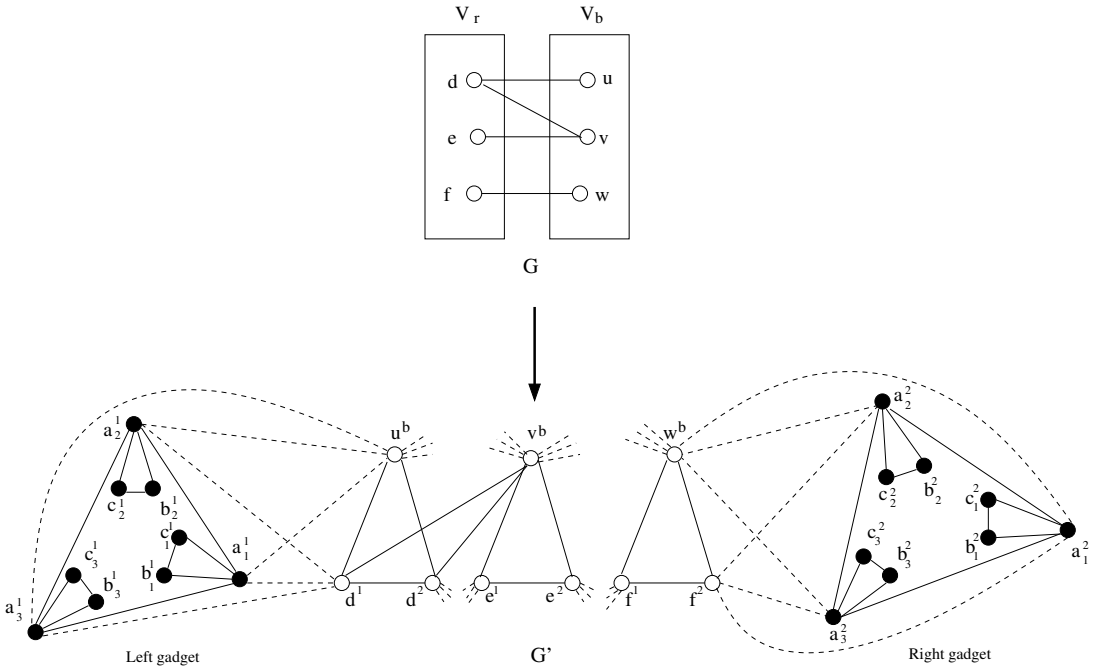


Figure 8.2: EDGE MONITORING gadget. For readability, some edges are drawn as dotted and for some of them, only one extremity is drawn. In the figure, the vertices u^b, v^b, w^b, d^1, e^1, and f^1 are connected to the three vertices a_1^1, a_2^1, and a_3^1 like u^b and d^1 are, and the vertices u^b, v^b, w^b, d^2, e^2, and f^2 are connected to the three vertices a_1^2, a_2^2, and a_3^2 like w^b and e^2 are.

Proof. Let $G = (V, E)$ be a graph, let (V_r, V_b) be a partition of V, and let k be an integer. We want to solve RED-BLUE DOMINATING SET on (G, V_r, V_b, k). Without lost of generality, we can assume that there is no isolated vertex.

We construct from (G, V_r, V_b, k) the graph $G' = (V', E')$ as depicted in Figure 8.2. Formally, $V' = V'_b \cup V'_e \cup V_a$ where $V'_b = \{v^b | v \in V_b\}$, $V'_e = \{v^1 | v \in V_r\} \cup \{v^2 | v \in V_r\}$, $V_a = \{a_j^i, b_j^i, c_j^i | i \in \{1,2\}, j \in \{1,2,3\}\}$, and $E' = \{\{v^1, v^2\} | v \in V_r\} \cup \{\{v^b, w^1\} | \{v, w\} \in E\} \cup \{\{v^b, w^2\} | \{v, w\} \in E\} \cup \{\{a_j^i, v^i\} | i \in \{1,2\}, j \in \{1,2,3\}\} \cup \{\{a_j^i, v^b\} | i \in \{1,2\}, j \in \{1,2,3\}\} \cup \{\{a_j^i, a_{j'}^i\} | i \in \{1,2\}, j, j' \in \{1,2,3\}, j \neq j'\} \cup \{\{a_j^i, b_j^i\}, \{a_j^i, c_j^i\}, \{b_j^i, c_j^i\} | i \in \{1,2\}, j \in \{1,2,3\}\}$.

We now show that solving RED-BLUE DOMINATING SET on (G, V_r, V_b, k) is equivalent to solving 1-UNIFORM EDGE MONITORING on $(G', k + 18)$. Let S be a

solution of Red-Blue Dominating Set on (G, V_r, V_b, k). Note that the existence of S implies that each vertex of V_r has at least one neighbor in V_b. Let $S' = \{v^b | v \in S\} \cup V_a$. Then S' is a solution of 1-uniform Edge Monitoring on $(G', k+18)$. Indeed, $|S'| \leq k + 18$ by definition of S and V_a. Let $e \in E'$. If $e = \{v^1, v^2\}$ with $v \in V_r$, then by definition of S, there exists $t \in S$ that is neighbor of v in G, so t^b monitors e in G'. If $e = \{v^b, w^1\}$ with $v \in V_b$ and $w \in V_r$, then a^1_1 monitors e. The same happens if $e = \{v^b, w^2\}$. If $e = \{a^i_j, v^i\}$ then $a^i_{(j \mod 3)+1}$ monitors e. As $\{a^i_1, a^i_2, a^i_3\}$ is a triangle where all the vertices are in S', all the edges are monitored. The same happens for the triangles $\{a^i_j, b^i_j, c^i_j\}$, $i \in \{1, 2\}$, $j \in \{1, 2, 3\}$.

Now let S' be a solution of 1-uniform Edge Monitoring on $(G', k+18)$. For each $i \in \{1, 2\}$ and $j \in \{1, 2, 3\}$, the edges $\{a^i_j, b^i_j\}$, $\{a^i_j, c^i_j\}$, and $\{b^i_j, c^i_j\}$ can be monitored only by the vertices c^i_j, b^i_j, and a^i_j respectively. So they need to be in S'. One can check that the only edges not monitored by V_a are the edges of the form $\{v^1, v^2\}$, and by construction of G' the only vertices that can monitor them are vertices from V'_b. It directly follows that $S = \{v \in V_b | v^b \in S'\}$ is a solution of Red-Blue Dominating Set on (G, V_r, V_b, k). □

8.3 Fixed parameter algorithms for edge monitoring

In this section, we will present algorithms that solve the Edge Monitoring problem. The first one is parameterized by the treewidth of the input graph and the second one, based on the first one, uses Bidimensionality to solve Edge Monitoring parameterized by the size of the solution when the input graph is apex-minor-free.

In this version, we allow only some selected monitors to be in the solution, and we impose that each edge is monitored by at least a given number of monitors.

From Theorem 8.2.8, we directly obtain the following.

Corollary 8.3.1 Edge Monitoring *is* $W[2]$*-hard parameterized by* k*.*

We now focus on the algorithms. First we present an FPT algorithm parameterized by the treewidth.

Lemma 8.3.2 *Let* $G = (V, E)$ *be a graph,* k *be an integer,* M *be a subset of* V*, and* $c : E \to \dots k$ *be a coloring of edges of* G*.* Edge Monitoring *on* (G, k, M, c) *can be solved in time* $2^{\mathcal{O}(\mathbf{tw}^2 . \log(\max_{e \in E} c(e)))} . n$*, where* **tw** *is the treewidth of* G*.*

Proof. Let $G = (V, E)$ be a triangulated graph, k be an integer, M be a subset of V, $c : E \to \dots k$ be a color function, and $(\mathcal{T}, \mu)$ be a nice tree-decomposition of G rooted at a node t_r of width **tw**.

For each $t \in V(\mathcal{T})$, we denote by V_t the set of vertices of all descendents of t, $G_t = G[V_t]$, and $E_t = E(G[B_t])$. Note that this graph may be disconnected.

We use a dynamic programming approach. The table we store at a node t will contain elements of the form (X, Y, p), where $X \subseteq B_t$ is the set of chosen vertices in B_t for this solution, $Y \subseteq E_t \times \mathbb{N}$ is the set of pairs (y, m) where the edge y still needs to be monitored m times in G_t, and p is the number of vertices we already have chosen. We will keep such an element in the table, if there exists a solution S of our problem of size at most k such that $S \cap B_t = X$, $|S \cap V_t| \leq p$, $S \cap V_t$ monitors all the edges of $E(G_t) \setminus \{y | \exists m \in \mathbb{N} : (y, m) \in Y\}$, and for each $(y, m) \in Y$, $S \cap V_t$ monitors $c(y) - m$ times the edge y. Formally, if $H = (V_h, E_h)$ is a graph, $B \subseteq V_h$, $X \subseteq B$, and $Y \subseteq E(H[B]) \times \{1, \dots, k\}$, we define $sol(H, B, X, Y, p, M) = true$, if and only if there exists a set $S \subseteq V_h \cap M$ of size at most p such that for each $(e, m) \in Y$, $|S \cap N(e)| = c(e) - m$, and for each $e \in E_h \setminus \{y | \exists m \in \mathbb{N} : (y, m) \in Y\}$, $|S \cap N(e)| = c(e)$, and $S \cap B = X$. Note that we add M as an argument of sol in order to obtain a function sol that is self-consistent. We define the table we store at each node $t \in V(\mathcal{T})$ to be $\mathcal{R}_t = \{(X, Y, p) | X \subseteq B_t, Y \subseteq E(G[B_t]) \times \{1, \dots, k\}, sol(G_t, B_t, X, Y, p, M), p \leq k\}$. Note that there is a solution of our problem if and only if $\mathcal{R}_{t_r} \neq \emptyset$. For convenience, if $(X, Y, p) \in \mathcal{R}_t$ and $(X, Y, q) \in \mathcal{R}_t$ with $p < q$ then our algorithm will keep only (X, Y, p), as the other entry is not relevant. Let $t \in V(T)$. We can compute $\mathcal{R}_t$ as follows:

- If t is a leaf then $G_t = (\emptyset, \emptyset)$ and $\mathcal{R}_t = \{(\emptyset, \emptyset, 0)\}$.

- If t is an introduce vertex v and $v \in M$, let t' be its child. Then $\mathcal{R}_t = \{(X \cup \{v\}, \{(y, m - |N(y) \cap \{v\}|) | (y, m) \in Y\} \cup \{(\{v, w\}, m') | w \in B_t, \{v, w\} \in E, m' = \max(c(\{v, w\}) - |N'\{v, w\} \cap X|, 0)\}, p+1) | (X, Y, p) \in \mathcal{R}_{t'}, p+1 \leq k\} \cup \{(X, Y \cup \{(\{v, w\}, m') | w \in B_t, \{v, w\} \in E, m' = \max(c(\{v, w\}) - |N'\{v, w\} \cap X|, 0)\}, p) | (X, Y, p) \in \mathcal{R}_{t'}\}$.

- If t is an introduce vertex v and $v \notin M$, let t' be its child. Then $\mathcal{R}_t = \{(X, Y \cup \{(\{v, w\}, m') | w \in B_t, \{v, w\} \in E, m' = \max(c(\{v, w\}) - |N'\{v, w\} \cap X|, 0)\}, p) | (X, Y, p) \in \mathcal{R}_{t'}\}$.

- If t is a forget vertex v, let t' be its child. Then $\mathcal{R}_t = \{(X \setminus \{v\}, Y \setminus \{(\{v,w\},0)|w \in B_t, (\{v,w\},0) \in Y\}, p)|(X,Y,p) \in \mathcal{R}_{t'}, \forall w \in X, m \in \{1,\ldots,k\} : (\{v,w\},m) \notin Y\}$. Note that if $v \notin X$ then $X \setminus \{v\} = X$.

- If t is a join vertex, let t' and t'' be its children. Then $\mathcal{R}_t = \{(X' \cup X'', \{(y,m)|(y,m') \in Y', (y,m'') \in Y'', m = c(y) - (c(y) - m') - (c(y) - m'') + |N(y) \cap (X' \cap X'')|\}, p' + p'' - |X' \cap X''|)|(X',Y',p') \in \mathcal{R}_{t'}, (X'',Y'',p'') \in \mathcal{R}_{t''}, p' + p'' - |X' \cap X''| \leq k\}$. Note that $(c(y) - m')$ (resp. $(c(y) - m'')$) is the number of times y has been monitored in $G_{t'}$ (resp. $G_{t''}$).

For all $t \in V(T)$, if $(X,Y,p) \in \mathcal{R}_t$ then $X \subseteq B_t$ and $Y \subseteq E_t \times \{1,\ldots,\max_{e \in E} c(e)\}$. Note that if (y,m) and (y,m') are in Y with $m < m'$, then we need to keep only (y,m). So we can see Y as a subset of all functions $E_t \to \{1,\ldots,\max_{e \in E} c(e)\}$. We obtain that $|Y| \leq 2^{\mathbf{tw}^2 \cdot \log(\max_{e \in E} c(e))}$. Thus, $|\mathcal{R}_t| \leq 2^{\mathbf{tw}} \cdot 2^{\mathbf{tw}^2 \cdot \log(\max_{e \in E} c(e))}$. So we can solve Edge Monitoring on (G,k) in time $2^{\mathcal{O}(\mathbf{tw}^2 \cdot \log(\max_{e \in E} c(e)))} \cdot n$. □

If G is apex-minor-free, then, there exists a constant a, depending only on the apex-graph, such that $|E| \leq a|V|$ [Tho01]. In particular, it implies that in the previous complexity analysis, if G is apex-minor-free, then Y is of size at most $a|V| \cdot \log(\max_{e \in E} c(e))$. This directly gives the following lemma.

Lemma 8.3.3 *Let $G = (V,E)$ be an apex-minor-free graph, k be an integer, M be a subset of V, and $c : E \to \{1,\ldots,k\}$ be a color function.* Edge Monitoring *on (G,k,M,c) can be solved in time $2^{\mathcal{O}(\mathbf{tw} \cdot \log(\max_{e \in E} c(e)))} \cdot n$.*

Theorem 8.3.4 ([FGT11]) *There exists a constant c such that for every apex-minor-free graph G and every integer k such that $k \leq \frac{\mathbf{tw}(G)}{c}$, the triangulated grid Γ_k is a contraction of G.*

Theorem 8.3.5 *Let $G = (V,E)$ be an apex-minor-free graph, k be an integer, c be a color function $c : E \to \{1,\ldots,k\}$, and M be a subset of V.* Edge Monitoring *on (G,k) can be solved in time $2^{\mathcal{O}(\sqrt{k} \cdot \log(\max_{e \in E} c(e)))} \cdot n$.*

Proof. Let $G = (V,E)$ be an apex-minor-free graph and k be an integer. Assume first that $\mathbf{tw} > c(2\lceil\sqrt{(k+1)}\rceil + 2)$. By Theorem 8.3.4, $\Gamma_{(2\lceil\sqrt{(k+1)}\rceil+2)}$ is a contraction of G. Let $L = \{\ell_{i,j}|i,j \in \mathbb{N}, 1 \leq i,j \leq (2\lceil\sqrt{(k+1)}\rceil + 2)\}$ be the vertex

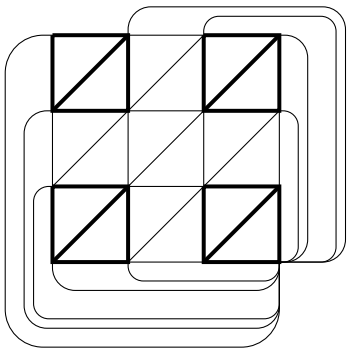


Figure 8.3: The considered squares in Γ_4 and their diagonals.

set of $\Gamma_{(2\lceil\sqrt{(k+1)}\rceil+2)}$, and let M be its edge set. Let $(R_u)_{u\in L}$ be a partition of V such that for all $u \in L$, R_u is not empty, $G[R_u]$ is connected, and such that $\{u_1, u_2\} \in E(\Gamma_{(2\lceil\sqrt{(k+1)}\rceil+2)})$ if and only if there exist $v_1 \in R_{u_1}$ and $v_2 \in R_{u_2}$ such that $\{v_1, v_2\} \in E$.

Consider the $\lceil\sqrt{k+1}\rceil^2$ squares $(2i, 2j)$, for $1 \leq i \leq \lceil\sqrt{k+1}\rceil$ and $1 \leq j \leq \lceil\sqrt{k+1}\rceil$. For simplicity we denote by $Q_{i,j}$ the square $(2i, 2j)$. The selected squares are illustrated in Figure 8.3. By construction, the squares $Q_{i,j}$ are pairwise vertex-disjoint. For each i, j, we arbitrarily choose $e_{i,j} = \{a_{i,j}, b_{i,j}\} \in E$ such that $a_{i,j} \in R_{2i+1,2j}$ and $b_{i,j} \in R_{2i,2j+1}$. We denote by $e_{i,j}$ the representative edge of $Q_{i,j}$. The edge $e_{i,j}$ can be monitored only by an element of $R_{\ell_{2i,2j}} \cup R_{\ell_{2i,2j+1}} \cup R_{\ell_{2i+1,2j}} \cup R_{\ell_{2i+1,2j+1}}$, because the other $\ell_{i',j'}$ are not connected to both $\ell_{2i+1,2j}$ and $\ell_{2i,2j+1}$. Thus, there are no two distinct representative edges in G that can be monitored by the same vertex of G. This means that the solution should be of size at least $k+1$, that is the number of squares we had consider. As we ask for a solution of size at most k, then we can safely answer that there is no such a solution.

Now assume that $\mathbf{tw}(G) \leq c(2\lceil\sqrt{(k+1)}\rceil + 2)$. By Lemma 8.3.3, we know that there is an algorithm in time $2^{\mathcal{O}(\mathbf{tw})\cdot\log(\max_{e\in E} c(e)))} \cdot n$ to solve the problem. In particular, this algorithm runs in time $2^{\mathcal{O}(\sqrt{k}\cdot\log(\max_{e\in E} c(e)))} \cdot n$. □

8.4 Edge monitoring on complete graphs

In this section, we give a complexity result related to complete graphs.

Theorem 8.4.1 Edge Monitoring *is* $W[1]$*-complete on complete graphs when the parameter is the size of the solution.*

Proof. The problem is clearly in NP. To prove that Edge Monitoring is NP-hard and $W[1]$-hard, we exhibit an FPT-reduction from Independent Set. Let $(G = (V, E), k)$ be an instance of Independent Set. Without loss of generality, we can assume that G is connected. Indeed, it is easily seen that Independent Set remains NP-complete under this restriction. We build an instance $(G' = (V, E'), c, k)$ of Edge Monitoring as follows: G' is a complete graph and for each edge $e \in E'$, we have $c(e) = k - 1$ if $e \in E$ and $c(e) = 0$ otherwise.

We show that (G, k) is a positive instance of Independent Set if and only if (G', c, k) is a positive instance of Edge Monitoring. First of all, notice that there is no monitoring set of size less than k. Indeed, assume, for the sake of contradiction, that there is a monitoring set S of size less than k. Since G is connected, there exists an edge e incident to a vertex in S and such that $c(e) = k - 1$. We have $M(e) \cap S < k - 1$ so there is a contradiction.

Now, let $S \subseteq V$ such that $|S| = k$. Then, we have:
S is a monitoring set of (G', c) iff for each $e \in E$, $|S \setminus e| \geq k - 1$ iff for each $e \in E$ in E, $|S \cap e| \leq 1$ iff S is a stable of G.

We will prove now that Edge Monitoring on complete graphs parametrized by k belongs to $W[1]$. To prove this, we will show that this problem can be reduced to Independent Set as described in the following algorithm.

First, let us prove that (G, c) admits a monitoring set of size at most k if Algorithm 8.1 returns True. We proceed by induction on k. If $k = 0$, it is clear that Algorithm 8.1 returns True if and only if $C = 0$. Now, assume that $k > 0$. If Line 7 returns True then (G, c) admits a monitoring set of size at most $k - 1$ by induction hypothesis. Assume now that Line 12 returns True. Then, there exists an independent set S of size k in G'. Thus, S is a monitoring set of (G, c). Indeed, (G, c) does not admit an edge e with $c(e) > k$ by Lines 3-4. Edges e with $c(e) = k$ have no extremities in S by construction of G'. Hence, these edges are monitored by S. Edges e with $c(e) = k-1$ have at most one extremity in S also by construction of G'. Thus, these edges are monitored by S. Edges e with $c(e) \leq k-2$ are necessarily monitored by S since $|S| = k$.

Now, let us prove that Algorithm 8.1 returns True if (G, c) admits a monitoring set S of size at most k. We proceed by induction on k. If $k = 0$ then necessarily $C = 0$. Thus, Algorithm 8.1 returns True. Now, assume that $k > 0$. If $|S| \leq k - 1$

then Algorithm 8.1 returns True in Line 7 by induction hypothesis. Assume now that $|S| = k$ then it is easily seen that S is an independent set of G' with $|S| = k$. Then Algorithm 8.1 returns True in Line 12. This completes the proof. □

Algorithm 8.1: Reduction of Edge Monitoring to Independent Set (Function ReducMStoIS(G, c, k)).

```
1:  Input: G = (V, E), c, k
2:    Let C = max{c(e) : e ∈ E}
3:    If C > k
4:        Return False
5:    Else
6:        If ReducMStoIS(G, c, k − 1) returns True
7:            Return True
8:        Else
9:            Let V' built from V by removing the extremities of edges e with
      c(e) = k
10:           Let E' = {uv ∈ E : c(uv) = k − 1 ∧ u ∈ V' ∧ v ∈ V'}
11:           If there exists an independent set of size k in G' = (V', E')
12:               Return True
13:           Else
14:               Return False
15:           End If
16:       End If
17:   End If
```

8.5 Conclusion

In this chapter, we proved that 1-uniform monitoring problem is NP-complete even when restricted to (P_5, C_4)-free 1-monitorable graphs. Moreover, we proved that 1-uniform Edge Monitoring cannot be approximated within $(1 - \varepsilon) \ln |V|$ for any $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{O(\log \log n)})$. We also proved that the 1-uniform edge monitoring problem is $W[2]$-hard when parameterized by the size of the solution. Afterwards, we proved that the edge monitoring problem is $W[1]$-complete on com-

plete graphs when the parameter is the size of the solution. Moreover, we showed that, in general graphs, we are unlikely to be able to solve our problem in FPT time when parameterized by the size of the solution. We used Bidimensionality to show that if the input graph has the topological restriction to be apex-minor-free, then our problem can be solved in time $2^{\mathcal{O}(\sqrt{k})} \cdot n$. We even show that the Edge Monitoring can be solved in a similar time, i.e., in time $2^{\mathcal{O}(\sqrt{k} \cdot \log(\max_{e \in E} c(e)))} \cdot n$, when the input graph is apex-minor-free.

Chapter 9

Weighted edge monitoring problems

Contents

In the two previous chapters, we studied the problem of *Edge Monitoring* and we gave a particular interest to 1-*uniform monitoring* (1-UMP). We presented some results on specific classes of graphs related to this problem and some complexity results are also discussed.

In this chapter, we consider a weighted version of the problem, it means a weight on vertices, called *weighted edge monitoring problem*. We study the problem on several classes of graphs as complete graphs, block graphs, interval graphs and cographs.

9.1 Introductory results on complete graphs

In this section, we give some results related to complete graphs.

Lemma 9.1.1 *Let $G = (V, E), w, c$ such that G is a complete graph, $C = \max\{c(e) : e \in E\}$ and $|V| \geq C + 2$. Then, $C \leq \gamma_m(G, c) \leq C + 2$. Moreover, every set $S \subseteq V$ such that $|S| \geq C + 2$ is a monitoring set of (G, c).*

Proof. Since there exists an edge e of color $c(e) = C$, we need C vertices to monitor it. Thus, $C \leq \gamma_m(G, c)$. Let $S \subseteq V$ be a set such that $|S| \geq C + 2$. Then, every edge is monitored by S. Indeed, let $e = \{u, v\} \in E$. Then, the set $S \setminus \{u, v\}$ of size at least $C \geq c(e)$ covers e. □

Lemma 9.1.2 *Let $G = (V, E), c$ be a colored graph such that G is a complete graph and c is k-uniform with $k > 0$ and $|V| \geq k + 2$. Then, $\gamma_m(G, c) = k + 2$.*

Proof. Assume, for the sake of contradiction, that there exists a set S that monitors G such that $|S| < k + 2$. If $|S| = 1$, let v be the unique element of S. Let e an edge incident to v. Then, e is not monitored by S. Otherwise, let u and v be two elements in S. Then, $M(\{u, v\}) \cap S = |S| - 2 < k$ so $\{u, v\}$ is not monitored by S. □

9.2 Polynomial algorithms for complete and block graphs

In this section we present some results of Weighted Edge Monitoring problem on complete and block graphs.

Let recall some definitions. A block graph is a graph where each biconnected component (block) is a clique. The block-cut tree T of a connected graph G is defined as follows. The vertices of T are the blocks and the articulation points of G. There is an edge in T for each pair of a block and an articulation point that belongs to that block.

Lemma 9.2.1 Weighted Edge Monitoring *can be solved in polynomial time on C-bounded weighted complete graphs.*

Proof. Let $(G = (V, E), w, c)$ such that G is a complete graph. By Lemma 9.1.1, $\gamma_m(G, c) \leq C + 2$. Therefore, it suffices to enumerate all sets $S \subseteq V$ that monitor G and such that $|S| \leq C + 2$. There are $O(n^{C+2})$ such sets. Thus, the problem can be computed in polynomial time. □

Lemma 9.2.2 Weighted Edge Monitoring *can be solved in quasi-linear time on uniform complete graphs.*

Proof. Let $(G = (V, E), w, c)$ such that G is a complete graph and c is k-uniform. By Lemma 9.1.2, $\gamma_m(G, c) = C + 2$ and by Lemma 9.1.1, every set $S \subseteq V$

of size $C + 2$ monitors G. Thus, if we choose S as the set of the $C + 2$ first elements in V sorted by increasing weight, we obtain an optimal solution for Weighted Edge Monitoring(G, w, c). We only need to sort V which can be done in time $|V| \log |V|$. □

The following lemma is useful to establish the connection between γ_m of a graph G and γ_m of its 2-connected components.

We denote $\gamma_m(G_1, w, c|u) = \min\{w(S) : S$ is a monitoring set of (G, c) and $u \in S\}$

Lemma 9.2.3 *Let $(G = (V, E), w, c)$ be a weighted graph, $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ two graphs and $u \in V$ such that $V = V_1 \cup V_2$, $E = E_1 \cup E_2$ and $V_1 \cap V_2 = \{u\}$. Let $d = \gamma_m(G_1, w, c|u) - \gamma_m(G_1, w, c)$. Let w' obtained from w by replacing the weight of u by d. Then $\gamma_m(G, w, c) = \gamma_m(G_1, w, c) + \gamma_m(G_2, w', c)$.*

Proof. Let S_1, S_1', S_2 be optimal solutions of Weighted Edge Monitoring(G_1, w, c), Weighted Edge Monitoring$(G_1, w, c|u)$, Weighted Edge Monitoring(G_2, w', c) respectively.

We first prove $\gamma_m(G, w, c) \leq \gamma_m(G_1, w, c) + \gamma_m(G_2, w', c)$: if $u \notin S_2$ then $S_1 \cup S_2$ is a solution of Weighted Edge Monitoring(G, w, c) having weight $w(S_1) + w'(S_2)$. If $u \notin S_2$ then $S_1' \cup S_2$ is a solution of Weighted Edge Monitoring(G, w, c) having weight $w(S_1') + w(S_2) - d = w(S_1) + w'(S_2)$. Thus we have $\gamma_m(G, w, c) \leq \gamma_m(G_1, w, c) + \gamma_m(G_2, w', c)$.

Now we prove $\gamma_m(G, w, c) \geq \gamma_m(G_1, w, c) + \gamma_m(G_2, w', c)$: let S^* be an optimal solution of Weighted Edge Monitoring(G, w, c). We have $S_1^* = S^* \cap V_1$ and $S_2^* = S^* \cap V_2$ are solutions of Weighted Edge Monitoring(G_1, w, c) and Weighted Edge Monitoring(G_2, w', c) respectively. We have to consider two cases:

$u \notin S^*$: We have $w(S_1^*) \geq w(S_1)$ and $w_2'(S_2^*) \geq w_2'(S_2)$ by optimality of S_1 and S_2. Since $w(S^*) = w(S_1^*) + w(S_2^*)$, $w(S^*) \geq w(S_1) + w(S_2)$.

$u \in S^*$: This implies that $w(S^*) = w(S_1^*) + w'(S_2^*) - d$. Since $w'(S_2^*) \geq w(S_2)$ and $w(S_1^*) \geq w(S_1')$, then

$$w(S^*) \geq w(S_1') + w_2'(S_2) - d = w(S_1) + w_2'(S_2)$$

Consequently we have $\gamma_m(G, w, c) \geq \gamma_m(G_1, w, c) + \gamma_m(G_2, w', c)$. This completes the proof of the lemma. □

Theorem 9.2.4 *The two statements hold:*

1. Weighted Edge Monitoring *can be solved in polynomial time on C-bounded weighted block graphs.*

2. Weighted Edge Monitoring *can be solved in quasi-linear time for block graphs $(G = (V, E), w, c)$ where c is uniform.*

Proof. Without loss of generality, we can assume that G is connected. We will prove the first statement. The proof of the second statement is similar. Let $(G = (V, E), w, c)$ be a C-bounded weighted block graph. We first compute the block-cut tree T of G. This can be done in linear time [HT73]. Then, we choose a clique V_1 that corresponds to a leaf of T and u the articulation point that is neighbor of V_1 in T. Let $G_1 = (V_1, E_1) = G[V_1]$ and $G_2 = (V_2, E_2) = G[(V \setminus V_1) \cup \{u\}]$. G_2 is also a block graph. Thus, we can apply Lemma 9.2.3. It suffices to compute $\gamma_m(G_1, w, c)$, $\gamma_m(G_1, w, c|u)$ and $\gamma_m(G_2, w', c)$. $\gamma_m(G_1, w, c)$ can be computed in polynomial time by using Lemma 9.2.1. Proof of Lemma 9.2.1 can be easily modified to compute $\gamma_m(G_1, w, c|u)$. $\gamma_m(G_2, w', c)$ can be computed by induction. □

9.3 Interval graphs

In this section we give a polynomial dynamic programming algorithm for computing Weighted Edge Monitoring on weighted interval graphs. First some definitions.

A graph $G = (V, E)$ is an interval graph if there exists $|V|$ intervals $(I_i)_{i \in V} = ([a_i, b_i])_{i \in V}$ of the real line such that $(i, j) \in E$ if and only if $I_i \cap I_j \neq \emptyset$ for every distinct vertices $i, j \in V$. We say that $(I_i)_{i \in [1,n]}$ is a realization of G. Without loss of generality, we can assume that there are no intervals I_i and I_j that have a common extremity.

Given an interval graph $G = (V, E)$ and a realization $(I_i)_{i \in V}$, we define a total order $<_L$ (resp. $<_R$) over V such that $i <_L j$ (resp. $i <_R j$) if $a_i < a_j$ (resp. $b_i < b_j$).

The following definition is a refinement of the nice tree decomposition introduced by Kloks [Klo94b] and used in Chapter 8.

Definition 12 *[FMN+15] Let $G = (V, E)$ be an interval graph and $(I_i)_{i \in V}$ be a realization of G. A nice path decomposition of G is a sequence of sets of vertices*

$B_0, \dots B_l$ such that

- *all sets B_i are cliques of G;*
- *every edge $e \in E$ appears in a set B_i,*
- *for every vertex $v \in E$, the set of indices i such that $v \in B_i$ is a segment of $[0, l]$.*
- *$B_0 = \emptyset$ and $B_l = \emptyset$;*
- *For every $i \in [1, l]$,*
  - *$B_i = B_{i-1} \cup \{v\}$ (i introduces the vertex v)*
  - *or $B_{i-1} = B_i \cup \{v\}$ (i forgets the vertex v).*
- *the order in which vertices are introduced corresponds to $<_L$*
- *the order in which vertices are forgotten corresponds to $<_R$*

For $i \in [0, l]$, F_i is the set of vertices appearing in some set B_j, $j < i$, but not in B_i. $V_i = F_i \cup B_i$ and $G_i = G[F_i \cup B_i]$.

Lemma 9.3.1 *[FMN+15] Let $G = (V, E)$ be an interval graph and $(I_i)_{i \in V}$ be a realization of G. Then G has a nice path-decomposition that can be computed in linear time.*

A set $S \subseteq V_i$ is an i-partial solution if every edge in G_i that has an extremity in F_i is monitored by S. The i-representant W of $S \subseteq V_i$, denoted by $repr_i(S)$, contains exactly the $C + 2$ greatest vertices in $S \cap N[B_i]$ w.r.t. $<_R$ or is $S \cap N[B_i]$ if $|S \cap N[B_i]| < C + 2$. We say that S extends W if W is the i-representant of S.

We denote by $\mathcal{F}_i^*$ the set of i-representants of i-partial solutions and w_i^* is a function $w_i^* : \mathcal{F}_i \to \mathbb{Q}^+$ such that $w_i^*(W) = \min\{w(S) : S \text{ is an } i\text{-partial solution that extends } W\}$.

Before presenting the algorithm, we introduce two lemmas. The second is the key of the algorithm.

Lemma 9.3.2 *Let $u \in B_i$, $v_1, v_2 \in V_i$ such that $v_1 <_R v_2$ and $v_1 \in N[u]$. Then $v_2 \in N[u]$.*

Proof. Let $[a_u, b_u]$, $[a_{v_1}, b_{v_1}]$ and $[a_{v_2}, b_{v_2}]$ the intervals that represent u, v_1 and v_2 respectively in the realization of G. Since $u \in B_i$ and $v_2 \in V_i$, $b_u > a_{v_1}$ and $b_u > a_{v_2}$. Since $v_1 \in N[u]$, we have $a_u < b_{v_1}$ and since $v_1 <_R v_2$, we have $a_u < b_{v_2}$. Thus $[a_u, b_u] \cap [a_{v_2}, b_{v_2}] \neq \emptyset$. Consequently, $v_2 \in N[u]$. □

Lemma 9.3.3 *Let $S \subseteq V_i$, $W = repr_i(S)$ and $v_1, v_2 \in B_i$ such $\{v_1, v_2\}$ is monitored by S. Then $\{v_1, v_2\}$ is monitored by W.*

Proof. First, notice that every $u \in V_i$ that belongs to $M(\{v_1, v_2\})$ belongs to $N[B_i]$. If $|S \cap N[B_i]| \leq C+2$, then $W = S \cap N[B_i]$ and the lemma is trivially verified. Now, assume that $|S \cap N[B_i]| > C + 2$ and let $u \in (S \setminus W) \cap M(\{v_1, v_2\})$. By Lemma 9.3.2, every vertex $u' \in W$ belongs to $N[v_1]$ and $N[v_2]$. So all elements in W except at most two (v_1 and v_2) belong to $M(\{v_1, v_2\})$. Thus $|M(\{v_1, v_2\}) \cap W| \geq C$ and $\{v_1, v_2\}$ is monitored by W. □

To solve Weighted Edge Monitoring on interval graphs, a naive algorithm consists to iterate over the sets B_i and to compute for each i the set of i-partial solutions. Unfortunately, the algorithm is non polynomial since the set of i-partial solutions can be exponential. The key of the algorithm is as follows: instead of considering all the i-partial solutions, we consider the representants of the i-partial solutions. Since the number of representants is polynomially bounded by $|V|$, the algorithm will run in polynomial time. Lemma 9.3.3 guarantees that we don't miss solutions. Indeed, let S be an i-partial solution. If $i+1$ introduces the node v, then S and $S \cup \{v\}$ are $(i+1)$-partial solutions. There is nothing to verify. If $i+1$ forgets the node v then S is an $(i+1)$-partial solution if and only if the forgotten edges i.e. the edges having v as extremity and the other extremity in B_{i+1} are monitored by S. But thanks to Lemma 9.3.3, it suffices to check that these edges are monitored by $repr_i(S)$.

We present now Algorithm 9.2. Functions w_i can be seen as associative arrays with default value $+\infty$: the instruction $w_i(W)$ returns $+\infty$ if the key W is not in the associative array w_i.

Lemma 9.3.4 *For every $i \in [0, l]$, after the run of Algorithm 9.2, it holds $\mathcal{F}_i = \mathcal{F}_i^*$ and $w_i(S) = w_i^*(S)$ for every $S \in \mathcal{F}_i$.*

Algorithm 9.1: Algorithm for Weighted Edge Monitoring on interval graphs

Input: An interval graph $G = (V, E)$.
Begin
Compute a nice path decomposition $B_0, ..., B_l$
$\mathcal{F}_0 \leftarrow \{\emptyset\}$
$w_0(\emptyset) = 0$
For each i from 1 to l **do**
    $\mathcal{F}_i \leftarrow \emptyset$
    If i forgets the node v
        For each $W \in \mathcal{F}_{i-1}$ **do**
            If all edges $\{u, v\}$ where $u \in B_i$ are monitored by W
                $W' \leftarrow repr_i(W)$
                $\mathcal{F}_i \leftarrow \mathcal{F}_i \cup \{W'\}$
                $w_i(W') \leftarrow \min\{w_i(W'), w_{i-1}(W)\}$
            End If
        End For
    Else If i introduces the node v
        For each $W \in \mathcal{F}_{i-1}$ **do**
            $W' \leftarrow repr_i(W)$
            $\mathcal{F}_i \leftarrow \mathcal{F}_i \cup \{W'\}$
            $w_i(W') \leftarrow \min\{w_i(W'), w_{i-1}(W)\}$
            $W' \leftarrow repr_i(W \cup \{v\})$
            $\mathcal{F}_i \leftarrow \mathcal{F}_i \cup \{W'\}$
            $w_i(W') \leftarrow \min\{w_i(W'), w_{i-1}(W) + w(v)\}$
        End For
    End If
End For
If $\mathcal{F}_l = \emptyset$
    Return $+\infty$
Else
    Return $\min\{w_l(W) : W \in \mathcal{F}_l\}$
End If
End

Proof. We prove by induction on i. The property is clearly verified for $i = 0$. Now, suppose that the property holds for i and prove it for $i + 1$.

$\mathcal{F}_{i+1} \subseteq \mathcal{F}^*_{i+1}$ and for each $W \in \mathcal{F}_{i+1}$, $w^*_{i+1}(W) \leq w_{i+1}(W)$: let $W' \in \mathcal{F}_{i+1}$. We consider two cases.

$i + 1$ forgets the vertex v: then, W' comes from some $W \in \mathcal{F}_i$ such that $W' = \mathit{repr}(W)$, $w_{i+1}(W') = w_i(W)$ and W' is added to $\mathcal{F}_{i+1}$ by Lines 11-13. Using the induction hypothesis, $W' \in \mathcal{F}^*_i$ and $w_i(W) = w^*_i(W)$. Let S be a i-partial solution of weight $w(S) = w^*_i(W)$ that extends W. By Line 10 of the algorithm, all edges $\{u, v\}$ where $u \in B_i$ are monitored by W and thus by S. Consequently, S is an $(i + 1)$-partial solution with $\mathit{repr}_{i+1}(S) = \mathit{repr}_{i+1}(W) = W'$. Thus, $W' \in \mathcal{F}^*_{i+1}$ and $w^*_{i+1}(W') \leq w(S) = w^*_i(W) = w_i(W) = w_{i+1}(W')$.

$i + 1$ introduces the vertex v: There are two possibilities.

$v \notin W'$: then W' comes from some $W \in \mathcal{F}_i$ such that $W' = \mathit{repr}(W)$, $w_{i+1}(W') = w_i(W)$ and W' is added to $\mathcal{F}_{i+1}$ by Lines 18-20. By induction hypothesis, $W \in \mathcal{F}^*_i$ and $w_i(W) = w^*_i(W)$. Let S be a i-partial solution of weight $w(S) = w^*_i(W)$ that extends W. S is an $(i + 1)$-partial solution with $\mathit{repr}_{i+1}(S) = \mathit{repr}_{i+1}(W) = W'$. Thus $W' \in \mathcal{F}^*_{i+1}$ and $w^*_{i+1}(W') \leq w(S) = w^*_i(W) = w_i(W) = w_{i+1}(W')$.

$v \in W$: W' comes from some $W \in \mathcal{F}_i$ such that $W' = \mathit{repr}(W + \{v\})$, $w_{i+1}(W') = w_i(W) + w(v)$ and W' is added to $\mathcal{F}_{i+1}$ by Lines 21-23. Let S be a i-partial solution of weight $w(S) = w^*_i(W)$ that extends W. $S' = S \cup \{v\}$ is an $(i+1)$-partial solution with $\mathit{repr}_{i+1}(S \cup \{v\}) = \mathit{repr}_{i+1}(W \cup \{v\}) = W'$. Thus $W' \in \mathcal{F}^*_{i+1}$ and $w^*_{i+1}(W') \leq w(S + \{v\}) = w^*_i(W) + w(v) = w_i(W) + w(v) = w_{i+1}(W')$.

$\mathcal{F}^*_{i+1} \subseteq \mathcal{F}_{i+1}$ and for each $W \in \mathcal{F}^*_{i+1}$, $w_{i+1}(W') \leq w^*_{i+1}(W')$: let $W' \in \mathcal{F}^*_{i+1}$ and S' be an $(i + 1)$-partial solution that extends W and such that $w(S') = w^*_{i+1}(W')$. We also consider two cases:

$i + 1$ forgets the vertex v: then S' is an i-partial solution. Let $W = \mathit{repr}_i(S')$. Then $W' = \mathit{repr}_{i+1}(W)$. Using the induction hypothesis, $W \in \mathcal{F}_i$ and $w_i(W) = w^*_i(W)$. By definition of a $(i+1)$-partial solution, all edges $\{u, v\}$ where $u \in B_i$ are monitored by S'. But, by applying Lemma 9.3.3, these edges are also monitored by W. Thus, Line 10 of the algorithm succeeds and $W' = \mathit{repr}_{i+1}(W)$ is added to $\mathcal{F}_{i+1}$ and by Line 13 $w_{i+1}(W') \leq w_i(W) = w^*_i(W) = w(S') = w^*_{i+1}(W')$.

$i + 1$ introduces the vertex v. There are two possibilities.

$v \notin S'$: then S' is an i-partial solution. Let $W = repr_i(S')$. Using the induction hypothesis, $W \in \mathcal{F}_i$ and $w_i(W) = w_i^*(W)$. Thus, $W' = repr_{i+1}(W)$ is added to $\mathcal{F}_{i+1}$ by Line 19 and by Line 20 $w_{i+1}(W') \leq w_i(W) = w_i^*(W) = w(S') = w_{i+1}^*(W')$.

$v \in S'$: let $S = S' - v$. Then S' is an i-partial solution. Let $W = repr_i(S)$. Using the induction hypothesis, $W \in \mathcal{F}_i$ and $w_i(W) = w_i^*(W)$. Thus, $W' = repr_{i+1}(S \cup \{v\}) = repr_{i+1}(W \cup \{v\})$ is added to $\mathcal{F}_{i+1}$ by Line 22 and by Line 23 $w_{i+1}(W') \leq w_i(W) + w(v) = w_i^*(W) + w(v) = w(S) + w(v) = w(S') = w_{i+1}^*(W')$. □

Theorem 9.3.5 Weighted Edge Monitoring *on C-bounded weighted interval graphs is in* P. *More precisely, it can be solved in time $O(|V|^{C+4})$.*

Proof. Thanks to Lemma 9.3.4, it is clear that Algorithm 9.2 is exact. Let prove that it runs in the expected time. The algorithm consists of a main loop that does $|V| + 1$ iterations. Within this loop, we have two possibilities: forgetting or introducing a vertex. In the two cases, we loop over the elements of $\mathcal{F}_{i-1}$. Each step of the loop can be done in time $O(N(B_i))$ (since C is bounded) in both cases. The size of $\mathcal{F}_{i-1}$ is bounded by $(N[B_{i-1}] \cap V_{i-1})^{C+2}$. Therefore the time spent within a step of the main loop is $O((N[B_{i-1}] \cap V_{i-1})^{C+3})$. Since $N[B_{i-1}] \cap V_{i-1}$ is bounded by $|V|$, Algorithm 9.2 runs in time $O(|V|^{C+4})$. □

The complexity of the algorithm can be refined in the case of unit interval graphs.

Lemma 9.3.6 *Let C be a clique of an unit interval graph $G = (V, E)$. Then $N[C] \leq 3\omega(G)$.*

Proof. Let $(I_i)_{i \in E}$ be a realization of G. Since G is an unit interval graph, we have $u \leq_L v \Leftrightarrow u \leq_R v$ for every $x, y \in V$. For every vertex $v \in V$, we denote by $N_{\leq}[v]$ (resp. $N_{\geq}[v]$) the set $\{u : u \in N[v] \wedge u \leq_L v\}$ (resp. $\{u : u \in N[v] \wedge u \geq_L v\}$). Let v_{min} (resp. v_{max}) be the minimal (resp. maximal) vertex of C w.r.t $\leq_L$. It is easily seen that $N[C] = N_{\leq}[v_{min}] \cup (N_{\geq}[v_{min}] \cap N_{\leq}[v_{max}]) \cup N_{\geq}[v_{max}]$ and that $N_{\leq}[v_{min}]$, $N_{\geq}[v_{min}] \cap N_{\leq}[v_{max}]$ and $N_{\geq}[v_{max}]$ are clique of G. Thus $N[C] \leq 3\omega(G)$. □

Theorem 9.3.7 Weighted Edge Monitoring *can be solved in time $O(\omega(G)^{C+3}|V|)$ on C-bounded weighted unit interval graphs.*

Proof. We refine the running time of Theorem 9.3.5. Thanks to Lemma 9.3.6, we can bound $N[B_{i-1}] \cap V_{i-1}$ by $3\omega(G)$. Thus, we deduce that the overall running time is $O(\omega(G)^{C+3}|V|)$ in weighted unit interval graphs. □

9.4 Cographs

Let $G_1 = (V_1, E_2)$ and $G_2 = (V_2, E_2)$ such that $V_1 \cap V_2 = \emptyset$. The join of G_1 and G_2 is the graph $G = (V_1 \cup V_2, E_1 \cup E_2 \cup \{\{u, v\} : u \in V_1 \wedge v \in V_2\})$. The class of cographs is defined by induction.

- The graph which contains one vertex is a cograph;
- The (disjoint) union and the join of two cographs are cographs.

Lemma 9.4.1 *Let $G = (V, E)$ be the join of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Let S be a total dominating set of G_1. Then, S monitors all edges between V_1 and V_2.*

Proof. Let $\{u, v\}$ be an edge between G_1 and G_2 such that $u \in V_1$. Then there exists a vertex $u_1 \in S$ adjacent to u. Thus, $\{u, v\}$ is monitored by S since $\{u, v, u_1\}$ is a triangle of G. □

Lemma 9.4.2 *Let $G = (V, E)$ be the join of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Let S be a monitoring set of G. Then $S \cap V_1$ is a total dominating set of G_1 or $S \cap V_2$ is a total dominating set of G_2.*

Proof. Assume for the sake of contradiction that S_1 is not a total dominating set of G_1 and S_2 is not a total dominating set of G_2. Then there exists an edge $\{u, v\} \in E$ such that u has no neighbor in S_1 and v has no neighbor in S_2. Thus, $\{u, v\}$ is not monitored by S. □

Lemma 9.4.3 *Let G be the join of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Let S be a minimal monitoring set of G. Then $|S \cap V_1| \leq 1$ or $|S \cap V_2| \leq 1$.*

Proof. Let S be a minimal monitoring set of G, $S_1 = S \cap V_1$ and $S_2 = S \cap V_2$. Assume, for the sake of contradiction, that $|S_1| \geq 2$ and $|S_2| \geq 2$. By Lemma 9.4.2, S_1 is a total dominating set of G_1 or S_2 is a total dominating set of G_2. By

symmetry, suppose that S_1 is a total dominating set of G_1. Then S_1 monitors all edges between V_1 and V_2 by Lemma 9.4.1 and all edges in V_2. Consequently, for every vertex $u \in V_2$, $S_1 \cup \{u\}$ is a monitoring set of G since u monitors all edges in V_1. Thus, S is not minimal. □

Lemma 9.4.4 *Let $G = (V, E)$ be a graph with no isolated vertices and S a monitoring set of G. Then, S is a total dominating set of G.*

Proof. Let v be a vertex in V. Since G has no isolated vertices, there is a vertex $e = (v, v_1)$ incident to v. Since S is a monitoring set of G, there is a vertex $v_2 \in S$ such that $\{v, v_1, v_2\}$ is a triangle in G. Thus, v is adjacent to a vertex in S. □

Combining Lemmas 9.4.1, 9.4.2 and 9.4.4, we obtain the following lemma.

Lemma 9.4.5 *Let $G = (V, E)$ be the join of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Let $S_1 = S \cap V_1$ and $S_2 = S \cap V_2$. The two statements hold.*

- *If $S_1 \neq \emptyset$ and $S_2 \neq \emptyset$, then S is a monitoring set of G if and only if S_1 is a total dominating set of G_1 or S_2 is a total dominating set of G_2.*
- *If $S_2 = \emptyset$ (resp. $S_1 = \emptyset$), then S is a monitoring set of G if and only if G_1 (resp. G_2) has no isolated vertices and S_1 (resp. S_2) is a monitoring set of G_1 (resp. G_2).*

The following lemma is a direct consequence of Lemma 9.4.3 and Lemma 9.4.5.

Lemma 9.4.6 *Let $G = (V, E)$ be a graph. If G is the (disjoint) union of two graphs G_1 and G_2. Then,*

$$\gamma_m(G, w, 1) = \gamma_m(G_1, w, 1) + \gamma_m(G_2, w, 1)$$

If G is the join of two graphs G_1 and G_2.

$$\gamma_m(G, w, 1) = \min \begin{cases} \gamma_t(G_1, w) + \min\{w(v) : v \in V_2\} \\ \min\{w(v) : v \in V_1\} + \gamma_t(G_2, w) \\ \gamma_m(G_1, w, 1) \text{ if } G_1 \text{ has no isolated vertices} \\ \gamma_m(G_2, w, 1) \text{ if } G_2 \text{ has no isolated vertices} \end{cases}$$

Lemma 9.4.6 combined with the fact that a cotree is computable in linear time [HP05] give us a linear time algorithm to compute Weighted Edge Monitoring on cographs.

Theorem 9.4.7 *1-uniform* Weighted Edge Monitoring *can be solved in linear time on cographs.*

9.5 Conclusion and summary of Part III

In this last part, we studied a recent problem, called Edge monitoring problem, in different classes of graphs. This parameter can be considered as a variant of dominating sets problem. Three variants of the problem have been studied in this part. We provided exact values of $\gamma_m(G, 1)$ on three types of graphs: powers of a cycle, powers of a path and square of trees. Moreover, we proved the NP-completeness of 1-uniform monitoring problem on split graphs, comparability graphs and planar unit disc graphs. Other results are also obtained for the edge monitoring and weighted edge monitoring problems in some graph classes such as complete graphs, block graphs, interval graphs, planar graphs and cographs.

The most important results, developed in this part, are summarized in the following points.

1. 1-Uniform Edge Monitoring

   - Exact values of $\gamma_m(C_n^m, 1)$ is given in power of cycle (C_n^m).
   - Exact values of $\gamma_m(P_n^m, 1)$ is given in power of path (P_n^m).
   - A linear time algorithm to find $\gamma_m(T^2, 1)$-sets is given.
   - NP-completeness on split graphs is proved.
   - NP-completeness on comparability graphs is proved.
   - NP-completeness on planar UDGs is proved.
   - NP-completeness even when restricted to (P_5, C_4)-free 1-monitorable graphs is proved.
   - Non approximability of the problem within $(1 - \varepsilon) \ln |V|$ for any $\varepsilon > 0$, unless NP $\subseteq$ DTIME$(n^{O(\log \log n)})$ is proved.

2. Edge monitoring

   - The problem is $W[1]$-complete on complete graphs when the parameter is the size of the solution.
   - The problem is $W[2]$-hard when the parameter is the size of the solution.
   - Two algorithms are presented. First one parameterized by the treewidth in time $2^{\mathcal{O}(\mathbf{tw}^2 \cdot \log(\max_{e \in E} c(e)))} \cdot n$ where $\mathbf{tw}$ is the treewidth of the input graph. The second one parameterized by k, the size of the solution, in time $2^{\mathcal{O}(\sqrt{k} \cdot \log(\max_{e \in E} c(e)))} \cdot n$ when the input graph is apex-minor-free, in particular, when it is planar.
   - Exact values of $\gamma_m(K_n, k)$ is given.

3. Weighted Edge Monitoring

   - The problem can be solved in polynomial time on C-bounded weighted complete graphs.
   - 1-uniform weighted edge monitoring can be solved in linear time in cographs.
   - The problem on C-bounded weighted interval graphs is in P. It can be solved in time $O(|V|^{C+4})$.
   - The problem can be solved in polynomial time on C-bounded weighted block graphs.
   - The problem can be solved in quasi-linear time for block graphs ($G = (V, E), w, c$) where c is uniform.

Chapter 10

Conclusions and Perspectives

In this thesis, we discussed three graph theory problems applied to different classes of graphs. This concluding chapter summarizes the results presented in the previous chapters and discusses several open questions and directions for future research to each contribution.

In the first part, we presented the graph decomposition problem, detailed two types of this problem and surveyed some of its famous results. Then, our focus was on the decomposition of complete multigraph. We discussed the problem of decomposition of complete multigraph λK_n into k-cycles and k-stars of size k and gave necessary and sufficient conditions for which this decomposition exists. Most cases have been treated. As future work, we plan to focus on the following issues:

1. We left some open subcases when $n < 2k$. The natural question is how to improve our results by using the proposed idea (or not) to prove the remaining cases ?

2. Investigate the same decomposition problem with a generalized point of view by considering the decomposition of complete multigraph λK_n into cycles of size l and stars of size k such that $l \geq 2$ and $k \geq 2$ are different.

3. It would be interesting to consider the directed version of the studied decomposition as the directed stars and circuit decomposition for the complete directed graph, the complete directed multigraph and the complete directed bipartite graph.

The second part of this thesis was devoted to study the $[i, j]$-dominating sets and its variant, namely $[i, j]$-total dominating sets. Several variants of the dominating sets problem have been defined in the literature. Then, we first presented dominating sets problem and reviewed some of its variants. Afterwards, we focused

on $[i, j]$-dominating sets and $[i, j]$-total dominating sets. We were particularly interested in these two variants for $i = 1$ and $j = 2$. We gave the exact value of the $[1, 2]$-dominating number and the $[1, 2]$-total dominating number for generalized Petersen graphs $P(n, k)$ when $k = 2$.

From these results, we can consider several directions for future work:

1. We would like to investigate whether the study of $[1, 2]$-dominating set problem on $P(n, k)$ for $k \geq 3$.

2. Since $\gamma_{t[1,2]}(P(n, 2)) = \gamma_{[1,2]}(P(n, 2))$ except for the case $n = 5$ and $n \equiv 1[6]$. The natural question is for each values of n and k this equation holds.

3. If there is some integer $1 < k \leq n$ such that $\gamma_{[1,2]}(P(n, k)) = \gamma(P(n, k))$.

4. If there is some integer $1 < k \leq n$ such that $\gamma_t[1, 2](P(n, k)) = \gamma_t(P(n, k))$.

5. There exists a simple algorithm to compute the exact value of $\gamma_{[1,j]}(P(n, k))$ and $\gamma_{t[1,j]}(P(n, k))$?

The last part of this thesis was devoted to study a recent problem namely edge monitoring problem. It can be considered as variant of dominating sets problem. This problem was originally motivated by its security application on wireless sensor networks. We presented the problem from theoretical point of view and detailed its variants. Then, some bounds on the edge monitoring number are given in general graphs. Afterwards, we first focused on 1-uniform monitoring problem by presenting some results on general graph and hardness proofs. The problem was also studied in particular classes of graphs: power of a cycle, power of a path, planar unit disc graph, split graph and comparability graph. An algorithm for finding the minimum 1-uniform edge monitoring set on the square of tree was also presented. Then, we studied the edge monitoring problem from the perspective of parameterized complexity. We proved that the edge monitoring problem is $W[2]$-hard when parameterized by the size of the solution. Moreover, we presented two algorithms that solve the problem in general graphs and in the particular case of apex-minor free graphs. Finally, we gave different study results about weighted edge monitoring problem on several graph classes: complete graphs, blocks graphs, interval graphs, cographs.

Even if this part contains several results on the edge monitoring problem and its variants, a number of issues need to be further investigated.

1. A natural extension is to study the problem in other types of graphs e.g. permutation graphs, strongly chordal graphs and maximal planar graphs.

2. Consider the variant of the problem in which each vertex can monitor only a fixed number of edges t, namely bounded edge monitoring problem.

Bibliography

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009. (Cited on page 16.)

[ABS90] B Alspach, J-C Bermond, and D Sotteau. Decomposition into cycles i: Hamilton decompositions. In *Cycles and Rays*, pages 9–18. Springer, 1990. (Cited on pages 28, 29 and 32.)

[ACG+12] Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer Science & Business Media, 2012. (Cited on page 18.)

[AG01] Brian Alspach and Heather Gavlas. Cycle decompositions of K_n and $K_n - I$. *Journal of Combinatorial Theory, Series B*, 81(1):77–99, 2001. (Cited on pages 31 and 34.)

[AGŠV03] Brian Alspach, Heather Gavlas, Mateja Šajna, and Helen Verrall. Cycle decompositions iv: complete directed graphs and fixed length directed cycles. *Journal of Combinatorial Theory, Series A*, 103(1):165–208, 2003. (Cited on page 31.)

[AL14] Atif A Abueida and Chester Lian. On the decompositions of complete graphs into cycles and stars on the same number of edges. *Discussiones Mathematicae Graph Theory*, 34(1):113–125, 2014. (Cited on pages 32, 36, 44 and 45.)

[ASSC02] Ian F Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer networks*, 38(4):393–422, 2002. (Cited on page 76.)

[AvG09] Rui Abreu and Arjan JC van Gemund. A low-cost approximate minimal hitting set algorithm and its application to model-based diagnosis. In *SARA*, volume 9, pages 2–9, 2009. (Cited on page 113.)

[BBP08] Arash Behzad, Mehdi Behzad, and Cheryl E Praeger. On the domination number of the generalized petersen graphs. *Discrete Mathematics*, 308(4):603–610, 2008. (Cited on page 54.)

[BCS10] Elizabeth J Billington, Nicholas J Cavenagh, and Benjamin R Smith. Path and cycle decompositions of complete equipartite graphs: 3 and 5 parts. *Discrete Mathematics*, 310(2):241–254, 2010. (Cited on page 31.)

[BDGP16] Arijit Bishnu, Kunal Dutta, Arijit Ghosh, and Subhabrata Paul. (1, j)-set problem in graphs. *Discrete Mathematics*, 339(10):2515–2525, 2016. (Cited on page 54.)

[BDTC05] Jeremy Blum, Min Ding, Andrew Thaeler, and Xiuzhen Cheng. Connected dominating set in sensor networks and manets. In Ding-Zhu Du and PanosM. Pardalos, editors, *Handbook of Combinatorial Optimization*, pages 329–369. Springer US, 2005. (Cited on page 50.)

[Bei96] LW Beineke. Graph decompositions. *Congressus Numerantium*, pages 213–226, 1996. (Cited on page 29.)

[Ber62] Claude Berge. *The theory of graphs*. Courier Corporation, 1962. (Cited on page 7.)

[Ber78] J-C Bermond. Hamiltonian decompositions of graphs, directed graphs and hypergraphs. *Annals of Discrete Mathematics*, 3:21–28, 1978. (Cited on pages 26 and 31.)

[Ber84] Alan A Bertossi. Dominating sets for split and bipartite graphs. *Information processing letters*, 19(1):37–40, 1984. (Cited on page 112.)

[BF76] Jean-Claude Bermond and V Faber. Decomposition of the complete directed graph into k-circuits. *Journal of Combinatorial Theory, Series B*, 21(2):146–155, 1976. (Cited on page 31.)

[BF11] Roberto Barrera and Daniela Ferrero. Power domination in cylinders, tori, and generalized petersen graphs. *Networks*, 58(1):43–49, 2011. (Cited on page 55.)

[BFM89] J-C Bermond, Odile Favaron, and Maryvonne Maheo. Hamiltonian decomposition of cayley graphs of degree 4. *Journal of Combinatorial Theory, Series B*, 46(2):142–153, 1989. (Cited on page 32.)

[BG12] János Barát and Dániel Gerbner. Edge-decomposition of graphs into copies of a tree with four edges. *arXiv preprint arXiv:1203.1671*, 2012. (Cited on page 32.)

[BHK15] Fairouz Beggas, Mohammed Haddad, and Hamamache Kheddouci. Decomposition of complete multigraphs into stars and cycles. *Discussiones Mathematicae Graph Theory*, 35(4):629–639, 2015. (Cited on pages 32 and 33.)

[BHMS11] Darryn Bryant, Daniel Horsley, Barbara Maenhaut, and Benjamin R Smith. Cycle decompositions of complete multigraphs. *Journal of Combinatorial Designs*, 19(1):42–69, 2011. (Cited on pages 31 and 34.)

[Big93] Norman Biggs. *Algebraic graph theory*. Cambridge university press, 1993. (Cited on page 13.)

[Bil99] Elizabeth J Billington. Decomposing complete tripartite graphs into cycles of lengths 3 and 4. *Discrete mathematics*, 197:123–135, 1999. (Cited on page 31.)

[Bil04] Elizabeth J Billington. Multipartite graph decomposition: cycles and closed trails. *Le Matematiche*, 59(I-II):53–72, 2004. (Cited on page 29.)

[BJ15] Fábio Botler and Andrea Jiménez. On path decompositions of 2k-regular graphs. *Electronic Notes in Discrete Mathematics*, 50:163–168, 2015. (Cited on page 31.)

[BJT10] Ali Behtoei, Mohsen Jannesari, and Bijan Taeri. A characterization of block graphs. *Discrete Applied Mathematics*, 158(3):219–221, 2010. (Cited on page 14.)

[BK98] Heinz Breu and David G. Kirkpatrick. Unit disk graph recognition is np-hard. *Comput. Geom.*, 9(1-2):3–24, 1998. (Cited on page 86.)

[BLW76] Norman Biggs, E Keith Lloyd, and Robin J Wilson. *Graph Theory, 1736-1936*. Oxford University Press, 1976. (Cited on page 20.)

[BM76] John Adrian Bondy and Uppaluri Siva Ramachandra Murty. *Graph theory with applications*, volume 290. Macmillan London, 1976. (Cited on pages 7 and 20.)

[BM04] Vladimir Batagelj and Andrej Mrvar. Pajekanalysis and visualization of large networks. In *Graph drawing software*, pages 77–103. Springer, 2004. (Cited on page 27.)

[BN16] Fairouz Beggas and Brahim Neggazi. A note on hamiltonian decomposition of bubble-sort graphs. *International Journal of Computer Mathematics*, 93(7):1074–1077, 2016. (Cited on page 32.)

[Bol13] Béla Bollobás. *Modern graph theory*, volume 184. Springer Science & Business Media, 2013. (Cited on page 20.)

[Bon91] Danail Bonchev. *Chemical graph theory: introduction and fundamentals*, volume 1. CRC Press, 1991. (Cited on page 20.)

[Bos90] Juraj Bosák. *Decompositions of graphs*, volume 47. Taylor & Francis, 1990. (Cited on page 30.)

[BS75] JC Bermond and D Sotteau. Graph decompositions and g-designs. In *5th British Combinatorial conference*, pages 53–72, 1975. (Cited on page 29.)

[BS+99] Andreas Brandstädt, Jeremy P Spinrad, et al. *Graph classes: a survey*, volume 3. Siam, 1999. (Cited on page 9.)

[BŠ07] Boštjan Brešar and Tadeja Kraner Šumenjak. On the 2-rainbow domination in graphs. *Discrete Applied Mathematics*, 155(17):2394–2400, 2007. (Cited on page 55.)

[BZ03] Vladimir Batagelj and Matjaz Zaversnik. An o (m) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049*, 2003. (Cited on page 27.)

[Cav98] Nicholas J Cavenagh. Decompositions of complete tripartite graphs into k-cycles. *Australasian Journal of Combinatorics*, 18:193–200, 1998. (Cited on page 31.)

[CB00] Nicholas J Cavenagh and Elizabeth J Billington. Decompositions of complete multipartite graphs into cycles of even length. *Graphs and Combinatorics*, 16(1):49–65, 2000. (Cited on page 31.)

[CC08] Miroslav Chlebík and Janka Chlebíková. Approximation hardness of dominating set problems in bounded degree graphs. *Information and Computation*, 206(11):1264–1275, 2008. (Cited on page 114.)

[CCJ90a] Brent N Clark, Charles J Colbourn, and David S Johnson. Unit disk graphs. *Discrete mathematics*, 86(1):165–177, 1990. (Cited on page 15.)

[CCJ90b] Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86(1-3):165–177, 1990. (Cited on page 87.)

[CDH80] Ernest J Cockayne, RM Dawes, and Stephen T Hedetniemi. Total domination in graphs. *Networks*, 10(3):211–219, 1980. (Cited on pages 51 and 55.)

[CDHH04] Ernie J Cockayne, Paul A Dreyer, Sandra M Hedetniemi, and Stephen T Hedetniemi. Roman domination in graphs. *Discrete Mathematics*, 278(1):11–22, 2004. (Cited on page 20.)

[CFK+15] Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*, volume 4. Springer, 2015. (Cited on page 19.)

[CG81] FRK Chung and RL Graham. Recent results in graph decompositions. *London Mathematical Society, Lecture Note Series*, 52:103–123, 1981. (Cited on page 29.)

[CH77a] Ernest J Cockayne and BL Hartnell. Edge partitions of complete multipartite graphs into equal length circuits. *Journal of Combinatorial Theory, Series B*, 23(2):174–183, 1977. (Cited on page 31.)

[CH77b] Ernest J Cockayne and Stephen T Hedetniemi. Towards a theory of domination in graphs. *Networks*, 7(3):247–261, 1977. (Cited on page 55.)

[CHHM13] Mustapha Chellali, Teresa W Haynes, Stephen T Hedetniemi, and Alice McRae. [1, 2]-sets in graphs. *Discrete Applied Mathematics*, 161(18):2885–2893, 2013. (Cited on pages 2, 53, 54 and 57.)

[CHR92] Charles J Colbourn, Dean G Hoffman, and CA Rodger. Directed star decompositions of the complete directed graph. *Journal of graph theory*, 16(5):517–528, 1992. (Cited on page 31.)

[CJMZ08] Gary Chartrand, Garry L Johns, Kathleen A McKeon, and Ping Zhang. Rainbow connection in graphs. *Mathematica Bohemica*, 133(1):85–98, 2008. (Cited on page 20.)

[Coc78] EJ Cockayne. Domination of undirected graphsa survey. In *Theory and Applications of Graphs*, pages 141–147. Springer, 1978. (Cited on page 50.)

[CT91] Edith Cohen and Michael Tarsi. Np-completeness of graph decomposition problems. *Journal of Complexity*, 7(2):200–212, 1991. (Cited on page 28.)

[Deo94] Narsingh Deo. Graph theory with applications to engineering and computer science. 1994. (Cited on page 20.)

[DF12] Rodney G Downey and Michael Ralph Fellows. *Parameterized complexity*. Springer Science & Business Media, 2012. (Cited on page 19.)

[DF13] Rodney G Downey and Michael R Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013. (Cited on pages 19 and 114.)

[DFHT04] Erik D. Demaine, Fedor V. Fomin, MohammadTaghi Hajiaghayi, and Dimitrios M. Thilikos. Subexponential parameterized algorithms on graphs of bounded-genus and H-minor-free graphs. In *In Proc. 15th SODA*, pages 830–839. Society for Industrial and Applied Mathematics, 2004. (Cited on page 109.)

[DH90] Shantanu Dutt and John P. Hayes. On designing and reconfiguring k-fault-tolerant tree architectures. *IEEE Transactions on Computers*, 39(4):490–503, 1990. (Cited on page 26.)

[DH08] Erik D. Demaine and MohammadTaghi Hajiaghayi. Bidimensionality. In *Encyclopedia of Algorithms*. 2008. (Cited on page 109.)

[DHHM00] Gayla S Domke, Johannes H Hattingh, Michael A Henning, and Lisa R Markus. Restrained domination in trees. *Discrete Mathematics*, 211(1):1–9, 2000. (Cited on page 51.)

[DHT04] ErikD. Demaine, MohammadTaghi Hajiaghayi, and DimitriosM. Thilikos. The bidimensional theory of bounded-genus graphs. In Ji Fiala, Vclav Koubek, and Jan Kratochvíl, editors, *Mathematical Foundations of Computer Science 2004*, volume 3153 of *Lecture Notes in Computer Science*, pages 191–203. Springer Berlin Heidelberg, 2004. (Cited on page 109.)

[DLL08] Dezun Dong, Yunhao Liu, and Xiangke Liao. Self-monitoring for sensor networks. In *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 431–440. ACM, 2008. (Cited on pages 3 and 77.)

[DLL+11] Dezun Dong, Xiangke Liao, Yunhao Liu, Changxiang Shen, and Xinbing Wang. Edge self-monitoring for wireless sensor networks. *Parallel and Distributed Systems, IEEE Transactions on*, 22(3):514–527, 2011. (Cited on pages 76, 77 and 87.)

[DO95] Guoli Ding and Bogdan Oporowski. Some results on tree decomposition of graphs. *Journal of Graph Theory*, 20(4):481–499, 1995. (Cited on page 32.)

[DT92] Dorit Dor and Michael Tarsi. Graph decomposition is npc-a complete proof of holyer's conjecture. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 252–263. ACM, 1992. (Cited on page 28.)

[EJM09] B Javad Ebrahimi, Nafiseh Jahanbakht, and Ebadollah S Mahmoodian. Vertex domination of generalized petersen graphs. *Discrete Mathematics*, 309(13):4355–4361, 2009. (Cited on page 54.)

[Eul41] L. Euler. Solutio problematis ad geometriam situs pertinentis. *Academiae Sci. I. Petropolitanae*, 8:128–140, 1741. (Cited on pages 1 and 20.)

[FB07] Santo Fortunato and Marc Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007. (Cited on page 27.)

[FD89a] Abdel Aziz Farrag and Robert J Dawson. Designing optimal fault-tolerant star networks. *Networks*, 19(6):707–716, 1989. (Cited on page 26.)

[FD89b] Abdel Aziz Farrag and Robert J Dawson. Fault-tolerant extensions of complete multipartite networks. In *Distributed Computing Systems, 1989., 9th International Conference on*, pages 143–150. IEEE, 1989. (Cited on page 26.)

[FGK10] Odile Favaron, François Genest, and Mekkia Kouider. Regular path decompositions of odd regular graphs. *Journal of Graph Theory*, 63(2):114–128, 2010. (Cited on page 31.)

[FGT11] Fedor V Fomin, Petr Golovach, and Dimitrios M Thilikos. Contraction obstructions for treewidth. *Journal of Combinatorial Theory, Series B*, 101(5):302–314, 2011. (Cited on page 118.)

[FH76] Stéphane Foldes and Peter L Hammer. *Split graphs*. Universität Bonn. Institut für Ökonometrie und Operations Research, 1976. (Cited on page 14.)

[FJ85] John F Fink and Michael S Jacobson. On n-domination, n-dependence and forbidden subgraphs. In *Graph theory with applications to algorithms and computer science*, pages 301–311. John Wiley & Sons, Inc., 1985. (Cited on page 96.)

[FMN+15] Florent Foucaud, George Mertzios, Reza Naserasr, Aline Parreau, and Petru Valicov. Algorithms and complexity for metric dimension and location-domination on interval and permutation graphs. In *WG*, 2015. (Cited on pages 126 and 127.)

[For79] Marsha Forman Foregger. *Decompositions of graphs.* University of Wisconsin–Madison, 1979. (Cited on page 30.)

[Fox88] Geoffrey C Fox. A review of automatic load balancing and decomposition methods for the hypercube. In *Numerical Algorithms for Modern Parallel Computer Architectures*, pages 63–76. Springer, 1988. (Cited on page 21.)

[FP75] H-YF Feng and Theodosios Pavlidis. Decomposition of polygons into simpler components: feature generation for syntactic pattern recognition. *IEEE Transactions on Computers*, 100(6):636–650, 1975. (Cited on page 21.)

[Fre85] Greg N Frederickson. Data structures for on-line updating of minimum spanning trees, with applications. *SIAM Journal on Computing*, 14(4):781–798, 1985. (Cited on page 21.)

[FYJ09] Xueliang Fu, Yuansheng Yang, and Baoqi Jiang. On the domination number of generalized petersen graphs p (n, 2). *Discrete Mathematics*, 309(8):2445–2451, 2009. (Cited on page 54.)

[Gab80] Catherine Lynn Gabel. *A Survey of Graph Decompositions.* PhD thesis, Pennsylvania State University., 1980. (Cited on page 29.)

[GHJ+08] Wayne Goddard, Stephen T Hedetniemi, David P Jacobs, Pradip K Srimani, and Zhenyu Xu. Self-stabilizing graph protocols. *Parallel Processing Letters*, 18(01):189–199, 2008. (Cited on page 50.)

[GHM16] AK Goharshady, MR Hooshmandasl, and M Alambardar Meybodi. [1, 2]-sets and [1, 2]-total sets in trees with algorithms. *Discrete Applied Mathematics*, 198:136–146, 2016. (Cited on page 54.)

[GJ77] M. R. Garey and David S. Johnson. The rectilinear steiner tree problem in NP complete. *SIAM Journal of Applied Mathematics*, 32:826–834, 1977. (Cited on page 87.)

[GJ02] Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. wh freeman New York, 2002. (Cited on page 16.)

[Hal02] Eran Halperin. Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. *SIAM Journal on Computing*, 31(5):1608–1623, 2002. (Cited on page 113.)

[Hei93] Katherine Heinrich. Path decomposition. *Le matematiche*, 47(2):241–258, 1993. (Cited on page 29.)

[Hen09] Michael A Henning. A survey of selected recent results on total domination in graphs. *Discrete Mathematics*, 309(1):32–63, 2009. (Cited on page 50.)

[HH00] Frank Harary and Teresa W Haynes. Double domination in graphs. *Ars Combinatoria*, 55:201–214, 2000. (Cited on pages 51 and 52.)

[HHS97] Teresa W Haynes, Stephen Hedetniemi, and Peter Slater. Domination in graphs: advanced topics. 1997. (Cited on page 50.)

[HHS98] Teresa W Haynes, Stephen Hedetniemi, and Peter Slater. *Fundamentals of domination in graphs*. CRC Press, 1998. (Cited on pages 22, 50 and 51.)

[Hil84] Anthony JW Hilton. Hamiltonian decompositions of complete graphs. *Journal of Combinatorial Theory, Series B*, 36(2):125–134, 1984. (Cited on page 32.)

[HJ88] Peter J Hansen and Peter C Jurs. Chemical applications of graph theory. part i. fundamentals and topological indices. *J. Chem. Educ*, 65(7):574, 1988. (Cited on page 20.)

[HJM99] K. Heinrich, Liu J., and Yu M. p_4-decomposition of regular graphs. *Journal of Graph Theory*, 31:135–143, 1999. (Cited on page 31.)

[HK10] Michael A Henning and Adel P Kazemi. k-tuple total domination in graphs. *Discrete Applied Mathematics*, 158(9):1006–1011, 2010. (Cited on page 75.)

[HL06] Chihfan Hsin and Mingyan Liu. Self-monitoring of wireless sensor networks. *Computer Communications*, 29(4):462–476, 2006. (Cited on page 77.)

[HM77] Stephen HY Hung and NS Mendelsohn. Handcuffed designs. *Discrete Mathematics*, 18(1):23–33, 1977. (Cited on page 27.)

[Hoc82a] Dorit S Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM Journal on computing*, 11(3):555–556, 1982. (Cited on page 18.)

[Hoc82b] Dorit S Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM Journal on Computing*, 11(3):555–556, 1982. (Cited on page 113.)

[Hol80] I. Holyer. *The Complexity of Graph Theory Problems*. Dissertation, Churchill College, 1980. (Cited on page 28.)

[HP05] Michel Habib and Christophe Paul. A simple linear time algorithm for cograph recognition. *Discrete Applied Mathematics*, 145(2):183–197, 2005. (Cited on page 134.)

[HR86] Anthony JW Hilton and Christopher A Rodger. Hamiltonian decompositions of complete regular s-partite graphs. *Discrete mathematics*, 58(1):63–78, 1986. (Cited on page 32.)

[HR06] Refael Hassin and Shlomi Rubinstein. An approximation algorithm for maximum triangle packing. *Discrete Applied Mathematics*, 154(6):971–979, 2006. (Cited on page 75.)

[HRCW72] Haim Hanani, Dwijendra K Ray-Chaudhuri, and Richard M Wilson. On resolvable designs. *Discrete Mathematics*, 3(4):343–357, 1972. (Cited on page 27.)

[HS93] Derek Allan Holton and John Sheehan. *The Petersen Graph*, volume 7. Cambridge University Press, 1993. (Cited on page 13.)

[HT73] John Hopcroft and Robert Tarjan. Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378, 1973. (Cited on page 126.)

[Hua91] Qingxue Huang. On the decomposition of kn into complete m-partite graphs. *Journal of graph theory*, 15(1):1–6, 1991. (Cited on page 32.)

[HWK+05] Xiaoyan Hong, Pu Wang, Jiejun Kong, Qunwei Zheng, and Jun Liu. Effective probabilistic approach protecting sensor traffic. In *Military Communications Conference, 2005. MILCOM 2005. IEEE*, pages 169–175. IEEE, 2005. (Cited on page 76.)

[I+78] Hideto Ikeda et al. Combinatorial file organization schemes and their experimental evaluation. *Hiroshima Mathematical Journal*, 8(3):515–544, 1978. (Cited on page 27.)

[JM14] Shanmugasundaram Jeevadoss and Appu Muthusamy. Decomposition of complete bipartite graphs into paths and cycles. *Discrete Mathematics*, 331:98–108, 2014. (Cited on page 32.)

[JM15] Shanmugasundaram Jeevadoss and Appu Muthusamy. Decomposition of complete bipartite multigraphs into paths and cycles having k edges. *Discussiones Mathematicae Graph Theory*, 35(4):715–731, 2015. (Cited on page 32.)

[JT11] Tommy R Jensen and Bjarne Toft. *Graph coloring problems*, volume 39. John Wiley & Sons, 2011. (Cited on page 20.)

[JWM09] Cao Jianxiang, Lin Weiguo, and Shi Minyong. Total domination number of generalized petersen graphs. *Intelligent Information Management*, 1(1):14–17, 2009. (Cited on page 55.)

[K+10] Adel P Kazemi et al. On the independent domination number of the generalized petersen graphs. *African Diaspora Journal of Mathematics. New Series*, 10(1):18–22, 2010. (Cited on page 55.)

[Kar72] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972. (Cited on page 113.)

[Kar05] George Karakostas. A better approximation ratio for the vertex cover problem. In *Automata, languages and programming*, pages 1043–1050. Springer, 2005. (Cited on page 113.)

[KL04] Ralf Klasing and Christian Laforest. Hardness results and approximation algorithms of k-tuple domination in graphs. *Information Processing Letters*, 89(2):75–83, 2004. (Cited on page 114.)

[Klo94a] Ton Kloks. *Treewidth: computations and approximations*, volume 842. Springer Science & Business Media, 1994. (Cited on page 111.)

[Klo94b] Ton Kloks. *Treewidth, Computations and Approximations*, volume 842 of *Lecture Notes in Computer Science*. Springer, 1994. (Cited on page 126.)

[KMW04] Fabian Kuhn, Thomas Moscibroda, and Rogert Wattenhofer. Initializing newly deployed ad hoc and sensor networks. In *Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 260–274. ACM, 2004. (Cited on page 50.)

[KNS09] Robert Krauthgamer, Joseph Seffi Naor, and Roy Schwartz. Partitioning graphs into balanced components. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 942–949. Society for Industrial and Applied Mathematics, 2009. (Cited on page 21.)

[Kot81] Anton Kotzig. Decompositions of complete graphs into isomorphic cubes. *Journal of Combinatorial Theory, Series B*, 31(3):292–296, 1981. (Cited on page 32.)

[Kub04] Marek Kubale. *Graph colorings*, volume 352. American Mathematical Soc., 2004. (Cited on page 20.)

[LA12] Peter A Lee and Thomas Anderson. *Fault tolerance: principles and practice*, volume 3. Springer Science & Business Media, 2012. (Cited on page 26.)

[Lap92] Gilbert Laporte. The traveling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2):231–247, 1992. (Cited on page 18.)

[LC03] Chung-Shou Liao and Gerard J. Chang. k-tuple domination in graphs. *Inf. Process. Lett.*, 87(1):45–50, 2003. (Cited on page 89.)

[LC13] Hung-Chih Lee and Yen-Po Chu. Multidecompositions of the balanced complete bipartite graph into paths and stars. *ISRN Combinatorics*, 2013, 2013. (Cited on page 32.)

[LC15] Hung-Chih Lee and Zhen-Chun Chen. Maximum packings and minimum coverings of multigraphs with paths and stars. *Taiwanese Journal of Mathematics*, 19(5):pp–1341, 2015. (Cited on page 32.)

[Lee13] Hung-Chih Lee. Multidecompositions of complete bipartite graphs into cycles and stars. *Ars Combinatoria*, 108:355–364, 2013. (Cited on page 32.)

[Lee15] Hung-Chih Lee. Decomposition of the complete bipartite multigraph into cycles and stars. *Discrete Mathematics*, 338(8):1362–1369, 2015. (Cited on pages 31 and 32.)

[LG10] Zhihe Liang and Jinping Guo. Decomposition of complete multigraphs into crown graphs. *Journal of Applied Mathematics and Computing*, 32(2):507–517, 2010. (Cited on page 32.)

[LGBK12] Hyo-Sang Lim, Gabriel Ghinita, Elisa Bertino, and Murat Kantarcioglu. A game-theoretic approach for high-assurance of data trustworthiness in sensor networks. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 1192–1203. IEEE, 2012. (Cited on page 76.)

[LH01] Bin Luo and Edwin R. Hancock. Structural graph matching using the em algorithm and singular value decomposition. *IEEE Transactions*

on Pattern Analysis and Machine Intelligence, 23(10):1120–1136, 2001. (Cited on page 21.)

[Lin10] Jenq-Jong Lin. Decomposition of balanced complete bipartite multigraphs into multistars. *Discrete Mathematics*, 310(5):1059–1065, 2010. (Cited on page 32.)

[LJ16] Jenq-Jong Lin and Min-Jen Jou. Cps,-decompositions of balanced complete bipartite multigraphs. *Open Journal of Discrete Mathematics*, 6(03):174, 2016. (Cited on page 32.)

[LL00] Chiang Lin and Jenq-Jong Lin. Cycle decompositions of crowns. *Discrete Mathematics*, 220(1):251–255, 2000. (Cited on page 31.)

[LL05] Hung-Chih Lee and Chiang Lin. Balanced star decompositions of regular multigraphs and λ-fold complete bipartite graphs. *Discrete mathematics*, 301(2):195–206, 2005. (Cited on page 31.)

[LL13] Hung-Chih Lee and Jenq-Jong Lin. Decomposition of the complete bipartite graph with a 1-factor removed into cycles and stars. *Discrete Mathematics*, 313(20):2354–2358, 2013. (Cited on page 32.)

[LLL09] Hung-Chih Lee, Ming-Ju Lee, and Chiang Lin. Isomorphic path decompositions of λkn, n, n (λk n, n, n) for odd n. *Taiwanese Journal of Mathematics*, 13(2A):pp–393, 2009. (Cited on page 31.)

[LLRKS85] Eugene L Lawler, Jan Karel Lenstra, AH-G Rinnooy-Kan, and David B Shmoys. traveling salesman problem.[the]. 1985. (Cited on page 20.)

[LLS+99] CA Lin, Jenq-Jong Lin, Tay-Woei Shyu, et al. Isomorphic star decompositions of multicrowns and the power of cycles. *Ars Combinatoria*, 53:249–256, 1999. (Cited on page 31.)

[Lon89] Zbigniew Lonc. Decompositions of graphs into trees. *Journal of graph theory*, 13(4):393–403, 1989. (Cited on page 32.)

[LR92] CC Lindner and CA Rodger. Decomposition into cycles ii: Cycle systems. *Contemporary design theory: a collection of surveys*, pages 325–369, 1992. (Cited on page 29.)

[LS90] Marilynn Livingston and Quentin F Stout. *Perfect dominating sets*. University of Michigan, Computer Science and Engineering Division, Department of Electrical Engineering and Computer Science, 1990. (Cited on page 51.)

[LS96] Chiang Lin and Tay-Woei Shyu. A necessary and sufficient condition for the star decomposition of complete graphs. *Journal of Graph Theory*, 23(4):361–364, 1996. (Cited on page 31.)

[Luc82] Édouard Lucas. *Récréations mathématiques*, volume 1. Gauthier-Villars, 1882. (Cited on page 30.)

[LXS13] Wen-Sheng Li, Hua-Ming Xing, and Moo Young Sohn. On the signed total domination number of generalized petersen graphs p (n, 2). *Bulletin of the Korean Mathematical Society*, 50(6):2021–2026, 2013. (Cited on page 55.)

[LY02] Lin Li and Jiang Yunfei. Computing minimal hitting sets with genetic algorithm. Technical report, DTIC Document, 2002. (Cited on page 113.)

[LZ14] Juan Liu and Xindong Zhang. The exact domination number of generalized petersen graphs p (n, k) with n= 2k and n= 2k+ 2ˆ{*}. *Computational and Applied Mathematics*, 33(2):497–506, 2014. (Cited on page 54.)

[Mar08] Dániel Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008. (Cited on page 17.)

[Mar12] Klas Markström. Even cycle decompositions of 4-regular graphs and line graphs. *Discrete Mathematics*, 312(17):2676–2681, 2012. (Cited on page 31.)

[MB98] Bruno T Messmer and Horst Bunke. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):493–504, 1998. (Cited on page 21.)

[MPS06] Jun Ma, Liqun Pu, and Hao Shen. Cycle decompositions of k_n,n-i. *SIAM Journal on Discrete Mathematics*, 20(3):603–609, 2006. (Cited on page 31.)

[MS06] Mariusz Meszka and Zdzisław Skupień. Decompositions of a complete multidigraph into nonhamiltonian paths. *Journal of Graph Theory*, 51(1):82–91, 2006. (Cited on page 31.)

[MS12] Mariusz Meszka and Zdzisław Skupień. Decompositions of a complete multidigraph into almost arbitrary paths. *Discussiones Mathematicae Graph Theory*, 32(2):357–372, 2012. (Cited on page 31.)

[NHTK14a] Brahim Neggazi, Mohammed Haddad, Volker Turau, and Hamamache Kheddouci. A self-stabilizing algorithm for edge monitoring problem. In *Stabilization, Safety, and Security of Distributed Systems*, pages 93–105. Springer, 2014. (Cited on page 76.)

[NHTK14b] Brahim Neggazi, Mohammed Haddad, Volker Turau, and Hamamache Kheddouci. A self-stabilizing algorithm for edge monitoring problem. In *Stabilization, Safety, and Security of Distributed Systems*, pages 93–105. Springer, 2014. (Cited on page 77.)

[OO62] Oystein Ore and Yystein Ore. *Theory of graphs*, volume 38. American Mathematical Society Providence, RI, 1962. (Cited on page 50.)

[Pap03] Christos H Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003. (Cited on page 16.)

[Par98] C. A. Parker. *Complete bipartite graph path decompositions*. Dissertation, Auburn University, 1998. (Cited on page 31.)

[Pet96] Vojislav Petrović. Decomposition of some planar graphs into trees. *Discrete Mathematics*, 150(1):449–451, 1996. (Cited on page 32.)

[PLH82] J. Pfaff, R.C. Laskar, and S.T. Hedetniemi. Np-completeness of total and connected domination and irredundance for bipartite graphs. Technical Report 428, Clemson University, Dept. Math. Sciences, 1982. (Cited on page 90.)

[PM09] HM Priyadharsini and A Muthusamy. (gm, hm)-multifactorization of λkm. *J. Combin. Math. Combin. Comput*, 69:145–150, 2009. (Cited on page 32.)

[PM12] HM Priyadharsini and A Muthusamy. (gm, hm)-multidecomposition of $k_{m,m}(\lambda)$. *Bull. Inst. Combin. Appl.*, 66:42–48, 2012. (Cited on page 32.)

[Pri13] HM Priyadharsini. *MULTIDECOMPOSITION AND MULTIFACTORIZATION OF MULTIGRAPHS*. PhD thesis, Bharathidasan University, 2013. (Cited on page 29.)

[PSM+11] Georgios A Pavlopoulos, Maria Secrier, Charalampos N Moschopoulos, Theodoros G Soldatos, Sophia Kossida, Jan Aerts, Reinhard Schneider, and Pantelis G Bagos. Using graph theory to analyze biological networks. *BioData mining*, 4(1):1, 2011. (Cited on page 20.)

[PvD10] Hof Pim vant and Paulusma Danil. A new characterization of p6 -free graphs. *Discrete Applied Mathematics*, 158(7):731–740, 2010. (Cited on page 112.)

[RIBJ15] Mohsen Rezvani, Aleksandar Ignjatovic, Elisa Bertino, and Somesh Jha. Secure data aggregation technique for wireless sensor networks in the presence of collusion attacks. *Dependable and Secure Computing, IEEE Transactions on*, 12(1):98–110, 2015. (Cited on page 76.)

[Rod91] Chris A Rodger. Graph decomposition. *Le Matematiche*, 45(1):119–140, 1991. (Cited on pages 29 and 30.)

[Šaj02] Mateja Šajna. Cycle decompositions III: Complete graphs and fixed length cycles. *Journal of Combinatorial Designs*, 10(1):27–78, 2002. (Cited on pages 31 and 34.)

[SC10] Lei Shi and Xuan Cai. An exact fast algorithm for minimum hitting set. In *Proceedings of the 3rd International Joint Conference on Computational Science and Optimization (CSO10). IEEE Computer Society*, pages 64–67, 2010. (Cited on page 113.)

[Sch75] J Schönheim. Partition of the edges of the directed complete graph into 4-cycles. *Discrete Mathematics*, 11(1):67–70, 1975. (Cited on page 31.)

[Sco12] John Scott. *Social network analysis*. Sage, 2012. (Cited on page 27.)

[Shy07] Tay-Woei Shyu. Path decompositions of $k_{n,n}$. *Ars Combinatoria*, 85:211–219, 2007. (Cited on page 31.)

[Shy10a] Tay-Woei Shyu. Decomposition of complete graphs into paths and stars. *Discrete Mathematics*, 310(15):2164–2169, 2010. (Cited on page 32.)

[Shy10b] Tay-Woei Shyu. Decompositions of complete graphs into paths and cycles. *Ars Combinatoria*, 97:257–270, 2010. (Cited on page 32.)

[Shy12] Tay-Woei Shyu. Decomposition of complete graphs into paths of length three and triangles. *Ars Combinatoria*, 107:209–224, 2012. (Cited on page 32.)

[Shy13] Tay-Woei Shyu. Decomposition of complete bipartite graphs into paths and stars with same number of edges. *Discrete Mathematics*, 313(7):865–871, 2013. (Cited on page 32.)

[Shy15] Tay-Woei Shyu. Decomposition of complete bipartite digraphs and complete digraphs into directed paths and directed cycles of fixed even length. *Graphs and Combinatorics*, 31(5):1715–1725, 2015. (Cited on page 32.)

[Sla96] Petr Slavík. A tight analysis of the greedy algorithm for set cover. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 435–441. ACM, 1996. (Cited on page 113.)

[SLY+14] Zehui Shao, Meilian Liang, Chuang Yin, Xiaodong Xu, Polona Pavlič, and Janez Žerovnik. On rainbow domination numbers of graphs. *Information Sciences*, 254:225–234, 2014. (Cited on page 55.)

[Sot80] Dominique Sotteau. *Decompositions de graphes et hypergraphes*. Dissertation, L'universite de Paris-sud, 1980. (Cited on page 29.)

[Sot81] Dominique Sotteau. Decomposition of $K_{m,n}$ $(K_{m,n})^*$ into cycles (circuits) of length $2k$. *Journal of Combinatorial Theory, Series B*, 30(1):75–81, 1981. (Cited on pages 31 and 34.)

[SS+73] Peter HA Sneath, Robert R Sokal, et al. *Numerical taxonomy. The principles and practice of numerical classification.* 1973. (Cited on page 20.)

[Ste99] Ian Stewart. Defend the roman empire! *Scientific American*, 281:136–138, 1999. (Cited on page 20.)

[Sti07] Douglas R Stinson. *Combinatorial designs: constructions and analysis*. Springer Science & Business Media, 2007. (Cited on page 26.)

[SXZF07] Jimeng Sun, Yinglian Xie, Hui Zhang, and Christos Faloutsos. Less is more: Compact matrix decomposition for large sparse graphs. In *SDM*, pages 366–377. SIAM, 2007. (Cited on page 21.)

[T+79] Shinsei Tazawa et al. Claw-decomposition and evenly-partite-claw-decomposition of complete multipartite graphs. *Hiroshima Mathematical Journal*, 9(2):503–531, 1979. (Cited on page 31.)

[Tar79] Michael Tarsi. Decomposition of complete multigraphs into stars. *Discrete Mathematics*, 26(3):273–278, 1979. (Cited on pages 28, 31 and 34.)

[Tar83] Michael Tarsi. Decomposition of a complete multigraph into simple paths: nonbalanced handcuffed designs. *Journal of Combinatorial Theory, Series A*, 34(1):60–70, 1983. (Cited on pages 28 and 31.)

[Tho01] Andrew Thomason. The extremal function for complete minors. *Journal of Combinatorial Theory, Series B*, 81(2):318–338, 2001. (Cited on page 118.)

[TLYL09] Chunling Tong, Xiaohui Lin, Yuansheng Yang, and Meiqin Luo. 2-rainbow domination of generalized petersen graphs p (n, 2). *Discrete Applied Mathematics*, 157(8):1932–1937, 2009. (Cited on page 55.)

[Tru85] Miroslaw Truszczynski. Note on the decomposition of $\lambda k_{m,n}(\lambda k)$ into paths. *Discrete Mathematics*, 55(1):89–96, 1985. (Cited on page 31.)

[Tve82] Helge Tverberg. On the decomposition of kn into complete bipartite graphs. *Journal of Graph Theory*, 6(4):493–494, 1982. (Cited on page 32.)

[Ush93] Kazuhiko Ushio. G-designs and related designs. *Discrete mathematics*, 116(1):299–311, 1993. (Cited on page 29.)

[UTY+78] Kazuhiko Ushio, Shinsei Tazawa, Sumiyasu Yamamoto, et al. On claw-decomposition of a complete multipartite graph. *Hiroshima Mathematical Journal*, 8(1):207–210, 1978. (Cited on page 31.)

[Val81] Leslie G. Valiant. Universality considerations in VLSI circuits. *IEEE Trans. Computers*, 30(2):135–140, 1981. (Cited on page 87.)

[Vaz13] Vijay V Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013. (Cited on page 18.)

[W+01] Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001. (Cited on pages 7 and 20.)

[Wag37] Klaus Wagner. Über eine eigenschaft der ebenen komplexe. *Mathematische Annalen*, 114(1):570–590, 1937. (Cited on page 12.)

[Wat69] Mark E Watkins. A theorem on tait colorings with an application to the generalized petersen graphs. *Journal of Combinatorial Theory*, 6(2):152–164, 1969. (Cited on pages 13 and 54.)

[Wes] Douglas B. West. Open problems - graph theory and combinatorics. http://www.math.illinois.edu/~dwest/openp/. (Cited on page 20.)

[WF94] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994. (Cited on pages 20 and 27.)

[Wil76] R.M. Wilson. Decomposition of a complete graph into subgraphs isomorphic to a given graph. *Proceedings of the 5th British Combinatorial Conference*, pages 647–659, 1976. (Cited on page 30.)

[WLZ08] Yu Wang, Xiang-Yang Li, and Qian Zhang. Efficient algorithms for p-self-protection problem in static wireless sensor networks. *Parallel and Distributed Systems, IEEE Transactions on*, 19(10):1426–1438, 2008. (Cited on page 77.)

[Wol62] ES Wolk. The comparability graph of a tree. *Proceedings of the American Mathematical Society*, 13(5):789–795, 1962. (Cited on page 112.)

[WZL07] Dan Wang, Qian Zhang, and Jiangchuan Liu. The self-protection problem in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 3(4):20, 2007. (Cited on page 77.)

[WZMX10] Guiyi Wei, Zhiqiang Zhu, Yunxin Mao, and Naixue Xiong. A distributed node self-monitoring mechanism in wireless sensor networks. In *Information Science and Engineering (ICISE), 2010 2nd International Conference on*, pages 1684–1687. IEEE, 2010. (Cited on page 76.)

[XK11] Guangjun Xu and Liying Kang. On the power domination number of the generalized petersen graphs. *Journal of combinatorial optimization*, 22(2):282–291, 2011. (Cited on page 55.)

[Xu09] Guangjun Xu. 2-rainbow domination in generalized petersen graphs p (n, 3). *Discrete Applied Mathematics*, 157(11):2570–2573, 2009. (Cited on page 55.)

[Xu13] Junming Xu. *Theory and application of graphs*, volume 10. Springer Science & Business Media, 2013. (Cited on page 20.)

[XYB09] Fu Xueliang, Yang Yuansheng, and Jiang Baoqi. Roman domination in regular graphs. *Discrete Mathematics*, 309(6):1528–1537, 2009. (Cited on page 55.)

[YISE+75] Sumiyasu Yamamoto, Hideto Ikeda, Shinsei Shige-Eda, Kazuhiko Ushio, Noboru Hamada, et al. On claw-decomposition of complete graphs and complete bigraphs. *Hiroshima Mathematical Journal*, 5(1):33–42, 1975. (Cited on pages 31 and 34.)

[YKR06] Ossama Younis, Marwan Krunz, and Srinivasan Ramasubramanian. Node clustering in wireless sensor networks: Recent developments and deployment challenges. *Network, IEEE*, 20(3):20–25, 2006. (Cited on page 50.)

[YKX09] Hong Yan, Liying Kang, and Guangjun Xu. The exact domination number of the generalized petersen graphs. *Discrete Mathematics*, 309(8):2596–2607, 2009. (Cited on page 54.)

[YW14] Xiaojing Yang and Baoyindureng Wu. [1, 2]-domination in graphs. *Discrete Applied Mathematics*, 175:79–86, 2014. (Cited on page 54.)

[Zel84] Bohdan Zelinka. On k-ply domatic numbers of graphs. *Mathematica Slovaca*, 34(3):313–318, 1984. (Cited on page 55.)

[Zel02] Bohdan Zelinka. Domination in generalized petersen graphs. *Czechoslovak Mathematical Journal*, 52(1):11–16, 2002. (Cited on page 55.)

List of Figures

List of Algorithms