On packing, colouring and identification problems

Petru Valicov

N° d'ordre: 4549

THÈSE

présentée à

L'UNIVERSITÉ BORDEAUX 1

École Doctorale de Mathématiques et Informatique de Bordeaux

par

Petru VALICOV

pour obtenir le grade de

DOCTEUR

SPÉCIALITÉ INFORMATIQUE

Problèmes de placement, de coloration et d'identification

Soutenue le 9 juillet 2012 au Laboratoire Bordelais de Recherche en Informatique

Après avis des rapporteurs :

M.	Frédéric Havet	CR (HDR)	INRIA Sophia-Antipolis
M.	Yannis Manoussakis	PR	Université Paris-Sud

Devant la commission d'examen composée de :

M.	Frédéric Havet	CR (HDR)	INRIA Sophia-Antipolis	**Rapporteur**
M.	Yannis Manoussakis	PR	Université Paris-Sud	**Rapporteur**
M.	Mickaël Montassier	MdC (HDR)	Université Bordeaux I	**Directeur**
M.	Arnaud Pêcher	PR	Université Bordeaux I	**Directeur**
M.	André Raspaud	PR	Université Bordeaux I	**Président**
M.	Éric Sopena	PR	Université Bordeaux I	**Directeur**
M.	Nicolas Trotignon	CR (HDR)	ENS Lyon	**Examinateur**
Mme.	Annegret Wagler	MdC (HDR)	Université Blaise-Pascal	**Examinatrice**

-2012-

Remerciements

Tout d'abord je tiens à remercier mes directeurs de thèse Arnaud, Mickaël et Éric. Merci Arnaud d'avoir encadré mon stage de M2 et d'avoir accepté d'encadrer ma thèse. Tu as supporté mon obstination pour le problème de placement et m'as laissé finaliser ce travail sans me pousser à faire autre chose. Tu m'as permis d'aborder les sujets que j'ai aimés, aussi différents qu'ils soient et tu m'as fait confiance durant ces trois années de thèse. Pour tout cela je te suis profondément reconnaissant.

Un grand merci à Mickaël de m'avoir aidé à colorier et à jouer avec les configurations réductibles. Merci également de m'avoir appris la rigueur même pour les preuves évidentes. L'attention avec laquelle tu lisais chaque fois mes notes m'a toujours surpris. J'espère maintenant, après m'avoir montré tant de failles, être attentif moi aussi!

J'ai fait la connaissance d'Éric au premier semestre de mon M2 quand je faisais mes premiers pas dans la recherche. J'ai été impressionné (et je le suis toujours !) par ta gentillesse et ta facilité à aborder les problèmes même difficiles. Je suis très heureux d'avoir eu l'occasion d'enseigner avec toi à plusieurs reprises. Et pour tout cela je te remercie.

Je tiens à remercier Frédéric Havet et Yannis Manoussakis d'avoir accepté d'être rapporteurs de cette thèse et pour leurs remarques constructives sur le manuscrit. Merci également à Nicolas Trotignon et Annegret Wagler d'avoir fait partie de mon jury. Enfin je remercie André Raspaud d'avoir présidé le jury de thèse. André m'a enseigné la théorie de graphes durant mes deux années de master, sa rigueur et son enthousiasme durant ses cours m'ont marqué définitivement et m'ont donné une motivation de plus pour étudier les graphes.

Je remercie tous les autres collègues avec qui j'ai eu l'occasion de travailler : Aline Parreau, Cédric Joncour, Eleonora Guerrini, Florent Foucaud, Hervé Hocquard, Matjaž Kovše, Pascal Ochem, Reza Naserasr, Sylvain Gravier.

Ces trois années de recherche ont été faites au sein de l'équipe Graphes et Applications de CombAlgo. Je remercie tous les membres pour les séminaires organisés et les discussions qui ont dynamisé notre groupe. Merci à l'ANR IDEA et plus particulièrement à Ralf, de nous avoir donné l'occasion de travailler sur les codes identifiants sur plusieurs sites en France et d'avoir organisé BWIC 2011.

Tout au long de mon doctorat j'ai eu la grande chance d'enseigner à l'IUT Bordeaux 1 au sein d'une superbe équipe pédagogique. Un grand merci à tous ces membres!

Je souhaite remercier également le personnel administratif du LaBRI et notamment Philippe Biais, Brigitte Cudeville, Lebna Mizani, Cathy Roubineau de m'avoir aidé dans toutes les démarches administratives (pour les problèmes de visa tunisien par exemple...).

À mes co-bureaux, qui se sont succédé au fils de ces trois années, c'était super de bosser dans cette ambiance! Merci Abbas, Anasua, Dominik, Guillaume, Nicolas, Radu et Sri de m'avoir supporté. Je tiens particulièrement à remercier Sri de m'avoir soutenu et poussé à travailler sur la rédaction de ma thèse au moment où j'en avais perdu un peu la motivation.

Je n'oublie pas tous mes amis (qu'ils soient du LaBRI ou pas) : Aditya, Alizée, Anna, Anne-Claire, Brice, Edita, Eleonora, Eugen, Florent, Gaël, Hervé, Lorenzo, Mathieu, Reza, Sagnik, Sid, Svetlana R. et T., Youssouf.

Je remercie l'AFoDIB pour l'organisation des événements en dehors du cadre de travail. Merci

à tous les doctorants avec qui on s'est vraiment amusés pendant les séminaires, apéros-docs ou les journées sportives. Je pense à Adrien, Allyx, Anaïs, Clément, Émilie, Eve, Gabriel, Lorijn, Min, Pierre, Razanne, Thomas, Vincent... et j'en oublie certainement quelques-uns.

Enfin j'ai une petite pensée pour ma famille. Ils m'ont soutenu pendant toute la durée de mes études en Moldavie et en France. Mes parents, mes grands-parents (dont certains n'ont pas réussi à voir cette thèse finalisée), mon frère et ma sœur et leurs familles. Pour tout cela je vous témoigne une profonde reconnaissance. Mulţumesc pentru tot!

Problèmes de placement, de coloration et d'identification

Résumé : Dans cette thèse, nous nous intéressons à trois problèmes issus de l'informatique théorique, à savoir le placement de formes rectangulaires dans un conteneur (OPP), la coloration dite "forte" d'arêtes des graphes et les codes identifiants dans les graphes.

L'OPP consiste à décider si un ensemble d'items rectangulaires peut être placé sans chevauchement dans un conteneur rectangulaire et sans dépassement des bords de celui-ci. Une contrainte supplémentaire est prise en compte, à savoir l'interdiction de rotation des items. Le problème est NP-difficile même dans le cas où le conteneur et les formes sont des carrés. Nous présentons un algorithme de résolution efficace basé sur une caractérisation du problème par des graphes d'intervalles, proposée par Fekete et Schepers. L'algorithme est exact et utilise les MPQ-arbres - structures de données qui encodent ces graphes de manière compacte tout en capturant leurs propriétés remarquables. Nous montrons les résultats expérimentaux de notre approche en les comparant aux performances d'autres algorithmes existants.

L'étude de la coloration forte d'arêtes et des codes identifiants porte sur les aspects structurels et de calculabilité de ces deux problèmes. Dans le cas de la coloration forte d'arêtes nous nous intéressons plus particulièrement aux familles des graphes planaires et des graphes subcubiques. Nous montrons des bornes optimales pour l'indice chromatique fort χ'_s des graphes subcubiques en fonction du degré moyen maximum et montrons que tout graphe planaire subcubique sans cycles induits de longueur 4 et 5 est coloriable avec neuf couleurs. Enfin nous confirmons la difficulté du problème de décision associé, en prouvant qu'il est NP-complet dans des sous-classes restreintes des graphes planaires subcubiques.

La troisième partie de la thèse est consacrée aux codes identifiants. Nous proposons une caractérisation des graphes identifiables dont la cardinalité du code identifiant minimum γ^{ID} est $n-1$, où n est l'ordre du graphe. Nous étudions la classe des graphes adjoints et nous prouvons des bornes inférieures et supérieures serrées pour le paramètre γ^{ID} dans cette classe. Finalement, nous montrons qu'il existe un algorithme linéaire de calcul de γ^{ID} dans la classe des graphes adjoints $L(G)$ où G a une largeur arborescente bornée par une constante. En revanche nous nous apercevons que le problème est NP-complet dans des sous-classes très restreintes des graphes parfaits.

Mots clefs : problème de placement orthogonal, graphes d'intervalles, MPQ-arbres, coloration forte d'arêtes, déchargement, graphes planaires, graphes parfaits, graphes adjoints, codes identifiants, complexité

Discipline : Informatique

LaBRI
Université Bordeaux 1
351 cours de la Libération,
33405 Talence Cedex (FRANCE)

On packing, colouring and identification problems

Abstract : In this thesis we study three theoretical computer science problems, namely the orthogonal packing problem (OPP for short), strong edge-colouring and identifying codes.

OPP consists in testing whether a set of rectangular items can be packed in a rectangular container without overlapping and without exceeding the borders of this container. An additional constraint is that the rotation of the items is not allowed. The problem is NP-hard even when the problem is reduced to packing squares in a square. We propose an exact algorithm for solving OPP efficiently using the characterization of the problem by interval graphs proposed by Fekete and Schepers. For this purpose we use some compact representation of interval graphs - MPQ-trees. We show experimental results of our approach by comparing them to the results of other algorithms known in the literature. We observe promising gains.

The study of strong edge-colouring and identifying codes is focused on the structural and computational aspects of these combinatorial problems. In the case of strong edge-colouring we are interested in the families of planar graphs and subcubic graphs. We show optimal upper bounds for the strong chromatic index χ'_s of subcubic graphs as a function of the maximum average degree. We also show that every planar subcubic graph without induced cycles of length 4 and 5 can be strong edge-coloured with at most nine colours. Finally, we confirm the difficulty of the problem by showing that it remains NP-complete even in some restricted classes of planar subcubic graphs.

For the subject of identifying codes we propose a characterization of non-trivial graphs having maximum identifying code number γ^{ID}, that is $n-1$, where n is the number of vertices. We study the case of line graphs and prove lower and upper bounds for γ^{ID} parameter in this class. At last we investigate the complexity of the corresponding decision problem and show the existence of a linear algorithm for computing γ^{ID} of the line graph $L(G)$ where G has the size of the tree-width bounded by a constant. On the other hand, we show that the identifying code problem is NP-complete in various subclasses of planar graphs.

Keywords : orthogonal packing problem, interval graphs, MPQ-trees, strong edge-colouring, discharging, planar graphs, perfect graphs, line graphs, identifying codes, complexity

Discipline : Computer Science

LaBRI
Université Bordeaux 1
351 cours de la Libération,
33405 Talence Cedex (FRANCE)

Contents

Chapter 1

Introduction

In this chapter we begin with some standard definitions and notations. We follow up with common concepts and classes of graphs which will be used later on in this thesis. At the end of the chapter we present an overview of the problems considered.

1.1 Preliminaries

Graph A graph $G = (V, E)$ is a set of elements V and a symmetric binary relation E defined on V. The elements of V are called *vertices* and the elements of E are called *edges*. To simplify notations, we will write $V(G)$ (respectively $E(G)$) for the set of vertices (respectively edges) of a graph G. Two vertices u and v are *adjacent* if there exists an edge $uv \in E$. Two distinct edges uv and xy are adjacent if they share an endpoint ($u = x$ for example). A graph is *finite* if the vertex set V is finite. A graph is *simple* if there can be at most one edge between every two vertices. Finally, a graph has a *loop* if there exists a vertex $v \in V$ such that $vv \in E$, that is, v has an edge back to itself. In this document, unless specified, the graphs which are considered will be finite, simple and without loops.

Subgraph A subgraph of a graph $G = (V, E)$ is a graph $H = (V', E')$ such that $V' \subseteq V$ and $E' \subseteq E(V')$. If for every edge $xy \in E(G)$ with $x, y \in V'$, $xy \in E(H)$, then H is said to be the subgraph of G *induced* by V' and is denoted $G[V']$.

Distance Two vertices u and v are at distance 1 if $uv \in E$. More generally, u and v are at distance k, if the length of a shortest path from u to v (the length being the number of edges) is k. For two vertices x and y of a graph G, $\text{dist}_G(x, y)$ (or $\text{dist}(x, y)$ if there is no ambiguity) denotes the distance between x and y in G.

Neighbourhood, degree The adjacent vertices of a vertex v are also called neighbours of v. The *open neighbourhood* of a vertex v, denoted by $N(v)$, is the set of vertices adjacent to v and the *closed neighbourhood*, denoted by $N[v]$, is defined as the union $N(v) \cup \{v\}$. Two vertices u and v are called *twins* in G if $N[u] = N[v]$. The number of vertices adjacent to a vertex v is the *degree* of v, denoted by $d_G(v)$ ($d(v)$ if no ambiguity). We denote the maximum degree of a graph G by $\Delta(G)$ (Δ if no ambiguity). A graph in which all vertices have the same degree k is said to

be *k-regular*. A class of graphs which we study in particular is the one of *subcubic* graphs which are graphs with maximum degree at most 3.

Connectivity A graph G is *connected* if for every pair of vertices u, v there exists a path between u and v in G. Otherwise the graph is said to be *disconnected*. A *k-connected* graph G is such that for every subset S of vertices with $|S| = k - 1$, $G - S$ is connected.

Girth The girth of a graph is the length of one of its shortest cycles.

Cliques and stable sets A *clique* is a set of vertices which induces a *complete graph* that is a graph in which all vertices are pairwise adjacent. A clique of a graph G is a set of vertices of G inducing a subgraph which is isomorphic to a complete graph. The clique number $\omega(G)$ is the order of a maximum clique of G. In contrast, a *stable* or *independent* set of a graph G is a subset of pairwise non-adjacent vertices of G. Given an integer $k \geq 2$, a subset $\mathcal{I}$ of vertices of G is called a *k-independent* set if for all distinct vertices x, y of $\mathcal{I}$, $\text{dist}_G(x, y) \geq k$. A 2-independent set is simply an independent set.

1.2 Some concepts on graphs

Operations on graphs

The complement of a graph G, denoted by $\overline{G}$, is the graph whose vertex set is $V(G)$ with two vertices being adjacent if and only if there is no edge between them in G.

Given a graph G, the *k-th power* G^k is the graph with vertex set $V(G)$ such that two vertices are adjacent in G^k if and only if their distance in G is at most k.

We denote by $G \setminus uv$ the graph obtained from G by removing the edge uv. Similarly, $G - v$ denotes the graph obtained from G by removing vertex v from $V(G)$ and all edges having v as an endpoint.

The *join* of two graphs, $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, denoted $G_1 \bowtie G_2$, is a graph whose vertex set is $V_1 \cup V_2$ and its edge set is $E_1 \cup E_2 \cup \{v_1v_2 \mid v_1 \in V_1, v_2 \in V_2\}$.

Vertex and edge-colourings

A *proper k-colouring* of vertices of a graph G is a mapping from the vertex set to the set of integers $\{1, \ldots, k\}$ (called colours) such that two adjacent vertices have distinct colours. A graph is *k-colourable* if it admits a proper k-colouring. The *chromatic number* of G, denoted $\chi(G)$, is the smallest integer k such that G admits a k-colouring.

The notion of edge-colouring is defined similarly. A *proper k-edge-colouring* of a graph G is a mapping from the edge set to the set of integers $\{1, \ldots, k\}$ such that two adjacent edges have distinct colours. A graph is *k-edge-colourable* if it admits a proper k-edge-colouring. The *chromatic index* of G, denoted $\chi'(G)$, is the smallest integer k such that G admits a proper k-edge-colouring.

Dominating sets and Vertex covers

Given a graph G, a subset of vertices $\mathcal{D} \subseteq V(G)$ is called a *dominating set* if for every vertex $v \in V(G)$, $\mathcal{D} \cap N[v] \neq \emptyset$.

A *vertex cover* of a graph G is a subset of vertices $\mathcal{C} \subseteq V(G)$ such that for every edge $uv \in E(G)$, $\{u, v\} \cap \mathcal{C} \neq \emptyset$.

Matchings

A *matching* is a set of pairwise non-adjacent edges, and a *perfect matching* is a matching which covers all the vertices of the graph.

1.3 Some classes of graphs

In this section we recall some known families of graphs. We first fix the notations for the common classes of graphs that we will use throughout this manuscript.

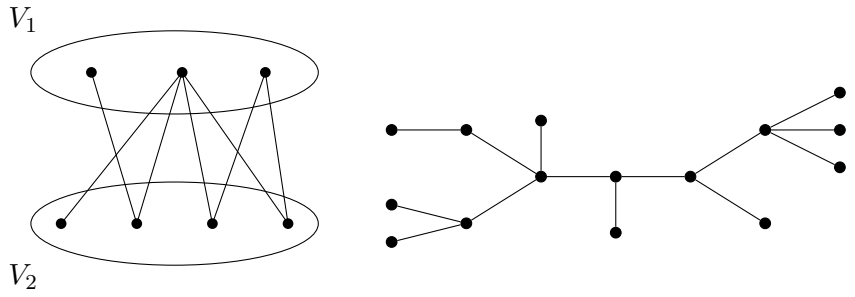
We denote by C_n, P_n and K_n the cycle, path and complete graph on n vertices respectively.

Trees, Bipartite graphs

A *tree* is a connected graph with no cycle. In particular, the path P_n is a tree. A graph whose connected components are trees is called a *forest*.

A *bipartite* graph is a graph $G = (V, E)$ such that the vertex set is partitioned $V = V_1 \dot\cup V_2$ such that V_1 and V_2 are independent sets. A *complete bipartite* graph $K_{n,m}$ is a bipartite graph with $|V_1| = n$ and $|V_2| = m$ such that $\forall u \in V_1$ and $\forall v \in V_2$, uv is an edge of this graph. In other words $K_{n,m}$ contains the maximum possible number of edges.

It is an easy fact that forests are bipartite. For examples of such graphs see Figure 1.1.

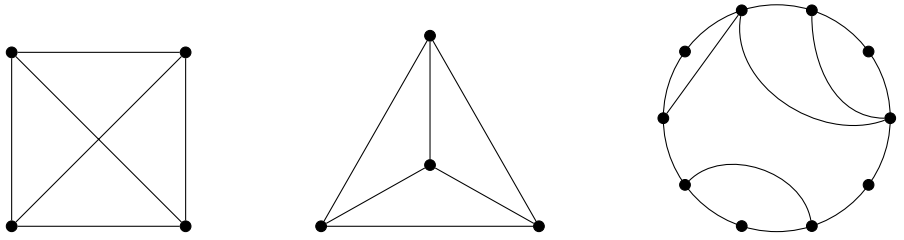


(a) A general bipartite graph (b) A tree

Figure 1.1: Bipartite graphs

Planar graphs and Outerplanar graphs

A *planar* graph is a graph that can be drawn on the plane without edges crossing (see Figure 1.2a for an example). A *face* of a planar graph G is the region bounded by the edges of G in a plane drawing of G. An *outerplanar* graph is a planar graph having a plane representation with all its vertices on the outer-face. An example is given in Figure 1.2b. Also, notice that the graph K_4 (Figure 1.2a) is not outerplanar.



(a) A non-planar and a planar drawing of K_4 (b) An outerplanar graph

Figure 1.2: Examples of planar graphs and outerplanar graphs

Hypercubes

The hypercube of dimension d, denoted $\mathcal{H}_d$, is a graph whose vertices are words of d bits such that two vertices are adjacent if the corresponding words differ on a single bit, (equivalently one can say that the *Hamming distance* between these words must be equal to 1). The hypercube of dimension d can also be viewed as a union of two disjoint copies H_0 and H_1 of $\mathcal{H}_{d-1}$, by adding a new bit with value 0 (respectively 1) to the left of the words representing the vertices of H_0 (respectively H_1) and by adding the edges between the corresponding vertices of two copies H_0 and H_1. The hypercubes for the first three dimensions are shown in Figure 1.3.

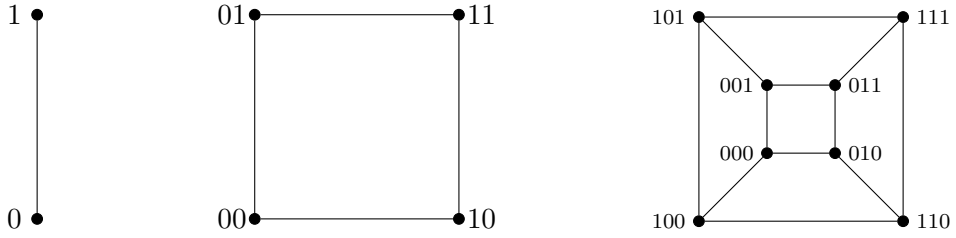


Figure 1.3: Hypercubes $\mathcal{H}_1$, $\mathcal{H}_2$, $\mathcal{H}_3$ respectively

Interval graphs

An *interval graph* is a graph $G = (V, E)$ such that there is an assignment of intervals I_v ($v \in V$) of the real line to the vertices of G such that for every pair of vertices u and v, uv is an edge if and only if $I_u \cap I_v \neq \emptyset$. An example is shown in Figure 1.4. This class of graphs will be the main topic of study in the second chapter of this manuscript, where we describe them in more details.

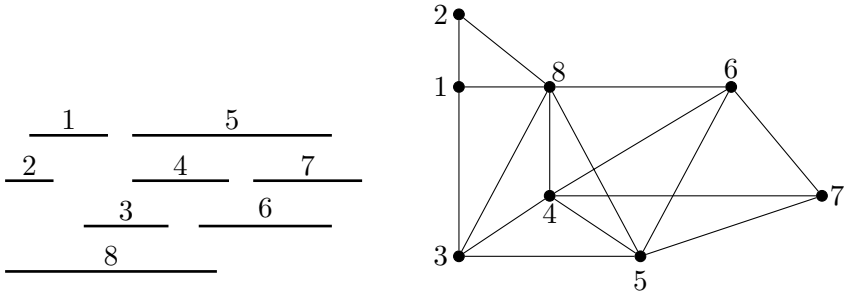


Figure 1.4: An example of interval graph

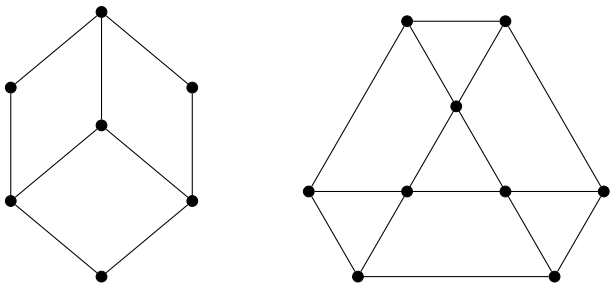
Split graphs, Chordal graphs

A graph G is a *split graph* if its set of vertices can be partitioned into two sets V_1 and V_2 such that V_1 is a clique and V_2 is an independent set.

A graph is *chordal* if it has no induced cycle of length $k \geq 4$. In particular, interval graphs and split graphs are chordal.

Line graphs

The *line* graph $L(G)$ of a graph G is the graph with vertex set $E(G)$, where two vertices of $L(G)$ are adjacent if the corresponding edges are adjacent in G. We provide an example in Figure 1.5.



(a) A graph G (b) The line graph $L(G)$ of G

Figure 1.5: An example of graph and its associated line graph

Perfect graphs

A graph G is *perfect* if for every induced subgraph H of G, $\chi(H) = \omega(H)$. The perfect graphs which we mentioned in this section are bipartite graphs, interval graphs, split graphs and chordal graphs.

1.4 Overview

The three major subjects of this thesis are *orthogonal packing*, *strong edge-colouring* and *identifying codes*. In this section we give a brief introduction to these topics. Further discussions on the problems considered are developed in the corresponding chapters. We observe that the orthogonal packing problem can be formulated as a set-packing problem, whereas strong edge-colouring and identifying codes both have a natural formulation as a set covering problem.

The set packing problem (SPP for short) is defined as follows.

Definition 1.1. Let U be a set of m elements and $\mathcal{S}$ be a family of subsets of U. A *packing* $\mathcal{P} \subseteq \mathcal{S}$ is a family of pairwise disjoint subsets $P_1, \ldots, P_k$. The *set packing* problem asks to find $\mathcal{P}$ of largest cardinality (that is the maximum number of subsets P_i). In the weighted version of the problem, each set P_i has a positive weight $w(P_i)$ and the goal is to find $\mathcal{P}$ of maximum total weight.

The set cover problem (SCP for short) is defined as follows.

Definition 1.2. Let U be a set of m elements and $\mathcal{S}$ be a family of subsets of U. A *cover* $\mathcal{C} \subseteq \mathcal{S}$ is a family of subsets $C_1, \ldots, C_k$ such that $U = C_1 \cup \ldots \cup C_k$. The *set cover* problem is to find $\mathcal{C}$ of smallest cardinality (that is the minimum number of subsets C_i). In the weighted version of the problem, each set C_i has a positive weight $w(C_i)$ and the goal is to find $\mathcal{C}$ of minimum total weight.

Set packing and set covering are two fundamental dual concepts in combinatorial optimization: each of them can be suitably transformed to the other. More precisely, let x_S be a binary decision variable such that

$$x_S = 1 \text{ if subset } S \subset \mathcal{S} \text{ is selected, 0 otherwise}$$

An integer linear programming formulation of the weighted set packing problem is:

$$\begin{aligned} \max & \sum_{S \in \mathcal{S}} w(S) x_S \\ \text{subject to} & \sum_{\substack{S \in \mathcal{S}: \\ e \in S}} x_S \leq 1, \forall e \in U \\ & x_S \in \{0,1\}, \forall S \in \mathcal{S} \end{aligned}$$

where the constraint is that for every element $e \in U$ there is at most one selected subset of $\mathcal{S}$ containing it.

The following dual formulation of the weighted set packing problem is an integer linear programming of the weighted set covering problem:

$$\begin{aligned} \min & \sum_{S \in \mathcal{S}} w(S) x_S \\ \text{subject to} & \sum_{\substack{S \in \mathcal{S}: \\ e \in S}} x_S \geq 1, \forall e \in U \\ & x_S \in \{0,1\}, \forall S \in \mathcal{S} \end{aligned}$$

and the constraint is that for every element $e \in U$, at least one subset of $\mathcal{S}$ containing it must be selected.

The Orthogonal Packing Problem

Let V be a set of D-dimensional rectangular items and let C be a D-dimensional rectangular container. Let $w_d(v) \in \mathbb{Q}^+$ and $w_d(C)$ be the length on dimension d of item $v \in V$ and C respectively. For a set of items S, we define $w_d(S) = \sum_{v \in S} w_d(v)$.

The problem which we are interested in here in this manuscript, is defined as follows:

Definition 1.3. The *D-dimensional orthogonal packing problem* (OPP-D) is to decide if the set of items V fits into the container C without overlapping and with no rotation of the items. Formally speaking, we have to find out whether there exists a function $f_d : V \to \mathbb{Q}^+$, $\forall d \in \{1, \ldots, D\}$, corresponding to the position of the left corner of each of the items on dimension d, such that:

$$\forall v \in V, f_d(v) + w_d(v) \leq w_d(C) \tag{1.1}$$

$$\forall v_1, v_2 \in V, (v_1 \neq v_2), [f_d(v_1), f_d(v_1) + w_d(v_1)) \ \cap \ [f_d(v_2), f_d(v_2) + w_d(v_2)) = \emptyset \tag{1.2}$$

Constraint 1.1 is that item v must be packed without exceeding the width of C and Constraint 1.2 models the non-overlapping of the items.

OPP-D is a sub-problem of a well-known general problem whose objective in the base case is to minimize the total unused space:

Definition 1.4. Let V be a set of D-dimensional items and C a D-dimensional container. For each $v \in V$, let p_v be an associated *profit* value. The *D-dimensional orthogonal knapsack problem* (OKP-D) consists in selecting a subset $V' \subseteq V$ such that V' is feasible and with maximum profit:

$$\max_{V' \subseteq V} \left\{ \sum_{v \in V'} p_v : V' \text{ satisfies 1.1 and 1.2} \right\}$$

We focus on the two-dimensional case as it is the most studied one. However, our approach is general enough to be applied to the d-dimensional case, for any $d \geq 1$.

We define the following binary decision variables:

$$\psi_{iab} = \begin{cases} 1 & \text{if the coordinate } (a,b) \text{ is covered by item } i \\ 0 & \text{otherwise} \end{cases}$$

$$l_{iab} = \begin{cases} 1 & \text{if the coordinate } (a,b) \text{ is covered by the bottom left corner of item } i \\ 0 & \text{otherwise} \end{cases}$$

A naive integer linear programming formulation of OKP uses space discretization ideas and can be seen as a set packing problem, which we provide below. By simplification, in the inequalities we ignore the containers' boundary constraints:

$$\max \sum_i \sum_a \sum_b p_i l_{iab} \tag{1.3}$$

$$\text{subject to } \psi_{i(a+j)(b+k)} \geq l_{iab}, \qquad \forall i,\ 0 \leq j \leq w_1(i) - 1,\ 0 \leq k \leq w_2(i) - 1 \tag{1.4}$$

$$\sum_i \psi_{iab} \leq 1, \qquad \forall a, \forall b \tag{1.5}$$

$$\sum_a \sum_b l_{iab} \leq 1, \qquad \forall i \tag{1.6}$$

$$\psi_{iab} \in \{0,1\}, l_{iab} \in \{0,1\}, \forall i, a, b \tag{1.7}$$

Constraint 1.4 ensures that a selected item i covering the coordinate (a,b) with its bottom left corner must cover all the coordinates $(a+j, b+k)$, for $0 \leq j \leq w_1(i) - 1,\ 0 \leq k \leq w_2(i) - 1$. Constraint 1.5 is an overlapping constraint: there cannot be more than one item covering coordinate (a,b). Constraint 1.6 is to ensure that an item i is used only once.

The first formulation of the problem in the bidimensional case was given by Beasley in [9] and it uses a discretization technique similar to the one described above. Since then other authors provided similar approaches [26, 7, 16]. Other mathematical programming formulations exploit the information given by the placement of items relative to each other. For a deep survey of these approaches we refer to the PhD thesis of Joncour [63].

The main line of our research (described in Chapter 2) on this problem is based on a graph-theoretic model. Specifically, inspired by the characterization of packing solutions using interval graphs given by Fekete and Schepers in [39, 40] and by the competitive results based on this characterization [41], we propose an equivalent combinatorial formulation using MPQ-trees - a data structure which gives a compact representation of interval graphs, introduced in [68]. We describe an algorithm to enumerate the MPQ-trees under several constraints, in order to produce a solution to OPP-2.

Strong edge-colouring

Previously we defined proper edge-colouring. We are interested in one variant of proper edge-colouring with an extra condition:

Definition 1.5. A *strong edge-colouring* of a graph G is a proper edge-colouring of G such that every two edges at distance exactly 2 are also assigned distinct colours. The *strong chromatic index* of G, denoted $\chi'_s(G)$, is the smallest integer k such that G can be strong edge-coloured with k colours.

Clearly, we have $\chi'_s(G) = \chi(L(G)^2)$, where χ is the classical chromatic number.

This observation gives rise to the following indirect but natural reformulation as an integer linear program of the strong edge-colouring problem

- Compute $L(G)^2$ (this can be done in polynomial time).

- Formulate $\chi(L(G)^2)$ as the set covering problem with $\mathcal{S}$ as the set of all stable sets of $L(G)^2$:

$$\begin{aligned} &\min \sum_{S\in\mathcal{S}} x_S \\ \text{subject to } &\sum_{S\in\mathcal{S}:v\in S} x_S \geq 1, && \forall v \in V(L(G)^2) \\ &x_S \in \{0,1\}, && \forall S \in \mathcal{S} \\ &\mathcal{S} = \left\{\text{stable sets of } L(G)^2\right\} \end{aligned}$$

Up to our knowledge, the strong chromatic index has not been studied yet with the help of linear programming techniques.

Strong edge-colouring is closely related to another concept which gives a trivial lower bound for the strong chromatic index:

Definition 1.6. An *antimatching* is a set of edges such that no two edges are at distance strictly greater than 2. Denote by $a(G)$, the size of the maximum antimatching of a graph G.

As in the case of strong edge-colouring, $a(G) = \omega(L(G)^2)$. Obviously we have $\chi'_s(G) \geq a(G)$.

Definition 1.7. Let $G = (V, E)$ be a graph. We define the closed neighbourhood of a clique Q, denoted $N[Q]$, as the set of edges S, such that each edge of S has at least one end in Q. The maximal clique-neighbourhood is defined as:

$$\begin{aligned} \omega'_s(G) &= \max\left\{|N[Q]|\,, Q \text{ maximal clique of } G\right\} \\ &= \max\left\{\sum_{v\in V(Q)} d(v) - \frac{|V(Q)|\,(|V(Q)|-1)}{2}, Q \text{ maximal clique of } G\right\} \end{aligned}$$

It is easy to see that for any graph G, $\omega'_s(G) \leq a(G) \leq \chi'_s(G)$. However, the inequalities can be strict: there are graphs such that $\omega'_s(G) < a(G) - \lambda$, for a large value of λ. This is illustrated in Figure 1.6, where all the edges of the graph form an antimatching of size $3\lambda + 4$ while $\omega'_s(G) = 2\lambda + 3$.

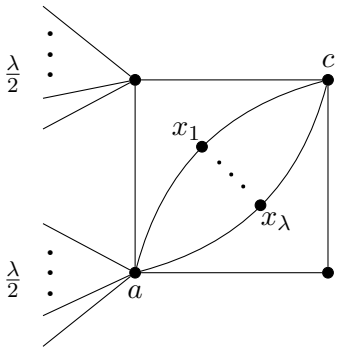


Figure 1.6: A graph G such that $\omega'_s(G) < a(G) - \lambda$

Proposition 1.8. If G is chordal, then $\chi'_s(G) = \omega'_s(G)$.

Proposition 1.8 is an immediate consequence of the two following theorems proved in [17]:

Theorem 1.9. (Cameron, 1989 [17]) If G is chordal, then $L(G)^2$ is chordal.

Theorem 1.10. (Cameron, 1989 [17]) If G is chordal, then $\omega'_s(G) = a(G)$.

Since chordal graphs are perfect, for a chordal graph G we have $\omega'_s(G) = a(G) = \omega(L(G)^2) = \chi(L(G)^2) = \chi'_s(G)$.

Lemma 1.11. If G has no induced C_4 and C_5, then $a(G) = \omega'_s(G)$.

Proof. Let S be an antimatching of G and $G[V(S)]$ be the graph induced by the endpoints of the edges of S. By hypothesis of the lemma and by definition of an antimatching, $G[V(S)]$ does not contain an induced cycle of length $k \geq 4$. Therefore, $G[V(S)]$ is chordal and, by Theorem 1.10, $\omega'_s(G[V(S)]) = a(G[V(S)])$.

Now, choose S of maximum size. We have: $\omega'_s(G[V(S)]) = |S| = a(G[V(S)]) = a(G)$. Moreover, $\omega'_s(G[V(S)]) \leq \omega'_s(G)$ and $\omega'_s(G) \leq a(G)$. So, $\omega'_s(G) = a(G)$ as claimed. □

A famous result of Grötschel, Lovász and Schrijver [56] states that the Lovász function $\vartheta(\overline{G})$ such that $\omega(G) \leq \vartheta(\overline{G}) \leq \chi(G)$ can be computed in polynomial time. Hence we can derive the following:

Remark 1.12. Let G be a graph such that $\chi'_s(G) = a(G)$. We have

$$\chi'_s(G) = \chi(L(G)^2) = \vartheta(\overline{L(G)^2}) = \omega(L(G)^2) = a(G)$$

Therefore, using the result of [56], $\chi'_s(G)$ can be computed in polynomial time by solving, in polynomial time, the linear program for the chromatic number.

The corresponding decision problem: given a graph G and an integer k, decide whether $\chi'_s(G) \leq k$, is known to be NP-hard [78]. A natural question would be to know how "deep" is this NP-hardness. That is to say, under which restrictions on graphs, the problem still remains NP-hard. Moreover, given this hardness result, a combinatorial point of view becomes all the more relevant: find polynomial-time computable lower and upper bounds for χ'_s on different classes of graphs. The main goal of Chapter 3 is to examine these two aspects: we show that strong edge-colouring remains NP-hard within very restricted subclasses of planar graphs; being motivated by several conjectures from the early 90's, we also study the strong chromatic index on the family of subcubic graphs and outerplanar graphs.

Identifying codes

An *identifying code* of a graph G is a subset $\mathcal{C}$ of vertices of G such that $\forall x \in V(G), N[x] \cap \mathcal{C} \neq \emptyset$ (*i.e.* $\mathcal{C}$ is a dominating set) and $\forall u, v \in V(G), N[u] \cap \mathcal{C} \neq N[v] \cap \mathcal{C}$ (*i.e.* $\mathcal{C}$ is a *separating set*) [66]. It is easy to see that a graph admits an identifying code if and only it has no twins.

One of the main questions is to find the size of a smallest identifying code of a graph G, denoted $\gamma^{ID}(G)$. Computing $\gamma^{ID}(G)$ is another instance of a set cover problem, which can be seen by the following reformulation:

Let $\mathcal{G}_{ID}$ be the identifying code hypergraph of G, where $V(G)$ is the vertex set of $\mathcal{G}_{ID}$ and there are two kinds of hyperedges: the closed neighbourhood of each vertex of G and the symmetric difference of each pair of vertices of G. More formally, $E(\mathcal{G}_{ID}) = \{N[v], N[u] \ominus N[v] | u, v \in V(G)\}$, where $N[u] \ominus N[v]$ is the symmetric difference of $N[u]$ and $N[v]$. It is then evident that finding $\gamma^{ID}(G)$ is equivalent to computing the size of a minimum *hitting set* of $\mathcal{G}_{ID}$, that is of a smallest subset of vertices covering all the hyperedges of $\mathcal{G}_{ID}$. Therefore, we have the following

straightforward integer linear programming formulation:

$$\min \sum_{v \in V} x_v \tag{1.8}$$

$$\text{subject to} \sum_{v \in N[u]} x_v \geq 1, \qquad \forall u \in V \tag{1.9}$$

$$\sum_{v \in N[u] \ominus N[w]} x_v \geq 1, \ \forall u, w \in V, u \neq w \tag{1.10}$$

$$x_u \in \{0,1\}^n, \qquad \forall u \in V \tag{1.11}$$

Here Constraints 1.9 and 1.10 are the domination and separation respectively.

Note that the identifying code problem was studied from the set cover perspective in [70]. Namely, it was proved by reduction from the set cover problem, that it is NP-hard to approximate $\gamma^{\text{ID}}(G)$ within a factor of $o(\log(n))$.

It is known that for every twin-free graph G, except for $\overline{K_n}$, $\gamma^{\text{ID}}(G) \leq |V(G)| - 1$ [11, 55]. Some conjectures have been proposed for the classification of graphs achieving this extremal bound. In our work, disproving these conjectures, we give a full characterization of these graphs. We also improve lower and upper bounds for the identifying code number for the family of line graphs. Finally, we study the complexity of the problem by showing that it remains NP-hard in some restricted classes of perfect graphs.

In summary, in this manuscript we have considered three instances of integer linear programming problems. For each of them, we have tried to find better solutions by considering the additional information we get from the structure behind the problem. In the case of OPP, we introduce an algorithm that, in practice, works better than previously known algorithms. For the other two problems we considered (strong edge-colouring and identifying codes), on the one hand we improve previously known bounds, sometimes by considering restricted families of graphs such as planar graphs of high girth or line graphs. On the other hand, we show that finding the exact solution for these problems remains NP-hard even for such restricted families of graphs.

Chapter 2

Orthogonal packing problem

Given a set of rectangular items of different sizes and a rectangular container, the aim of the two-dimensional Orthogonal Packing Problem (OPP-2), is to decide whether there exists a non-overlapping packing of the items in that container. The rotation of items is not allowed. This problem was proved to be NP-hard even in the restricted case, were a set of squares must be packed into a bigger square [73].

In this chapter, we present a new exact algorithm for solving OPP-2, detailed in Section 2.2.1. This procedure is based on the characterization of solutions using interval graphs proposed by Fekete and Schepers [39, 40] and described in Section 2.1. The algorithm uses MPQ-trees as data structures, which were introduced by Korte and Möhring in [67, 68] to recognize interval graphs. In Section 2.3 we establish the effectiveness of this algorithm on standard benchmarks. The main results we obtained and describe in this chapter are published in [65].

Let V be a set of D-dimensional rectangular shapes. For $d \in \{1, \ldots, D\}$ and every $v \in V$, let $w_d(v) \in \mathbb{Q}^+$ (resp. $w_d(C)$) be the length of v (resp. of the container C) with respect to the dimension d. For every subset of items $S \subseteq V$, let $w_d(S) = \sum_{v \in S} w_d(v)$. Let $W(v) = \prod_{d=1}^{D} w_d(v)$ and $W(C) = \prod_{d=1}^{D} w_d(C)$ be the volumes of the item v and of the container C respectively.

Formally OPP is defined as follows.

Definition 2.1. The *D-dimensional orthogonal packing problem* (OPP-D) is to decide if the set of items V fits into the container C without overlapping (if true, V is said to be *feasible*). Formally speaking, we have to find out whether $\forall d \in \{1, \ldots, D\}$ there exists a function $f_d : V \to \mathbb{Q}^+$, such that:

$$\forall v \in V, f_d(v) + w_d(v) \leq w_d(C) \tag{2.1}$$

$$\forall v_1, v_2 \in V, (v_1 \neq v_2), [f_d(v_1), f_d(v_1) + w_d(v_1)) \cap [f_d(v_2), f_d(v_2) + w_d(v_2)) = \emptyset \tag{2.2}$$

where $f_d(v)$ denotes the coordinate of the left corner of item v with respect to dimension d.

A natural generalization of the problem is the well-known orthogonal knapsack problem:

Definition 2.2. Let V be a set of D-dimensional items and C a D-dimensional container. For each $v \in V$, let p_v be an associated *profit* value. The *D-dimensional orthogonal knapsack problem* (OKP-D) consists in selecting a subset $V' \subseteq V$ such that V' is feasible and with maximum profit:

$$\max_{V' \subseteq V} \left\{ \sum_{v \in V'} p_v : V' \text{ satisfies 2.1 and 2.2} \right\}$$

A major step for solving the OPP, especially on hard instances, was made by Fekete and Schepers in late 90's when they proposed to model the problem from a new graph-theoretic perspective. In the following section we explain their main idea as it is the basis of our approach.

2.1 Formulation using interval graphs

Fekete and Schepers [39, 40] introduced a powerful characterization of feasible packings, based on interval graphs: given a feasible packing of a set of items V, for every dimension d, let $G_d = (V, E_d)$ be the interval graph with vertex set V and edge set E_d, such that ij is an edge if and only if the projections of the packing of items $i \in V$ and $j \in V$ onto the dimension d intersect (see Fig. 2.1 for an illustration).

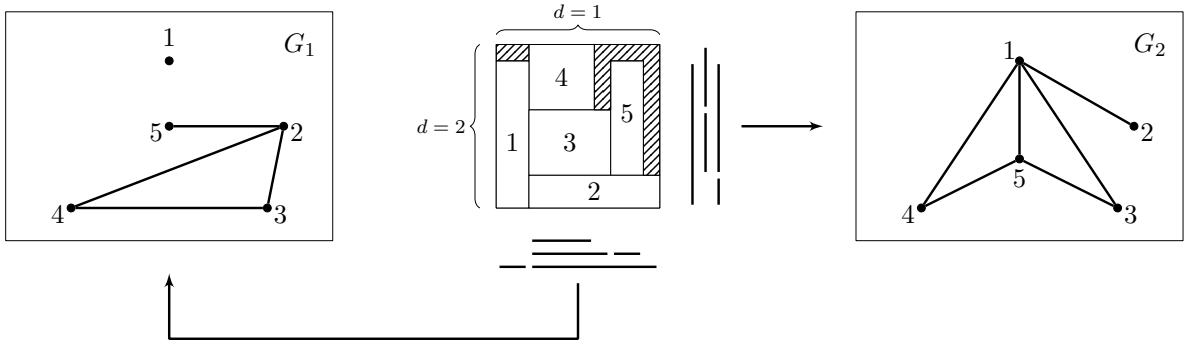


Figure 2.1: Example of 2D packing and its associated interval graphs

Hence every feasible packing induces a D-tuple of interval graphs which capture the relative positions of items. Fekete and Schepers established that the converse also holds:

Theorem 2.3 (Fekete and Schepers [39, 40])**.** Given a D-dimensional container C, a set of items V is feasible, if and only if there is a set of D graphs $G_d = (V, E_d)$, with $d \in \{1, \ldots, D\}$, such that:

(P1) Every graph G_d is an interval graph;

(P2) For every stable set S of G_d, $w_d(S) \leq w_d(C)$;

(P3) $\bigcap_{d=1}^{D} E_d = \emptyset$.

Definition 2.4. The D-tuple of graphs G_d satisfying the properties (P1), (P2) and (P3) of Theorem 2.3, is called a *packing class*.

Fekete, Schepers and van der Veen gave an efficient algorithm for solving OKP-D by solving its subproblem OPP-D [41]. The underlying idea of their algorithm is an exhaustive generation of tuples of interval graphs in order to find a packing class. To perform this generation efficiently, they used the following characterization of interval graphs:

Theorem 2.5 (Ghouilà-Houri [52], Gilmore and Hoffman [53])**.** Let a *2-chordless cycle* be a cycle $v_0, \ldots, v_{n-1}, v_0$ such that there is no edge $v_i v_j$ for $i, j \in \{0, \ldots, n-1\}$ and $|i-j| \mod n = 2$. A graph G is an interval graph if and only if it contains no induced C_4 and its complement contains no 2-chordless cycle of odd length.

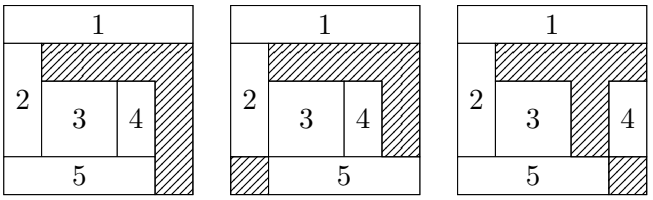


Figure 2.2: Symmetrical solutions in Fekete and Schepers' model

Despite its efficiency, their algorithm may enumerate symmetrical solutions. An example is given in Figure 2.2 where "almost" similar packing configurations are modeled by different pairs of interval graphs. Moreover, there are some degeneracy issues of Fekete and Schepers' algorithm pointed out in [42], implying the generation of some unnecessary pairs of interval graphs. In Figure 2.3, we illustrate a packing configuration together with its associated interval graphs. These graphs are obtained with Fekete and Schepers' algorithm and satisfy the properties of Theorem 2.3. However, there are some equivalent solutions which can be obtained by removing some edges in G_1 or G_2. For example, the edge between vertices 3 and 6 in G_1 or between 1 and 3 in G_2 can be removed.

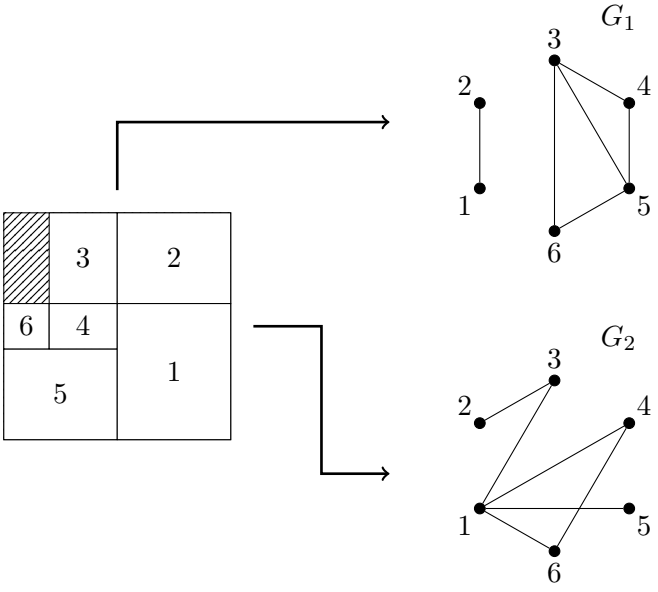


Figure 2.3: Degeneracy issues in Fekete and Schepers' algorithm

To handle the issues mentioned above, Joncour and Pêcher [64] designed an algorithm based on some other characterization of interval graphs proposed by Fulkerson and Gross [50]. Before stating this characterization, we need the following definition.

Definition 2.6. Let $G = (V, E)$ be a graph and $Q_1, \ldots, Q_m$ its maximal cliques. A consecutive arrangement of the maximal cliques of G is an order $\prec$ over the maximal cliques such that $Q_i \prec Q_j$ if:

$\forall v \in V$, if $v \in Q_i$ and $v \in Q_j$, then $v \in Q_k$ for all k s.t. $Q_i \prec Q_k \prec Q_j$.

Theorem 2.7 (Fulkerson and Gross [50])**.** A graph G is an interval graph if and only if the maximal cliques of G can be linearly ordered to obtain a consecutive arrangement.

Definition 2.8. Let $Q_1, \ldots, Q_m$ be the maximal (inclusionwise) cliques of a graph G. The matrix $M \in \mathcal{M}_{n,m}(\{0,1\})$ defined by $M_{ij} = 1$ if and only if vertex i belongs to clique Q_j is the *vertex/clique matrix* of G. If there exists a consecutive arrangement of $Q_1, \ldots, Q_m$, then on every row of M, the 1's occur consecutively and in this cases M is said to be a *consecutive-ones matrix*.

In other words, one can consider a vertex/clique matrix M which is a 0-1 matrix were rows represent vertices, columns represent maximal cliques and $M[i,j] = 1$ if and only if vertex i belongs to clique j. Therefore, Theorem 2.7 states that a graph G is an interval graph if and only if M associated to G has the consecutive-ones property. The idea of the algorithm of Joncour and Pêcher was to generate consecutive ones matrices satisfying a condition equivalent to Property (P2) of Theorem 2.3. The main issue of this approach is that the same interval graph can be represented by several distinct vertex/clique matrices having the consecutive-ones property.

In the following section we detail our approach which uses a compact data-structure, designed to capture isomorphic interval graphs - MPQ-trees. These trees are an enriched representation of PQ-trees which were invented for testing the consecutive-ones property as well as graph planarity [15].

2.2 MPQ-trees

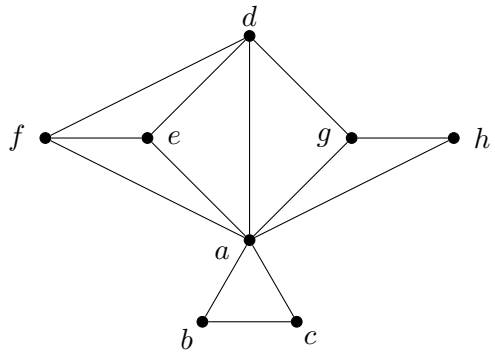
We start with main definitions and some useful results on MPQ-trees.

Let M be a set of elements and $\mathcal{M}$ a set of subsets of M. A PQ-tree is a data-structure representing all permutations of M in which the elements of each $M' \in \mathcal{M}$ occur consecutively. It was introduced by Booth and Lueker in [14, 15]. In the case of an interval graph, M is the set of maximal cliques, and $\mathcal{M}$ is the set of all $\mathscr{C}(v)$, for every vertex v, where $\mathscr{C}(v)$ denotes the set of all maximal cliques containing vertex v. Namely, the constraints over $\mathcal{M}$ are designed to model the consecutive arrangement of the maximal cliques according to Theorem 2.7. For a better understanding we give an example in Figure 2.4. For the interval graph of this figure, there are four possible permutations of the maximal cliques such that in the representation of the associated vertex/clique matrix with the columns given by these orders, the 1's occur consecutively:

$$
\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{array}
\begin{array}{c} \begin{array}{cccc} c_1 & c_2 & c_3 & c_4 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}
\quad
\begin{array}{c} \begin{array}{cccc} c_1 & c_4 & c_3 & c_2 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}
\quad
\begin{array}{c} \begin{array}{cccc} c_2 & c_3 & c_4 & c_1 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}
\quad
\begin{array}{c} \begin{array}{cccc} c_4 & c_3 & c_2 & c_1 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \end{array}
$$

Before explaining the way how these permutations are captured by the PQ-tree illustrated on the figure, we provide few definitions from [14, 15]:

Definition 2.9. A PQ-tree is a planar drawing of a rooted tree with two types of internal vertices: P and Q, represented by circles and rectangles respectively. The leaves of a PQ-tree are in one-to-one correspondence with the elements of M (the maximal cliques of an interval graph G). For an example see Figure 2.4. To avoid ambiguity between vertices of a graph and vertices of a PQ-tree, we call the latter ones *nodes*.



Maximal cliques:

$c_1 = \{a, b, c\}$
$c_2 = \{a, d, e, f\}$
$c_3 = \{a, d, g\}$
$c_4 = \{a, g, h\}$

$$M = \{c_1, c_2, c_3, c_4\}$$

$$\mathscr{C}(a) = \{c_1, c_2, c_3, c_4\},\ \mathscr{C}(b) = \mathscr{C}(c) = \{c_1\}$$
$$\mathscr{C}(d) = \{c_2, c_3\},\ \mathscr{C}(e) = \mathscr{C}(f) = \{c_2\}$$
$$\mathscr{C}(g) = \{c_3, c_4\},\ \mathscr{C}(h) = \{c_4\}$$

$$\mathcal{M} = \{\{c_1, c_2, c_3, c_4\}, \{c_1\}, \{c_2\}, \{c_2, c_3\}, \{c_3, c_4\}, \{c_4\}\}$$

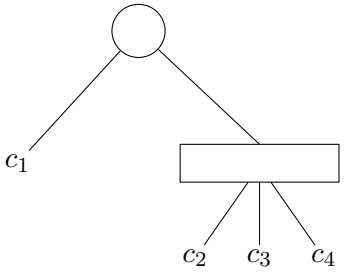


Figure 2.4: An interval graph, its consecutive constraints and a PQ-tree representation.

The difference between P- and Q-nodes consists in the possible permutations of the reading orders of their sons:

Definition 2.10. The *frontier* $F(T)$ of a PQ-tree T, represents the permutation of the maximal cliques obtained by the ordering of the leaves of T from left to right. Two PQ-trees T and T' are *equivalent*, if one can be obtained from the other by applying the following rules a finite number of times:

1. Arbitrarily permute the children of a P-node
2. Reverse the order of children of a Q-node

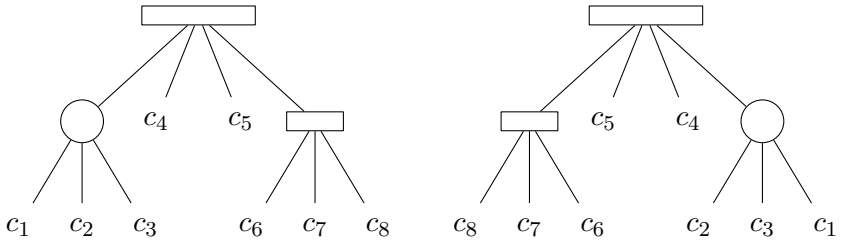


Figure 2.5: Equivalent PQ-trees

Figure 2.5 shows an example of two equivalent PQ-trees: the frontier of the right picture is obtained from the frontier of the left one, by reversing the order of the sons of both Q-nodes and by permuting the leaves c_1, c_2 and c_3, sons of the P-node.

Now, let us come back to the PQ-tree of Figure 2.4. The main constraint over $\mathcal{M}$ of the corresponding interval graph is that the maximal cliques c_2, c_3 and c_4 can appear only in two possible orders c_2, c_3, c_4 and the reversed order c_4, c_3, c_2 (because of vertices d and g). Therefore, it can be easily checked that c_1, c_2 and c_3 must be sons of a Q-node as it is the only way to model exactly these two permutations.

A *proper* PQ-tree is a PQ-tree for which every P-node has at least two children and the Q-node has at least three children. From now on, by the term PQ-tree we will consider a proper PQ-tree.

An important result of Booth and Lueker is the following characterization of interval graphs using PQ-trees:

Theorem 2.11 (Booth and Lueker [14, 15])**.** A graph G is an interval graph if and only if there exists a PQ-trees T such that $F(T)$ is a consecutive arrangement of the maximal cliques of G.

In the same articles, Booth and Lueker gave an algorithm recognizing interval graphs in linear time. The input of this algorithm is a graph G and it returns a PQ-tree associated to G if G is an interval graph and rejects the input otherwise. We do not describe their approach as in our algorithm for solving OPP-2 we use an extension of the notion of PQ-trees, introduced by Korte and Möhring in [67] for the same purpose of recognition of interval graphs. This extension allows to store more information about the vertices of the maximal cliques in the nodes of the tree and gives an easier method of recognizing interval graphs than the method using classical PQ-trees proposed by Booth and Lueker. We provide its definition below:

Definition 2.12. *A modified PQ-tree* (MPQ-tree for short) associated with a graph G is an extension of a PQ-tree where the nodes of the tree are labelled with some subsets of vertices of G, such that each branch of the MPQ-tree represents a maximal clique. A P-node is assigned only one set, while a Q-node is assigned a set for each of its children. Here are the rules of the labelling:

- A P-node is labelled by the set of vertices of G which are *only* contained in *all* cliques represented by the subtree of T rooted in this node.
- A leaf is labelled by the set of vertices of G contained *only* in the clique represented by this leaf.
- A Q-node, with m children $F_1, \ldots, F_m$, is labelled by a list of sets S_k, for $k \in \{1, \ldots, m\}$, each of them being called a *section* such that a section S_k corresponds to the child F_k in the left-to-right order. Each section S_k ($k \in \{1, \ldots, m\}$) is a subset S_k of vertices of G such that S_k is contained in *all* cliques represented by the subtree rooted in F_k. Additionally, every vertex of S_k must belong to *all* cliques represented by the subtree rooted in some other child of the Q-node, say F_l, where $l \in \{1, \ldots, m\}$ and $l \neq k$.

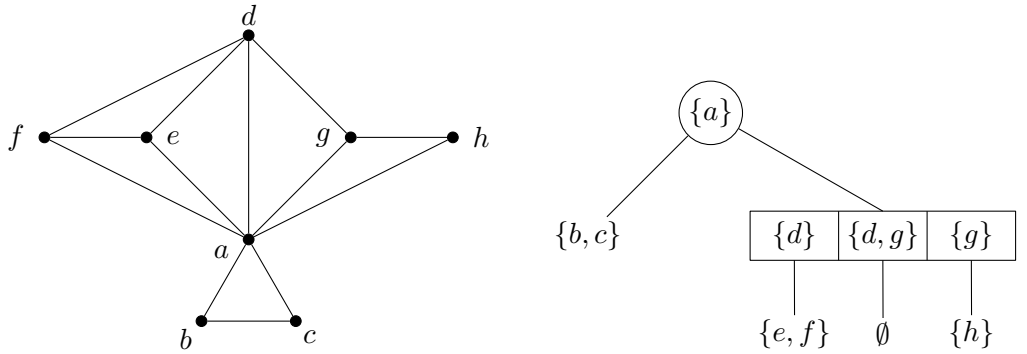
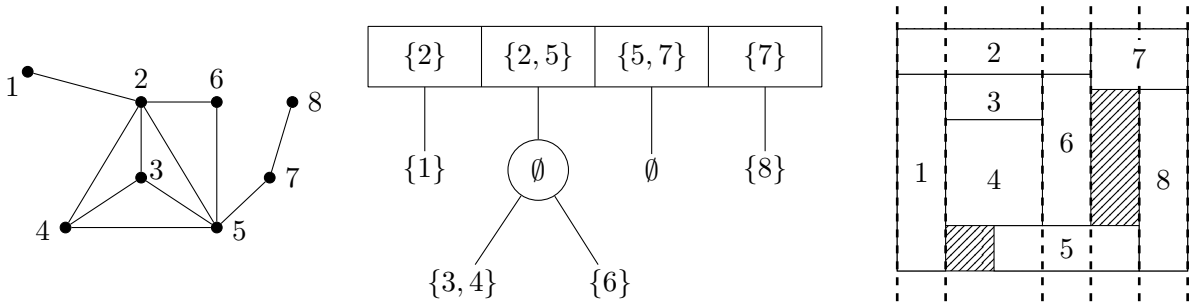


Figure 2.6: The interval graph of Figure 2.4 and an MPQ-tree representation.

We will say that a node N of an MPQ-tree associated with a graph G *contains* a vertex v of G if $v \in V_N$, where V_N is the vertex set or section (in case of a Q-node) of the label of N. Notice that from the definition of an MPQ-tree, a vertex v of G is contained in exactly one MPQ-tree node. However, v can be contained in more than one section of a Q-node. Figure 2.6 gives an illustration of an MPQ-tree based on the interval graph of Figure 2.4. Observe that the structure

of both the PQ-tree of Figure 2.4 and the MPQ-tree of Figure 2.6 are the same, only the labels of the nodes are different, the branches of the MPQ-tree being the maximal cliques.

Figure 2.7 shows an example of an MPQ-tree associated with an interval graph G, where both representations model the intersections in dimension 1 of a feasible packing in terms of maximal cliques (the intersections are given by the strips in the picture).



Maximal cliques: $\{1,2\}$, $\{2,3,4,5\}$, $\{2,5,6\}$, $\{5,7\}$, $\{7,8\}$

Figure 2.7: An interval graph, an associated MPQ-tree and the packing configuration

Similarly as in the case of PQ-trees, Korte and Möhring established the connection between MPQ-trees and interval graphs by proving the following characterization of interval graphs:

Theorem 2.13 (Korte and Möhring [67, 68])**.** A graph G is an interval graph if and only if there exists an MPQ-tree associated with G.

Therefore, we may consider MPQ-trees instead of interval graphs as Theorem 2.13 gives an equivalence between the two structures. The following lemma is crucial for our algorithm since it is the basis for an incremental generation of MPQ-trees.

Lemma 2.14 (Korte and Möhring [68])**.** Let G be an interval graph and T its associated MPQ-tree. Then $G+u$ (where u is a vertex added to G) is an interval graph if and only if the following holds:

1. All vertices adjacent to u are contained in a unique path of T.
2. For each Q-node N, labelled with sections $S_1, \ldots, S_m$, let $S = S_1 \cup \ldots \cup S_m$. Then $S \cap N(u) \subseteq S_1$ or $S \cap N(u) \subseteq S_m$

In other words, Lemma 2.14 says that while building an MPQ-tree incrementally from an interval graph (i.e. adding vertices of the graph one by one) only one path of this tree must be updated and due to the properties of the reading orders of children of nodes of an MPQ-tree, this path can be chosen to be the **leftmost one**. This restricts considerably the number of cases to consider while updating the MPQ-tree.

To describe the idea of Korte and Möhring's algorithm for the recognition of interval graphs, an idea which is reused in our approach, we first need to provide the following definition:

Definition 2.15. Given a graph G, a *simplicial elimination scheme* (also called a perfect vertex elimination scheme) is an ordering $\sigma = [v_1, \ldots, v_n]$ of the vertices of G such that the graph induced by the vertices of $N(v_i) \cap \{v_{i+1}, \ldots, v_n\}$ is a clique.

An important fact is that a graph is a chordal graph if there exists a *simplicial elimination scheme* for the vertices of this graphs [33, 50, 87]. Therefore, interval graphs being chordal must admit one too.

A simplicial elimination scheme for the vertices of a graph G can be obtained by running a *LexBFS* algorithm on G (introduced in [88]). In particular, if a graph is chordal, then a LexBFS algorithm produces an order of the vertices $\sigma = [v_1, \ldots, v_n]$, such that $[v_n, v_{n-1}, \ldots, v_1]$ is a simplicial elimination scheme.

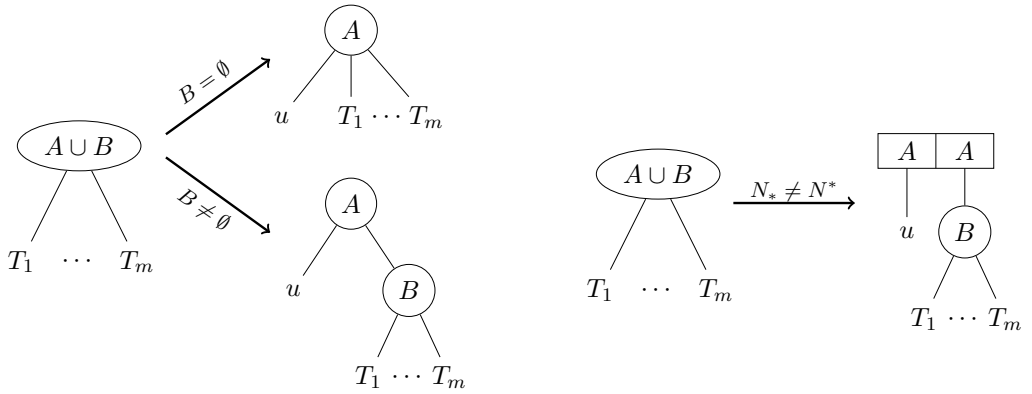
Using Theorem 2.13 and Lemma 2.14, Korte and Möhring gave a much simpler linear algorithm than the one of Booth and Lueker, recognizing interval graphs, by building iteratively an associated MPQ-tree [68]. Their algorithm uses a LexBFS-ordering of the vertices of the input graph to iteratively build a corresponding MPQ-tree. Let $G = (V, E)$ be an interval graph with an associated MPQ-tree T_G and u the vertex to be inserted in T_G to obtain the MPQ-tree T_{G+u} associated with the graph $G + u$. Since vertices are inserted in LexBFS-order, $N(u)$ must induce a clique. Let N be a node of T_G. If N is of type P or a leaf, then let V_N be its associated vertex set and if N is a Q-node, let V_N be the vertex set corresponding to the first section. The principle of Korte and Möhring's algorithm is the following:

- Find the unique path P of the current MPQ-tree having a node N such that $V_N \cap N(u) \neq \emptyset$. This path can be either leftmost or rightmost.
- Find the first node N_* in the bottom-up traversal such that $\exists j \in V_{N_*}, j \in N(u)$.
- Select the corresponding pattern and apply the suitable replacement: let $V_{N_*} = A \cup B$ be the partition of V_{N_*} such that $A = V_{N_*} \cap N(u)$ and $B = V_{N_*} \setminus A$. Let N^* be the highest node in P such that $V_{N^*} \setminus N(u) \neq \emptyset$ if it exists and let $N^* = N_*$ otherwise. Due to Lemma 2.14, the patterns which can be applied are described in Figures 2.8, 2.9 and 2.10. In the case when $N_* \neq N^*$ the patterns are applied recursively by rewriting the current tree starting from N_* up to N^*, to obtain a valid MPQ-tree. We omit details which are rather technical. For an elaborate explanation, see the original paper [68]. We just strengthen the fact that the process is deterministic e.g. at each step only one pattern can apply. If no pattern can be applied to the current configuration, then $G + u$ is not an interval graph.

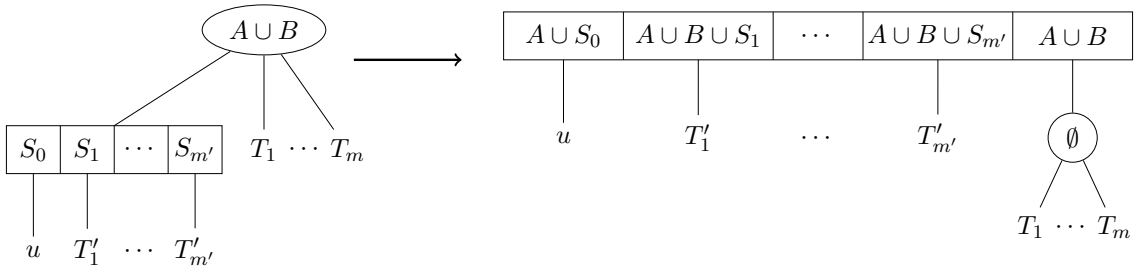
Figure 2.8: Templates for a leaf: $V_{N_*} = A \cup B$

For an example of construction of an MPQ-tree from an interval graph using Korte and Möhring's templates, we illustrate the successive application of patterns in Figure 2.11. The LexBFS order applied in the figure is $\sigma = [8, 7, 5, 6, 2, 3, 4, 1]$. The building of the MPQ-tree is done in eight step (from s1 to s8 in the picture), provided the default operation in the very first step s0: an empty MPQ-tree T is created and vertex 8 is inserted in T as a leaf. Below, we give a detailed explanation of which of the templates of Figures 2.8, 2.9, 2.10 is applied in each of the steps together with the value of each of the parameters V_{N_*}, B:

- s1: a leaf template of Figure 2.8 with $V_{N_*} = \{8\}, B = \emptyset, N_* = N^*$;
- s2: a leaf template of Figure 2.8 with $V_{N_*} = \{7, 8\}, B \neq \emptyset, N_* = N^*$;
- s3: since vertex 6 is not adjacent to vertex 7, N^* is the root P-node with $V_{N^*} = \{7\}$ and hence $N_* \neq N^*$. Therefore, the leaf template of Figure 2.8 with $V_{N_*} = \{5\}, N_* \neq N^*$ is applied. In the following step the higher nodes of the tree must be changed in order to obtain $N_* = N^*$;

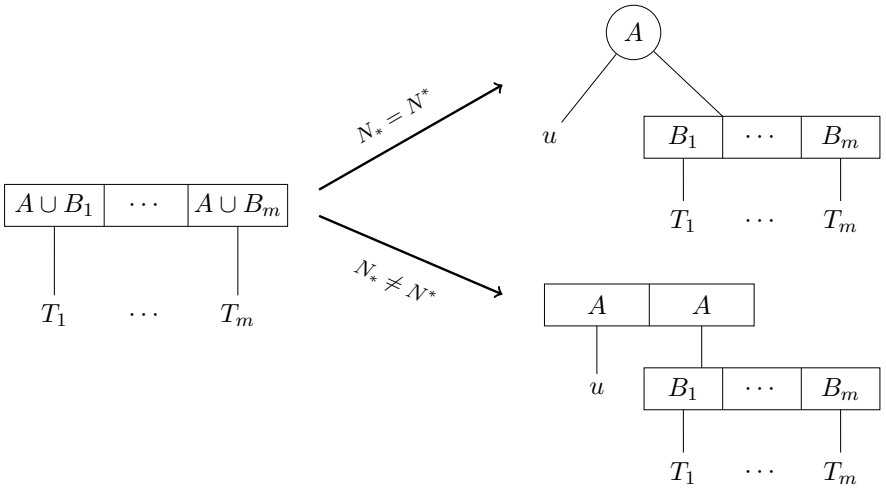


(a) P1 and P2: $V_{N_*} = A \cup B$



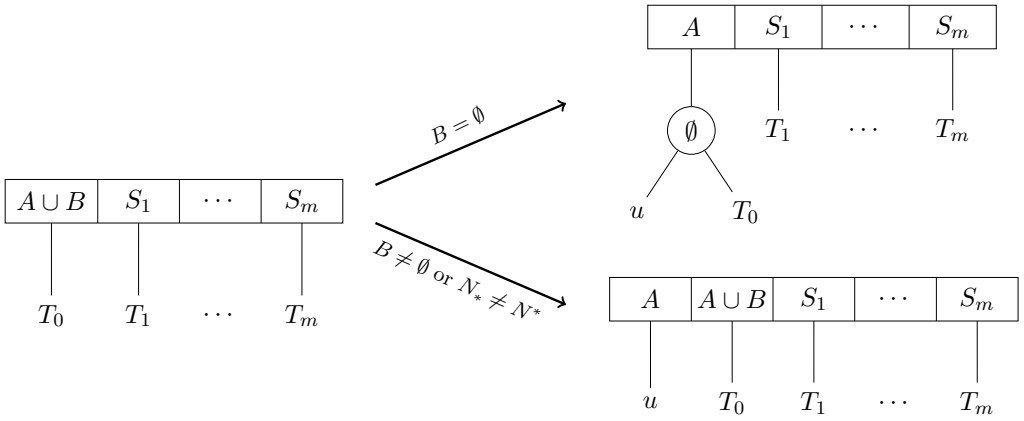
(b) P3: $V_{N_*} = \{u\}$ and $N_* \neq N^*$

Figure 2.9: Templates for a P-node

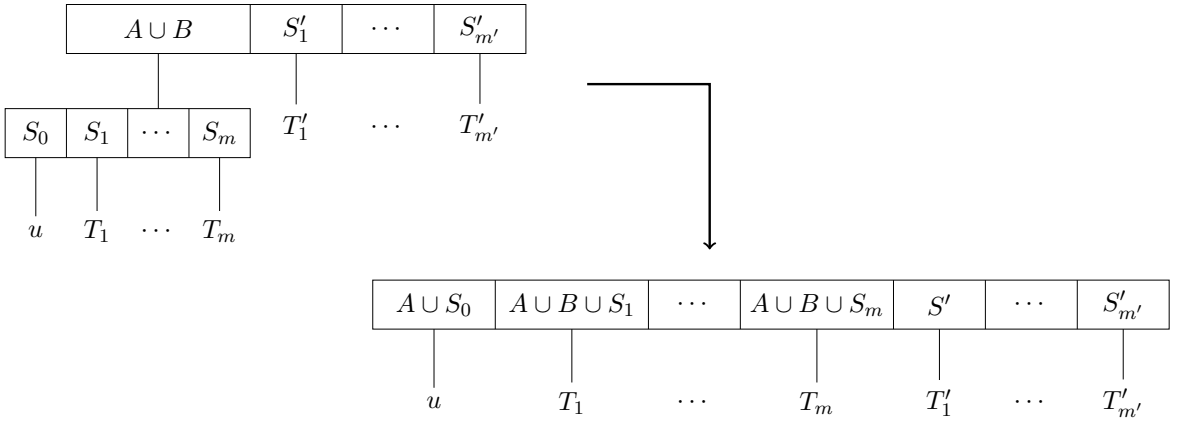


(a) Q1

Figure 2.10: Templates for a Q-node



(b) Q2



(c) Q3

Figure 2.10: Templates for a Q-node

- s4: the P-node template P3 of Figure 2.9b with $V_{N_*} = \{5\}, N_* \neq N^*, B \neq \emptyset$ is applied and at the end we obtain $N* = N^*$;
- s5: a leaf template of Figure 2.8 $V_{N_*} = \{6\}, B = \emptyset, N_* = N^*$;
- s6: a leaf template of Figure 2.8 $V_{N_*} = \{2, 6\}, B \neq \emptyset, N_* = N^*$;
- s7: a leaf template of Figure 2.8 $V_{N_*} = \{3\}, B = \emptyset, N_* = N^*$;
- s8: a leaf template of Figure 2.8 $V_{N_*} = \{3, 4\}, N_* \neq N^*$, then the higher nodes of the tree must be changed in order to obtain $N_* = N^*$;
- s9: the Q-node template Q3 of Figure 2.10c is applied.

The order σ is not the only LexBFS order possible. For instance, one could start building from vertex 1, in which case the obtained MPQ-tree would be the equivalent of the one of Figure 2.11 (with the reading order from right to left).

In order to formulate Property (P2) of Theorem 2.3 with respect to MPQ-trees we define the width of the nodes:

Definition 2.16. For every node N of an MPQ-tree associated with a dimension d of a packing configuration, the *width* $\lambda_N \in \mathbb{Q}^+$ of N is as described below:

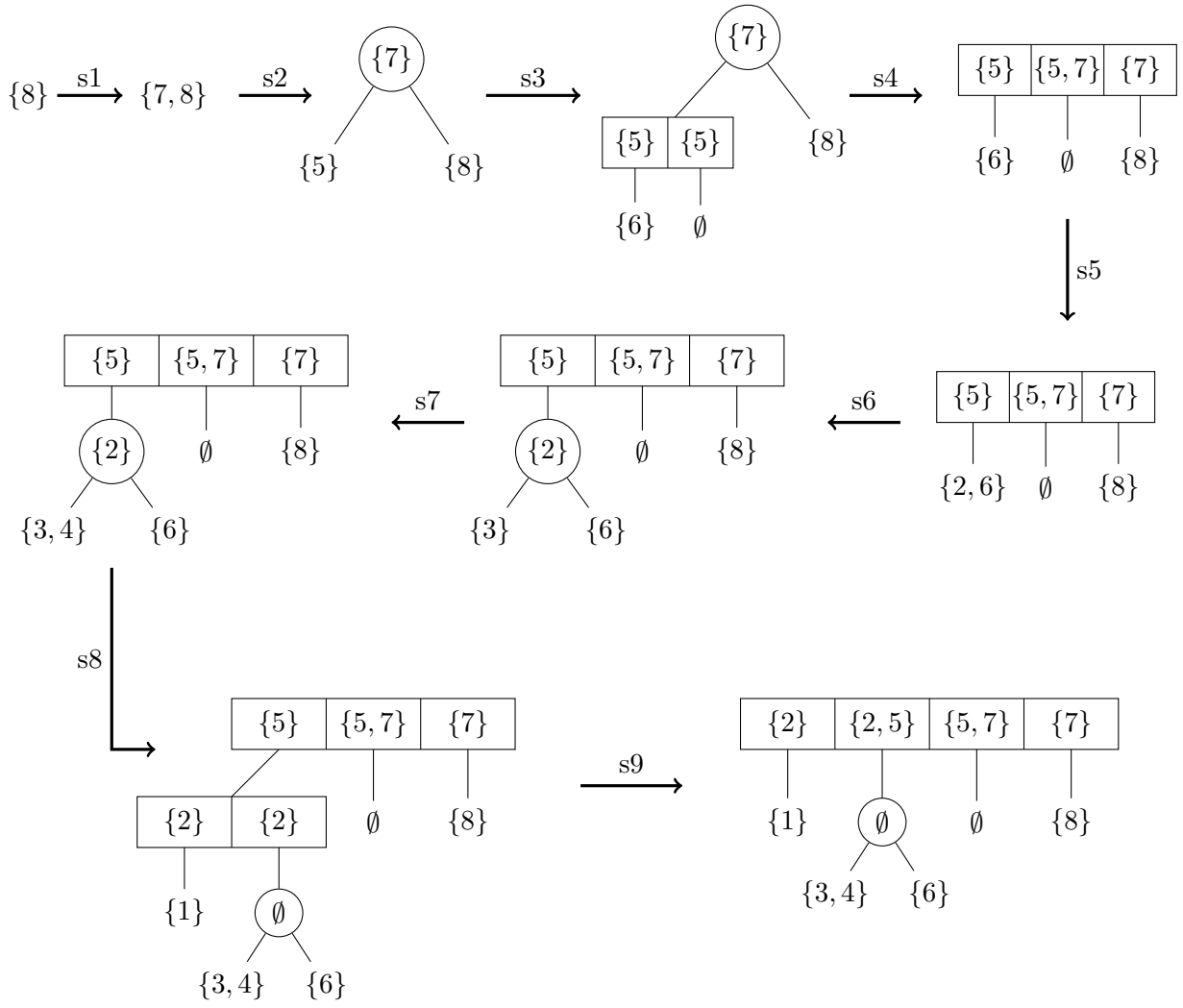


Figure 2.11: Building the MPQ-tree of Figure 2.7 with the templates from [68]

- If N is a **leaf** L labelled with the set V_L, then $\lambda_L = \max_{i \in V_L}\{w_d(i)\}$ if $V_L \neq \emptyset$ and $\lambda_L = 0$ otherwise.
- If N is a **P-node** P labelled with the set V_P, and $\lambda_{f_1}, \ldots, \lambda_{f_m}$ are the widths of each of its children, then

$$\lambda_P = \max\left\{\sum_{j=1}^{m} \lambda_{f_j}, \max_{i \in V_P}\{w_d(i)\}\right\}$$

- If N is a **Q-node** Q, let $S_1, \ldots, S_m$ be its sections and $\lambda_{f_1}, \ldots, \lambda_{f_m}$ the widths of each of the children of the node. In order to define the width λ_Q, we first define, recursively from m to 1, the widths of its sections: $\lambda_{S_j}, \forall 1 \leq j \leq m$:

$$\lambda_{S_m} = \lambda_{f_m}$$

Suppose $\lambda_{S_{k+1}}, \ldots, \lambda_{S_m}$ are defined, then

$$\lambda_{S_k} = \max\left\{\lambda_{f_k}, \max_{i \in S_k, i \notin S_{k-1}}\left\{w_d(i) - \sum_{h>k,\, i \in S_h} \lambda_{S_h}\right\}\right\}$$

The width of N is given by the following formula

$$\lambda_Q = \sum_{k=1}^{m} \lambda_{S_k}$$

It follows from definition that for every node N with children $f_1, \ldots, f_m$, $\lambda_N \geq \sum_{k=1}^{m} \lambda_{f_k}$.

In Figure 2.12 we illustrate the notion of widths on the MPQ-tree and the corresponding packing configuration from Figure 2.7. The sizes of the items in dimension 1 are given in the right part of the picture.

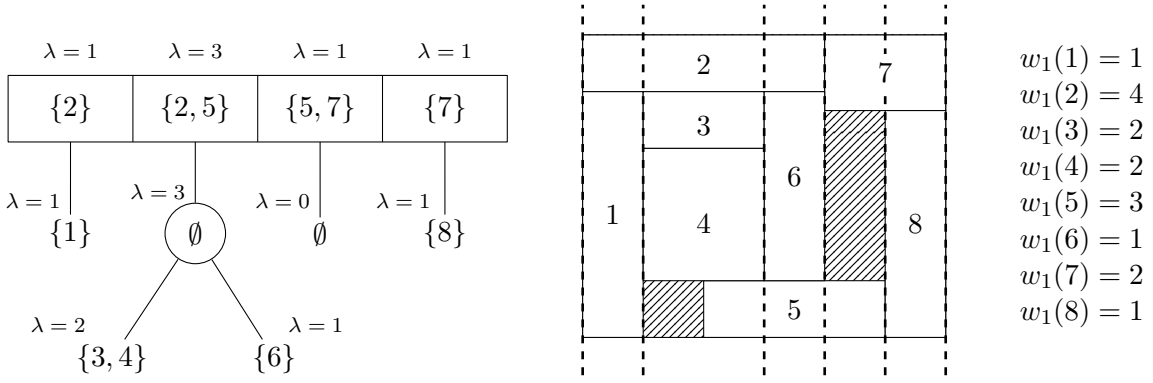


Figure 2.12: An MPQ-tree with the widths of its nodes and a corresponding packing

We now establish the inequality relation between the width of an item and the width of the MPQ-tree node containing this item:

Lemma 2.17. Let $G_d = (V, E_d)$ be an interval graph of a packing class and T_d an associated MPQ-tree. Let v be an item of V. For every P-node or leaf N such that v belongs to the label of N, we have $w_d(v) \leq \lambda_N$. For every Q-node N containing v in the labels of some of its sections, we have $w_d(v) \leq \sum_{h=k}^{l} \lambda_{S_h}$, where $S_k, \ldots, S_l$ are the sections of N containing v.

Proof. The cases of a P-node and of a leaf are obvious. Suppose N is a Q-node with sections $S_1, \ldots, S_m$ and children $f_1, \ldots, f_m$. Since branches associated with sections $S_k, \ldots, S_l$ of N are all the cliques containing v, we have $\lambda_{S_k} \geq w_d(v) - \sum_{h=k+1}^{l} \lambda_{S_h}$. Hence, $w_d(v) \leq \sum_{h=k}^{l} \lambda_{S_h}$. □

The following proposition translates Property (P2) of Theorem 2.3 in terms of widths of the MPQ-tree nodes, the previous lemma being helpful to prove the converse part of the equivalence.

Proposition 2.18. Let G be an interval graph. The graph G satisfies Property (P2) if and only if for every associated MPQ-tree T of G, the width λ_R of the root of T verifies $\lambda_R \leq w_d(C)$.

Proof.
Suppose G satisfies Property (P2) and consider an MPQ-tree T of G with root R. We will prove that $\lambda_R \leq w_d(C)$.

The proof is by induction on the distance between R and the nodes of T. Let $H(k)$ be the assertion: "If N is a node of T at distance k from R, then there is a stable set S of $G[N]$ such that $\lambda_N \leq w_d(S)$, where $G[N]$ is the subgraph of G induced by the set of vertices of G contained in the labels of the subtree of T rooted in N."

- Let M be the maximal distance from R in T. Let N be any node of T at distance M from R. Hence N is a leaf. Due to the definition of a leaf, there is i in V_N such that $w_d(i) = \lambda_N$. Hence, $S = \{i\}$ and $H(M)$ is true.

- Assume that $H(k)$ is true for $k \leq M$ (induction hypothesis). Let N be any node of T at distance $k-1$ from R.

  - If N is a leaf, then the above argument applies again.

  - If N is a P-node with m sons, then we distinguish two cases. If $\max_{j \in V_N}\{w_d(j)\} \leq \sum_{k=1}^{m} \lambda_{f_k}$, then let $S = \bigcup_{k=1}^{m} Z_k$, where Z_k is the stable set associated to son f_k such that, due to induction hypothesis, $\lambda_{f_k} \leq w_d(Z_k)$. Since in the graph G_d there are no edges between two vertices contained in two different children of N, S is clearly an independent set of $G[N]$. Applying the induction hypothesis, $\lambda_N \leq \sum_{k=1}^{m} \lambda_{f_k} \leq w_d(S)$ and we are done.
    If $\max_{j \in V_N}\{w_d(j)\} > \sum_{k=1}^{m} \lambda_{f_k}$ then let $S = \{i\}$ where i is such that $w_d(i) = \max_{j \in V_N}\{w_d(j)\}$. Thus $\lambda_N = w_d(S)$ and we are done.

  - If N is a Q-node with sections $S_1, \ldots, S_m$, then let Z_k be the stable set of G built by iteration of k from 1 to m:

    * $k = 1$. If the width of S_1 is equal to the width of its child f_1, then due to the induction hypothesis, Z_1 is a stable set of $G[N]$ such that $\lambda_{S_1} \leq w_d(Z_1)$; otherwise there is a vertex $b \in S_1$ such that $\lambda_{S_1} = w_d(b) - \sum_{h>1, b \in S_h} \lambda_{S_h}$, in which case define $Z_1 = \{b\}$.
    * $k \geq 2$. If there is a vertex $b \in S_k, b \in S_{k-1}$ and $Z_{k-1} = \{b\}$, define $Z_k = \{b\}$; otherwise, if there is a vertex $b \in S_k, b \notin S_{k-1}$ and $\lambda_{S_k} = w_d(b) - \sum_{h>k, b \in S_h} \lambda_{S_h}$, then $Z_k = \{b\}$; otherwise, necessarily $\lambda_{S_k} = \lambda_{f_k}$, in which case, due to the induction hypothesis, Z_k can be chosen to be the stable set of $G[f_k]$ such that $\lambda_{f_k} \leq w_d(Z_k)$.

    Let $S = \bigcup_{k=1}^{m} Z_k$. Clearly, S is a stable set of G_N such that $\lambda_N = \sum_{k=1}^{m} \lambda_{S_k} \leq \sum_{s \in S} w_d(s) = w_d(S)$.

  Hence $H(k-1)$ is true.

Due to the induction, $H(0)$ is true, and since $w_d(S) \leq w_d(C)$ (Property (P2) of Theorem 2.3), we have $\lambda_R \leq w_d(C)$.

Conversely, suppose $\lambda_R \leq w_d(C)$. Consider an independent set S of G_d. By definition of an MPQ-tree, $\forall u \in S$, in T_d there exist exactly two distinct cases for the node N of T_d containing u:

1. N is a Q-node. Let $S_k, \ldots, S_l$ be the sections of N containing u. In this case, let $\Lambda_u = \sum_{h=k}^{l} \lambda_{S_h}$.

2. N is a P-node or a leaf. In this case, let $\Lambda_u = \lambda_{N_u}$.

Since S is an independent set, no two vertices of S are contained in the same branch of T_d. Hence, by definition of λ_R, we have $\sum_{s \in S} \Lambda_s \leq \lambda_{R_d}$. By Lemma 2.17, we have $\forall x \in V$, $w_d(x) \leq \Lambda_x$. Hence,

$w_d(S) \leq \lambda_{R_d}$. Applying the hypothesis, $w_d(S) \leq w_d(C)$ which is exactly the Property (P2) of Theorem 2.3.

□

Notice that Property (P3) of Theorem 2.3 may be easily translated with respect to MPQ-trees, since the branches of an MPQ-tree represent the maximal cliques of its corresponding interval graph.

2.2.1 Algorithm to check feasibility

The main idea of our approach for solving the orthogonal packing problem is to build an MPQ-tree T_d for each dimension d such that $\lambda_R^{T_d} \leq w_d(C)$ and Property (P3) of Theorem 2.3 is verified. The construction of the valid MPQ-tree is done through an exhaustive enumeration of all possible MPQ-trees containing all items, until one is found. If the MPQ-tree cannot be built, then the algorithm stops rejecting the input. The procedure is applied for all dimensions.

More precisely, for a given dimension and a given order on items, the (pseudo) Algorithm 1 constructs all the MPQ-trees to get the one representing this dimension. The items are added one by one in the MPQ-tree with respect to the templates of Figures 2.8, 2.9, 2.10 and such that for every node N, $\lambda_N \leq w_d(C)$. Observe that the algorithm may have to consider all possible orders σ. If this happens, the generation will not be efficient at all. However, in next sections, we explain that in fact we have to consider only a reduced subset of orders.

Algorithm 1 recursive enumeration of MPQ-trees to check feasibility

Require: order σ, (int) `n`, MPQ-tree T
Ensure: TRUE if there exists a feasible packing, FALSE otherwise

```
recurse(MPQ-node currentNode, int i)
if i > n then
  return TRUE
end if
repeat
  for all corresponding patterns do
    apply the modification with vertex σ(nrVertex)
    currentNode ← new created leaf L
    for all N in leftmost branch of T do
      update λ_N
    end for
    if λ_R ≤ w_d(C) then
      if recurse(currentNode,i+1) then
        return TRUE
      end if
    end if
    for all N in leftmost branch of T do
      backup λ_N
    end for
    undo the modification
  end for
  currentNode ← currentNode.getFather()
until currentNode ≠ NULL
return FALSE
```

2.2.2 Symmetries and early unfeasibility detection

The structure of an MPQ-tree provides some partial information about the position of the items which can be easily exploited. For instance, its frontier gives the order of appearance of items in the packing configuration. In addition, the higher an item is contained in the tree associated with a dimension d, the more items it "covers" in d (i.e. the more are the intersections of its projection on d with the projections of the other items on d). Being able to compute at each step the widths of each node, we provide some optimizations of the algorithm by adding some valid constraints, breaking some symmetries and early detecting some unfeasible configurations.

Two-dimensional case: remaining areas, widths and branches heights

In the bidimensional case, a few basic valid constraints may be applied when enumerating MPQ-trees in dimension 1:

1. Assume that at the step i of the algorithm, an MPQ-tree T_1 containing $\sigma(1), \ldots, \sigma(i)$ was constructed with the root having m children $f_1, \ldots, f_m$. After the step i, the widths of the children $f_2, \ldots, f_m$ cannot be modified (since only the leftmost branch can be modified), therefore by Proposition 2.18, the remaining items $\sigma(i+1), \ldots, \sigma(n)$ have to be packed in the area $W(C) - w_2(C) * \sum_{h=2}^{m} \lambda_{f_h}$. Hence the following constraint is valid:

$$\sum_{h=i+1}^{n} W(\sigma(h)) + w_2(C) * \sum_{h=2}^{m} \lambda_{f_h} \leq W(C)$$

2. The width of any item among $\sigma(i+1), \ldots, \sigma(n)$ cannot exceed the remaining width of the container after removing $\lambda_{f_2}, \ldots, \lambda_{f_m}$:

$$\sum_{h=2}^{m} \lambda_{f_h} + \max\{w_1(\sigma(i+1)), \ldots, w_1(\sigma(n))\} \leq w_1(C)$$

3. Let $\mathcal{B}$ be a branch of T_1. Since all the items contained in this branch overlap in the dimension 2, the sum of the size of these items in dimension 2 cannot exceed $w_2(C)$:

$$\sum_{i \in \mathcal{B}} w_2(i) \leq w_2(C)$$

Positive widths

The left part of the Figure 2.13 is an example of packing which is somehow similar to the right part. In the following we give a precise definition of this equivalence notion:

Definition 2.19. Two packings with (x_i, y_i) and (x'_i, y'_i) being the coordinates of the right upper corner of item i in each of them, are said to be *equivalent* if either

$$x_i = x'_i \text{ and } y_i \geq y'_i, \forall i \text{ or}$$
$$y_i = y'_i \text{ and } x_i \geq x'_i, \forall i$$

Next lemma establishes a sufficient condition for avoiding the enumeration of this type of equivalent packing configurations. Hence, our algorithm generates only the right packing of Figure 2.13.

Lemma 2.20. Let G_d be an interval graph of a packing class. There is an associated MPQ-tree of G_d such that for every internal node N with m sons $f_1, \ldots, f_m$, the following holds:

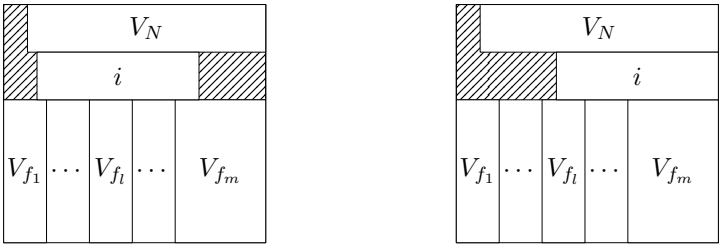


Figure 2.13: Example of "similar" packing configurations

- If N is a P-node, then $\forall i \in V_N$, $w_d(i) - \sum_{k=2}^{m} \lambda_{f_k} > 0$
- If N is a Q-node with sections $S_1, \ldots, S_m$, then $\forall k \in \{1, \ldots, m\}$, $\forall i \in S_k$, $w_d(i) - \sum_{h>k, i \in S_h} \lambda_{S_h} > 0$

Proof. Recall that, from Definition 2.16, for every node or section X, $\lambda_X \geq 0$.

Let T' be an MPQ-tree associated with G_d and let N' be an internal node of T' not satisfying the statement.

- Suppose N' is a P-node. Then there is $i \in V_{N'}$, such that $w_d(i) - \sum_{k=2}^{m} \lambda_{f_k} \leq 0$.

  We choose i such that $w_d(i) = \min_{j \in V_{N'}} \{w_d(j)\}$. Let $l = \max\{h \geq 2 \mid w_d(i) \leq \sum_{k=h}^{m} \lambda_{f_k}\}$. We distinguish two cases for the value of l.

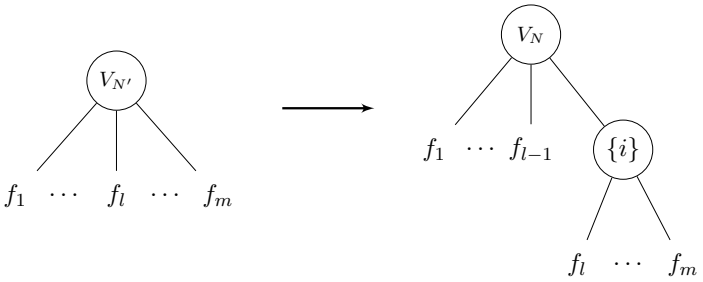


Figure 2.14: P-node replacement

  - Suppose $l < m$. Let T be the MPQ-tree obtained from T' by replacing N' as shown in Figure 2.14. Observe that the MPQ-trees associated to dimension 1 of the packing configurations of Figure 2.13 are the MPQ-trees of Figure 2.14 when $f_1, \ldots, f_m$ are leaves. In T, N is the P-node replacing N' of T' and having $V_N = V_{N'} \setminus \{i\}$ as a labelling set, $\{i\}$ is the labelling set of a new P-node, say A, having $f_l, \ldots, f_m$ as children. In T, $\lambda_A = \sum_{k=l}^{m} \lambda_{f_k}$ and, therefore, $\sum_{k=1}^{m} \lambda_{f_k} = \sum_{k=1}^{l-1} \lambda_{f_k} + \lambda_A$. Hence, $\lambda_N = \lambda_{N'}$. Finally, we note that $w_d(i) - \sum_{k=l+1}^{m} \lambda_{f_k} > 0$.
  - Suppose $l = m$. Let T be the MPQ-tree obtained from T' by removing i from the set $V_{N'}$ and modifing f_m as follows. If f_m is a leaf or a P-node, then we insert i in the labelling set of f_m. If f_m is a Q-node, then we insert i in every section of f_m. Now, if f_m is a leaf, then we are done. Otherwise, if i does not satisfy the statement for f_m, then we apply the proof on f_m.

- Suppose N' is a Q-node with sections $S'_1, \ldots, S'_m$. Then there are i and a section S'_k with $i \in S'_k$, such that $w_d(i) - \sum_{h>k, i \in S'_h} \lambda_{S'_h} \leq 0$. We choose k such that $k = \min\{h \mid i \in S_h\}$. Let $l = \min\{h \mid w_d(i) > \sum_{h'>h, i \in S'_{h'}} \lambda_{S'_{h'}}\}$. We have $w_d(i) - \sum_{h \geq l, i \in S'_h} \lambda_{S_h} \leq 0$. We distinguish two cases for the value of l.

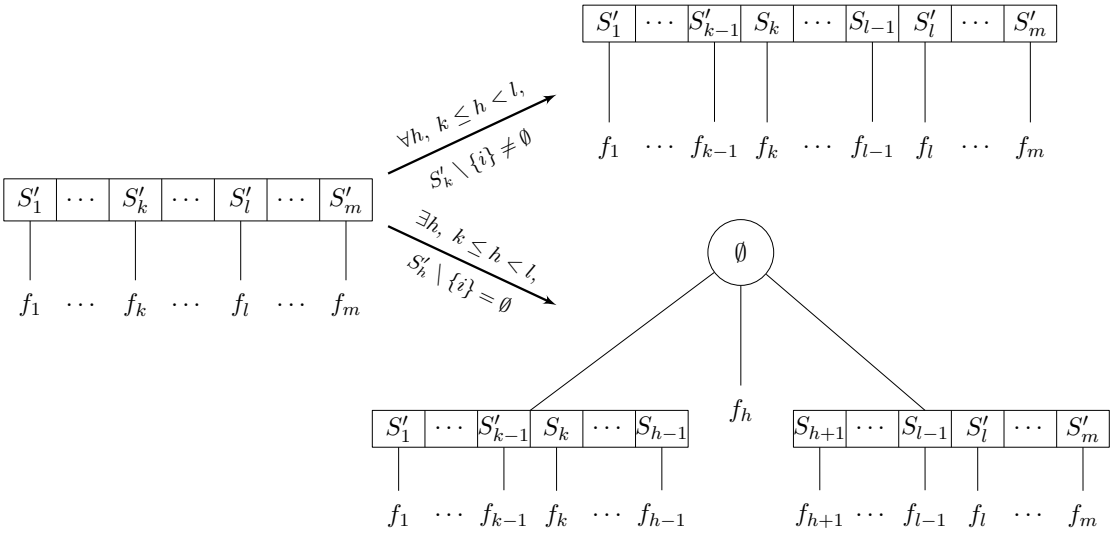


Figure 2.15: Q-node replacement

- Suppose $l < m$. Intuitively, one has to remove i from every section of N' which does not satisfy the statement of the lemma in case of a Q-node. We apply the replacement of Figure 2.15 to construct from T' an MPQ-tree T, in which $\forall h \in \{k, \ldots, l-1\}$, $S_h = S'_h \setminus \{i\}$ and the other sections do not change. To distinguish the widths of T and T', let λ^T_M (resp. $\lambda^{T'}_M$) denote the width of any node (or section) M of T (resp. of T').

  Consider the upper case of the replacement of Figure 2.15. We have $\forall h \in \{l, \ldots, m\}$, $\lambda^{T'}_{S'_h} = \lambda^T_{S'_h}$. Since $w_d(i) - \sum_{h \geq l, i \in S_h} \lambda^{T'}_{S_h} \leq 0$, we have $\forall h \in \{k, \ldots, l-1\}$, $\lambda^T_{S_h} = \lambda^{T'}_{S'_h}$ and, therefore, $\forall h \in \{1, \ldots, k-1\}$, $\lambda^T_{S'_h} = \lambda^{T'}_{S'_h}$. Hence $\lambda^T_N = \lambda^{T'}_N$.

  Consider the bottom case. We suppose that there exists only one section S_h containing only item i. If there are more than one section of this type, one can easily use the same technique of replacement. Let N be the new created P-node and N_1 and N_2 be the two created Q-nodes. Obviously, $\forall h' \in \{l, \ldots, m\}$, $\lambda^{T'}_{S'_{h'}} = \lambda^T_{S'_{h'}}$. By the same arguments as in the upper case of the figure, we have $\forall h' \in \{h+1, \ldots, l-1\}$, $\lambda^{T'}_{S'_{h'}} = \lambda^T_{S_{h'}}$. Now, since $S_{h-1} \cap S_{h+1} = \emptyset$, $\forall h' \in \{k, \ldots, h-1\}$, $\lambda^{T'}_{S'_{h'}} = \lambda^T_{S_{h'}}$ and, therefore, $\forall h' \in \{1, \ldots, k-1\}$, $\lambda^{T'}_{S'_{h'}} = \lambda^T_{S'_{h'}}$. In T', $S'_h = \{i\}$ and the subtree rooted in f_h is not empty, because otherwise the branch of T' containing S'_h and f_h would not represent a maximal clique. Thus $\lambda^{T'}_{S'_h} = \lambda^T_{f_h} > 0$ and $\lambda^T_{f_h} = \lambda^{T'}_{f_h}$. We have $\lambda_{N_1} = \sum_{h'=1}^{h-1} \lambda^{T'}_{S_{h'}}$ and $\lambda_{N_2} = \sum_{h'=h+1}^{m} \lambda^{T'}_{S_{h'}}$. Hence, $\lambda_N = \lambda_{N_1} + \lambda^T_{f_h} + \lambda_{N_2} = \lambda^{T'}_{N'}$.

  Notice that in both cases of the replacement, every item satisfying the condition of the lemma in T', satisfies it also in T.

- Suppose $l = m$. Similarly to the case when $l = m$ and N' is a P-node, we build an MPQ-tree T from T'. For this purpose, we remove i from every section of N' containing i and modify f_m by inserting i as in the case when N' is a P-node. Now, if $S'_m = \{i\}$, then we apply the replacement of Figure 2.16. The case when $m > 3$ (the upper case of the Figure) is straightforward. Consider the case when $m = 3$. Recall that by definition of a Q-node, every item of any section must be contained in at least two sections of the same Q-node. Hence, $S'_1 \subset S'_2$, $S'_3 \subset S'_2$ and $S'_2 = S'_1 \cup S'_3$. Therefore, $S'_1 = S'_2 \setminus \{i\} = A$ and N' is replaced by a P-node as shown in the bottom case of the replacement of Figure 2.16.

  Finally, if f_m is a leaf, we are done. Otherwise, if i does not satisfy the statement of the lemma for f_m, we apply the proof for the new node f_m.

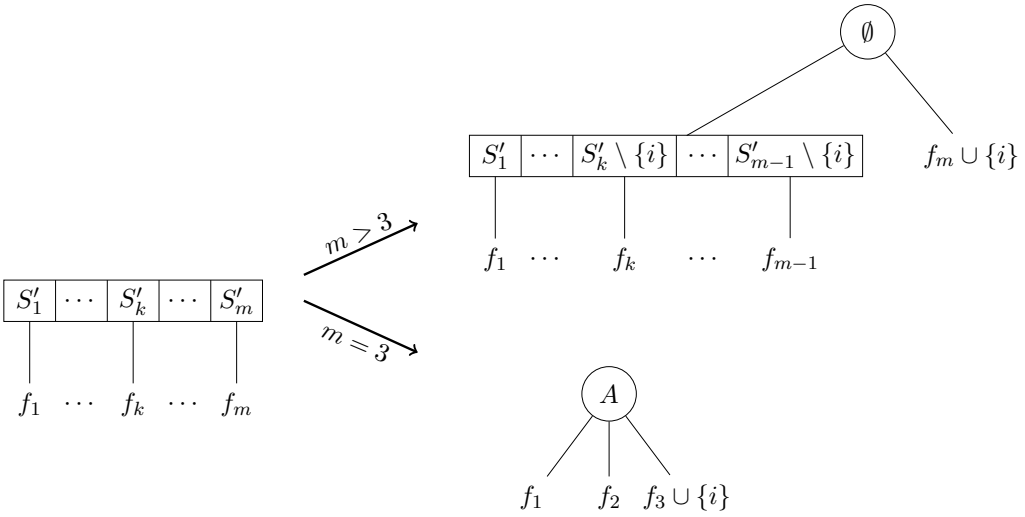


Figure 2.16: Q-node replacement when $S'_m = \{i\}$

By applying the replacement of Figures 2.14, 2.15, 2.16 for all internal nodes (P-type or Q-type) of T', for every item i not satisfying the statement of the lemma, we can construct another MPQ-tree associated with a feasible packing in which every item satisfies this statement. □

Order on the children of a node

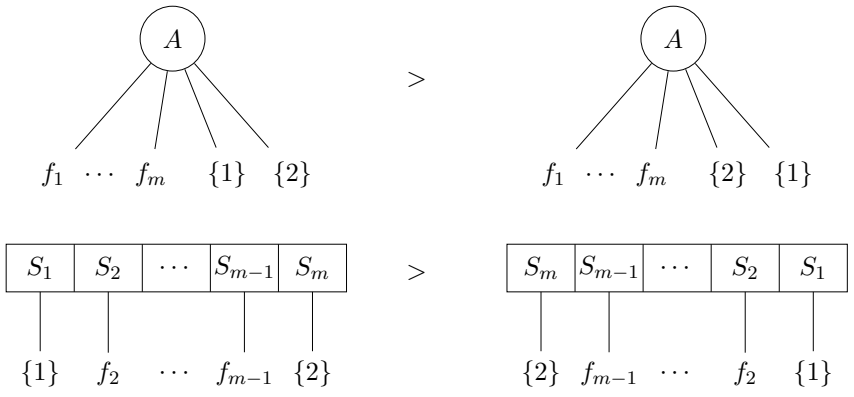


Figure 2.17: Lexicographic order between equivalent MPQ-trees

Using the general properties of PQ-trees, we impose an order on the children of a node during

the generation. We define for any subtree T_s of an MPQ-tree T, a variable $m_T = \min\{i \mid i \in T_s\}$. We say that T is *lexicographically ordered* if:

- For any two subtrees T_1 and T_2 children of the same P-node such that T_1 is the first in the right to left reading order, we have $m_{T_1} > m_{T_2}$.
- For two subtrees T_1 and T_2 children of the same Q-node where T_1 is the rightmost and T_2 is the leftmost, we have $m_{T_1} > m_{T_2}$.

An example of lexicographically ordered MPQ-trees is depicted in Figure 2.17. Notice that due to the definition of PQ-trees, for every MPQ-tree, there is an equivalent lexicographically ordered MPQ-tree with respect to Definition 2.10. Therefore, our algorithm to check feasibility generates all lexicographically ordered possible MPQ-trees.

2.3 Computational results and perspectives

We report the performance of our algorithm on 37 classical benchmarks for OKP-2 from [9, 8, 26, 41] (Tables 2.1) and on 42 benchmarks for OPP-2 defined in [28] (Table 2.2). For OKP-2 instances, we used a basic branch-and-bound procedure to select the items to be checked for feasibility designed in [64].

The program was implemented in Java 6 and was tested on a PC (Pentium IV, 3GHz). These conditions of experimentation are identical to the ones of Joncour and Pêcher [64] and are quite similar to the ones used by Fekete and Schepers [41] (PC with Pentium IV processor, 2,8 GHz, using C++).

Table 2.1 shows the running times of our algorithm along with the ones existing in the literature, as reported in [41]. The first column (JPV) gives our runtimes. The column JP gives the runtimes of the algorithm from [64] and the column FS gives the runtimes of the algorithm from [41]. The column BB corresponds to the algorithm of Baldacci and Boschetti [5], implemented in Visual Digital Fortran 6.0 and run on a Laptop equipped with an Intel Pentium IV, 2.5 GHz. The columns A0, A1, A2 and A3 correspond to the algorithms of Caprara and Monaci, as depicted in [18] and which were implemented in ANSI C and run on a Pentium III 800 MHz. We also report the number of unsolved benchmarks (within the time limit of 1800 seconds) and the average time (computed on the set of instances `cgcut`, `gcut` and `okp`), with the convention that an unsolved benchmark counts for 1800 seconds.

The running times of our algorithm are of interest since it is one of the two algorithms to solve all, but one, of the benchmarks within the time limit of 1800 seconds (`gcut13` is still open, the optimal value being unknown). Compared to Fekete and Schepers', on an average, our running times turned out to be smaller and were significantly better for 6 instances (`cgcut2`, `gcut3`, `gcut8`, `gcut11`, `gcut12` and `okp1`), though Fekete and Schepers' algorithm outperforms ours for the 2 instances `okp2` and `okp5`. Compared to Joncour and Pêcher's, the times are significantly better for 5 okp instances and are significantly worse for 3 instances: cgcut2, gcut4 and gcut8. Additionally, compared to other authors', our running times are considerably better. Note that considering the big difference between our processors and the ones used by Caprara and Monaci for their experiments, one can see that algorithms A1 and A3 are also competitive.

Table 2.2 shows the running times on benchmarks defined in [28]. The third column corresponds to the algorithm of Clautiaux, Carlier and Moukrim [28] implemented in C on a PC (Pentium IV, 2.6 GHz). The size of the containers is (20,20) and there are 10 to 23 items to be packed. These benchmarks are designed to check feasibility (OPP-2) and therefore relevant to our approach. However, the size of containers being small, our approach as well as Joncour and Pêcher's and Fekete and Schepers' is less appropriate compared to the methods using a space discretization. The F (resp. N) character in the name of the instance stand for "feasible" (resp. "non feasible") and X character is used when there exists another instance being of the same

type (feasible or unfeasible) and with the same number of items to be packed. In the case of feasible instances our algorithm has generally a better performance than Fekete and Schepers' and Carlier and Moukrim's, however for half of feasible instances they are slower than Joncour and Pecher's. For unfeasible instances, the running times are generally better that those of the algorithm of Joncour and Pêcher and about half of them are equivalent to those of the other authors. However, for six instances the running times are less competitive than those of the other two algorithms, while only one instance is not solved. In average, our algorithm is thrice faster than Fekete and Schepers', slightly faster than Joncour and Pêcher's but significantly slower than Clautiaux, Carlier and Moukrim's.

Solving the instance `gcut13` is still a challenging issue, the best solution being obtained in [64]. The particularity of this benchmark is that the size of the container is big (3000 by 3000) while there are many items (32) of relatively small sizes to be packed.

We underline the fact that our algorithm is an exact one and that at any moment of the execution we do not use heuristic-like techniques. This is contrary to Fekete and Schepers' algorithm which consists mainly of two parts: in the first one it tries to guess with a heuristic a feasible packing and when it fails, it launches the main procedure which generates the interval graphs in order to check the feasibility (and this is the second part). Nevertheless, our results are competitive, thus have a significant potential to be reused within heuristics. For instance, for `gcut13` as well as for all other instances for which our algorithm is not efficient, one could adapt it by cutting the enumeration scheme. In this perspective, recall that an advantage of the approach using MPQ-trees is that at any moment, it is possible to obtain the current packing configuration. Therefore, it is possible to detect the total size P of portions of the container which are not available any more while building an MPQ-tree. Consequently, intuitively one could assume that for any other MPQ-tree containing the same set of items, the total size of not available portions must not exceed P. We think, that in this case, the obtained results should not omit too much of the potential solutions.

Benchmark	**JPV**[1]	**JP**[1]	**FS**[2]	**BB**[3]	**A0**[4]	**A1**[4]	**A2**[4]	**A3**[4]
ngcut1 ... ngcut11	0	0	0	0				
hccut2 ... hccut5	0	0	0	0				
wang20	0	0	0	-	6	6	17	2
cgcut1	0	0	0	0	0	1	1	1
cgcut2	108	39	>1800	>1800	>1800	>1800	533	531
cgcut3	1	0	0	95	23	23	4	4
gcut1	0	0	0	0	0	0	0	0
gcut2	0	0	0	0	0	0	25	0
gcut3	0	0	4	2	>1800	2	276	3
gcut4	154	28	195	46	>1800	346	>1800	376
gcut5	0	0	0	0	0	0	0	0
gcut6	0	0	0	1	0	0	9	0
gcut7	0	0	2	3	1	0	354	1
gcut8	67	17	253	186	1202	136	>1800	168
gcut9	0	0	0	0	0	0	0	0
gcut10	0	0	0	0	0	0	6	0
gcut11	2	1	8	3	16	14	>1800	16
gcut12	7	3	109	12	63	16	>1800	25
gcut13	>1800	>1800	>1800	>1800	>1800	>1800	>1800	>1800
okp1	1	1	10	779	24	25	72	35
okp2	54	477	20	288	>1800	>1800	1535	1559
okp3	1	7	5	0	21	1	465	10
okp4	1	23	2	14	40	2	0	4
okp5	274	>1800	11	190	40	>1800	513	488
# unsolved	1	2	2	2	5	4	5	1
Average time	66	113	114	248	474	353	582	228

[1] Java, Pentium IV, 3GHz
[2] C++, Pentium IV, 2.8GHz
[3] Visual Digital Fortran 6.0, Pentium IV, 2.5 GHz
[4] ANSI C, Pentium III, 800 MHz

Table 2.1: Running times in seconds for OKP-2 benchmarks

Benchmark	**JPV**[1]	**JP**[1]	**FS**[2]	**CCM**[3]
E02F17	0	0	7	12
E02F20	2	0	> 1800	12
E02F22	2	0	167	4
E04F15	1	2	0	1
E04F17	19	0	13	26
E04F19	15	0	560	7
E04F20	0	0	22	3
E05F15	0	0	0	3
E05F18	0	3	0	126
E05F20	6	0	491	2
E07F15	0	0	0	1
E08F15	0	0	0	117
E20F15	2	0	0	1
E00X23	> 1800	> 1800	> 1800	289
E03X18	26	17	0	22
E05X15	144	336	2	0
E07X15	94	236	0	1
E10X15	46	162	0	1
E13X15	3	7	0	0
E20X15	5	3	0	44
E00N10	0	9	0	0
E00N15	2	1	0	2
E00N23	103	45	> 1800	86
E02N20	0	0	0	1
E03N10	0	0	0	0
E03N15	26	21	0	1
E03N16	48	22	2	32
E03N17	59	57	0	4
E04N15	7	5	0	1
E04N17	3	35	0	1
E04N18	8	26	10	7
E05N15	4	27	0	0
E05N17	1	42	0	1
E07N10	0	0	0	0
E07N15	0	6	0	0
E08N15	4	74	0	1
E10N10	0	0	0	0
E10N15	0	8	0	0
E13N10	0	0	0	0
E13N15	0	0	0	0
E15N10	0	0	0	0
E15N15	0	1	0	0
# unsolved	1	1	3	0
Average time	55	73	158	19

[1]Java, Pentium IV, 3GHz
[2]C++, Pentium IV, 2.8GHz
[3]C, Pentium IV, 2.6 GHz

Table 2.2: Running times in seconds for OPP-2 benchmarks

Chapter 3

Strong edge-colouring

In this chapter we study a problem of graph colouring which models a conflict-free channel assignment in radio networks. We consider two major families of graphs: subcubic graphs (Section 3.1 and outerplanar graphs (Section 3.2). In the third section, we discuss complexity issues and the last section is devoted to open problems.

We first present some definitions, few of them being already mentioned in Chapter 1.

A *strong edge-colouring* c of a graph G is a proper edge-colouring of G, such that for every path $uvxy$ of length 3, we have $c(uv) \neq c(xy)$. We denote by $\chi'_s(G)$ the *strong chromatic index* of G which is the smallest integer k such that G can be strongly edge-coloured with k colours.

We will say that two edges uv and xy are at distance 2 if these edges are not adjacent and there exists a path $uvxy$ of length 3.

An easy observation (mentioned in Chapter 1) is that $\chi'_s(G) = \chi(L(G)^2)$, where χ is the chromatic number and $L(G)^2$ is the square of the line graph of G.

Strong edge-colouring was introduced by Fouquet and Jolivet in [47, 48]. This type of colouring can be used to represent the conflict-free channel assignment in radio networks. Imagine a simplified model of a radio network where transceivers communicate among each other over different frequencies. Since it is a radio network every message sent by a transceiver u on a given frequency is received by all transceivers "close" to u. This observation yields the two following interference situations:

1. Suppose transceivers u and w send a message to v. If both u and w use the same frequency, then v will not be able to understand their messages as they will interfere with each other. Hence they must use distinct frequencies.

2. Suppose now that transceiver u wants to communicate with transceiver v, transceiver w wants to communicate with transceiver x, and v and w are close. If u and w send their messages on a same frequency, then v will receive both messages on the same frequency and so messages will interfere with each other. This type of interference is illustrated in Figure 3.1

If we consider the graph whose vertices are the transceivers, and there is an edge if the corresponding transceivers are close, then solving the frequency assignment problem is equivalent to finding a strong edge-colouring of the graph. For more details on applications and protocols we refer the reader to [6, 75, 77, 82, 85].

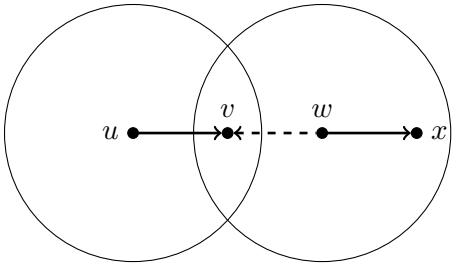


Figure 3.1: The second type of interference. The conflict is on node v.

One of the main questions in studying strong edge-colouring is the following:

"Does there exist a bound of the strong chromatic index in terms of Δ?"

A natural answer for the lower bound can be seen by a simple counting argument on the maximal number of edges adjacent to an edge in the graph:

$$\chi'_s(G) \geq \max_{xy \in E(G)} \{d(x) + d(y) - 1\}$$

This bound was proved to be tight for trees [38] and in the case of some sparse random graphs:

Theorem 3.1 (Frieze *et al.*, 2005 [49])**.** Let $G = G(n,p)$ be the random graph on n vertices where every edge appears with probability p such that $np < \frac{1}{100}\sqrt{\log n / \log\log n}$. Then with high probability $\chi'_s(G) = \max_{xy \in E(G)} \{d(x) + d(y) - 1\}$.

As described in the first chapter of this thesis, another natural lower bound for $\chi'_s(G)$ is the size of the maximum antimatching $a(G)$: $\chi'_s(G) \geq a(G) \geq \max_{xy \in E(G)} \{d(x) + d(y) - 1\}$. In [78] Mahdian proved that the antimatching problem is NP-complete for the class of simple graphs and for bipartite multigraphs, and polynomially solvable for graphs without 4-cycles.

As of upper bounds for the strong chromatic index, the natural one can be obtained by a greedy algorithm and is the maximum number of edges at distance at most 2 from a given edge. Therefore, we have:

$$\chi'_s(G) \leq 2\Delta(\Delta - 1) + 1$$

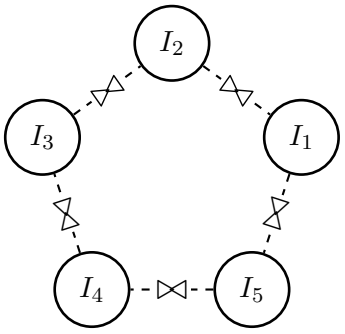
One of the major attempts in refining this bound was proposed in 1985 by Erdős and Nešetřil who conjectured the following:

Conjecture 3.2 (Erdős and Nešetřil, 1985 [35, 37])**.** For every graph G,

$$\chi'_s(G) \leq \begin{cases} \frac{5}{4}\Delta^2 & \text{if } \Delta \text{ is even} \\ \frac{1}{4}(5\Delta^2 - 2\Delta + 1) & \text{if } \Delta \text{ is odd} \end{cases}$$

The bounds of the conjecture are tight as the authors gave a construction of graphs reaching them (see Figure 3.2). The conjecture is still open, the only case being solved is when $\Delta \leq 3$ [1, 61]. For some special classes of graphs, such as trees, chordal graphs or Kneser graphs, the conjecture holds [77]. In the case when $\Delta = 4$, Conjecture 3.2 states that $\chi'_s(G) \leq 20$ and the best proved upper bound is 22 [31].

In [27] Chung *et al.* proved the following result which goes as a support to Conjecture 3.2.



Every I_j is an independent set

If $\Delta = 2k$, then $|I_j| = k$

If $\Delta = 2k+1$, then $|I_1| = |I_2| = |I_3| = k$
and $|I_4| = |I_5| = k+1$

Figure 3.2: Erdős and Nešetřil's construction.

Theorem 3.3 (Chung *et al.*, 1990 [27])**.** Let G be a graph with no induced path of four edges. Then $|E(G)| \leq \frac{5}{4}\Delta^2$ if Δ is even and $|E(G)| \leq \frac{1}{4}(5\Delta^2 - 2\Delta + 1)$, if Δ is odd.

In particular, Theorem 3.3 shows that the construction of Figure 3.2 is a biggest antimatching possible and hence if Conjecture 3.2 is not true, the counterexample must be a graph where there are edges at distance strictly greater than 2.

Several authors tried to give upper bounds for the strong chromatic index in terms of the maximum degree using probabilistic methods.

Theorem 3.4 (Molloy and Reed, 1997 [79])**.** For sufficiently large Δ, every graph G satisfies $\chi'_s(G) \leq 1.998\Delta^2$.

In [37] Faudree *et al.* stated the following conjecture.

Conjecture 3.5 (Faudree *et al.*, 1990 [38])**.** Every bipartite graph has a strong edge-colouring with Δ^2 colours.

The first results on planar graphs were proved in the same paper. The authors proved the following upper bound for planar graphs:

Theorem 3.6 (Faudree *et al.*, 1990 [38])**.** For every planar graph G, $\chi'_s(G) \leq 4\Delta + 4$.

However, until now it is not known whether this bound is tight as the authors showed a construction of planar graphs with $\chi'_s(G) = 4\Delta - 4$ and no better lower and upper bounds for this family of graphs were found since then. In Section 3.2 we improve Theorem 3.6 in the case of outerplanar graphs by showing that every non-trivial outerplanar graph can be strongly edge-coloured with at most $3\Delta - 3$ colours (Theorem 3.22). The obtained result is tight.

The class of Halin graphs was studied in [90, 91, 69, 74, 20] and tight upper bounds were given for different subfamilies of this class. The case of planar 3-regular graphs was first studied in [48]. In the case of subcubic planar graphs, Faudree *et al.* stated the following conjecture:

Conjecture 3.7 (Faudree *et al.*, 1990 [38])**.** For every planar subcubic graph G, $\chi'_s(G) \leq 9$.

The bound the authors proposed is tight, the example proving it being the prism graph (or the complement of a C_6, $\overline{C_6}$) depicted in Figure 3.3. Moreover, this graph is 3-regular and to our knowledge it is the only subcubic planar graph with $\chi'_s = 9$. That is why this conjecture is cited in many articles as a conjecture on cubic graphs (i.e. 3-regular graphs).

Conjecture 3.7 is a special case of an older conjecture stated by Wegner in 1977:

Conjecture 3.8 (Wegner, 1977 [98])**.** If G is a planar graph with maximum degree Δ, then

$$\chi(G^2) \leq \begin{cases} 7 & \text{if } \Delta = 3 \\ \Delta + 5 & \text{if } 4 \leq \Delta \leq 7 \\ \frac{3\Delta}{2} + 1 & \text{if } \Delta \geq 8 \end{cases}$$

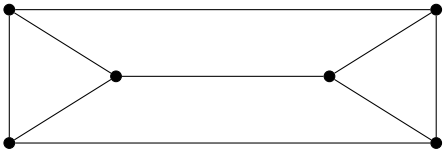


Figure 3.3: The graph $\overline{C_6}$ has $\chi'_s(\overline{C_6}) = 9$

For instance, since a line graph of a planar subcubic graph is a planar graph of maximum degree 4, Conjecture 3.8 for the case when $\Delta \leq 4$ implies Conjecture 3.7.

For cubic Halin graphs Conjecture 3.7 holds [74, 20]. In Section 3.1.3 we give some progress towards Conjecture 3.7, by showing that it holds in a large subclass of planar subcubic graphs. More precisely, in Theorem 3.20 we prove that every planar subcubic graph with no induced cycles of length 4 or 5 has the strong chromatic index at most 9.

Motivated by the same conjecture, we studied the case of general sparse subcubic graphs (Section 3.1.1) by giving upper bounds on the strong chromatic index in terms of maximum average degree and by deriving some immediate consequences for the case of subcubic planar graphs. To this regard, in Theorem 3.9, we show that every subcubic graph with maximum average degree strictly less than $\frac{7}{3}$ (resp. $\frac{5}{2}$, $\frac{8}{3}$, $\frac{20}{7}$) can be strongly edge-coloured with six (resp. seven, eight, nine) colours. In the following section (3.1.2) we discuss on the optimality of the upper bounds on the maximum average degree. We show that except for the bound $\frac{8}{3}$, the other ones are optimal. Some of the results we obtained in this section have been published in [59].

From the computational point of view, deciding whether a bipartite graph with girth g is strongly k-edge-colourable for every $k \geq 4$ was proved to be NP-complete by Mahdian [78]. The same author showed that the strong edge-colouring problem can be solved in polynomial time for chordal graphs and co-comparability graphs. In [89] Salavatipour showed a polynomial algorithm for graphs of bounded tree-width. Bunde *et al.* proved that strong 5-edge-colouring is NP-complete in the class of bipartite 2-degenerate graphs with girth 6 and maximum degree 3 [36]. However, for planar graphs no complexity result was stated. In Section 3.3 we show that the strong k-edge-colouring problem for $k = 4, 5, 6$ remains NP-complete in some restricted subclasses of planar graphs of maximum degree 3.

A closely related notion to strong edge-colouring is *induced matching*. An induced matching $\mathcal{I}$ of a graph G is a set of non-adjacent edges of G (matching) such that no two of them are joined by an edge in G. In other words, the graph induced by the endpoints of the edges of $\mathcal{I}$ has maximum degree 1. The strong edge-colouring can be seen as a partition of the set of edges into a set of induced matchings (see [47, 37, 38]). Induced matchings are also known as strong matchings, see [47]. Clearly from the definitions, one can observe that finding the size of the maximum induced matching in a graph G is equivalent to finding the maximum independent set in $L(G)^2$.

Recall that an *antimatching* of a graph G is a subset of its edges which are pairwise at distance at most 2. As showed in the first chapter, finding the size of the maximum antimatching is equivalent to finding the size of the maximum clique in $L(G)^2$. Thus we have a clear correspondence between the strong edge-colouring, induced matchings and antimatchings and respectively the classical vertex-colouring, independent sets and cliques of $L(G)^2$.

The notion of induced matchings was first studied by Stockmeyer and Vazirani in [95]. They proved that the maximum induced matching problem is NP-hard even for bipartite graphs of maximum degree 4 [95]. Cameron proved in [17] that finding a maximum induced matching in chordal graphs can be done in polynomial time and that the problem is NP-complete for bipartite graphs of girth at least g, for every g positive integer. Lozin proved in [76] that deciding whether a set of edges is a maximum induced matching is NP-complete for bipartite graphs of maximum degree 3. Duckworth *et al.* proved in [34] that the induced matching problem is NP-complete

even when restricted to planar cubic graphs. In [58] we unified all these results and proved that the problem is NP-complete for bipartite planar graphs of maximum degree 3 with an arbitrarily large girth g.

This chapter has four sections. In Section 3.1 we consider two restrictions of subcubic graphs: sparse graphs and planar graphs. Section 3.2 is devoted to the case of outerplanar graphs and in Section 3.3 we study some complexity aspects of the problem. We end the chapter by a discussion on our results and some open problems for future work (Section 3.4).

3.1 Subcubic graphs

In this section we concentrate on the family of graphs with maximum degree 3. We prove lower and upper bounds for the strong chromatic index of different subclasses of this family. First, we study the case of sparse graphs.

3.1.1 Sparse graphs through maximum average degree (mad)

Let $\text{mad}(G)$ be the maximum average degree of the graph G defined as follows:

$$\text{mad}(G) = \max\left\{\frac{2|E(H)|}{|V(H)|},\ H \subseteq G\right\}$$

In some sense, the maximum average degree of a graph measures its sparseness. This parameter can be computed in polynomial time [29, 62]. We study the family of subcubic graphs in terms of the maximum average degree. Namely, we prove the following results:

Theorem 3.9. Let G be a subcubic graph:

(i) If $\text{mad}(G) < \frac{7}{3}$, then $\chi'_s(G) \leq 6$.

(ii) If $\text{mad}(G) < \frac{5}{2}$, then $\chi'_s(G) \leq 7$.

(iii) If $\text{mad}(G) < \frac{8}{3}$, then $\chi'_s(G) \leq 8$.

(iv) If $\text{mad}(G) < \frac{20}{7}$, then $\chi'_s(G) \leq 9$.

An interesting fact about these upper bounds on the maximum average degree is that for parts (i), (ii) and (iv) they are tight. We show it on examples of graphs after proving the theorem.

Recall that if a graph G has $\text{mad}(G) < 2$, then G is a forest. We observe that every subcubic graph G with $\text{mad}(G) < 2$ verifies $\chi'_s(G) \leq 5$ and the bound on the maximum average degree is best possible (sine there exist graphs with $\text{mad}(G) = 2$ and $\chi'_s(G) = 6$, see Figure 3.4).

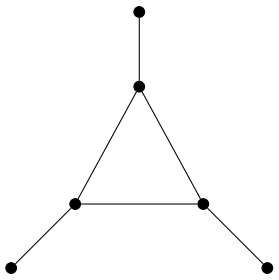


Figure 3.4: A graph G with $\text{mad}(G) = 2$ and $\chi'_s(G) = 6$

The following lemma that belongs to folklore gives the relationship between the maximum average degree and the girth of a planar graph.

Lemma 3.10 (Folklore)**.** Let G be a planar graph with girth at least g. Then, $\text{mad}(G) < \frac{2g}{g-2}$.

Proof. Let G be a connected planar graph with girth at least g. Let H be a subgraph of G. Note that H is planar and has girth at least g. Hence, $g|F(H)| \leq 2|E(H)|$, where $F(H)$ is the set of faces of H. According to Euler's Formula, we obtain:

$$2g - g|V(H)| + g|E(H)| = g|F(H)| \leq 2|E(H)|$$

Hence,

$$2g + (g-2)|E(H)| \leq g|V(H)|$$

$$2|E(H)|(2g + (g-2)|E(H)|) \leq 2|E(H)|g|V(H)|$$

$$\frac{2|E(H)|}{|V(H)|} \leq \frac{2g|E(H)|}{2g + (g-2)|E(H)|} < \frac{2g}{g-2}$$

for every subgraph H of G. □

According to Lemma 3.10 and Theorem 3.9, one can derive the following result:

Corollary 3.11. Let G be a planar subcubic graph with girth g:

(i) If $g \geq 14$, then $\chi'_s(G) \leq 6$.

(ii) If $g \geq 10$, then $\chi'_s(G) \leq 7$.

(iii) If $g \geq 9$, then $\chi'_s(G) \leq 8$.

(iv) If $g \geq 7$, then $\chi'_s(G) \leq 9$.

We will later improve part (iv) by showing that every planar graph G without cycles of length 4 and 5 verifies $\chi'_s(G) \leq 9$ (Theorem 3.20 of Section 3.1.3).

Notations Let G be a graph. A vertex of degree k (resp. at most k) is called a k-vertex (resp. k^--vertex). A *good* 2*-vertex* is a vertex of degree 2 being adjacent to two 3-vertices, otherwise it is a *bad* 2*-vertex*. A 3_k-vertex is a 3-vertex adjacent to exactly k 2-vertices. Two edges are at distance 1 if they are adjacent. Two edges are at distance 2 if they are not at distance 1 and there exists an edge adjacent to both of them. We define $N_2(uv)$ as the set of edges at distance at most 2 from the edge uv and we denote by $SC(N_2(uv))$ the set of colours used by edges in $N_2(uv)$. Finally, we use $[\![n]\!]$ to denote the set of integers $\{1, 2, \ldots, n\}$.

The proof of Theorem 3.9 uses the method of reducible configurations and the discharging technique. The proof is done by minimum counterexample. In each of the cases, for the minimum counterexample H, we prove the non-existence of some configurations *i.e.* a set $\mathcal{S}$ of subgraphs which cannot appear in H. We define the weight function $\omega : V(H) \to \mathbb{R}$ with $\omega(x) = d(x) - m$ (where $m \in \mathbb{R}$ is the value of the upper bound on the maximum average degree given by Theorem 3.9). It follows from the hypothesis on the maximum average degree that the total sum of weights is strictly negative. In the next step, we define discharging rules to redistribute weights and once the discharging process is finished, a new weight function ω^* will be produced. During the discharging process the total sum of weights is kept fixed. Nevertheless, by the non-existence of $\mathcal{S}$, we can show that $\omega^*(x) \geq 0$ for all $x \in V(H)$. This leads to the following contradiction:

$$0 \leq \sum_{x \in V(H)} \omega^*(x) = \sum_{x \in V(H)} \omega(x) < 0$$

and hence, this counterexample cannot exist.

Proof of (i) of Theorem 3.9

Let H be a counterexample to part (i) of Theorem 3.9 minimizing $|E(H)| + |V(H)|$: H is not strongly edge-colourable with six colours, $\text{mad}(H) < \frac{7}{3}$ and $\forall e \in E(H)$, $\chi'_s(H \setminus e) \leq 6$. One can assume that H is connected; otherwise, by minimality of H, we can colour independently each connected component. A 3-vertex adjacent to a 1-vertex is a *light* 3*-vertex*. Otherwise it is a *heavy* 3*-vertex*.

Claim 3.12. The minimal counterexample H to part (i) of Theorem 3.9 satisfies the following properties:

1. H does not contain a 1-vertex adjacent to a 2-vertex.
2. H does not contain a 3-vertex adjacent to a 1-vertex and a 2^--vertex.
3. H does not contain a path uvw where u, v and w are 2-vertices.
4. H does not contain a path uvw where u, v and w are three light 3-vertices.
5. H does not contain a triangle xyz, where x is a light 3-vertex.
6. H does not contain a path $stuvw$ where s, t, v and w are four light 3-vertices, u is a 3-vertex adjacent to a third light 3-vertex x.

Proof. In the following, we give a proof for each of the items of the claim.

1. Suppose H contains a 1-vertex u adjacent to a 2-vertex v. Let us consider $H' = H \setminus \{uv\}$, which by minimality of H is strongly edge-colourable with six colours. By counting the number of available colours to extend a colouring of H' to H, it is easy to see that we have at least three colours left for uv.

2 Trivial by a counting argument.

3. Suppose H contains a path uvw where u, v and w are 2-vertices. Let us consider $H' = H \setminus \{uv, vw\}$, which by minimality of H is strongly edge-colourable with six colours. By counting the number of available colours to extend a colouring of H' to H, it is easy to see that we have at least two colours left for uv and at least one colour left for vw (after the colouring of uv).

4. Suppose H contains a path $xuvwy$ where u, v and w are three light 3-vertices. Call u_1 (resp. v_1, w_1) the neighbour of u (resp. v, w) of degree 1. Assume $N(x) = \{u, x_1, x_2\}$, $N(u) = \{x, u_1, v\}$, $N(v) = \{u, v_1, w\}$, $N(w) = \{v, w_1, y\}$, $N(y) = \{w, y_1, y_2\}$ (see Figure 3.5). Let us consider $H' = H \setminus \{uu_1, uv, vv_1, vw, ww_1\}$. By minimality of H, there exists a strong edge-colouring ϕ of H', using six colours. We will extend this colouring to H. Suppose first, $\phi(ux) = \phi(wy)$. We colour uv, vw, uu_1, ww_1 and vv_1 in this order, which is possible by counting for each edge the number of available colours to extend the colouring. Suppose now, $\phi(ux) \neq \phi(wy)$. W.l.o.g. we can assume that $\phi(ux) = 5$ and $\phi(wy) = 6$. First, we try to colour the edge uu_1 with the colour 6. If it is possible, then we assign the colour 6 to uu_1 and we colour uv, vw, ww_1 and vv_1 in this order, which is possible by counting the number of available colours to extend the colouring. If we cannot colour uu_1 with the colour 6, we are sure that the colour 6 appears in the neighbourhood of x. W.l.o.g. we can assume that $\phi(xx_1) = 6$. By applying the same reasoning on ww_1, we can assume w.l.o.g. that $\phi(yy_1) = 5$. We assign now the same colour α to uu_1 and ww_1, with $\alpha \notin \{\phi(xx_2), 5, 6, \phi(yy_2)\}$. Finally, we colour uv, vw and vv_1 in this order, which is possible by counting the number of available colours to extend the colouring. In each case the extension of ϕ to H is possible which is a contradiction.

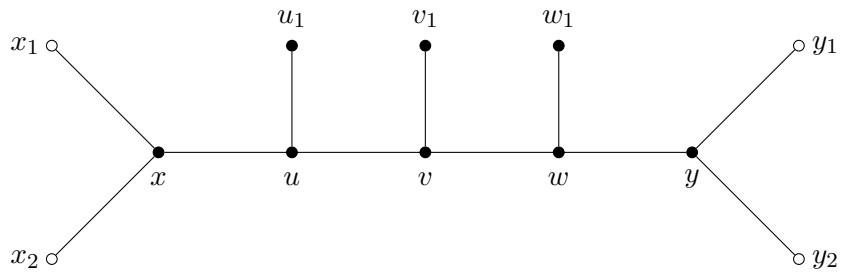


Figure 3.5: The configuration of Claim 3.12(4)

5. Suppose H contains a triangle xyz, where x is a light 3-vertex and let x_1 be the 1-vertex neighbour of x. By minimality of H, the graph $H \setminus xx_1$ can be strongly edge-coloured with at most six colours. Since $|N_2(xx_1)| \leq 5$, every colouring of $H \setminus xx_1$ using the minimum number of colours can be extended to H.

6. Suppose H contains a path $stuvw$ where s, t, v and w are four light 3-vertices, u is a 3-vertex adjacent to a light 3-vertex x distinct from s, t, v, w. Call s_1 (resp. t_1, v_1, w_1, x_1) the neighbour of s (resp. t, v, w, x) of degree 1. Let r (resp. y, z) be the third neighbour of s (resp. x, w). Also, for $i = 1, 2$, let r_i (resp. y_i, z_i) be the neighbours of r (resp. y, z), other than s (resp. x, w). By Claim 3.12(2), r, y and z are 3-vertices. By Claims 3.12(4) and 3.12(5), we can assume that Figure 3.6 illustrates the given configuration (with r, y, z possibly not distinct).

   Let us consider $H' = H \setminus \{ss_1, tt_1, vv_1, ww_1, xx_1, st, tu, uv, vw, ux\}$. By minimality of H, there exists a strong edge-colouring ϕ of H', using six colours. We show how to extend this colouring to H.

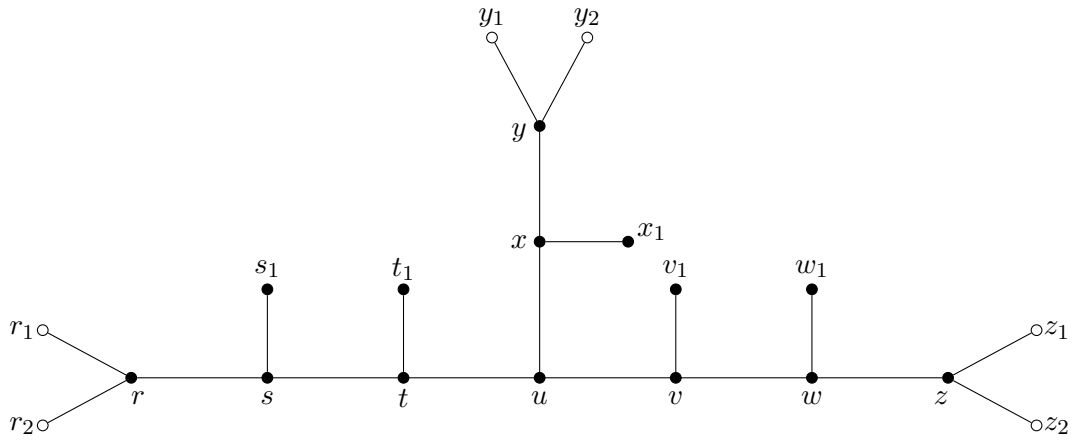


Figure 3.6: The configuration of Claim 3.12(6)

   Without loss of generality we can suppose that $\phi(xy) = 1$, $\phi(yy_1) = 2$ and $\phi(yy_2) = 3$. First, we colour edge ux and we distinguish two cases:

   (a) Suppose ux can be coloured with the same colour as rs, say colour 4. We colour now uv with a colour of yy_1, yy_2 that does not appear on wz. Finally, we consider the remaining edges in the following order: vw, ww_1, vv_1, tu, st, ss_1, tt_1 and xx_1. At each step, there exists an available colour for the corresponding edge.

   (b) Suppose ux cannot be coloured with the same colour as rs or wz. Hence $\phi(rs), \phi(wz) \in \{1, 2, 3\}$. Then it is easy to observe that there exists a colour, say 4, such that ux and ss_1, can be coloured with 4. We fix $\phi(ux) = \phi(ss_1) = 4$.

Next, we distinguish the following cases for $\phi(rs)$ and $\phi(wz)$:

- Suppose $\phi(rs) \in \{\phi(yy_1), \phi(yy_2)\} = \{2, 3\}$ and $\phi(wz) \neq \phi(rs)$. Without loss of generality we can suppose that $\phi(rs) = \phi(yy_1) = 2$. Then we fix $\phi(uv) = 2$. We colour the remaining edges in the following order: vw, ww_1, vv_1, tu, st, tt_1 and xx_1. Note that at each step, there exists a colour left for the corresponding edge.
- Suppose $\phi(rs) \in \{\phi(yy_1), \phi(yy_2)\} = \{2, 3\}$ and $\phi(wz) = \phi(rs)$. Without loss of generality we can suppose that $\phi(wz) = \phi(rs) = 2$ and we fix $\phi(uv) = 3$.
  Suppose we can assign $\phi(ww_1) = 4$. Next, we colour first st and then tu (note that at each step there is at least one colour left). If there is a colour left for vw, then the colouring of H can be finished easily as edges tt_1, vv_1 and xx_1 are pairwise at distance 3 and for each of these edges there would be a colour left. Therefore, there is no colour left for vw which implies that $\{\phi(tu), \phi(zz_1), \phi(zz_2)\} = \{1, 5, 6\}$ and since tu cannot be coloured 1, without loss of generality we can assume $\phi(tu) = 5$, $\phi(zz_1) = 1$ and $\phi(zz_2) = 6$. Similarly, by uncolouring st, recolouring tu with 6 and assigning to vw colour 5, we conclude that $\{\phi(rr_1), \phi(rr_2)\} = \{1, 5\}$. But then we do the following reassignment of colours $\phi(ss_1) = 6$, $\phi(st) = 4$, $\phi(tu) = 5$, $\phi(ux) = 6$, $\phi(vw) = 4$, $\phi(ww_1) = 5$. Finally there is a colour left for each of the edges tt_1, vv_1 and xx_1, thus we are done.
  Suppose ww_1 cannot be coloured with 4. Therefore, without loss of generality $\phi(zz_1) = 4$. On the other hand, recall that $\phi(ss_1) = \phi(ux) = 4$ and by previous paragraphs, it is not possible to colour ss_1, ux and ww_1 with the same colour (4, 5 or 6). Hence we must have $\{5, 6\} \subseteq \{\phi(zz_2), \phi(rr_1), \phi(rr_2)\}$. Obviously one of the colours 5 or 6, say 5, is not assigned to zz_2 and is assigned to either rr_1 or rr_2 (we can suppose $\phi(rr_1) = 5$). Hence we can assign $\phi(ww_1) = \phi(tu) = 5$ and we colour the remaining edges in the following order: vw, vv_1, st, tt_1, xx_1. Observe that at each step there exists at least one colour left for every edge.
- Suppose $\{\phi(rs), \phi(wz)\} \cap \{\phi(yy_1), \phi(yy_2)\} = \emptyset$. Hence, $\phi(rs) = \phi(wz) = 1$. We fix $\phi(tu) = 2$ and then we colour the following edges in the given order: st, vw, uv, vv_1, tt_1 and xx_1. It remains to colour ww_1. If we have a colour left for ww_1, then we are done. Otherwise, $\{\phi(uv), \phi(vv_1), \phi(vw), \phi(zz_1), \phi(zz_2)\} = \{2, 3, 4, 5, 6\}$. Therefore, $\{\phi(uv), \phi(vv_1), \phi(vw)\} = \{3, 5, 6\}$ and $\{\phi(zz_1), \phi(zz_2)\} = \{2, 4\}$. But then we permute the colours of tu and uv and we obtain a free colour for ww_1 (which is the same as the colour of tu). A contradiction.

□

As a corollary from the proof of Claim 3.12 we derive the following:

Corollary 3.13. The minimal counterexample H to part (i) of Theorem 3.9 does not contain a path $stuvw$ where s, t, v and w are either light 3-vertices or 2-vertices and u is a 3-vertex adjacent to a vertex x which is either a light 3-vertex x or a 2-vertex.

Let H'' be the graph obtained from H by removing all 1-vertices of H, *i.e.* $H'' = H \setminus \{v \in V(H),\ d_H(v) = 1\}$. Clearly, H'' is connected and $\text{mad}(H'') < \frac{7}{3}$.

One can derive the following structural properties of H'':

Claim 3.14. Due to Claim 3.12 and to Corollary 3.13, H'' has the following properties:

1. $\delta(H'') \geq 2$, where $\delta(H'')$ is the minimum degree of H'' (from Claim 3.12(1) and 3.12(2)).
2. H'' does not contain a path uvw where u, v, w are 2-vertices (from Claims 3.12(2), 3.12(3) and 3.12(4)).
3. H'' does not contain a 3_3-vertex adjacent to two bad 2-vertices (from Corollary 3.13).

For each vertex x of H'', we assign a charge $w(x)$ equal to $d(x)-\frac{7}{3}$. We apply now a discharging procedure on H'' with the following rules:

(R1) Every 3-vertex gives $\frac{1}{3}$ to each adjacent bad 2-vertex.

(R2) Every 3-vertex gives $\frac{1}{6}$ to each adjacent good 2-vertex.

Let $v \in V(H'')$ be a k-vertex. By Claim 3.14(1), $k \geq 2$. Consider the following cases:

Case k = 2. Observe that $\omega(v) = -\frac{1}{3}$. Suppose v is a bad 2-vertex. By Claim 3.14(2), v is adjacent to a 3-vertex. Hence, by (R1), $\omega^*(v) = -\frac{1}{3} + \frac{1}{3} = 0$. If v is a good 2-vertex, then $\omega^*(v) = -\frac{1}{3} + 2 \times \frac{1}{6} = 0$ by (R2).

Case k = 3. Observe that $\omega(v) = \frac{2}{3}$. Suppose v is adjacent to a bad 2-vertex. By Claim 3.14(2), v is not adjacent to another bad 2-vertex. Hence, by (R1) and (R2), $\omega^*(v) \geq \frac{2}{3} - 1 \times \frac{1}{3} - 2 \times \frac{1}{6} = 0$. If v is not adjacent to a bad 2-vertex, then $\omega^*(v) \geq \frac{2}{3} - 3 \times \frac{1}{6} > 0$ by (R2).

Therefore, H'' cannot exist and consequently H does not exist neither. This completes the proof.

Proof of (ii) of Theorem 3.9

Let H be a counterexample to part (ii) of Theorem 3.9 minimizing $|E(H)| + |V(H)|$: H is not strongly edge-colourable with seven colours, $\text{mad}(H) < \frac{5}{2}$ and for any edge e, $\chi'_s(H \setminus e) \leq 7$. Recall that $\omega(x) = d(x) - \frac{5}{2}$. In this subsection a 3-vertex adjacent to a 2-vertex is a *light* 3*-vertex*. Otherwise it is a *heavy* 3*-vertex*.

Claim 3.15. The minimal counterexample H satisfies the following properties:

1. H does not contain 1^--vertices.
2. H does not contain a path uvw where u, v and w are 2-vertices.
3. H does not contain a 3-vertex adjacent to two 2-vertices one of them being bad.
4. H does not contain two 3_3-vertices having a 2-vertex as a common neighbour.
5. H does not contain a 3_3-vertex u with one of the neighbours, say v, adjacent to a 3_2-vertex w having as neighbours a 2-vertex w_1 and a 3-vertex w_2, such that:
   (a) either w_1 is adjacent to a 3_3-vertex
   (b) or w_2 is a 3_2-vertex
   (c) or w_2 is a light 3-vertex

Proof.

1-2. Trivial.

3. Suppose H contains a 3-vertex u having a bad 2-vertex v as a neighbour. Call w, the bad 2-vertex adjacent to v. Let us consider $H' = H \setminus \{uv, vw\}$, which by minimality of H is strongly edge-colourable with seven colours. By counting the number of available colours to extend a colouring of H' to H, it is easy to see that we have at least one colour left for uv and at least one colour left for vw (after the colouring of uv).

4. Suppose H contains two 3_3-vertices u and w having a 2-vertex v as a common neighbour. $N(u) = \{u_1, u_2, v\}$, $N(w) = \{w_1, w_2, v\}$, $N(u_1) = \{u, x\}$, $N(u_2) = \{u, y\}$, $N(x) = \{u_1, x_1, x_2\}$, $N(y) = \{u_2, y_1, y_2\}$, $N(w_1) = \{w, t\}$, $N(w_2) = \{w, z\}$, $N(t) = \{w_1, t_1, t_2\}$, $N(z) = \{w_2, z_1, z_2\}$ (see Figure 3.7). Let us consider $H' = H \setminus \{uv, vw\}$. Since H is a minimum counterexample, $\chi'_s(H') \leq 7$ and there exists a strong edge-colouring of H', ϕ using seven colours. We will extend this colouring to H. First, we want to colour vw. Observe that $|[\![7]\!] \setminus SC(N_2(vw))| \geq 1$, so we pick the colour left and we colour vw. Next, if we cannot colour uv, then $|[\![7]\!] \setminus SC(N_2(uv))| = 0$ and without loss of generality, we can assume that $\phi(vw) = 1$, $\phi(ww_1) = 2$, $\phi(ww_2) = 3$, $\phi(uu_1) = 4$, $\phi(uu_2) = 5$, $\phi(u_1x) = 6$, $\phi(u_2y) = 7$. If we can recolour recolour vw, then we could assign $\phi(uv) = 1$. Therefore, without loss of generality $\phi(w_1t) = 6$, $\phi(w_2z) = 7$. If it is possible to recolour ww_1 with 4 or 5, then we could assign $\phi(uv) = 2$ and we would complete the colouring of H. Therefore, we have $\phi(tt_1) = 5$, $\phi(tt_2) = 4$. Similarly, we cannot recolour ww_2 with 4 or 5 and hence $\phi(zz_1) = 5$, $\phi(zz_2) = 4$. Symmetrically, we continue to try to recolour in the same manner the edges uu_1 and uu_2. If in one of the steps, the recolouring is possible, then we will have a colour free to use for uv. At the end, without loss of generality we obtain the following colours: $\phi(xx_1) = 2$, $\phi(xx_2) = 3$, $\phi(yy_1) = 3$, $\phi(yy_2) = 2$. Next, having this knowledge about the colours of the edges, we can recolour some of the edges as follows: $\phi(uu_2) = \phi(ww_1) = 1$, $\phi(vw) = 5$, $\phi(uv) = 2$. It is easy to see that there are no "conflicts" between the colours. Hence the extension of ϕ to H is possible which is a contradiction.

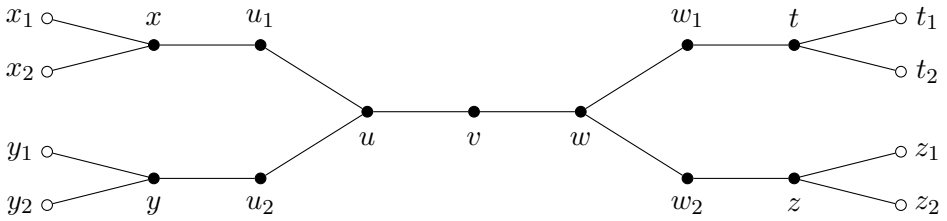


Figure 3.7: The configuration of Claim 3.15(4)

5. Suppose H contains a path uvw where u is a 3_3-vertex, v is a 2-vertex, w is a 3_2-vertex and w is adjacent to a 2-vertex w_1 (distinct from v) and to a 3-vertex w_2. Let $H' = H \setminus \{u_1u, u_2u, uv, vw\}$. By minimality of H, $\chi'_s(H') \leq 7$ and there exists a strong edge-colouring ϕ of H', which uses seven colours. We will extend this colouring to H. We colour the edges vw, uv and u_2u in this order. Note that at each step there exists at least one colour left for the corresponding edge. In order to complete the strong edge-colouring of H, we must assign a colour to u_1u. If $|[\![7]\!] \setminus SC(N_2(u_1u))| \geq 1$, then we are done. Hence $|[\![7]\!] \setminus SC(N_2(u_1u))| = 0$ and since $|N_2(u_1u)| = 7$, all the colours of $SC(N_2(u_1u))$ must be distinct. Next, observe that it is possible to colour u_1u with the colour of u_2u, uncolour u_2u and then apply the same argument as previously to show that all the colours of $SC(N_2(u_2u))$ must be distinct. And then similarly, it is possible to colour u_1u with the colour of uv, to uncolour uv and if there is no colour left for uv, then all the colours of $SC(N_2(uv))$ must be distinct. We conclude that $w_1 \neq u_1, u_2$ and $w_2 \neq x, y, x_1, x_2, y_1, y_2$. Hence the configuration and its fixed precolouring of edges is as depicted in Figure 3.8.

5.a Suppose w_1 is adjacent to a 3_3-vertex t as in Figure 3.9.

Let us consider the edge ww_1. Observe that $SC(N_2(ww_1))$ contains the colours 1, 4 and 5. Otherwise, we can recolour ww_1 with 1 (or 4, or 5), vw with 6 and u_1u with 3. This extends the colouring to whole H which is a contradiction.

Observe that $3 \in \{\phi(tt_1), \phi(tt_2)\}$. Otherwise, we can permute the colours of vw and ww_1, and assign colour 3 to u_1u. Similarly, if we could permute the colours of ww_1

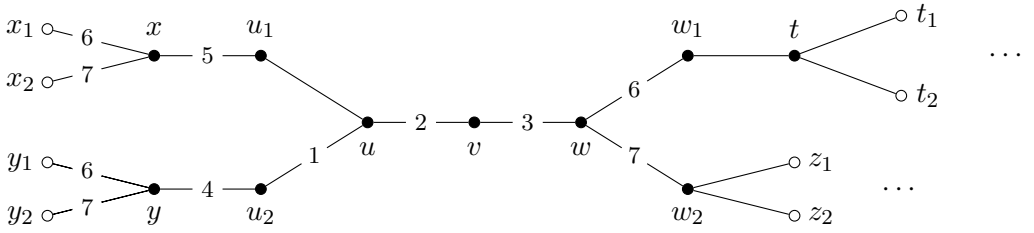


Figure 3.8: A fixed precolouring of the configuration of Claim 3.15(5)

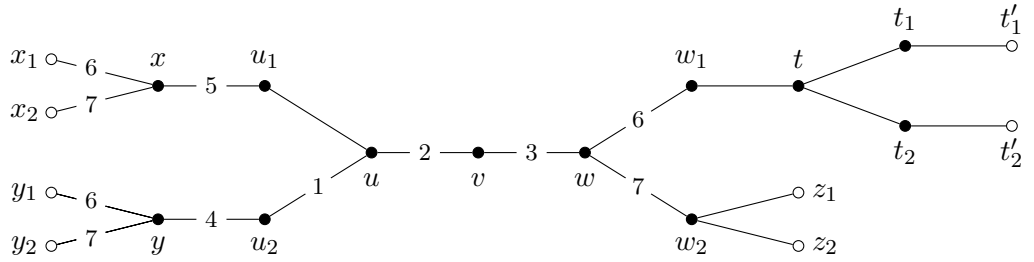


Figure 3.9: The configuration of Claim 3.15(5.a)

and uv, then u_1u could be coloured with 2, which is impossible. Therefore, we have $2 \in \{\phi(w_1t), \phi(tt_1), \phi(tt_2), \phi(w_2z_1), \phi(w_2z_2)\}$.

We conclude that $\{\phi(w_1t), \phi(tt_1), \phi(tt_2), \phi(w_2z_1), \phi(w_2z_2)\} = \{1, 2, 3, 4, 5\}$ and $3 \in \{\phi(tt_1), \phi(tt_2)\}$ (w_1t, tt_1, tt_2, w_2z_1, w_2z_2 are assigned pairwise distinct colours).

Suppose that $\{\phi(t_1t'_1), \phi(t_2t'_2)\} \neq \{\phi(w_2z_1), \phi(w_2z_2)\}$. Let $\alpha \in \{\phi(w_2z_1), \phi(w_2z_2)\} \setminus \{\phi(t_1t'_1), \phi(t_2t'_2)\}$, $\alpha \in \{1, 2, 4, 5\}$ ($3 \in \{\phi(tt_1), \phi(tt_2)\}$).

We do the following assignment of colours (in the given order): $\phi(u_1u) = 2$, $\phi(uv) = 3$, $\phi(vw) = 6$, $\phi(ww_1) = \phi(w_1t)$, $\phi(w_1t) = \alpha$.

It follows that $\{\phi(t_1t'_1), \phi(t_2t'_2)\} = \{\phi(w_2z_1), \phi(w_2z_2)\}$ and $6 \notin \{\phi(t_1t'_1), \phi(t_2t'_2)\}$. But then we permute the colours of w_1t and ww_1, recolour uv with colour 6 and assign to u_1u colour 2. We obtain a strong edge-colouring of H and this is a contradiction.

5.b Suppose w_2 is a 3_2-vertex as depicted in Figure 3.10.

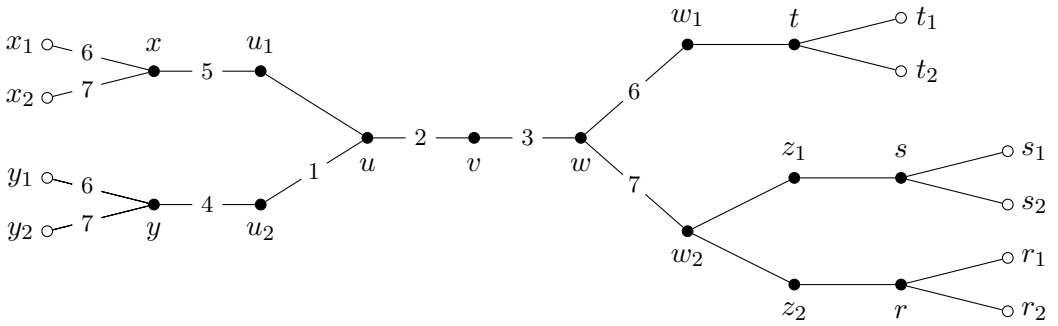


Figure 3.10: The configuration of Claim 3.15(5.b)

Consider in this case the edge ww_2.

Observe that $3 \in \{\phi(z_1s), \phi(z_2r)\}$. If not, we can permute the colours of vw and ww_2, and assign 3 to u_1u. Observe that $1, 2, 4, 5 \in \{\phi(w_1t), \phi(w_2z_1)\phi(w_2z_2), \phi(z_1s), \phi(z_2r)\}$. Otherwise we can recolour ww_2 with 1 or 2 or 4 or 5, uv with 7, and assign colour 2 to u_1u. Hence $\{\phi(w_1t), \phi(w_2z_1)\phi(w_2z_2), \phi(z_1s), \phi(z_2r)\} = \{1, 2, 3, 4, 5\}$. Observe that $3 \in \{\phi(tt_1), \phi(tt_2)\}$. Otherwise we can permute the colours of vw and ww_1, and assign colour 3 to u_1u. Hence, without loss of generality we can assume $\phi(tt_1) = \phi(z_1s) = 3$.

Moreover, we prove that $\phi(tt_2) = \phi(z_2r)$. By contradiction, assume that $\phi(z_2r) = \alpha \neq \phi(tt_2)$ ($\alpha \in \{1, 2, 4, 5\}$). We recolour ww_1 with α, uv with 6, and assign 2 to u_1u.

Now let us uncolour uv and assign colour 2 to u_1u. Observe that $7 \in \{\phi(ss_1), \phi(ss_2)\}$. Otherwise we can permute the colours of ww_2 and w_2z_1, and assign colour 7 to uv. Observe that $\phi(w_1t) \in \{\phi(ss_1), \phi(ss_2)\}$. Otherwise we use $\phi(w_2z_1)$ to recolour ww_1, we recolour w_2z_1 with $\phi(w_1t)$ (recall that $\{\phi(tt_1), \phi(tt_2)\} = \{\phi(z_1s), \phi(z_2r)\}$), and assign colour 6 to uv. It follows that $\{7, \phi(w_1t)\} = \{\phi(ss_1), \phi(ss_2)\}$ ($\phi(w_1t) \neq 6$).

Finally we permute the colours of w_2z_1 and ww_1 and assign 6 to uv. A contradiction.

5.c Suppose w_2 is a light 3-vertex as depicted in Figure 3.11.

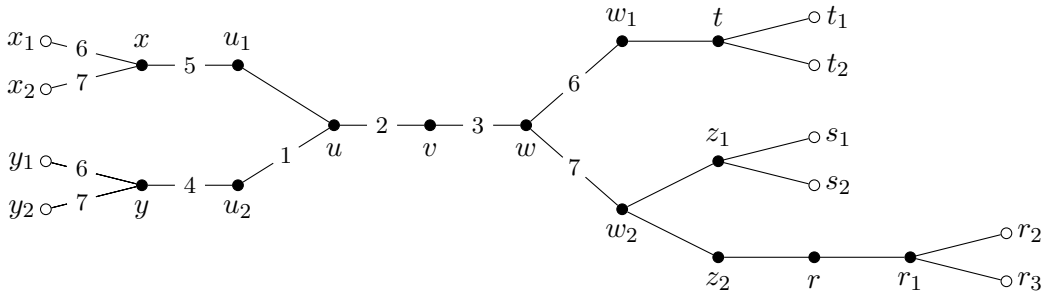


Figure 3.11: The configuration of Claim 3.15(5.c)

Exactly as the first part of the proof of Claim 3.15(5.a) we have:

$$\{\phi(w_1t), \phi(tt_1), \phi(tt_2), \phi(w_2z_1), \phi(w_2z_2)\} = \{1, 2, 3, 4, 5\},\ 3 \in \{\phi(tt_1), \phi(tt_2)\}.$$

Let us uncolour uv and assign to u_1u colour 2. If the permutation of the colours of ww_2 and w_2z_2 is possible, then uv can be recoloured with 7. Hence $\phi(rr_1) = 7$.

Now we uncolour vw and assign colour 3 to uv. Observe that $\{\phi(w_1t), \phi(tt_1), \phi(tt_2)\} = \{\phi(z_1s_1), \phi(z_1s_2), \phi(z_2r)\}$. By contradiction, let us suppose that there exists $\alpha \in \{\{\phi(w_1t), \phi(tt_1), \phi(tt_2)\} \setminus \{\phi(z_1s_1), \phi(z_1s_2), \phi(z_2r)\}$. Recall that $\alpha \in \{1, 2, 3, 4, 5\}$ and $\phi(w_2z_2) \neq 3$. We colour vw with 7, assign colour $\phi(w_2z_2)$ to ww_2, and recolour w_2z_2 with α.

Finally we permute the colours of ww_1 and w_2z_2, assign colour 6 to uv and colour 3 to vw.

□

The discharging rules are defined as follows:

(R1) Every 3_3-vertex gives $\frac{1}{6}$ to each adjacent good 2-vertex.

(R2) Every 3_2-vertex and 3_1-vertex gives $\frac{1}{4}$ to each adjacent good 2-vertex if this 2-vertex is not adjacent to a 3_3-vertex.

(R3) Every 3_0-vertex gives $\frac{1}{12}$ to each adjacent 3_2-vertex if any.

(R4) Every 3_1-vertex u gives $\frac{1}{12}$ to each adjacent 3_2-vertex v if v has a 2-neighbour w adjacent to a 3_3-vertex.

(R5) Every 3-vertex gives $\frac{1}{3}$ to each adjacent 2-vertex which is a neighbour of 3_3-vertex.

(R6) Every 3-vertex gives $\frac{1}{2}$ to each adjacent bad 2-vertex.

Let $v \in V(H)$ be a k-vertex. By Claim 3.15(1), $k \geq 2$.

Case k = 2. Observe that $\omega(v) = -\frac{1}{2}$. Suppose v is a good 2-vertex. If v is adjacent to a 3_3-vertex, then v cannot be adjacent to another 3_3-vertex by Claim 3.15(4). Hence, $\omega^*(v) \geq -\frac{1}{2} + 1 \times \frac{1}{6} + 1 \times \frac{1}{3} = 0$ by (R1) and (R5). If v is not adjacent to a 3_3-vertex, then $\omega^*(v) \geq -\frac{1}{2} + 1 \times \frac{1}{4} + 1 \times \frac{1}{4} = 0$ by (R2). Suppose v is bad. Vertex v is adjacent to one 3-vertex by Claim 3.15(2). Hence, $\omega^*(v) = -\frac{1}{2} + 1 \times \frac{1}{2} = 0$ by (R6).

Case k = 3. Observe that $\omega(v) = \frac{1}{2}$. We have the following cases for v:

- Vertex v is adjacent to three 2-vertices. By Claim 3.15(3) these 2-vertices are good. Moreover, by Claim 3.15(4) none of these 2-vertices is adjacent to another 3_3-vertex. Hence, $\omega^*(v) = \frac{1}{2} - 3 \times \frac{1}{6} = 0$ by (R1).
- Vertex v is adjacent to exactly two 2-vertices. By Claim 3.15(3), none of these 2-vertices is bad. Suppose that none of these 2-vertices are adjacent to a 3_3-vertex. Hence, $\omega^*(v) \geq \frac{1}{2} - 2 \times \frac{1}{4} = 0$ by (R2). Assume now that one of the 2-vertices adjacent to v is adjacent to a 3_3-vertex (note that among the 2-vertices adjacent to v, at most one can be adjacent to a 3_3-vertex by Claim 3.15(5.a)). Hence v cannot be adjacent to a 3_2-vertex by Claim 3.15(5.b). Then, v must have either a 3_1-vertex or a 3_0-vertex as a neighbour. Hence, $\omega^*(v) \geq \frac{1}{2} - 1 \times \frac{1}{4} + 1 \times \frac{1}{12} - 1 \times \frac{1}{3} = 0$ by (R2), (R3) (or (R4)) and (R5).
- Vertex v is adjacent to exactly one 2-vertex u. If u is a bad 2-vertex, then by Claim 3.15(5.c), v cannot be adjacent to a 3_2-vertex w which has a 2-neighbour y adjacent to a 3_3-vertex. Hence, $\omega^*(v) \geq \frac{1}{2} - 1 \times \frac{1}{2} = 0$ by (R6). Suppose u is a good 2-vertex. Let w be the other neighbour of u ($d(w) = 3$). If w is a 3_3-vertex, then $\omega^*(v) \geq \frac{1}{2} - 2 \times \frac{1}{12} - 1 \times \frac{1}{3} = 0$ by (R4) and (R5). If w is not a 3_3-vertex then, $\omega^*(v) \geq \frac{1}{2} - 2 \times \frac{1}{12} - 1 \times \frac{1}{4} > 0$ by (R2) and (R4).
- Vertex v is a 3_0-vertex. Hence, $\omega^*(v) \geq \frac{1}{2} - 3 \times \frac{1}{12} > 0$ by (R3).

This completes the proof.

Proof of (iii) of Theorem 3.9

Let H be a counterexample to part (iii) of Theorem 3.9 minimizing $|E(H)| + |V(H)|$: H is not strongly edge-colourable with eight colours, $\text{mad}(H) < \frac{8}{3}$ and for any edge e, $\chi'_s(H \setminus e) \leq 8$. Recall that $\omega(x) = d(x) - \frac{8}{3}$.

Claim 3.16. The minimal counterexample H to part (iii) of Theorem 3.9 satisfies the following properties:

1. H does not contain 1^--vertices.
2. H does not contain two adjacent 2-vertices.
3. H does not contain a 3-vertex adjacent to three 2-vertices.
4. H does not contain a 2-vertex adjacent to two 3_2-vertices.

Proof.

1. Trivial.
2. Suppose H contains a 2-vertex u adjacent to a 2-vertex v. Let t and w be the other neighbours of u and v respectively. By minimality of H, the graph $H' = H \setminus \{tu, uv, vw\}$ is strongly 8-edge-colourable. Consequently, there exists a strong edge-colouring ϕ of H' with eight colours. Observe that $|[\![8]\!] \setminus SC(N_2(tu))| \geq 2$, $|[\![8]\!] \setminus SC(N_2(uv))| \geq 4$ and $|[\![8]\!] \setminus SC(N_2(vw))| \geq 2$. Therefore, the colouring ϕ can be easily extended to H, which is a contradiction.

3. Suppose H contains a 3-vertex v adjacent to three 2-vertices u, w and t. By minimality of H, there exists a strong edge-colouring ϕ of $H' = H \setminus \{vt, vu, vw\}$ with eight colours. Observe that $|[\![8]\!] \setminus SC(N_2(vt))| \geq 3$, $|[\![8]\!] \setminus SC(N_2(vu))| \geq 3$ and $|[\![8]\!] \setminus SC(N_2(vw))| \geq 3$. Therefore, we can extend ϕ to H, which is a contradiction.

4. Suppose H contains two 3_2-vertices having a 2-vertex as a common neighbour. Hence, there exists a path of five vertices in H, $uvwxy$ such that u, w and y are 2-vertices and v, x are 3_2-vertices. Let v_1 (respectively x_1) be the third neighbour of v (respectively x) other than u and w (respectively w and y). Let us consider $H' = H \setminus \{uv, vw, wx, xy, vv_1, xx_1\}$. Since H is a minimum counterexample, $\chi'_s(H') \leq 8$ and there exists a strong 8-edge-colouring ϕ of H'. We extend this colouring to H. Let us first colour the edges vv_1 and xx_1. Next, we colour uv and xy. Each of these edges has two colours left to use: c^1_{uv}, c^2_{uv} for uv and c^1_{xy}, c^2_{xy} for xy. Suppose, there exists at least one colour in common: $c^1_{uv} = c^1_{xy}$. We choose these colours to colour uv and xy. After the colouring of these edges, vw and wx have each at least two colours left and we can colour them easily. Suppose now that c^1_{uv}, c^2_{uv}, c^1_{xy} and c^2_{xy} are all different. Let us colour uv with c^1_{uv} and xy with c^1_{xy}. Since vw has three colours left to use at the beginning of the process, in the worst case there exists one colour non used, c_{vw}. So, we colour vw with this colour. At the last step we need to colour wx. If it is not possible, then all three colours left to use for this edge at the beginning of the process of extension of ϕ to H, were used by uv, vw and xy. In this case if $c^2_{uv} \neq c_{vw}$, then we change the colour of uv to c^2_{uv}. Otherwise we change the colour of xy to c^2_{xy} (which is possible since c^1_{uv}, c^2_{uv}, c^1_{xy} and c^2_{xy} are all different). Hence, we have a colour left for wx, to complete the colouring of H.

□

Claim 3.17. The minimal counterexample H to part (iii) of Theorem 3.9 does not contain:

1. A 3-vertex adjacent to a 3_2-vertex and to a 2-vertex.

2. A 3-vertex adjacent to two 3_2-vertices.

Before proving Claims 3.17(5) and 3.17(6), we need to introduce some definitions and notations. Let ϕ be a partial strong 8-edge-colouring of H. For an edge uv, we denote by $PC_\phi(uv)$ the set of permissible colours that would extend ϕ to uv. Let $SC(N_1(uv))$ be the set of coloured edges at distance 1 from uv.

Observation 3.18. Suppose H contains a 3_2-vertex x. Let u and r be its adjacent 2-neighbours, and let y be its adjacent 3-neighbour. Also let v and s be the other neighbours (distinct from x) of u and r respectively. Finally let z and t be the other neighbours of y (distinct from x).

Consider ϕ a strong 8-edge-colouring of $H' = (V(H), E(H) \setminus \{xy, xu, uv, xr, rs\})$. Then ϕ satisfies the following:

O1. $PC_\phi(uv) \cap PC_\phi(rs) = \emptyset$. Otherwise, let α be a colour of the intersection. First we colour uv and rs with α, then we colour xy ($|PC_\phi(xy)| \geq 2$; hence it remains at least one colour). Finally we colour ux ($|SC(N_2(ux))| = 7$) and xr ($|SC(N_2(ux))| = 8$, but colour α is repeated twice).

O2. $\{\phi(zy), \phi(yt)\} \cap (PC_\phi(uv) \cup PC_\phi(rs)) = \emptyset$. Otherwise assume that $\phi(zy) \in PC_\phi(uv)$. We colour uv with $\phi(zy)$. Then we colour xy ($|PC_\phi(xy)| \geq 2$; hence it remains at least one colour). We colour sequentially rs (at least one available colour), xr (at least two available colours, since zy and uv have the same colour), and ux (at least one available colour, again zy and uv have the same colour).

O3. $PC_\phi(xy) \subseteq PC_\phi(uv) \cup PC_\phi(rs)$. By contradiction. Observe that $|PC_\phi(uv)| \geq 2$. Let $\alpha, \beta \in PC_\phi(uv)$. Similarly, let $\gamma, \lambda \in PC_\phi(rs)$. Finally let $\zeta \in PC_\phi(xy) \setminus (PC_\phi(uv) \cup PC_\phi(rs))$. Assign ζ to xy, α to uv, β to ux, γ to rs. Finally, by *O1* and *O2* we can assign colour λ to xr.

O4. $SC(N_1(uv)) = SC(N_1(rs))$. By contradiction. Colour first xy, then uv and rs. Count the number of available colours for ux and xr. If one of them has two available colours, then we colour it the last. So each has one available colour. Suppose these two colours are the same. Then we have $SC(N_1(uv)) = SC(N_1(rs))$.

O5. $|PC_\phi(uv)| = 2 = |PC_\phi(rs)|$. By contradiction suppose $|PC_\phi(uv)| \geq 3$ and $\alpha, \beta, \gamma \in PC_\phi(uv)$. Suppose $PC_\phi(xy) \nsubseteq PC_\phi(uv)$. Colour first xy with a colour that does not appear in $PC_\phi(uv)$, then rs. Assign α to uv, β to ux, and γ to xr (possible by *O1* and *O2*). Now suppose that $PC_\phi(xy) \subseteq PC_\phi(uv)$ and $PC_\phi(xy)$ contains α, β. Colour xy with α, ux with β, uv with γ, xr and rs with the colours of $PC_\phi(rs)$ (that is possible by *O1*).

O6. $SC(N_1(uv)) \cap \{\phi(zy), \phi(yt)\}) = \emptyset$. Otherwise colour sequentially xy, uv, rs, xr, ux.

To summarize one can assume without loss of generality that: $PC_\phi(uv) = \{1, 2\}, PC_\phi(rs) = \{3, 4\}, \phi(zy) = 5, \phi(yt) = 6, SC(N_1(uv)) = SC(N_1(rs)) = \{7, 8\}, PC_\phi(xy) \subseteq \{1, 2, 3, 4\}$.

Now, we prove Claim 3.17.

Proof.

1. This follows from the previous discussion in Observation 3.18. By contradiction suppose t is a 2-vertex. Observe that the edge yt is coloured with α and has also an other available colour, say β (at most six other coloured edges at distance at most 2). Now $\beta \notin PC_\phi(uv) \cup PC_\phi(rs)$. Otherwise we permute α and β, and this contradicts *O2*. It suffices then to colour xy with β, the edges uv and ux with the colours of $PC_\phi(uv)$, and the edges rs and xr with the colours of $PC_\phi(rs)$. This extends ϕ to whole H.

2. By contradiction, suppose H contains a 3-vertex y adjacent to two 3_2-vertices x_1 and x_2. Let u_i and r_i be the two 2-neighbours of x_i $(i = 1, 2)$. Finally let v_i and s_i be the two other neighbours of u_i and r_i respectively $(i = 1, 2)$. Consider $H' = (V(H), E(H) \setminus \{yx_1, x_1u_1, u_1v_1, x_1r_1, r_1s_1\})$. By minimality of H, H' admits a strong 8-edge-colouring ϕ. By the previous discussion and without loss of generality one can assume that $PC_\phi(u_1v_1) = \{1, 2\}$, $PC_\phi(r_1s_1) = \{3, 4\}$, $\phi(zy) = 5$, $\phi(yx_2) = 6$, $SC(N_1(u_1v_1)) = SC(N_1(r_1s_1)) = \{7, 8\}$, $PC_\phi(yx_1) \subseteq \{1, 2, 3, 4\}$.

   Hence observe that if we can change the colour of yx_2, then we will be able to extend the colouring (by *O2* or *O6*). To do this uncolour x_2r_2. Recolour yx_2 with an available colour distinct from 6. Colour x_2r_2. We are done.

□

The discharging rule is defined as follows :

(R) Every 3-vertex gives $\frac{1}{3}$ to each adjacent 2-vertex and to each adjacent 3_2-vertex.

Let $v \in V(H)$ be a k-vertex. By Claim 3.16(1), $k \geq 2$.

Case k = 2. Observe that $\omega(v) = -\frac{2}{3}$. By Claim 3.16(2), the neighbours of v have degree 3. Hence v receives twice $\frac{1}{3}$ by (R), and so $\omega^*(v) = -\frac{2}{3} + 2 \times \frac{1}{3} = 0$.

Case k = 3. Observe that $\omega(v) = \frac{1}{3}$. We have the following cases for v:

- If v is not adjacent to any 2-vertices, then v is adjacent to at most one 3_2-vertex by Claim 3.17(5), and so gives at most $\frac{1}{3}$ by (R); it follows $\omega^*(v) \geq \frac{1}{3} - \frac{1}{3} = 0$.
- If v is adjacent to exactly one 2-vertex, then its 3-neighbours are not 3_2-vertices by Claim 3.16(4). It follows that $\omega^*(v) = \frac{1}{3} - \frac{1}{3} = 0$ by (R).
- If v is a 3_2-vertex, then it receives $\frac{1}{3}$ from its 3-neighbour (which is not a 3_2-vertex by Claim 3.16(4)) and gives $\frac{1}{3}$ to each adjacent 2-vertex. Hence $\omega^*(v) = \frac{1}{3} + \frac{1}{3} - 2 \times \frac{1}{3} = 0$.
- The case where v is adjacent to three 2-vertices does not appear by Claim 3.16(3).

This completes the proof.

Proof of (iv) of Theorem 3.9

Let H be a counterexample to part (iv) of Theorem 3.9 minimizing $|E(H)| + |V(H)|$: H is not strongly edge-colourable with nine colours, $\mathrm{mad}(H) < \frac{20}{7}$ and for any edge e, $\chi'_s(H \setminus e) \leq 9$. Recall that $\omega(x) = d(x) - \frac{20}{7}$.

Claim 3.19. The minimal counterexample H to part (iv) of Theorem 3.9 satisfies the following properties:

1. H does not contain 1^--vertices.
2. H does not contain two adjacent 2-vertices.
3. H does not contain a 3-vertex adjacent to two 2-vertices.
4. H does not contain two adjacent 3_1-vertices.
5. H does not contain a triangle.
6. H does not contain a path of three 3-vertices ztu where z and u are 3_1-vertices.

Proof.

1-2. Trivial by counting argument.

3. Suppose H contains a 3-vertex v adjacent to two 2-vertices u and w. Call t the third neighbour of v. By minimality of H, there exists a strong edge-colouring ϕ of $H' = H \setminus \{vt, vu, vw\}$ with nine colours. We show that we can extend this colouring to H. One can observe that $|[\![9]\!] \setminus SC(N_2(vt))| \geq 3$, $|[\![9]\!] \setminus SC(N_2(vu))| \geq 3$ and $|[\![9]\!] \setminus SC(N_2(vw))| \geq 3$. Obviously, we can extend the colouring ϕ to H, which is a contradiction.

4. Suppose H contains two adjacent 3_1-vertices. Let u and v be these 3_1-vertices and x and y respectively, their adjacent 2-vertices.

   (a) Suppose $x = y$. Let z be the third adjacent vertex of u. By minimality of H, there exists a strong edge-colouring ϕ of $H' = H \setminus \{zu, ux, xv, vu\}$. By counting the number of available colours for each of the edges zu, ux, xv, vu, one can easily extend ϕ to H.

   (b) Suppose $x \neq y$. Let t be the 3-vertex adjacent to x. Consider the path $txuvy$. By minimality of H, there exists a strong edge-colouring ϕ of $H' = H \setminus \{tx, xu\}$. We will extend ϕ to H. First, we uncolour the edges uv and vy. The edges tx, vy and uv have each two colours left to use. Moreover, xu have three colours left to use. Suppose we can colour tx with $\alpha \in \{a_1, a_2\}$ and vy with $\beta \in \{a_3, a_4\}$. We distinguish two cases for the available colours of tx and vy:

i. There exists at least one colour in common, say $a_1 = a_3$. We colour tx and vy with a_1 (since these edges are at distance 3, they can have the same colour). Then, we have at least one colour left for uv and we colour this edge with this colour. The edge xu has at least one colour left to use and we choose it. This extends the colouring to H and this is a contradiction.

ii. All four colours are different and without loss of generality $a_1 = 1$, $a_2 = 2$, $a_3 = 3$ and $a_4 = 4$. Let us colour tx and vy with 1 and 3, respectively. We try to colour uv: either we have at least one colour left, say b, and we assign it to uv or $SC(N_2(uv)) \setminus \{1, 3\} = [\![9]\!] \setminus \{1, 3\}$. We distinguish these two cases:

- Suppose there exists a colour b which can be used for uv. If xu has one colour left, ϕ would be extendible to H, which would be a contradiction. Hence, all three possible choices of colours we had initially for xu are $1, 3$ and b. If we can recolour tx with 2, we are done, since we would obtain a new colour for xu. Therefore, we have $b = 2$. But then we can change the colour of vy to 4 and we have again, a colour left for xu to extend ϕ to H. A contradiction.
- Suppose $SC(N_2(uv)) = [\![9]\!]$. We recolour tx with 2 and we colour uv with 1. Now, we try to colour xu: either we are done or this means that all three possible choices we had initially for the colouring of xu are $1, 2$ and 3. In that case, we recolour vy with 4 and we colour xu with 3. This extends the colouring to H - a contradiction.

5. Suppose H contains a triangle xyz.

   If $d(x) = 2$, then by minimality of H, the graph $H \setminus xy$ can be strongly edge-coloured with at most nine colours. Since $|N_2(xy)| \leq 6$, there exists at least three colours left for xy. Hence $d(x) = d(y) = d(z) = 3$.

   Let u, v and t be the neighbours of x, y and z respectively (u, v and t being outside the triangle). Let $H' = H \setminus x$. By minimality of H, we have $\chi'_s(H') \leq 9$. Consider a strong edge-colouring ϕ of H' using the minimum number. We show how to extend it to H. We colour xu and xy (in each case there exists a free colour). If we have a colour left for xz, then we are done. Therefore, $|N_2(xz)| = 9$ and $|SC(N_2(xz))| = 9$, which implies that all edges in $N_2(xz)$ are assigned pairwise distinct colours. Now, one of the following assignment of colours is possible:

   - Assign $\phi(xz) = \phi(yz)$ and recolour yz with a free colour
   - Assign $\phi(xz) = \phi(xy)$ and recolour xy with a free colour

   This is a contradiction.

6. Let ztu be such a path and y and v be the 2-vertices neighbours of z and u respectively. Let x be the neighbour of y distinct from z and w be the neighbour of v distinct from u. By Claims 3.19(1), 3.19(2) and 3.19(3), x and w are 3-vertices. By Claims 3.19(3) and 3.19(4), t is a 3_0-vertex. Let z_1, t_1 and u_1 be the neighbours of z, t and u respectively. Since H has no triangles (by Claim 3.19(5)), we have the configuration depicted in Figure 3.12. Note that in H there might exist edges z_1t_1, t_1u_1 or z_1u_1 and the representation of the given figure is a general one.

   Let $H' = H - y$. By minimality of H, H' can be strongly edge-coloured with at most nine colours. Let us consider such a colouring ϕ. We show how to extend ϕ to H.

   In order to complete the colouring of H one need to assign a colour to xy and yz. By counting the number of edges in $N_2(xy)$, it is easy to see that there is at least one colour left for xy, so we assign it to this edge. Now, if there is a colour left for yz, then we are done.

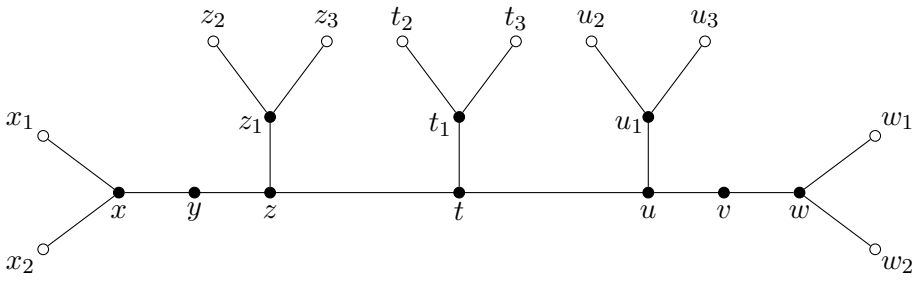


Figure 3.12: The configuration of Claim 3.19(5)

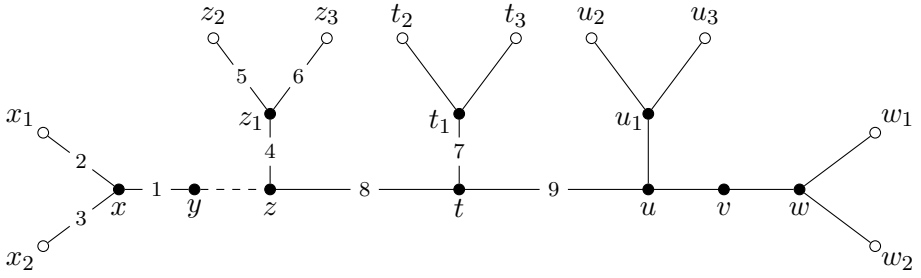


Figure 3.13: The initial fixed colouring of the edges of the configuration of Claim 3.19(5). Edge yz is the only non-coloured edge.

Therefore, since $|N_2(yz)| = 9$, all the colours of $[\![9]\!]$ must appear exactly once in $N_2(yz)$ and without loss of generality we can fix the colours of all edges of $N_2(yz)$ as depicted in Figure 3.13.

Observe that $\{5, 6, 7, 9\} \subseteq SC(N_2(xy))$ as otherwise one could recolour xy with one of these colours and assign $\phi(yz) = 1$. Therefore, all edges incident to x_1 and x_2 for which we did not fix a colour yet, must have distinct colours from the set $\{5, 6, 7, 9\}$. If one could recolour tu with 1, 2, 3, 5 or 6 then yz could be coloured 9 which is impossible. Hence $\{1, 2, 3, 5, 6\} \subseteq SC(N_2(tu))$. Observe that t cannot be neither x_1 nor x_2 as none of the edges incident to t is coloured 2 or 3. Also, since u is adjacent to v which is a 2-vertex, by Claim 3.19(4) u cannot be neither x_1 nor x_2. Therefore, if one could permute the colours of tu and zt, then 8 would not belong to $SC(N_2(xy))$ any more, thus xy could be recoloured with 8 and yz could be assigned colour 1. Therefore, $8 \in \{\phi(u_1u_2), \phi(u_1u_3), \phi(vw)\}$.

Observe that recolouring zz_1 with 2 or 3 must not be possible as otherwise colour 4 could be used for yz to complete the colouring of H. Hence out of all the edges incident to z_2 and z_3, two of them must be coloured 2 and 3 respectively. Let us uncolour edges xy, zz_1, zt and tu. We claim that it is not possible to assign colour 4 to tu. Indeed, if tu could be coloured with 4, then we assign $\phi(tu) = \phi(xy) = 4$ and by using the fact that two out of all the edges incident to z_2 and z_3 must be coloured 2 and 3 respectively, one of the following assignments of colours would be valid:

- $\phi(yz) = 1$, $\phi(zz_1) = 9$ and $\phi(zt) = 8$
- $\phi(yz) = 9$, $\phi(zz_1) = 1$ and $\phi(zt) = 8$
- $\phi(yz) = 1$, $\phi(zz_1) = 8$ and $\phi(zt) = 9$

Therefore, tu cannot be assigned colour 4 and we must have the following statement:

$$\{\phi(t_1t_2), \phi(t_1t_3), \phi(uu_1), \phi(u_1u_2), \phi(u_1u_3), \phi(uv), \phi(vw)\} = \{1, 2, 3, 4, 5, 6, 8\} \qquad (\star)$$

Observe that in $(\star)$ both sets have the same cardinality and hence $\phi(t_1t_2)$, $\phi(t_1t_3)$, $\phi(uu_1)$, $\phi(u_1u_2)$, $\phi(u_1u_3)$, $\phi(uv)$, $\phi(vw)$ are pairwise distinct and this implies that there is no edge between t_1 and u_1.

Consider the edge uv. Since at the beginning of the proof we have fixed $\phi(tt_1) = 7$, $\phi(zt) = 8$ and $\phi(tu) = 9$, obviously $\phi(uv) \in [\![6]\!]$.

We distinguish the following cases for $\phi(uv)$:

(a) Suppose $\phi(uv) \in \{1, 2, 3, 5, 6\}$. We will denote this colour a. By $(\star)$ we know that $\phi(t_1t_2) \neq a$ and $\phi(t_1t_3) \neq a$. We uncolour uv and do the following assignment of colours: $\phi(xy) = 1$, $\phi(yz) = 9$, $\phi(zt) = 8$ and $\phi(tu) = a$. If we manage to colour uv, then we are done. In order to do this, observe that by $(\star)$, $\{9, \phi(t_1t_2), \phi(t_1t_3)\} \cap \{\phi(uu_1), \phi(u_1u_2), \phi(u_1u_3), \phi(vw)\} = \emptyset$. Therefore we could use one of the three colours 9, $\phi(t_1t_2)$ or $\phi(t_1t_3)$ for uv, distinct from the colours assigned to ww_1 and ww_2. A contradiction.

(b) We have $\phi(uv) = 4$. Let us come back to the fixed colouring of Figure 3.13. If one could recolour zt with 2 or 3, then yz could be coloured 8. Therefore, two of the three edges t_1t_2, t_1t_3 and uu_1 must be coloured with 2 and 3 respectively. Without loss of generality we can assume that $\phi(t_1t_2) = 2$. Moreover, by $(\star)$ $\phi(uv) \notin \{\phi(t_1t_2), \phi(t_1t_3)\} = \{2, \phi(t_1t_3)\}$, which means that $\{\phi(ww_1), \phi(ww_2)\} = \{2, \phi(t_1t_3)\}$ and without loss of generality we can fix $\phi(ww_1) = 2$. Now, we uncolour zz_1, zt, uv and assign $\phi(xy) = \phi(tu) = 4$. We obtain a valid partial stong edge-colouring of the configuration as depicted in Figure 3.14.

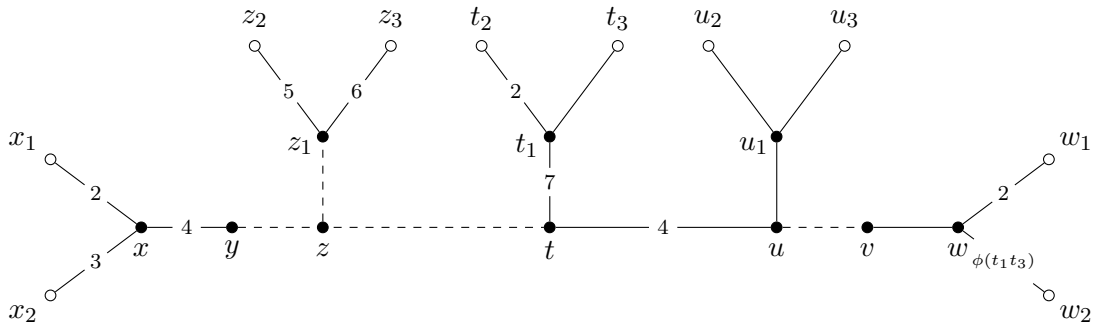


Figure 3.14: Case (b) of the proof of Claim 3.19(5). The dashed edges are not coloured.

Since $\phi(t_1t_3) \neq 9$ and none of the other edges of $N_2(uv)$ is coloured 9, we assign $\phi(uv) = 9$.

We claim that out of all edges incident to z_2 and z_3, two of them must be coloured with 1 and 9. Indeed, if it is not the case then by assigning $\phi(zt) = 8$, we could assign either $\phi(zz_1) = 1$ and $\phi(yz) = 9$ or $\phi(zz_1) = 9$ and $\phi(yz) = 1$ and we would be done. Therefore, four edges incident to z_2 and z_3 except z_1z_2 and z_1z_3 must have distinct colours which are namely 2, 3, 1 and 9. If none of the edges t_1t_3 and uu_1 is coloured 1 then one of the following assignments of colours would be a valid strong edge-colouring:

- $\phi(yz) = 1$, $\phi(zt) = 8$ and $\phi(zz_1) = 9$
- $\phi(yz) = 9$, $\phi(zt) = 8$ and $\phi(zz_1) = 1$
- $\phi(yz) = 9$, $\phi(zt) = 1$ and $\phi(zz_1) = 8$

Therefore, one of the edges t_1t_3 and uu_1 must be coloured 1. On the other hand, as proved previously, one of these edges must be coloured 3 and hence $\{\phi(t_1t_3), \phi(uu_1)\} = \{1, 3\}$. Recall from the previous paragraph that in $N_2(zz_1)$, none of the edges is coloured 7. We recolour zz_1 with 7 and uncolour edges tt_1 and tu. Observe that

by ($\star$) $7 \notin \{\phi(uu_1), \phi(u_1u_2), \phi(u_1u_3), \phi(vw), \phi(ww_1), \phi(ww_2)\}$, so we recolour uv with 7. Moreover, we assign colour 1 to yz and obtain the partial strong edge-colouring of the configuration as depicted in Figure 3.15.

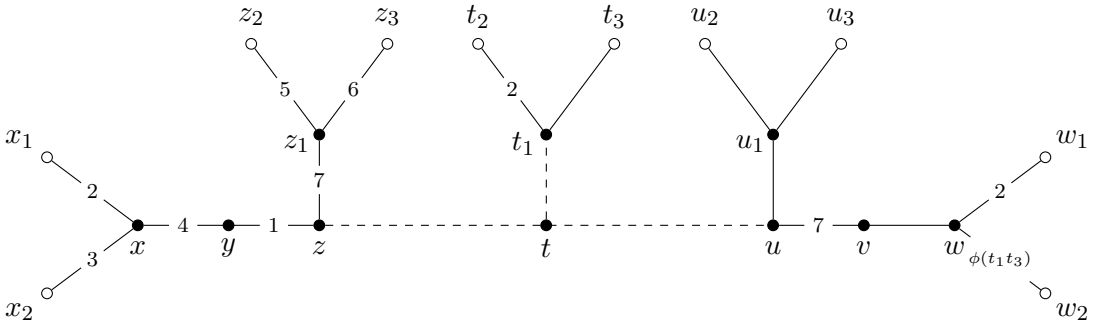


Figure 3.15: A partial strong edge-colouring of the configuration of Claim 3.19(5). The dashed edges are not coloured.

In order to finish the colouring of H we must assign colours to zt, tt_1 and tu. We know that $\phi(t_1t_3) \in \{1, 3\}$ and $\phi(t_1t_2) = 2$. Let us consider temporarily the colouring given in Figure 3.13. If one could recolour edge tt_1 with 5 or 6, then colour 7 could be assigned to yz which implies that out of all the edges incident to t_2 and t_3, two of them must be coloured 5 and 6 respectively. Applying these to the colouring given in Figure 3.15, we conclude that one of the following assignments of colours is valid:

- $\phi(zt) = 8$, $\phi(tt_1) = 4$ and $\phi(tu) = 9$
- $\phi(zt) = 8$, $\phi(tt_1) = 9$ and $\phi(tu) = 4$
- $\phi(zt) = 9$, $\phi(tt_1) = 8$ and $\phi(tu) = 4$

This is a contradiction.

□

We carry out the discharging procedure as follows:

(R) Every 3-vertex gives $\frac{1}{7}$ to each 2-vertex at distance at most 2 from itself.

Let $v \in V(H)$ be a k-vertex. By Claim 3.19(1), $k \geq 2$.

Case k = 2. Observe that $\omega(v) = -\frac{6}{7}$. By Claims 3.19(2), 3.19(3), 3.19(4), 3.19(6), v has six 3-vertices at distance at most two. Hence, $\omega^*(v) = -\frac{6}{7} + 6 \times \frac{1}{7} = 0$.

Case k = 3. Observe that $\omega(v) = \frac{1}{7}$. By Claims 3.19(3), 3.19(4) and 3.19(6), v has at most one 2-vertex at distance at most two. Hence $\omega^*(v) \geq \frac{1}{7} - \frac{1}{7} = 0$.

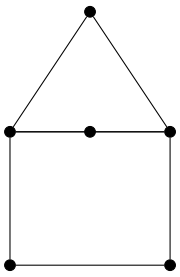
This completes the proof.

3.1.2 Optimality of the bounds on the mad

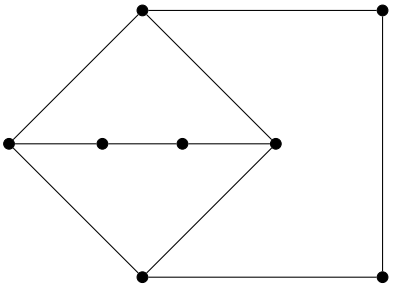
In order to emphasize the relevance of the upper bounds proved in Theorem 3.9 we illustrate the graphs of Figure 3.16. Observe that the graph of Figure 3.16c is the graph of Figure 3.2 for $\Delta = 3$.

Let $f(n) = inf\{\text{mad}(G) \mid \chi'_s(G) > n\}$. By Theorem 3.9, we have the following lower bounds for this function:

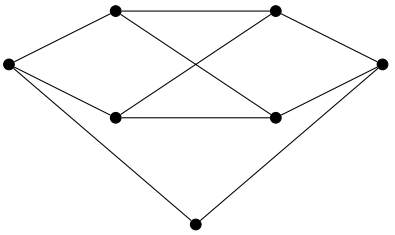
$$\frac{7}{3} \leq f(6),\ \frac{5}{2} \leq f(7),\ \frac{8}{3} \leq f(8) \text{ and } \frac{20}{7} \leq f(9)$$



(a) A graph G with $\text{mad}(G) = \frac{7}{3}$ and $\chi'_s(G) > 6$



(b) A graph G with $\text{mad}(G) = \frac{5}{2}$ and $\chi'_s(G) > 7$



(c) A graph G with $\text{mad}(G) = \frac{20}{7}$ having $\chi'_s(G) > 9$

Figure 3.16: Graphs proving the optimality of the bounds of parts 1, 2 and 4 of Theorem 3.9

On the other hand, the graphs of Figure 3.16 satisfy respectively $\text{mad}(G) = \frac{7}{3}, \frac{5}{2}, \frac{20}{7}$ and $\chi'_s(G) > 6, 7, 9$. It follows:

$$f(6) \leq \frac{7}{3},\ f(7) \leq \frac{5}{2} \text{ and } f(9) \leq \frac{20}{7}$$

and so parts (i), (ii) and (iv) of Theorem 3.9 are optimal.

The case of $f(8)$ remains open. The problem of finding this value is even more challenging since until now we do not have an example of subcubic graph G having $\chi'_s(G) = 9$ and $\text{mad}(G) < \frac{20}{7}$.

3.1.3 Subcubic planar graphs

In this section we prove that Conjecture 3.7 holds for a large class of subcubic planar graphs.

Theorem 3.20. Let G be a planar subcubic graph with no induced k-cycles with $k \in \{4, 5\}$. Then $\chi'_s(G) \leq 9$.

Proof. The proof is done by contradiction. Suppose the statement is not true and let H be a counterexample minimizing $|V(H)| + |E(H)|$. Let us prove some structural properties of H.

First, observe that Claim 3.19 of Theorem 3.9 holds in the case of planar subcubic graphs, as no planarity argument is used in the proof of this claim.

Claim 3.21. H has no 6-cycle $C = xyztuvx$ where y is a 2-vertex.

Proof. Suppose there exists such a cycle C as depicted in Figure 3.17. Observe that x, z, t, u, v, x_1, z_1, t_1, v_1 are 3-vertices by Claims 3.19(2), 3.19(3), 3.19(4) and u_1 is a 3-vertex by Claim 3.19(6).

Consider the graph $H' = H - y$. Consider a strong edge-colouring ϕ of H' using at most nine colours. We will extend ϕ to H in order to obtain a contradiction. Observe that $|SC(N_2(xy))| \leq 8$, thus there exists a colour left for xy. If we can colour yz, then we are done. Therefore,

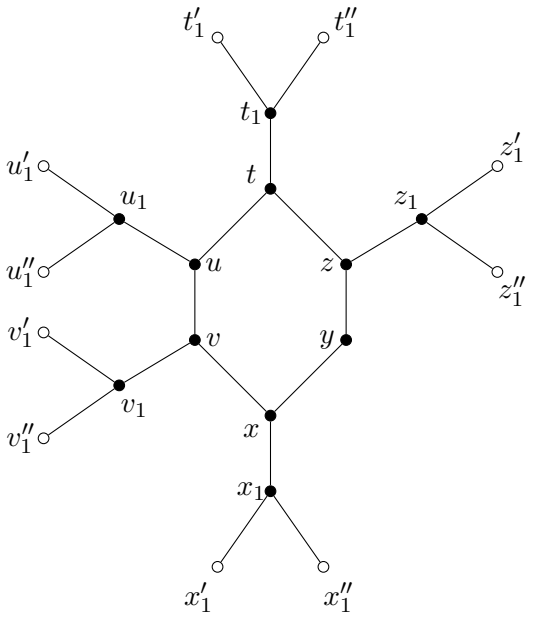


Figure 3.17: An induced cycle C of length 6 of H having a 2-vertex on its boundary

since $|N_2(yz)| = 9$, we must have $SC(N_2(yz)) = [\![9]\!]$ and every colour is used exactly once in $N_2(yz)$. Therefore, we claim that $|SC(N_2[xy])| = 9$ as otherwise one could recolour xy with another colour and obtain a free colour for yz. Without loss of generality we can assume that $\phi(zt) = 1$, $\phi(zz_1) = 2$, $\phi(xx_1) = 3$, $\phi(vx) = 4$, $\phi(uv) = 5$, $\phi(vv_1) = 6$, $\phi(x_1x'_1) = 7$, $\phi(x_1x''_1) = 8$ and $\phi(xy) = 9$. Since $SC(N_2(yz)) = [\![9]\!]$ we have $\{\phi(tu), \phi(tt_1), \phi(z_1z'_1), \phi(z_1z''_1)\} = \{5, 6, 7, 8\}$. Observe that since $5 \in SC(N_2(tu))$ and $5 \in SC(N_2(tt_1))$, without loss of generality we can assume that $\phi(z_1z'_1) = 5$. Also, $6 \in SC(N_2(tu))$ and therefore $\phi(tu) \in \{7, 8\}$. Since colours 7 and 8 are fixed only on edges $x_1x'_1$ and $x_1x''_1$ respectively, we can assume without loss of generality that $\phi(tu) = 7$ and therefore $\{\phi(tt_1), \phi(z_1z''_1)\} = \{6, 8\}$. Figure 3.18 resumes the unique colouring (up to permutation) of the edges described previously.

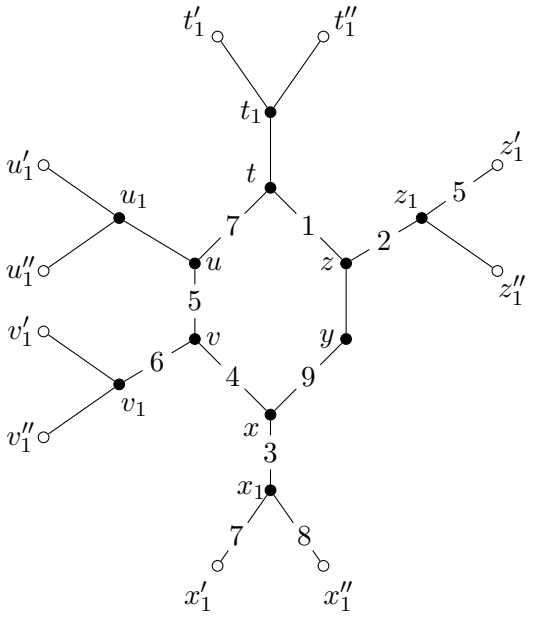


Figure 3.18: The unique colouring of $C - yz$ in H'

We claim that one of the edges $v_1v'_1$ or $v_1v''_1$, say $v_1v'_1$, must have the same colour as the edge

zt (colour 1 in Figure 3.18). Otherwise, one could change the colour of vx to the colour of zt and colour yz with 4. Similarly, $2 \in \{\phi(v_1v_1''), \phi(uu_1)\}$ (we can assign 2 to vx and 4 to yz). Observe that one can use the same argument conversely (by trying to assign to tz the colour of vx) by recalling from the previous paragraph that $\{\phi(tt_1), \phi(z_1z_1'')\} = \{6, 8\}$. Hence, we conclude that one of the edges t_1t_1' or t_1t_1'', say t_1t_1', must have the same colour as the edge vx (colour 4 in Figure 3.18). If it is possible to permute the colours of edges uv and vx, one could obtain a free colour (colour 4) for yz, thus either uu_1' or uu_1'' must have the same colour as vx (colour 4 in Figure 3.18). Without loss of generality $\phi(u_1u_1') = 4$. If it is possible to permute the colours of edges tu and uv (7 and 5 respectively), then one could obtain a free colour for yz. Hence either $\phi(v_1v_1'') = 7$ or $\phi(t_1t_1'') = 5$ (or both).

1. Suppose $\phi(v_1v_1'') = 7$. Hence $\phi(uu_1) = 2$. If one can permute the colours of tu and zt, such that tu is assigned colour 1 and zt is assigned colour 7, then xy could be recoloured with 1 and colour 9 would be free for yz. Hence $\phi(u_1u_1'') = 1$. But now it is possible to permute the colours of xy and uv and to use colour 9 for yz. A contradiction.

2. Suppose $\phi(t_1t_1'') = 5$. If it is possible to change the colour of edge zt (which is 1) to the colour of the edge xx_1 (which is 3), then yz could be coloured with 1. Hence $\phi(uu_1) = 3$ and therefore, $\phi(v_1v_1'') = 2$. By permuting the colours of edges vx and xy (4 and 9 respectively) and by recolouring zt with 9, we can colour yz with 1. A contradiction.

□

By Claim 3.19(5), H has no triangle and by hypothesis of the theorem H contains no induced cycle of length 4 nor cycle of length 5. Hence, the counterexample H must have girth $g \geq 6$.

Consider now the graph H_1 obtained from H by replacing each path of two edges xyz, where y is a 2-vertex and x, z are 3-vertices, by an edge xz. Clearly, H_1 is planar. By Claim 3.19(5), H has no triangle and since it does not contain an induced 4-cycle, H_1 is simple. Moreover, since H has no 1^--vertices (Claim 3.19(1)) and no two adjacent 2-vertices (Claim 3.19(2)), H_1 is 3-regular. Therefore, H_1 must contain a face of length at most 5, say C' (this can be easily seen using Euler's formula). Recall that H has girth at least 6, thus by Claims 3.19(2), 3.19(3) and 3.19(4), C' cannot be obtained from a cycle of H of length $l \geq 7$. Therefore, in H there exists a cycle C of length 6 having a vertex of degree 2 on its boundary. But this is impossible by Claim 3.21. Hence H cannot exist. □

3.2 Outerplanar graphs

Theorem 3.22. For every outerplanar graph G with maximum degree $\Delta \geq 3$, $\chi_s'(G) \leq 3\Delta - 3$.

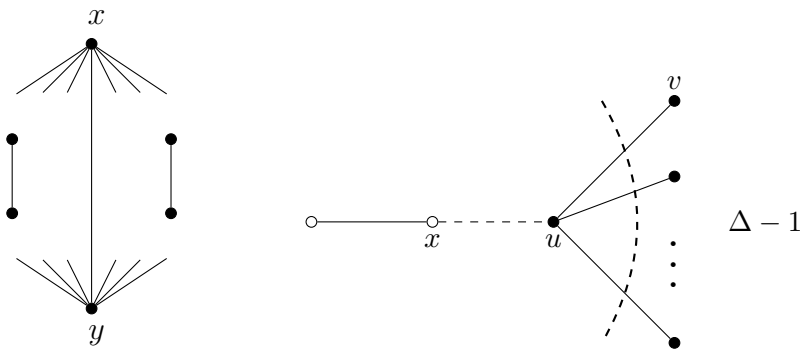
Proof. In this proof, the term *pendant edge* is an edge incident to a vertex of degree 1. We define the partial order $\preceq$ on graphs such that $G_1 \prec G_2$ if and only if

- $|E(G_1)| < |E(G_2)|$ or
- $|E(G_1)| = |E(G_2)|$ and G_1 contains strictly more pendant edges than G_2.

Let $k \geq 3$ be an integer and G be an outerplanar graph with maximum degree k such that $\chi_s'(G) > 3k - 3$ and that is minimal with respect to $\preceq$.

We first show that G does not contain Configuration 1 depicted in Figure 3.19a. That is, two adjacent vertices x and y, such that the graph G' obtained from G by removing the set S of edges incident to x or y ($G' = G - \{x, y\}$) contains two edges in distinct connected components.

Suppose that G contains Configuration 1. Let $G_1, \cdots, G_k$ be the connected components of G'. Since they contain fewer edges than G, by minimality of G with respect to $\preceq$, the graphs induced by the edges of $G_i \cup S$ admit a strong edge-colouring with at most $3k - 3$ colours. Since



(a) Configuration 1 (b) Configuration 2

Figure 3.19: Forbidden configurations

the colours of the edges of S are distinct, we can permute the colours in the colouring of $G_i \cup S$ so that the colouring of S is the same in every $G_i \cup S$. By gluing up the graphs $G_i \cup S$, we obtain a valid strong edge-colouring of G since the distance between an edge in G_i and an edge in $G_{i'}$ for $i \neq i'$ is at least 3. This is a contradiction.

Configuration 2 depicted in Figure 3.19b consists of a vertex u adjacent to at most one vertex x with degree at least 2 and to at least one vertex v of degree 1. The graph G does not contain Configuration 2 since otherwise we could obtain a colouring of G by extending a colouring of $G \setminus \{uv\}$.

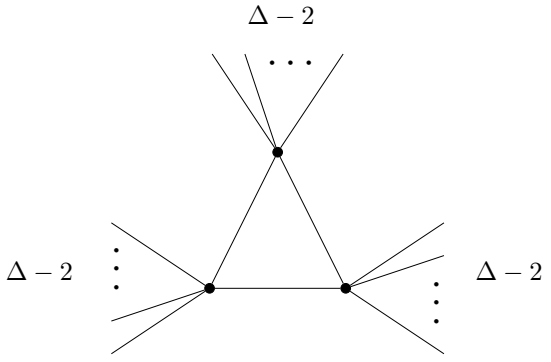


Figure 3.20: Example of outerplanar graph such that $\chi'_s(G) = 3\Delta - 3$.

Let G' be the graph induced by the vertices of G of degree at least 2. Since Configuration 2 is forbidden in G, G' has minimum degree 2.

We claim that G' is 2-connected. Suppose the contrary and let v be a vertex of G' such that $G' - v$ is disconnected. Let $G'_1, \ldots, G'_k$ be the connected components of $G' - v$. Observe that each of the graphs $G'_i \cup N[v]$ with $i \in \{1, \ldots, k\}$ is smaller than G with respect to $\preceq$ and thus for each of them there exists a strong edge-colouring ϕ_i using at most $3k - 3$ colours. One can permute the colours of the edges incident to v for every ϕ_i so that the colouring of the edges of $N[v]$ is the same in every $G'_i \cup N[v]$. The colourings $\phi_1, \ldots, \phi_k$ provide a valid strong edge-colouring of G' and this is a contradiction.

Let C be the cycle of the outer-face. Again, since Configuration 1 is forbidden in G, the chords of C join vertices at distance 2 in the cyclic order. It is easy to check that if C contains at most four vertices, then the theorem holds. So C contains n vertices ($n \geq 5$) $v_1, \ldots, v_n$ in cyclic order and G is the graph induced by the vertices of the cycle C which can be adjacent to some vertices of degree 1.

Let us suppose that C contains a chord, say the edge v_1v_3. Notice that since Configuration 1 is forbidden, v_2 is only adjacent to v_1 and v_3. The graph G' is obtained from G by splitting the

vertex of degree 2 v_2 into v_2', which is only adjacent to v_1, and v_2'', which is only adjacent to v_3. Notice that G' and G have the same number of edges but G' has two more pendant edges than G, so $G' \prec G$. The graph G' thus admits a valid strong edge-colouring using $3k-3$ colours and this colouring remains valid if we identify v_2' and v_2'' to form G. This shows that vertices of degree at least 2 in G form a chordless cycle.

To finish the proof, we have to consider only the worst case of graphs of this form, where every vertex on the chordless cycle is incident to $\Delta(G)-2$ pendant edges. It is easy to check that if $\Delta(G) = k = 3$, then we can colour G using at most $3k-3 = 6$ colours. We iteratively construct a suitable colouring for larger values of k: when k is incremented by 1, there is at most one new pendant edge for each vertex on the cycle and three more available colours. We use the three new colours to colour the new edges such that two new edges incident to adjacent vertices get distinct colours. □

3.3 Complexity

A natural question in the study of any colouring problem is what could be said about its complexity. In this section we study the complexity of the problem in the class of planar graphs. The STRONG k-EDGE-COLOURING problem is defined as follows:

INSTANCE: A graph G.

QUESTION: Does G have a strong edge-colouring with k colours?

The 3-COLOURING problem is defined as follows:

INSTANCE: A graph G.

QUESTION: Does G have a proper vertex colouring with three colours?

3-COLOURING was proved to be NP-complete even when restricted to planar graphs of maximum degree 4 [51].

Theorem 3.23. STRONG 4-EDGE-COLOURING is NP-complete for planar bipartite graphs with maximum degree 3 and with an arbitrarily large girth.

Proof. The problem is clearly in NP since it can be checked in polynomial time whether a given assignment of colours to edges is a strong edge-colouring. We will prove the theorem by reduction from 3-COLOURING of planar graphs of maximum degree 4.

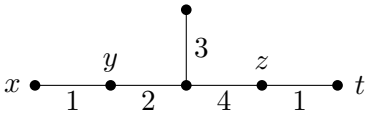
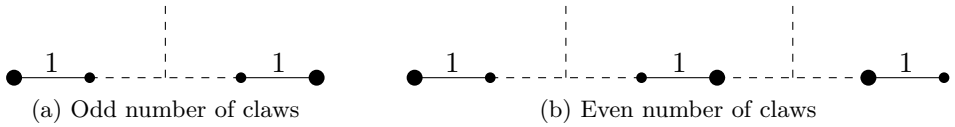


Figure 3.21: Forcing a colour



(a) Odd number of claws (b) Even number of claws

Figure 3.22: Transportation of a colour

First let us observe that in a strong 4-edge-colouring of the graph of Figure 3.21, the edges xy, zt must receive the same colours. By gluing several copies of this graph as shown in Figures 3.22a, 3.22b, we can increase the distance between the edges that must be coloured the same. Moreover, by choosing an odd or an even number of copies of the graph of Figure 3.21, we

may force the two end vertices of the constructed graph to be in a same or in different parts of a bipartition of the graph. In Figures 3.22a and 3.22b, the bipartitions are given by small and big vertices.

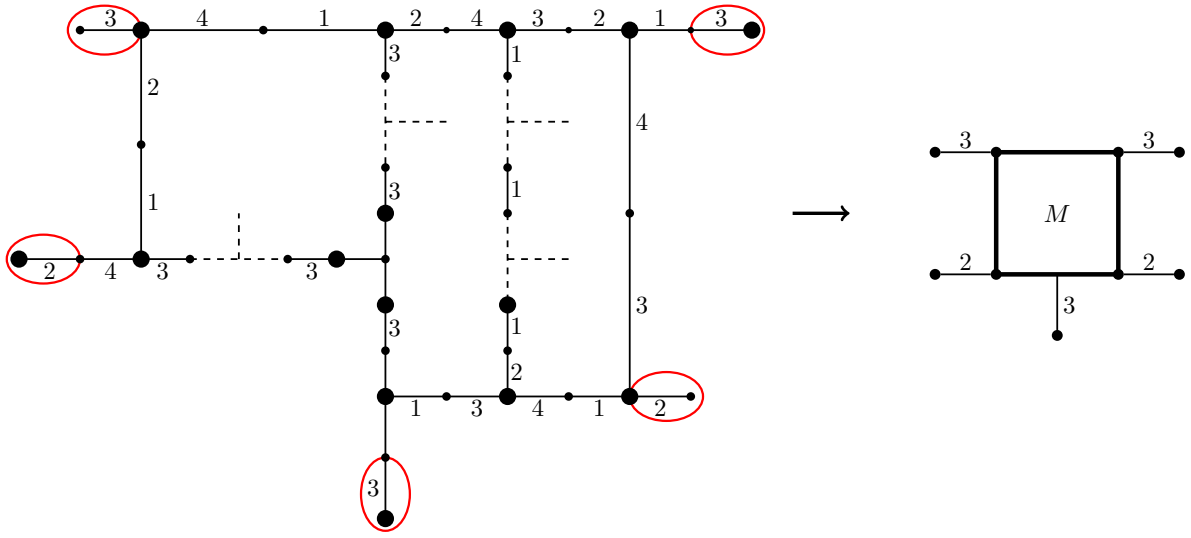
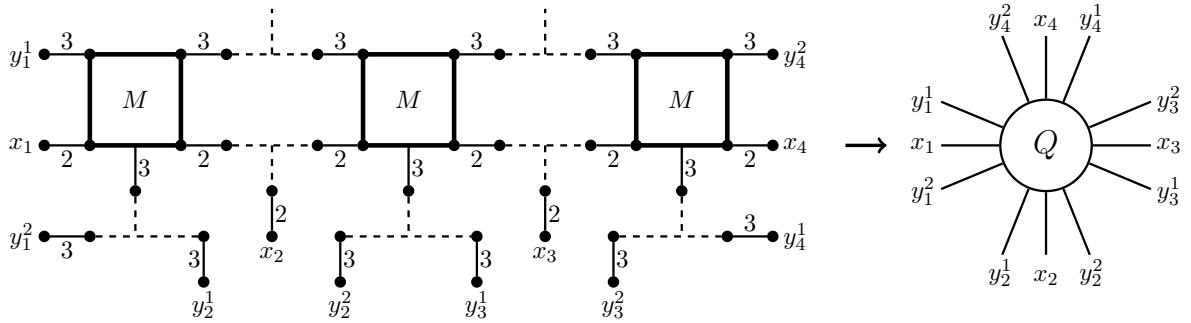
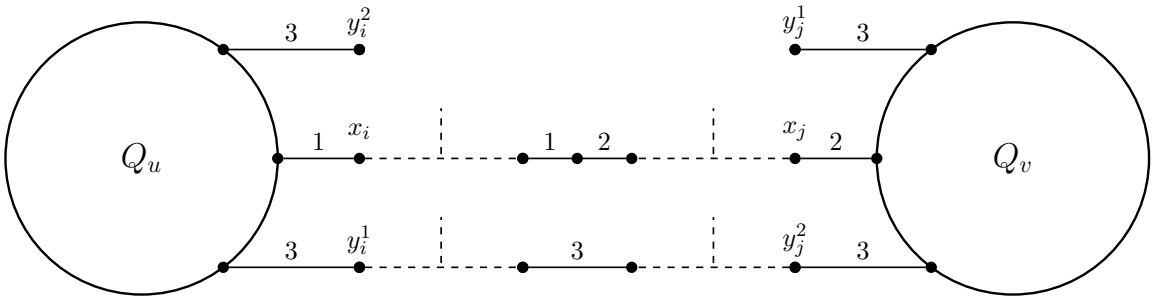


Figure 3.23: Sub-gadget M

Now, we are ready to build the generic sub-gadget M (Figure 3.23) which will be used in our reduction. It can be checked by case analysis that up to permutation of colours, the strong 4-edge-colouring of M, given in Figure 3.23, is unique. Also M is bipartite (the bipartition is given in the picture by big and small vertices) and of arbitrarily large girth.



(a) Vertex gadget Q



(b) Connecting two vertex gadgets in G'

Figure 3.24: Vertex and edge gadgets

Given a planar graph G with maximum degree 4, we construct a graph G' as follows. Every vertex v of G is replaced by a copy Q_v of the graph Q depicted in Figure 3.24a which contains three copies of the sub-gadget M. Notice that since M is bipartite and of arbitrarily large girth,

so is Q.

For every edge uv in G, we choose an index i for Q_u and j for Q_v and join x_i of Q_u with x_j of Q_v and one of the vertices y_i^1, y_i^2 with one of the vertices y_j^1, y_j^2. These connections are done using an arbitrarily large number of claws as depicted in Figure 3.24b. It is easy to observe that we can make the choice of connections such that the obtained graph is planar. Furthermore, by construction the obtained graph G' is bipartite and of arbitrarily large girth.

We say that the *colour of* Q is the colour of the edges incident to the vertices x_i in Q (colour 2 in Figure 3.24a). Also, the *forbidden colour of* Q is the colour of the edges incident to y_i^1 and y_i^2 (colour 3 in Figure 3.24a).

Figure 3.24b shows that for every edge $uv \in G$, Q_u and Q_v are assigned distinct colours and the same forbidden colour. Since G is connected, all copies of Q have same forbidden colour, say 3, and thus no copy of Q is coloured 3.

If G is 3-colourable, then for every vertex $v \in G$, we can assign the colour of v to Q_v and extend this to a strong 4-edge-colouring of G'. Conversely, given a strong 4-edge-colouring of G', we obtain a 3-colouring of G by assigning the colour of Q_v to the vertex v. So G' is strongly 4-edge-colourable if and only if G is 3-colourable and this completes the proof.

□

Theorem 3.24. STRONG 5-EDGE-COLOURING is NP-complete for planar bipartite graphs with maximum degree 3 and girth 8, and for planar graphs with maximum degree 3 and girth 9.

Proof. In the following we will give the proof for the case of girth 8 since the same argument applies for the case of girth 9.

The problem is clearly in NP since it can be checked in polynomial time whether a given edge-colouring is a strong edge-colouring. As in the case of Theorem 3.23, we will reduce 3-COLOURING of planar graphs of degree 4 to STRONG 5-EDGE-COLOURING.

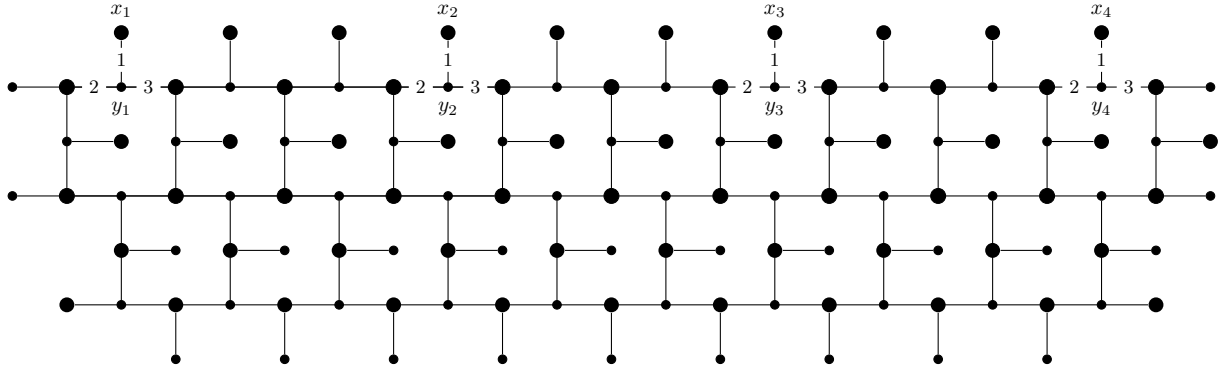


Figure 3.25: Vertex gadget for the case of girth 8 of Theorem 3.24

Given a planar graph G with maximum degree 4, we construct a graph G' as follows. Every vertex v in G is replaced by a copy Q_v of the vertex gadget Q depicted in Figure 3.25. For every edge uv in G, we identify a vertex x_i of Q_u with a vertex x_j of Q_v and add a vertex of degree 1 adjacent to the common vertex of Q_u and Q_v, as depicted in Figure 3.26. We identify vertices in such a way that the obtained graph G' is planar. Observe that Q is bipartite and the bipartition is given by small and big vertices in Figure 3.25. Hence, it is easy to see that G' is bipartite too.

Moreover, G' has no cycle of size strictly less than eight, hence G' has girth 8.

We will show that up to permutation of colours, the strong 5-edge-colouring of Q given in Figure 3.25 is unique. To do this, first observe that the strong 5-edge-precolouring of the subgraph of Q depicted in Figure 3.27a, cannot be extended to the whole subgraph without using a sixth colour. Therefore if it is possible to give a strong 5-edge-colouring of Q, the only way to do it is with the strong 5-edge-colouring of the subgraph of Q as depicted in Figure 3.27b.

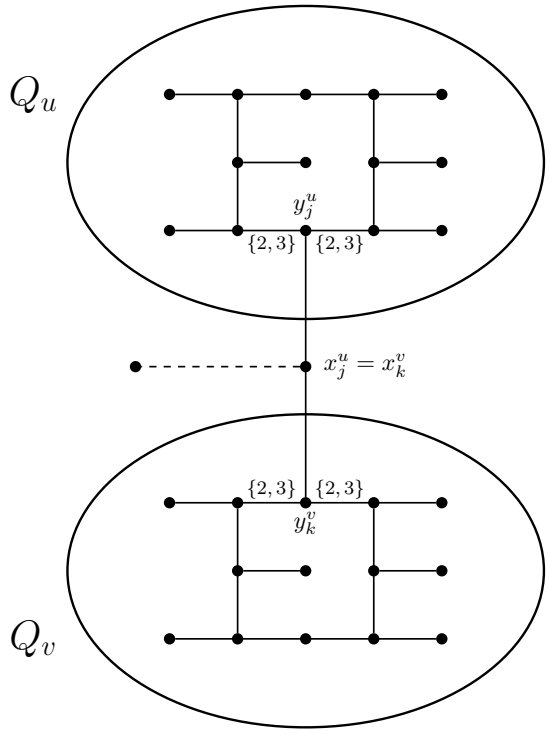
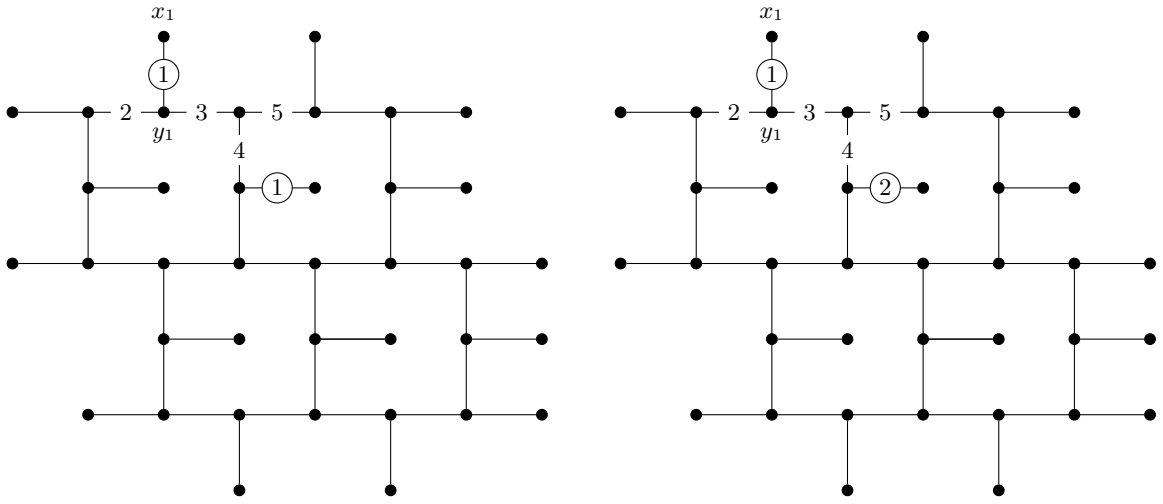


Figure 3.26: Edge gadget for the case of girth 8 of Theorem 3.24



(a) Non valid strong 5-edge-precolouring

(b) Valid strong 5-edge-precolouring

Figure 3.27: Two possible precolourings of a subgraph of Q

Using this observation it is easy to prove that up to permutation of colours the strong 5-edge-colouring of Q is unique.

We say that the *colour of* Q is the colour of the edges x_iy_i in Q (colour 1 in Figure 3.25). Also, the *forbidden colours of* Q are the colours of the edges incident to y_j in Q, different from x_iy_i (colours 2 and 3 in Figure 3.25). Figure 3.26 shows that for every edge $uv \in G$, Q_u and Q_v have distinct colours and same forbidden colours. Since G is connected, all copies of Q have same forbidden colours, 2 and 3, and thus no copy of Q is coloured 2 or 3.

If G is 3-colourable, then for every vertex $v \in G$, we can assign the colour of v to Q_v and

extend this to a strong 5-edge-colouring of G'. Conversely, given a strong 5-edge-colouring of G', we obtain a 3-colouring of G by assigning the colour of Q_v to the vertex v. So G' is strongly 5-edge-colourable if and only if G is 3-colourable, which completes the proof.

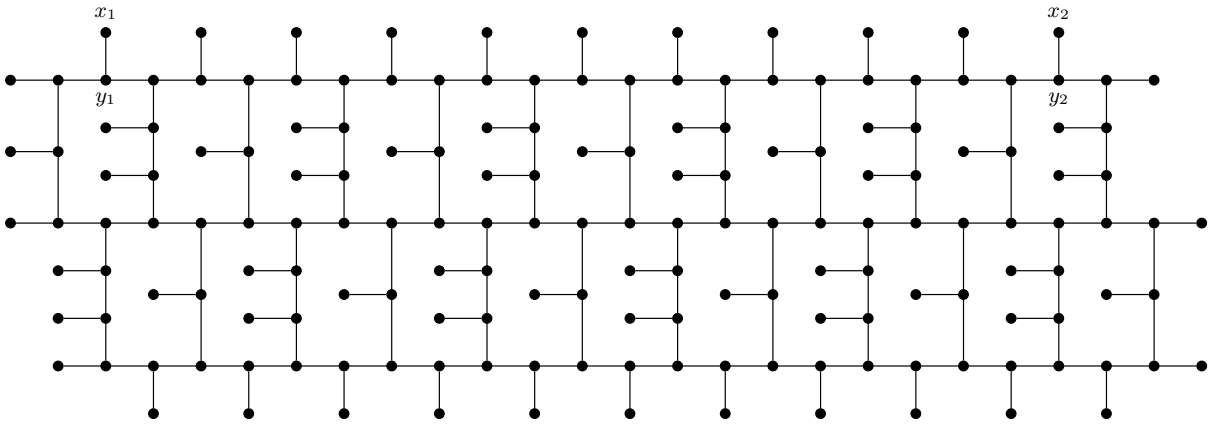


Figure 3.28: Vertex gadget for the case of girth 9 of Theorem 3.24

For the case of girth 9 the same argument applies by using the vertex gadget Q obtained by gluing three copies of the graph of Figure 3.28, in order to obtain four pairs of vertices (x_k, y_k) $(k \in \{1, \ldots, 4\})$. The edge gadget is the same as in the case of girth 8. Checking the strong edge-colouring of Q is much more tedious than in the case of the vertex gadget of girth 8. We will give the idea of the proof that up to permutation of colours, the strong 5-edge-colouring of Q is unique. To do this first observe that the strong 5-edge-precolourings of the subgraphs of Q as depicted in Figures 3.29a, 3.29b, 3.29c cannot be extended to the whole subgraph without using a sixth colour. Therefore if it is possible to give a strong 5-edge-colouring of this graph, the only way to do it is when the subgraph of Q has the colours assigned as in Figure 3.30. Moreover, the same configuration of colours is repeated as shown in the figure, thus it can be extended up to the whole graph Q.

□

An interesting fact about Theorem 3.24 is that for subcubic planar graphs with girth $g \geq 31$, $\chi'_s(G) \leq 5$ [19] and thus there exists a trivial polynomial-time algorithm for this class of graphs. Before the result of [19] was proved, we were trying to increase the value of the girth as much as possible and see whether the problem still was remaining NP-complete. Although, there is a gap between the values 9 and 31 for the girth, it explains partially why it is not easy to increase the size of the girth and prove that the problem remains NP-complete.

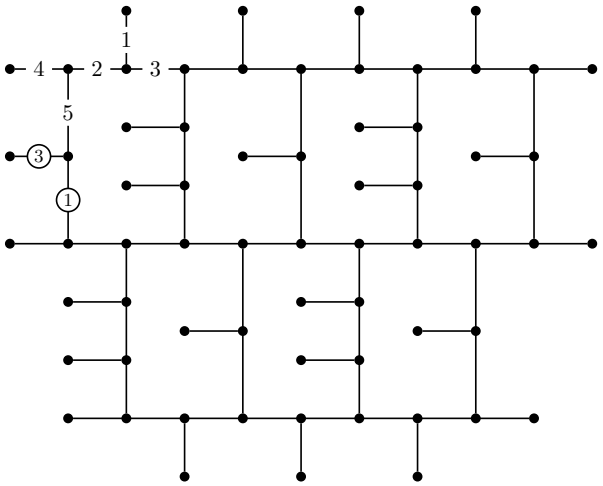
Theorem 3.25. STRONG 6-EDGE-COLOURING is NP-complete for planar bipartite graphs with maximum degree 3.

Proof. The problem is clearly in NP since it can be checked in polynomial time whether a given edge-colouring is a strong edge-colouring. Again, we prove the theorem by reduction from 3-COLOURING of planar graphs with maximum degree 4. For a graph G of instance of 3-COLOURING of planar graphs with maximum degree 4 we will give a construction of a graph G' such that G is 3-colourable if and only if G' is strongly 6-edge-colourable.

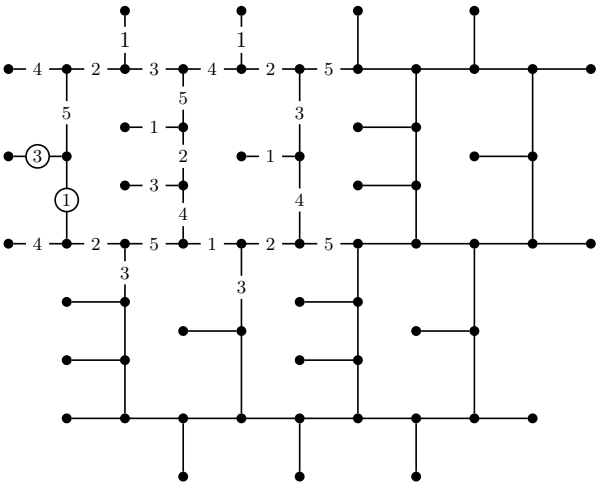
We first want to point out two easy but very useful observations.

Observation 3.26. In any strong edge-colouring with six colours of the graphs of Figure 3.31a, the colours of edges at distance 3 have to be the same (colours 1 and 2 are forced).

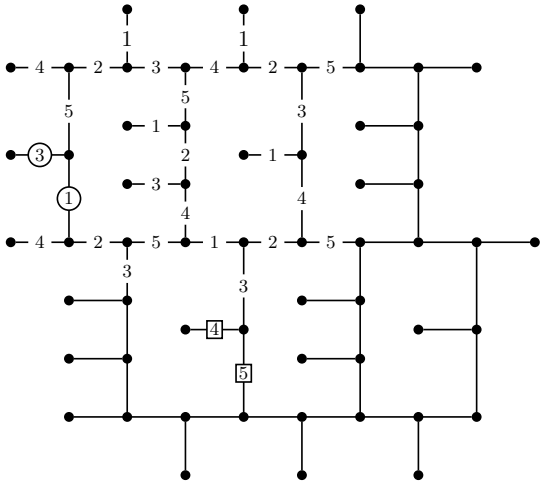
Observation 3.27. For any strongly 6-edge-colourable subcubic graph with an embedding such that two edges coloured distinctly cross each other in the plane, there exists a strongly 6-edge-colourable subcubic planar graph obtained by replacing this crossing as depicted in Figure 3.31b.

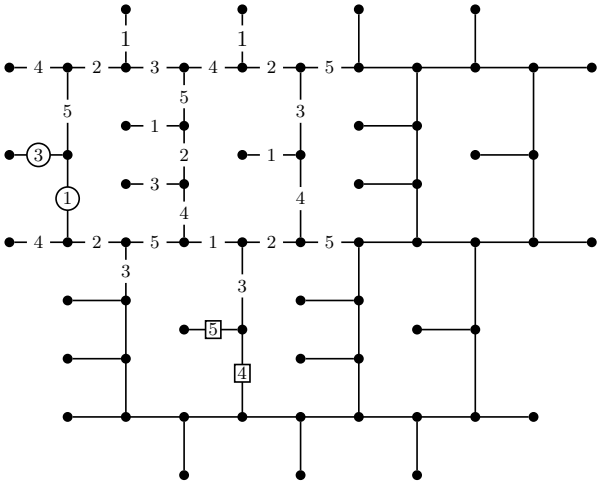


(a) First choice of circled colours

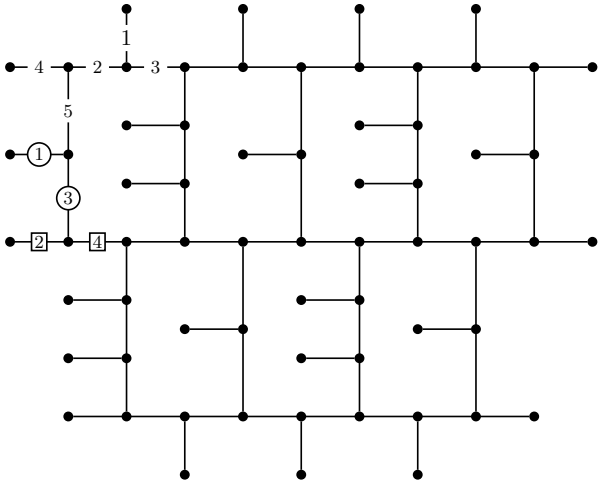


(b) The obtained forced strong 5-edge-precolouring





(c) For any choice of colours in squares, the given colouring cannot be extended to the given subgraphs



(d) The last non-valid precolouring

Figure 3.29: Non-valid strong 5-edge-precolourings of the graph of Figure 3.28

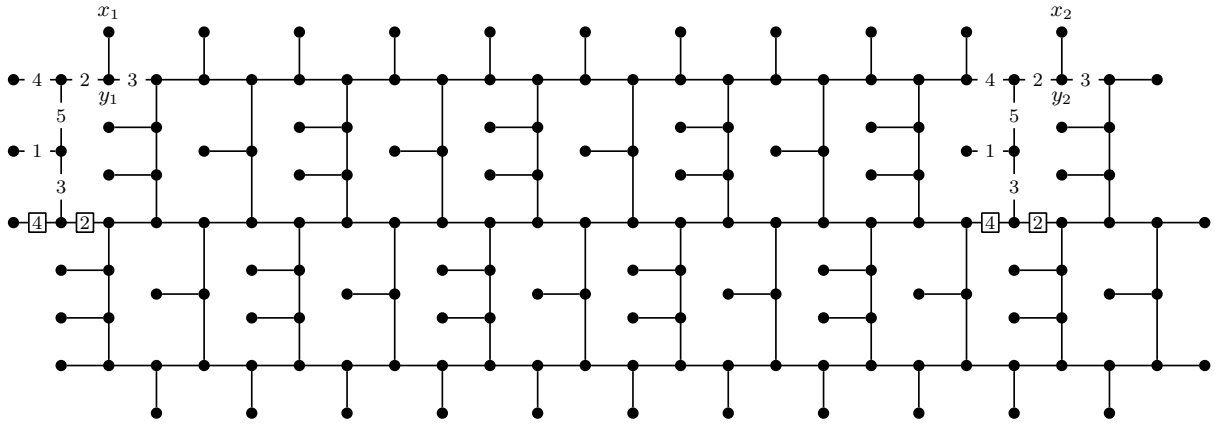
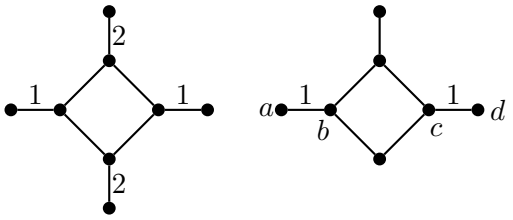
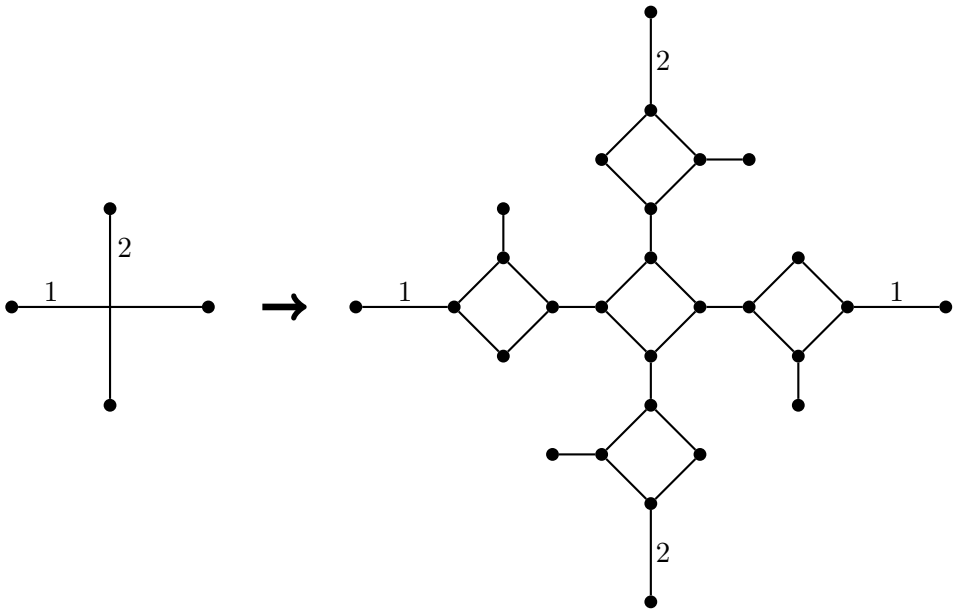


Figure 3.30: A valid and unique strong 5-edge-precolouring of the gadget of Figure 3.28



(a) Colours 1 and 2 are forced



(b) Replacing crossing edges

Figure 3.31: Transportation of colours in a strong 6-edge-colouring

Let us take a look at the graph P of Figure 3.32, obviously it has $\chi'_s(P) \geq 6$ and it can be easily strongly edge-coloured with six colours. Following Observation 3.26, up to a permutation, in any strong 6-edge-colouring of this graph the colours 1 and 2 are forced. We call *pendant* edges the edges of P incident to a vertex of degree one and coloured 1 and 2 in the figure. Notice that P is bipartite.

Next, we construct the vertex gadget *i.e.* the graph G_v that will replace a vertex v of G in G', as depicted in Figure 3.33. First we build the graph of Figure 3.33a:

Take two copies of graph P with six pairs of pendant edges and connect them in order to

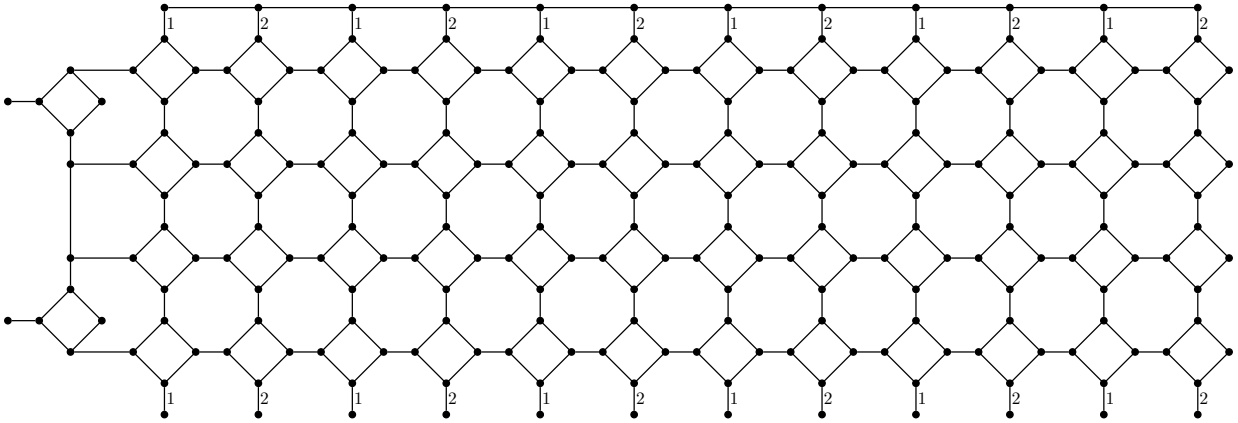


Figure 3.32: Graph P and some of its forced colours

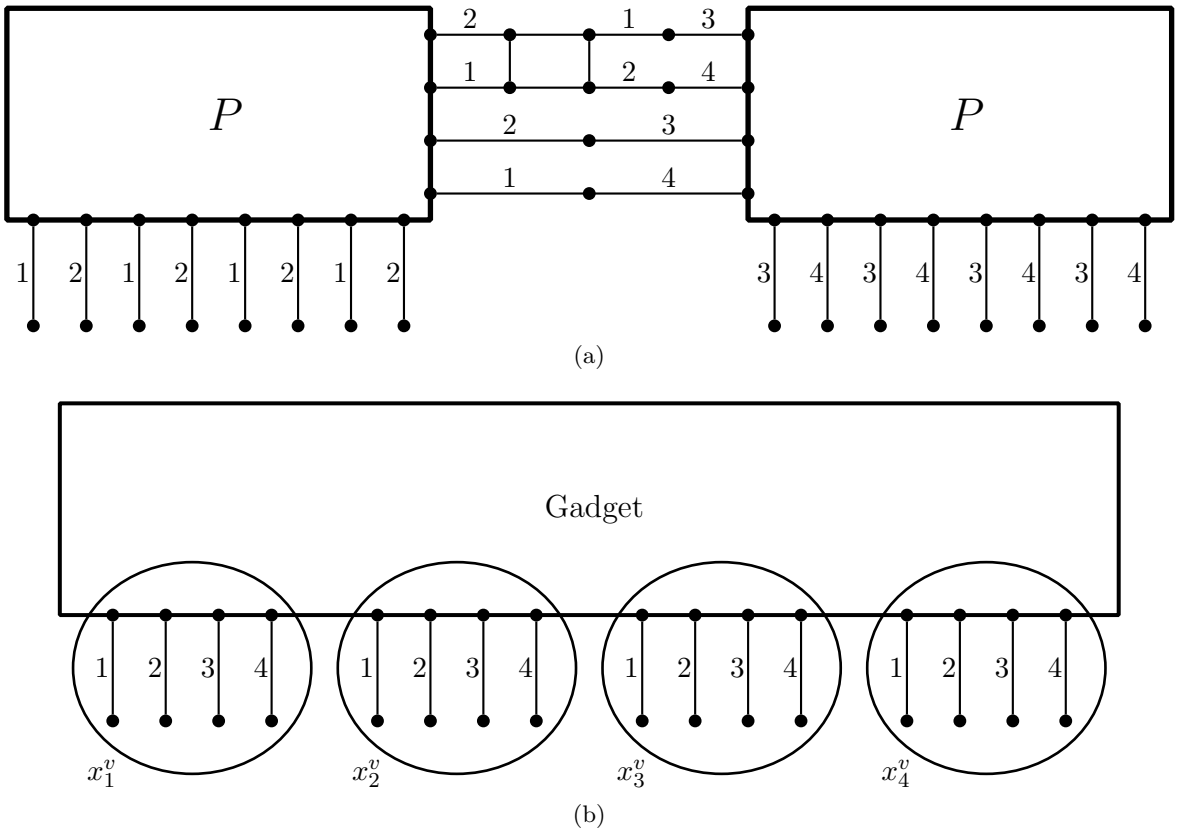


Figure 3.33: Vertex gadget Q for Theorem 3.25

obtain a graph with eight pairs of pendant edges, as shown in the figure. Observe that due to the connection between these copies, the colours of pendant edges of the left copy must be distinct from the colours of pendant edges of the right copy ($\{1, 2\}$ and $\{3, 4\}$ respectively). It is easy to see that the obtained graph Q is planar, bipartite, subcubic, and such that $\chi'_s(Q) = 6$.

Now, we show how to obtain the final representation of this gadget (the one of Figure 3.33b): Take a copy of the graph Q and choose an embedding such that there are four quadruples of edges coloured 1, 2, 3 and 4 in this order. Note that according to Observation 3.27, each crossing of edges is replaced as shown in Figure 3.31b, such that the obtained graph is planar. Consider four pendant edges (one for each quadruple) of the obtained graph having the same colour, say colour 1. For each of these edges, label its incident vertex of degree 1, x_k^v ($1 \leq k \leq 4$).

For an edge uv of G, in G' we identify vertices x_i^u et x_j^v and connect the other three pairs of

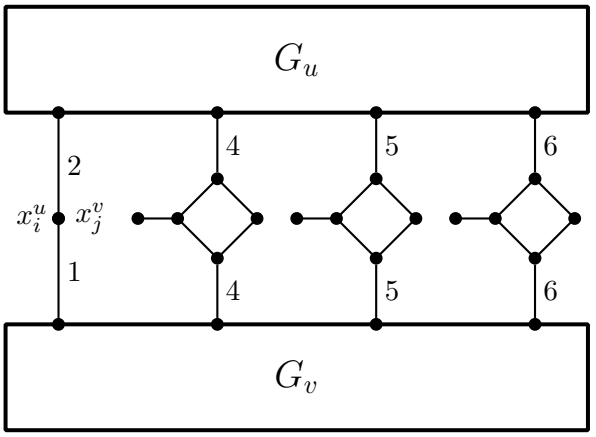


Figure 3.34: Edge gadget for Theorem 3.25 and its forced colouring

edges as shown in Figure 3.34. We remark that edges of G_u adjacent to the edge coloured with 4 (5 or 6) can be coloured with the same colours as the edges of G_v adjacent to the second edge coloured with 4 (5 or 6). This would make the strong 6-edge-colouring of the gadget impossible. In order to avoid this type of conflicts while transporting a colour, according to Observation 3.26, one could add an additional cycle C_4 between edges coloured 4, 5 and 6.

Note that if G' is connected, in any strong 6-edge-colouring of G' the colours 4, 5 and 6 used in Figure 3.34 are not used to colour any edge incident to some vertex labelled x_i^v.

We claim that the obtained graph G' is strongly 6-edge-colourable if and only if G is 3-colourable. Similar to the proof of Theorem 3.23, the forbidden colours in a strong 6-edge-colouring of the graph G' are the colours of edges not incident to some vertex labelled x_i^v. Hence, in G' there are three forbidden colours. If G is 3-colourable then we can assign the colour of a vertex v of G to the pendant edge G_v incident to x_i^v in G' and extend this colouring to a valid strong 6-edge-colouring of G'. Conversely, given a strong 6-edge-colouring of G', since there are three forbidden colours for G', we can use the colour of the edge incident to x_i^v in the graph G_v, to colour v in G. □

3.4 Open problems

In part (iii) of Theorem 3.9 we proved that every subcubic graph G with $\mathrm{mad}(G) < \frac{8}{3}$ satisfies $\chi'_s(G) \leq 8$. By part (iv) of the same theorem we know that if $\mathrm{mad}(G) < \frac{20}{7}$ then $\chi'_s(G) \leq 9$. Moreover, this bound is tight as there exists a subcubic graph with $\mathrm{mad}(G) = \frac{20}{7}$ and $\chi'_s(G) = 10$. Even though all graphs with $\mathrm{mad}(G) < \frac{20}{7}$ are strongly 9-edge-colourable, we do not know any example of subcubic graph with $\mathrm{mad}(G) < \frac{20}{7}$ and $\chi'_s(G) = 9$. This leaves a natural open question of moving the bound $\frac{8}{3}$ closer to $\frac{20}{7}$:

Question 3.28. Does there exist a subcubic graph G with $\mathrm{mad}(G) < \frac{20}{7}$ and having $\chi'_s(G) = 9$?

We do not believe that the same local discharging technique as used in our proof of part (iii) of Theorem 3.9, would help to answer the above question. It might be interesting to investigate this problem through global discharging approach.

Derived from Theorem 3.9, Corollary 3.11 states that every subcubic planar graph with girth g at least 14 (resp. 10, 9, 7) can be strongly edge-coloured with six (resp. seven, eight, nine) colours. The case of $g \geq 7$ and $\chi'_s(G) \leq 9$ is improved by Theorem 3.20 but we do not know if it is optimal. As of the relevance of the lower bounds on the girth parameter provided by the other three parts of Corollary 3.11, we think that there is scope for further research:

Question 3.29. What is the minimum size of the girth of a subcubic planar graph G such that $\chi'_s(G)$ is at most 6, 7, 8 respectively? In case if Conjecture 3.7 is not true, what is the minimum size of the girth of G such that $\chi'_s(G) \leq 9$?

Conjecture 3.7 still remains a challenging open problem. For this purpose studying subcubic planar graphs without C_4 or without C_5 could be the next step. For instance, can one use discharging method and Euler's formula to prove the conjecture for these restricted cases?

We recall that $\overline{C_6}$ is the only known subcubic planar graph, having the strong chromatic index equal to 9. The other subcubic graphs with $\chi'_s(G) = 9$ we found are not planar, thus another question related to Conjecture 3.7 would be:

Question 3.30. Does there exist a subcubic planar graph G other than $\overline{C_6}$ of Figure 3.3, such that $\chi'_s(G) = 9$? If yes, is it possible to find G having at least one vertex v with $d(v) \leq 2$?

In Section 3.3 we proved that STRONG k-EDGE-COLOURING is NP-complete for various restrictions of subcubic planar graphs when $k = 4, 5, 6$. When $k = 4$ this restriction is the class of planar subcubic bipartite graphs of an arbitrarily large girth. There is not much room for improvement in this case since for the intuitive smaller class of trees the problem is clearly polynomial (quadratic in the number of edges in the worst case). For $k \geq 5$ the problem becomes polynomial when the input graph is subcubic planar and of girth at least 31 [19]. On the other hand, when $k = 5$, we showed that the problem is NP-complete when restricted to subcubic planar graphs of girth 9. For $k = 6$, the problem is yet again NP-complete for the class of subcubic planar bipartite graphs. According to Corollary 3.11, when the girth is at least 14, every subcubic graph is strongly 6-edge-colourable. It is very likely that the span between the bounds 9 and 31 is not optimal in the case of STRONG 5-EDGE-COLOURING and between the bounds 4 and 14 in the case of STRONG 6-EDGE-COLOURING. That is:

Question 3.31. Unless P=NP, what is the smallest value of the girth of a subcubic planar graph, such that STRONG k-EDGE-COLOURING is polynomial for $k = 5, 6$?

The question whether there exists a polynomial algorithm for $k = 7, 8, 9$ for the class of subcubic planar graphs is not studied. However, if Conjecture 3.7 is solved, the case when $k = 9$ becomes trivial. If the answer for Question 3.30 is negative for one of the questions, the case when $k = 8$ becomes also trivial.

Finally, since for chordal graphs the problem is polynomial [17], it would also be interesting to examine the complexity of the problem for other classes of perfect graphs.

Chapter 4

Identifying codes

Given a graph G, a *vertex-identifying code* of G or simply *identifying code*, is a subset $\mathcal{C}$ of vertices of G such that every vertex of G is uniquely determined within $\mathcal{C}$. Formally speaking:

Definition 4.1. The subset $\mathcal{C}$ of $V(G)$ is an identifying code of G if $\mathcal{C}$ is both:

- a *dominating set* of G, i.e. $\forall x \in V(G), N[x] \cap \mathcal{C} \neq \emptyset$, and
- a *separating set* of G, i.e. $\forall u, v \in V(G)\ (u \neq v), N[u] \cap \mathcal{C} \neq N[v] \cap \mathcal{C}$.

In this chapter, we investigate vertex-identifying codes, and identifying codes of line graphs, called also edge-identifying codes (Sections 4.1 and 4.2). We also discuss the complexity issues of these types of identification (Section 4.3). We conclude the chapter with some open questions and remarks (Section 4.4).

Identifying codes have been widely studied since the introduction of the concept in [66], and have been applied to problems such as fault-diagnosis in multiprocessor systems [66], compact routing in networks [71], emergency sensor networks in facilities [86] or the analysis of secondary RNA structures [57].

The concept of identifying codes of graphs is related to several other concepts, such as locating-dominating sets [94, 93] for graphs or Bondy's theorem on induced subsets [12, 46].

Before giving the state of the art, we introduce some additional notations. A vertex x of G is *universal* if $N[x] = V(G)$. Given a subset S of $V(G)$, we say that a vertex x is *S-universal* if $S \subseteq N[x]$. The symmetric difference of two sets A and B is denoted by $A \ominus B$. We recall that two vertices x and y are called *twins* in G if $N[x] = N[y]$ and a graph is called *twin-free* if it has no pair of twin vertices.

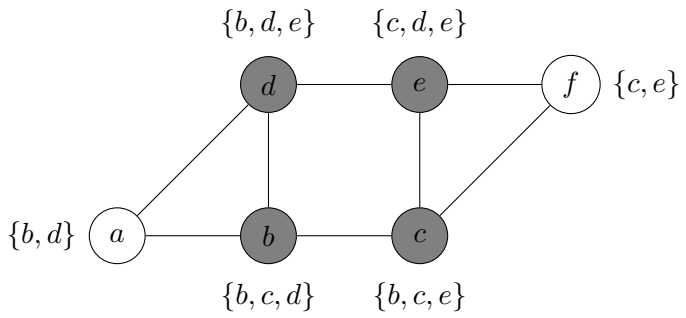


Figure 4.1: An example of identifying code (vertices in gray). The sets represent the neighbours within the identifying code.

A subset S *separates* two vertices x and y, if $N[x] \cap S \neq N[y] \cap S$. A subset S of vertices is a *separating set* if it separates any pair of distinct vertices. The cardinalities of a minimum separating set and minimum identifying code of G are denoted $\gamma^{\text{S}}(G)$ and $\gamma^{\text{ID}}(G)$, respectively. If S is dominating and separates vertices of $V(G) \setminus S$, it is called a *locating-dominating set*, this concept was introduced in [94]. The cardinality of a minimum locating-dominating set of a graph G is denoted $\gamma^{\text{LD}}(G)$. Observe that we have the following relation: $\gamma^{\text{LD}}(G) \leq \gamma^{\text{ID}}(G)$. Although identifying codes and locating-dominating sets are closely related, in this chapter we will focus mainly on identifying codes.

Observe that to admit a separating set, a graph must be twin-free. Moreover, this condition is sufficient, as $V(G)$ is a separating set of G. A graph G admits a separating set if and only if it is twin-free. Therefore, a graph admits an identifying code if and only if it is twin-free. In contrast to the case of identifying codes, observe that every graph admits a locating-dominating set.

Since a set $\mathcal{C}$ of k elements can uniquely identify at most 2^k elements and every vertex of the graph must be dominated, the parameter $\gamma^{\text{ID}}(G)$ has a natural lower bound:

Theorem 4.2 (Karpovsky *et al.*, [66])**.** Let G be a twin-free graph G on n vertices. Then $\gamma^{\text{ID}}(G) \geq \lceil \log_2(n+1) \rceil$.

Moreover, equality holds for infinitely many graphs and this collection of graphs was classified in [80].

When speaking about upper bounds, in the case of locating-dominating sets it has been shown that for every graph G with at least one edge, $\gamma^{\text{LD}}(G) \leq |V(G)| - 1$ [93, 25]. Moreover, in [93] it was proved that for a connected graph G, $\gamma^{\text{LD}}(G) = |V(G)| - 1$ if and only if G is either $K_{1,t}$ or a complete graph.

Naturally, the same question was raised and studied in the case of identifying codes and a similar result was shown:

Theorem 4.3 (Bertrand, 2001 [11], Gravier and Moncel, 2007 [55])**.** Let G be a twin-free graph on n vertices having at least one edge. Then $\gamma^{\text{ID}}(G) \leq n - 1$.

This bound is tight. For example, it can be easily checked that $\gamma^{\text{ID}}(K_{1,t}) = t$ for $t \geq 2$. In [21] Charbit *et al.* conjectured that the only graphs with $\gamma^{\text{ID}}(G) = |V(G)| - 1$ are the star $K_{1,t}$ and the complete graph K_n minus a maximum matching. In [92] Skaggs proposed a different conjecture stating that these graphs are exactly the closure of $\mathcal{P} = \{P_4\}$ under operation $\bowtie$ and $\mathcal{P} \bowtie K_1$. In Section 4.1.2, we disprove these conjectures by characterizing all graphs with $\gamma^{\text{ID}}(G) = |V(G)| - 1$. We note that contrary to locating-dominating sets, this class is a much richer family.

Conjecture 4.4 (Foucaud *et al.*, [45])**.** For every connected twin-free graph G on n vertices, there exists a constant such that $\gamma^{\text{ID}}(G) \leq n - \frac{n}{\Delta} + c$.

In support of this conjecture, in [44] we proved the following weaker upper bound for the size of a minimum identifying code of a twin-free graph:

Theorem 4.5 ([44])**.** Let G be a connected twin-free graph on n vertices.
Then $\gamma^{\text{ID}}(G) \leq n - \frac{n}{\Theta(\Delta^5)}$. Moreover, if G is Δ-regular, then $\gamma^{\text{ID}}(G) \leq n - \frac{n}{\Theta(\Delta^3)}$.

In the case of triangle-free graphs, this bound is improved in [45]:

Theorem 4.6 (Foucaud *et al.*, [45])**.** Let G be a connected twin-free graph on n vertices and with no triangles. Then $\gamma^{\text{ID}}(G) \leq n - \frac{n}{\Delta + o(\Delta)}$.

From a computational point of view, finding the exact value of γ^{ID} for a graph G is known to be NP-hard [23]. In fact, it remains NP-hard for many subclasses of graphs such as bipartite graphs [24] or some restricted subclasses of planar graphs [3, 4]. Furthermore, approximating $\gamma^{\text{ID}}(G)$ is not easy neither as shown in [70, 54, 96]. We point out that the proof of hardness of approximation given in [54, Theorem 3] can actually be adapted in order to show that finding the minimum size identifying code in the class of chordal graphs is NP-hard: the authors use an L-reduction from the total dominating set problem, which is known to be NP-hard in the class of chordal graphs [72]; moreover, one can check that their proof preserves chordality and thus we conclude that the identifying code problem is NP-hard for chordal graphs. In Section 4.3 we improve this result by showing that the problem remains NP-hard for an even smaller subclass of chordal graphs, namely split graphs. We also prove that the identifying code problem is NP-hard for another restricted subclass of perfect graphs: perfect 3-colourable planar line graphs.

4.1 Vertex-identifying codes

This section is dedicated to the study of upper bounds for the minimum size of identifying codes. In particular, we characterize all graphs such that $\gamma^{\text{ID}}(G) = |V(G)| - 1$. The results presented in this section are published in [44].

The next section provides a set of preliminary results, necessary for proving the main result of this section (Theorem 4.15).

4.1.1 Preliminary results

We start by providing necessary tools (observations and propositions) for proving Theorem 4.15.

Observation 4.7. Let G be a twin-free graph and let $\mathcal{C}$ be an identifying code of G. Then, any set $\mathcal{C}' \subseteq V(G)$ such that $\mathcal{C} \subseteq \mathcal{C}'$ is an identifying code of G.

The next proposition is useful in proving upper bounds on minimum identifying codes by induction.

Proposition 4.8. Let G be a twin-free graph and $S \subseteq V(G)$ such that $G - S$ is twin-free. Then $\gamma^{\text{ID}}(G) \leq \gamma^{\text{ID}}(G - S) + |S|$.

Proof. Take a minimum code $\mathcal{C}_0$ of $G - S$. Consider the vertices of S in an arbitrary order $(x_1, \ldots, x_{|S|})$. Using induction we extend $\mathcal{C}_0$ to a subset $\mathcal{C}_i$ of G which identifies the vertices in $V_i = V(G) \setminus \{x_{i+1}, \ldots, x_{|S|}\}$. To do this, if $\mathcal{C}_{i-1}$ identifies all the vertices of V_i, we are done. Otherwise, since all the vertices in V_{i-1} are identified, either $N[x_i] \cap \mathcal{C}_{i-1} = N[y] \cap \mathcal{C}_{i-1}$ for exactly one vertex y in V_{i-1}, or x_i is not dominated by $\mathcal{C}_{i-1}$. In the first case x_i and y are separated in G by some vertex, say u, so let $\mathcal{C}_i = \mathcal{C}_{i-1} \cup \{u\}$. In the second case, let $\mathcal{C}_i = \mathcal{C}_{i-1} \cup \{x_i\}$. Now, in both cases, $\mathcal{C}_i$ identifies all the vertices of V_i. At step $|S|$, $\mathcal{C}_{|S|}$ is an identifying code of G of size at most $|\mathcal{C}_0| + |S| \leq \gamma^{\text{ID}}(G - S) + |S|$. □

We derive the following special case of the previous proposition.

Corollary 4.9. Let G be a connected graph with $\gamma^{\text{ID}}(G) = |V(G)| - 1$, $G \ncong K_{1,2}$, then there is a vertex x of G such that $G - x$ is still connected and $\gamma^{\text{ID}}(G - x) = |V(G - x)| - 1$.

Proof. If $G \cong K_{1,t}$, $t \neq 2$, then any leaf vertex works. Thus, we may suppose $G \ncong K_{1,t}$. Then by Theorem 4.3, there is a vertex x of G such that $V(G-x)$ is an identifying code of G and thus $G-x$ is twin-free and $G-x \ncong \overline{K_n}$. By Proposition 4.8, we have $\gamma^{\text{ID}}(G-x) \geq \gamma^{\text{ID}}(G)-1 = |V(G-x)|-1$. Equality holds since otherwise $\gamma^{\text{ID}}(G - x) = |V(G - x)|$. To complete the proof, we show that x can be chosen such that $G - x$ is connected. To see this, assume $G - x$ is not connected. Since $\gamma^{\text{ID}}(G - x) = |V(G - x)| - 1$, except one component, every component of $G - x$ is an isolated vertex. If there are two or more such isolated vertices, then either one of them can be the vertex we want. Otherwise there is only one isolated vertex, call it y. Now if $G - y$ is twin-free, then y is the desired vertex, else there is a vertex x' such that $N[x'] = N[x] - y$. Then $G - x'$ is connected and twin-free. □

Lemma 4.10. Let G be a twin-free graph and let $v \in V(G)$. Let x, y be a pair of twins in $G - v$. If $G - x$ or $G - y$ has a pair of twins, then v must be one of the vertices of the pair.

Proof. Since v separates x and y, it is adjacent to one of them (say x) and not to the other. Suppose z, t are twins in $G - x$. Suppose z is adjacent to x and t is not. If $z \neq v$, then y is also adjacent to z and, therefore, t is also adjacent to y which implies x being adjacent to t. This contradicts the fact that x separates z and t. The other case is proved similarly. □

Proposition 4.11. Let G_1 and G_2 be twin-free graphs such that for every minimum separating set S there is an S-universal vertex. If $G_1 \bowtie G_2$ is twin-free, then $\gamma^{\text{S}}(G_1 \bowtie G_2) = \gamma^{\text{S}}(G_1)+\gamma^{\text{S}}(G_2)+1$. Furthermore, if S is a separating set of size $\gamma^{\text{S}}(G_1) + \gamma^{\text{S}}(G_2) + 1$ of $G_1 \bowtie G_2$, then there is an S-universal vertex.

Proof. Let S be a minimum separating set of $G_1 \bowtie G_2$. Since vertices of G_2 do not separate any pair of vertices in G_1, $S \cap V(G_1)$ is a separating set of G_1. By the same argument $S \cap V(G_2)$ is a separating set of G_2. Therefore, $|S| \geq \gamma^{\text{S}}(G_1) + \gamma^{\text{S}}(G_2)$. But if $|S| = \gamma^{\text{S}}(G_1) + \gamma^{\text{S}}(G_2)$, then there is an $[S \cap V(G_1)]$-universal vertex x in G_1 and an $[S \cap V(G_2)]$-universal vertex y in G_2. But then, x and y are not separated by S.

Given a separating set S_1 of G_1 and a separating set S_2 of G_2, the set $S_1 \cup S_2$ separates all pairs of vertices except the S_1-universal vertex of G_1 from the S_2-universal vertex of G_2. But since $G_1 \bowtie G_2$ is twin-free, we could add one more vertex to $S_1 \cup S_2$ to obtain a separating set of $G_1 \bowtie G_2$ of size $\gamma^{\text{S}}(G_1) + \gamma^{\text{S}}(G_2) + 1$.

For the second part assume S is a separating set of size $\gamma^{\text{S}}(G_1)+\gamma^{\text{S}}(G_2)+1$ of $G_1 \bowtie G_2$. Then we have either $|S \cap V(G_1)| = \gamma^{\text{S}}(G_1)$ or $|S \cap V(G_2)| = \gamma^{\text{S}}(G_2)$. Without loss of generality assume the former. Then there is an $[S \cap V(G_1)]$-universal vertex z of G_1. Since z is also adjacent to all the vertices of G_2, it is an S-universal vertex of $G_1 \bowtie G_2$. □

We remark that in Proposition 4.11 if $G_1 \ncong K_1$ and $G_2 \ncong K_1$, then $\gamma^{\text{ID}}(G_1 \bowtie G_2) = \gamma^{\text{S}}(G_1 \bowtie G_2) = \gamma^{\text{S}}(G_1) + \gamma^{\text{S}}(G_2) + 1$.

4.1.2 Graphs having maximum possible identifying code number

In this section we classify all graphs G for which $\gamma^{\text{ID}}(G) = |V(G)| - 1$. As already mentioned, some of them are stars and joins of copies of P_4 are examples of such graphs. To classify the rest we show that special powers of paths are the basic examples of such graphs. Then we show that any other example is mainly obtained from the join of some basic elements.

Definition 4.12. For an integer $k \geq 1$, let $A_k = (V_k, E_k)$ be the graph with vertex set $V_k = \{x_1, \ldots, x_{2k}\}$ and edge set $E_k = \{x_i x_j \mid |i - j| \leq k - 1\}$.

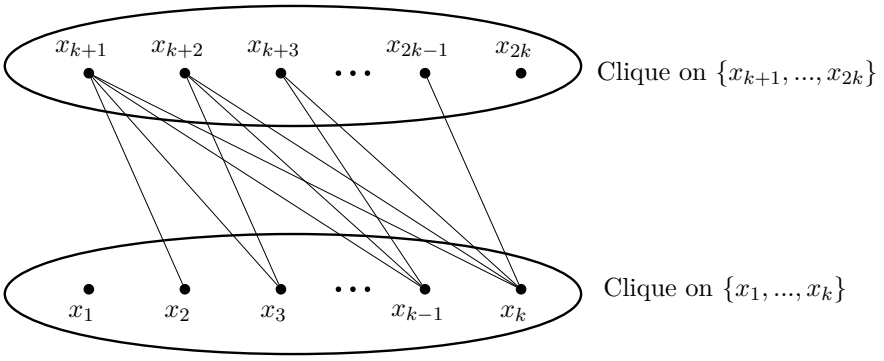


Figure 4.2: The graph A_k which needs $|V(A_k)| - 1$ vertices for any identifying code

An illustration of graph A_k is given in Figure 4.2. We note that for $k \geq 2$ we have $A_k = P_{2k}^{k-1}$ and $A_1 = \overline{K_2}$. It is also easy to check that the only non-trivial automorphism of A_k is the mapping $x_i \to x_{2k+1-i}$. It is not hard to observe that A_k is twin-free, $\Delta(A_k) = 2k - 2$ and that A_k and $\overline{A_k}$ are connected if $k \geq 2$.

Proposition 4.13. For $k \geq 1$, we have: $\gamma^{\mathrm{S}}(A_k) = 2k - 1$ with $N[x_k]$ and $N[x_{k+1}]$ being the only separating sets of size $2k - 1$ of A_k. Furthermore, if $k \geq 2$, $\gamma^{\mathrm{ID}}(A_k) = 2k - 1$.

Proof. Let S be a separating set of A_k. For $i < k$, we have $\ominus(x_i, x_{i+1}) = \{x_{i+k}\}$ and for $k < i \leq 2k - 1$, we have $\ominus(x_i, x_{i+1}) = \{x_{i-k+1}\}$. Thus, $\{x_2, \ldots, x_{2k-1}\} \subset S$. But to separate x_k and x_{k+1}, we must add x_1 or x_{2k}. It is now easy to see that $V_k \setminus \{x_1\} = N[x_{k+1}]$ and $V_k \setminus \{x_{2k}\} = N[x_k]$, each is a separating set of size $2k - 1$. If $k \geq 2$, then they both dominate A_k and therefore are also identifying codes. □

As a corollary of the previous proof, we have:

Corollary 4.14. For $k \geq 1$ every minimum separating set S of A_k has a S-universal vertex.

Let $\mathcal{A}$ be the closure of $\{A_i \mid i = 1, 2, \ldots\}$ with respect to operation $\bowtie$.

Theorem 4.15. Given a connected graph G, we have $\gamma^{\mathrm{ID}}(G) = |V(G)| - 1$ if and only if $G \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$ and $G \ncong A_1$.

In the following we prove this Theorem.

Proof of Theorem 4.15

First we show that elements of $\mathcal{A}$ are also extremal graphs with respect to both separating sets and identifying codes.

Proposition 4.16. For every graph $G \in \mathcal{A}$, we have $\gamma^{\mathrm{S}}(G) = |V(G)| - 1$. Furthermore, every minimum separating set S of G has an S-universal vertex.

Proof. The proposition is true for basic elements of $\mathcal{A}$ by Proposition 4.13 and by Corollary 4.14. For a general element $G = G_1 \bowtie G_2$ it is true by Proposition 4.11 and by induction. □

Corollary 4.17. If $G \in \mathcal{A}$ and $G \ncong A_1$, then $\gamma^{\mathrm{ID}}(G) = |V(G)| - 1$.

Further examples of graphs extremal with respect to separating sets and identifying codes can be obtained by adding a universal vertex to each of the graphs in $\mathcal{A}$, as we prove below.

Proposition 4.18. For every graph G in $\mathcal{A} \bowtie K_1$ we have $\gamma^{\mathrm{ID}}(G) = \gamma^{\mathrm{S}}(G) = |V(G)| - 1$.

Proof. Assume $G = G_1 \bowtie K_1$ with $G_1 \in \mathcal{A}$, and assume u is the vertex corresponding to K_1. Suppose S is a minimum separating set of G. We first note that since $S \cap V(G_1)$ is a separating set of G_1, we have $|S \cap V(G_1)| \geq |V(G_1)| - 1$. But if $|S \cap V(G_1)| = |V(G_1)| - 1$, then by Proposition 4.16, there is an $[S \cap V(G_1)]$-universal vertex y of G_1. Then y is not separated from x. Thus $|S \cap V(G_1)| = |V(G_1)|$ and therefore $S = V(G_1)$. It is easy to check that S is also an identifying code. □

It was proved in [25] that $\gamma^{\mathrm{ID}}(K_n \setminus M) = n - 1$ where $K_n \setminus M$ is the complete graph minus a maximal matching. We note that this graph, for even values of n, is the join of $\frac{n}{2}$ disjoint copies of A_1, thus it belongs to $\mathcal{A}$. For odd values of n, it is built from the previous graph by adding a universal vertex.

So far we have seen that $\gamma^{\mathrm{ID}}(G) = |V(G)| - 1$ for $G \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$, $G \ncong A_1$. We also know that $\gamma^{\mathrm{ID}}(\overline{K_n}) = n$. More examples of graphs with $\gamma^{\mathrm{ID}}(G) = |V(G)| - 1$ can be obtained by adding isolated vertices. In the next theorem we show that for any other twin-free graph G we have $\gamma^{\mathrm{ID}}(G) \leq |V(G)| - 2$.

Proposition 4.19. Let G be a connected twin-free graph such that $\gamma^{\mathrm{ID}}(G) = |V(G)| - 1$. Then $G \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$ and $G \ncong A_1$.

Proof. We proceed by induction on the number of vertices of G. For graphs on at most 4 vertices this is easy to check. Assume the claim is true for graphs on at most $n - 1$ vertices and, by contradiction, let G be a twin-free graph on $n \geq 5$ vertices such that $\gamma^{\mathrm{ID}}(G) = n - 1$ and $G \notin \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$.

By Corollary 4.9 there is a vertex $x \in V(G)$ such that $G - x$ is connected and $\gamma^{\mathrm{ID}}(G - x) = |V(G - x)| - 1$. By the induction hypothesis we have $G - x \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$. Depending on which one of these three sets $G - x$ belongs to, we will have three cases.

Case 1, $G - x \in \{K_{1,t} \mid t \geq 2\}$. In this case we consider a minimum identifying code $\mathcal{C}$ of $G - x$. If $\mathcal{C}$ does not already identify x, then either $deg(x) \leq 3$ or $deg(x) \geq n - 2$. We leave it to the reader to check that in each of these cases, there is an identifying code of size $n - 2$.

Case 2, $G - x \in \mathcal{A}$. We consider two subcases. Either $G - x \cong A_k$ for some k or $G - x = G_1 \bowtie G_2$, with $G_1, G_2 \in \mathcal{A}$.

(1) $G - x \cong A_k$, for some $k \geq 2$. If x is adjacent to all the vertices of $G - x$, then $G \in \mathcal{A} \bowtie K_1$ and we are done. Otherwise there is a pair of consecutive vertices of A_k, say x_i and x_{i+1}, such that one is adjacent to x and the other is not. By the symmetry of A_k we may assume $i \leq k$. We claim that one of the sets $\mathcal{C} = V(G) \setminus \{x_{2k}, x\}$, $\mathcal{C}' = V(G) \setminus \{x_1, x\}$ or $\mathcal{C}'' = V(G) \setminus \{x_k, x_{k+1}\}$ is an identifying code of G. This would contradict our assumption. Note that for each of the sets $\mathcal{C}$, $\mathcal{C}'$ and $\mathcal{C}''$, vertices of $V(G - x)$ are all separated. If x is also separated from all the vertices of $G - x$ then we are done. Otherwise there will be two possibilities.

First we consider the possibility: x is not adjacent to x_i and adjacent to x_{i+1}. Let us choose $\mathcal{C}$ as a potential identifying code. Each vertex x_j, $j > i + k$, is separated from x by x_{i+1} and each vertex x_j, $j < i + k$, is separated from x by x_i. Since there must exist a vertex from which x must not be separated, we conclude that x is not separated from x_{i+k} and therefore x must be adjacent to $x_{i+1}, \ldots, x_{2k-1}$. On the other hand, if we consider $\mathcal{C}'$, then x is not separated only from x_k (it is the only vertex adjacent to x_{2k-1} but not to x_{2k}). Therefore, x is adjacent to exactly $x_2, \ldots, x_{2k-1}$. And in this case $\mathcal{C}''$ is an identifying code of G. Indeed, x is separated from all other vertices of G and also it separates x_1 from x_2 and x_{2k-1} from x_{2k}.

In the other possibility, x is adjacent to x_i and not adjacent to x_{i+1}. If we consider $\mathcal{C}$, a similar argument implies that x is separated from every vertex but x_{2k}. Then $\mathcal{C}'$ would be an identifying code.

(2) $G - x \cong G_1 \bowtie G_2$ with $G_1, G_2 \in \mathcal{A}$. If x is adjacent to all the vertices of $G - x$, then $G \in \mathcal{A} \bowtie K_1$ and we are done. Thus there is a vertex, say y, that is not adjacent to x. Without loss of generality, we can assume $y \in V(G_1)$. Let $\mathcal{C}_1$ be an identifying code of size $\gamma^{\text{ID}}(G_1) = |V(G_1)| - 1$ of G_1 which contains y. The existence of such an identifying code becomes apparent from the proof of Proposition 4.16. Then $\mathcal{C} = \mathcal{C}_1 \cup V(G_2)$ is an identifying code of $G_1 \bowtie G_2$ of size $|V(G_1 \bowtie G_2)| - 1 = |V(G)| - 2$. Thus $\mathcal{C}$ does not separate a vertex of $G_1 \bowtie G_2$ from x. Call this vertex z. Since $y \in \mathcal{C}$, z is not adjacent to y, hence $z \in V(G_1)$. Therefore, z is adjacent to all the vertices of G_2. So x should also be adjacent to all the vertices of G_2. Thus we have $G = (G_1 + x) \bowtie G_2$ and any minimum identifying code of $G_1 + x$ together with all vertices of G_2 would form an identifying code of G. This proves that $\gamma^{\text{ID}}(G_1 + x) = |V(G_1 + x)| - 1$. Since $G_1 + x$ has less vertices than G, by induction hypothesis, we have $G_1 + x \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$ and $G \ncong A_1$. Since $G_1 \in \mathcal{A}$, and since x is not adjacent to a vertex of G_1, we should have $G_1 + x \in \mathcal{A}$ but all graphs in $\mathcal{A}$ have an even number of vertices and this is not possible.

Case 3, $G - x \in \mathcal{A} \bowtie K_1$. Suppose $G - x \cong A_{i_1} \bowtie A_{i_2} \bowtie \ldots \bowtie A_{i_j} \bowtie K_1$ and let u be the vertex corresponding to K_1.

If x is also adjacent to u, then u is a universal vertex of G and $G-u$ is also twin-free. In this case we apply the induction on $G-u$: by Proposition 4.8, $\gamma^{\text{ID}}(G-u) = |V(G-u)|-1$ and by induction hypothesis $G - u \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$. But if $G - u \in \{K_{1,t} \mid t \geq 2\} \cup (\mathcal{A} \bowtie K_1)$, there will be two universal vertices, and therefore twins. Thus $G - u \in \mathcal{A}$ and $G \in \mathcal{A} \bowtie K_1$.

We now assume x is not adjacent to u and we repeat the argument with $G-u$ if it is twin-free. In this case if $G - u \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A}$, we apply Case 1 or Case 2. If $G - u \in \mathcal{A} \bowtie K_1$ with u' being the vertex of K_1, then u and u' induce an isomorphic copy of A_1 and $G \in \mathcal{A}$.

If $G - u$ is not twin-free, then by Lemma 4.10, x must be one of the twin vertices. Let x' be its twin and suppose $x' \in V(A_{i_1})$ with $V(A_{i_1}) = \{z_1, z_2, \ldots, z_{2k}\}$. Without loss of generality we may assume $x' = z_l$ with $l \leq k$. If $l \geq 2$, then we claim $\mathcal{C} = V(G) \setminus \{z_l, z_{2k}\}$ is an identifying code of G which is a contradiction. To prove our claim notice first that vertices of $A_{i_2} \bowtie \cdots \bowtie A_{i_j}$ are already identified from each other and from the other vertices. Now each pair of vertices of A_{i_1} is separated by a vertex in $V(A_{i_1}) \cap \mathcal{C}$ except z_{l+k-1} and z_{l+k} which are separated by x. The vertex x is also separated from all the other vertices by u. It remains to show that u is separated from vertices of A_{i_1}. It is separated from vertices in $\{z_1, \ldots, z_{l+k-1}\}$ by x and from $\{z_{k+1}, \ldots, z_{2k}\}$ by z_1 ($l \geq 2$). Thus $x' = x_1$ and now it is easy to see that the subgraph induced by $V(A_{i_1})$, u and x is isomorphic to A_{i_1+1} and, therefore, $G \cong A_{i_1+1} \bowtie A_{i_2} \bowtie \ldots \bowtie A_{i_j}$. □

Since every graph $G \in \{K_{1,t} \mid t \geq 2\} \cup \mathcal{A} \cup (\mathcal{A} \bowtie K_1)$ has maximum degree $|V(G)| - 2$, we obtain the following:

Corollary 4.20. Let G be a twin-free connected graph on $n \geq 3$ vertices and maximum degree $\Delta \leq n - 3$. Then $\gamma^{\text{ID}}(G) \leq n - 2$.

We remark that the definition of an identifying code can be easily extended to infinite graphs. In contrast to finite graphs there are non-trivial infinite graphs for which the whole vertex set is needed to form an identifying code. Examples of these graphs are given in [25]. In [44] we classified all such infinite graphs.

4.2 Edge-identification

In this section, we study the identification of edges by edges or *edge-identifying codes*. The results of this section are published in [43].

Let $\mathcal{S}_E$ be a subset of edges of a graph G. We define the graph induced by $\mathcal{S}_E$ as the graph with the set of all endpoints of the edges of $\mathcal{S}_E$ as its vertex set and $\mathcal{S}_E$ as its edge set.

Given a graph G and an edge e of G, we define $N[e]$ to be the set of edges adjacent to e together with e itself. An *edge-identifying code* of a graph G is a subset $\mathcal{C}_E$ of edges such that for each edge e the set $N[e] \cap \mathcal{C}_E$ is non-empty and uniquely determines e:

Definition 4.21. The subset $\mathcal{C}_E$ of $E(G)$ is an edge-identifying code of G if $\mathcal{C}_E$ is both:

- an *edge-dominating set* of G, i.e. for each edge $e \in E(G)$, $N[e] \cap \mathcal{C}_E \neq \emptyset$, and
- an *edge-separating set* of G, i.e. for each pair $e, f \in E(G)$ $(e \neq f)$, $N[e] \cap \mathcal{C}_E \neq N[f] \cap \mathcal{C}_E$.

One can easily observe that an edge-identifying code of a graph G is an identifying code of the line graph of G, $\mathcal{L}(G)$. Therefore, a graph G admits an edge-identifying code if and only if $\mathcal{L}(G)$ is twin-free.

We say that an edge e separates two edges f and g if either e belongs to $N[f]$ but not to $N[g]$, or vice-versa. To avoid trivial cases, when considering edge-identifying codes, we assume that the edge set of the graph is non-empty. A pair of twins in $\mathcal{L}(G)$ may correspond in G to a pair of: parallel edges, or a pair of adjacent edges whose non common ends are of degree 1, or a pair of adjacent edges whose non common ends are of degree 2 but they are connected to each other. Here parallel edges are not allowed and we consider simple graphs only. A pair of edges of the other two types will be called *edge-twins*. We illustrate the possible pairs of edge-twins in Figure 4.3. Therefore, it is easy to see that a graph is *edge-identifiable* if and only if it is *edge-twin-free*. The smallest size of an edge-identifying code of an edge-identifiable graph G is denoted by $\gamma^{\text{EID}}(G)$ and is called the *edge-identifying code number* of G.

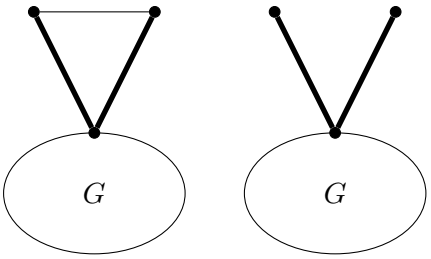


Figure 4.3: Two possible pairs of edge-twins (thick edges) in G.

In Figure 4.4, we show an edge-identifying code of the Petersen graph. The lower bound of Theorem 4.2 proves that $\gamma^{\text{EID}}(P) \geq 4$. Later, by improving this bound for line graphs, we will see that in fact $\gamma^{\text{EID}}(P) = 5$ (see Theorem 4.25 of Section 4.2.1).

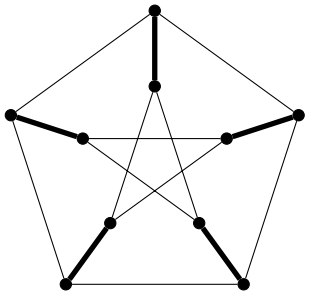


Figure 4.4: An edge-identifying code of the Petersen graph.

4.2.1 Preliminary results

In this section we first give some easy tools which help for finding the minimum edge-identifying code of graphs. We then apply these tools to determine the exact values of γ^{EID} for some basic

families of graphs.

Lemma 4.22. Let G be a simple graph with girth at least 5. Let $\mathcal{C}_E$ be a subset of edges of G that covers vertices of G (that is, an *edge cover* of G), such that the graph $(V(G), \mathcal{C}_E)$ is edge-twin-free. Then $\mathcal{C}_E$ is an edge-identifying code of G. In particular, if G has a perfect matching M, then M is an edge-identifying code of G.

Proof. The code $\mathcal{C}_E$ is an edge-dominating set of G because it covers all the vertices of G. To complete the proof, we need to prove that $\mathcal{C}_E$ is also an edge-separating set. Let e_1, e_2 be two edges of G. If $e_1, e_2 \in \mathcal{C}_E$, then $\mathcal{C}_E \cap N[e_1] \neq \mathcal{C}_E \cap N[e_2]$ because $(V(G), \mathcal{C}_E)$ is edge-twin-free. Otherwise, we can assume that $e_2 \notin \mathcal{C}_E$. If $e_1 \in \mathcal{C}_E$ and $\mathcal{C}_E \cap N[e_1] = \mathcal{C}_E \cap N[e_2]$, then e_2 must be adjacent to e_1. Let u be their common vertex and $e_2 = uv$. Since $\mathcal{C}_E$ is an edge cover, there is an edge $e_3 \in \mathcal{C}_E$ which is incident to v. However, e_3 cannot be adjacent to e_1 because G is triangle-free. Therefore e_3 separates e_1 and e_2. Finally, we assume neither of e_1 and e_2 is in $\mathcal{C}_E$. Then there are two edges of $\mathcal{C}_E$, say e_3 and e_4, adjacent to the two ends of e_1. But since G has neither C_3 nor C_4 as a subgraph, e_3 and e_4 cannot both be adjacent to e_2 and, therefore, e_1 and e_2 are separated. □

We note that in the previous proof the absence of C_4 is only used at the last part where e_1, e_2, e_3, e_4 could induce a C_4 which is not adjacent to any other edge of $\mathcal{C}_E$. Thus, we have the following stronger statement:

Lemma 4.23. Let G be a triangle-free graph. Let $\mathcal{C}_E$ be a subset of edges of G that covers vertices of G, such that $\mathcal{C}_E$ is edge-twin-free. If for no pair xy, uv of isolated edges in $\mathcal{C}_E$, the set $\{x, y, u, v\}$ induces a C_4 in G, then $\mathcal{C}_E$ is an edge-identifying code of G.

We will also need the following lemma about edge-twin-free trees.

Lemma 4.24. If T is an edge-twin-free tree on more than two vertices, then T has two vertices of degree 1 each adjacent to a vertex of degree 2.

Proof. Take a longest path in T, then it is easy to verify that both ends of this path satisfy the condition of the lemma. □

The following general theorem together with its corollary, will be useful for determining the edge-identifying code number of several classes of graphs.

Theorem 4.25. Let G be a connected edge-twin-free graph. We have:

$$\gamma^{\mathrm{EID}}(G) \geq \frac{|V(G)|}{2}$$

Proof. Let $\mathcal{C}_E$ be an edge-identifying code of G. Let G' be the subgraph induced by $\mathcal{C}_E$ and let $G_1, \ldots, G_s$ be the connected components of G'. Let n_i be the order of G_i and k_i be its size (thus $\sum_{i=1}^{s} k_i = |\mathcal{C}_E|$). Let $X = V(G) \setminus V(G')$ and n'_i be the number of vertices in X that are joined to a vertex of G_i in G. We show that $n'_i + n_i \leq 2k_i$. If $k_i = 1$, then clearly $n'_i = 0$ and $n'_i + n_i = 2 = 2k_i$. If G_i is a tree, then $n_i = k_i + 1$ and, by Lemma 4.24, G_i must have two vertices of degree 2 each having a vertex of degree 1 as a neighbour. Then no vertex of X can be adjacent to one of these two vertices in G. Moreover, each other vertex of G_i can be adjacent to at most one vertex in X. So $n'_i \leq k_i - 1$, and finally $n_i + n'_i \leq 2k_i$. If G_i is not at tree, we have $n_i \leq k_i$ and $n'_i \leq n_i$ and, therefore, $n'_i + n_i \leq 2k_i$. Finally, since G is connected, each vertex in X is connected to at least one G_i. Hence by counting the number vertices of G we have:

$$|V(G)| \leq \sum_{i=1}^{s} (n_i + n'_i) \leq 2 \sum_{i=1}^{s} k_i \leq 2|\mathcal{C}_E|$$

□

Theorem 4.25 together with Lemma 4.23 leads to the following result:

Corollary 4.26. Let G be an edge-identifiable triangle-free graph. Suppose G has a perfect matching M with the property that for any pair xy, uv of edges in M, the set $\{x, y, u, v\}$ does not induce a C_4. Then M is an optimal edge-identifying code and $\gamma^{\text{EID}}(G) = \frac{|V(G)|}{2}$.

This shows for example that the edge-identifying code of the Petersen graph given in Figure 4.4 is optimal. This result can be extended to graphs with girth 4 which have a perfect matching with no pair of edges of the matching that induces a C_4.

4.2.2 Edge-identification for some classes of graphs

In this section we determine γ^{EID} of some families of graphs.

Proposition 4.27. We have $\gamma^{\text{EID}}(K_n) = \begin{cases} 5, & \text{if } n = 4 \text{ or } 5 \\ n-1, & \text{if } n \geq 6 \end{cases}$. Furthermore, let $\mathcal{C}_E$ be an edge-identifying code of K_n of size $n-1$ ($n \geq 6$) and let $G_1, G_2, \ldots, G_k$ be the connected components of $(V(K_n), \mathcal{C}_E)$. Then exactly one component, say G_i, is isomorphic to K_1 and every other component G_j ($j \neq i$) is isomorphic to a cycle of length at least 5.

Proof. We note that $\mathcal{L}(K_4)$ is isomorphic to $K_6 \setminus M$, where M is a perfect matching of K_6. One can check that this graph has identifying code number 5. By a case analysis, we can show that K_5 does not admit any edge-identifying code of size 4. Indeed, since an edge-identifying code must be edge-twin-free, there are only two graphs possibly induced by an edge-identifying code of this size: a path P_5 or a cycle C_4. In both cases, there are edges which are not separated. Edges of a C_5 form an edge-identifying code of size 5 of K_5, hence $\gamma^{\text{EID}}(K_5) = 5$. Furthermore, it is not difficult to check that the set of edges of a cycle of length $n-1$ ($n \geq 6$) identifies all edges of K_n. Thus we have $\gamma^{\text{EID}}(K_n) \leq n-1$.

Let us prove $\gamma^{\text{EID}}(K_n) \geq n-1$. Let $\mathcal{C}_E$ be an edge-identifying code of K_n of size $n-1$ or less ($n \geq 6$). Let $G' = (V(K_n), \mathcal{C}_E)$. Let $G_1, G_2, \ldots, G_k$ be the connected components of G'. Since G' has n vertices but at most $n-1$ edges, at least one component of G' is a tree. On the other hand we claim that at most one of these components can be a tree and that such tree would be isomorphic to K_1. Let G_i be a tree. First we show that $|V(G_i)| \leq 2$. If not, by Lemma 4.24, there is a vertex x of degree 1 in G_i with a neighbour u of degree 2. Let v be the other neighbour of u. Then the edges xv and uv are not identified. If $V(G_i) = \{x, y\}$ then for any other vertex u, the edges ux and uy are not separated. Finally, if there are G_i and G_j with $V(G_i) = \{x\}$ and $V(G_j) = \{y\}$, then the edge xy is not dominated by $\mathcal{C}_E$. Thus exactly one component of G', say G_1, is a tree and $G_1 \cong K_1$. This implies that $\gamma^{\text{EID}}(K_n) \geq n-1$. Therefore, $\gamma^{\text{EID}}(K_n) = n-1$ and, furthermore, each G_i, ($i \geq 2$), is a graph with a unique cycle.

It remains to prove that each G_i, $i \geq 2$ is isomorphic to a cycle of length at least 5. By contradiction suppose one of these graphs, say G_2, is not isomorphic to a cycle. Since G_2 has a unique cycle, it must contain a vertex v of degree 1. Let t be the neighbour of v in G_2 and let u be the vertex of G_1. Then the edges tv and tu are not separated by $\mathcal{C}_E$. Finally we note that such cycle cannot be of length 3 or 4, because C_3 is not edge-twin-free and in C_4, the two chords (which are edges of K_n) would not be separated. □

The following examples show that if true, the upper bound of Conjecture 4.4 is tight even in the class of line graphs. These examples were first introduced in [45] but without using the notion of edge-identifying codes.

Proposition 4.28. Let G be a k-regular multigraph ($k \geq 3$). Let G_1 be obtained from G by subdividing each edge exactly once. Then $\gamma^{\text{EID}}(G_1) = (k-1)|V(G)| = |E(G_1)| - \frac{|E(G_1)|}{2k-2} = |V(\mathcal{L}(G_1))| - \frac{|V(\mathcal{L}(G_1))|}{\Delta(\mathcal{L}(G_1))}$.

Proof. Let x be a vertex of G_1 of degree at least 3 (an original vertex from G). For each edge e_i^x incident to x, let $e_i'^x$ be the edge adjacent to e_i^x but not incident to x and let $A_x = \{e_i'^x\}_{i=1}^k$. Then $\{A_x \mid x \in V(G)\}$ is a partition of $E(G_1)$. For any edge-identifying code $\mathcal{C}_E$ of G_1, if two elements of A_x, say $e_1'^x$ and $e_2'^x$, are both not in $\mathcal{C}_E$, then e_1^x and e_2^x are not separated. Thus $|\mathcal{C}_E \cap A_x| \geq k-1$. This proves that $|\mathcal{C}_E| \geq (k-1)|V(G)|$.

We now build an edge-identifying code of this size by choosing one edge of each set A_x, in such a way that for each vertex x originally from G, exactly one edge incident to x is chosen. Then the set of non-chosen edges will be an edge-identifying code. To select this set of edges, one can consider the incident bipartite multigraph H of G: the vertex set of H is $V \cup V'$ where V and V' are copies of $V(G)$ and there is an edge xx' in H if $x \in V$, $x' \in V'$ and $xx' \in E(G)$. The multigraph H is k-regular and bipartite, thus it has a perfect matching M. For each vertex $x \in V$, let $\rho(x)$ be the vertex in V' such that $x\rho(x) \in M$. Let now $e_M'^x$ be the edge of G_1 that belongs to the set A_x and is incident to $\rho(x)$ (in G_1). Finally, let $\mathcal{C}_E = E(G_1) \setminus \{e_M'^x\}_{x \in V(G)}$. Exactly one element of each A_x is not in $\mathcal{C}_E$, and for each vertex x, exactly one edge incident to x is not in $\mathcal{C}_E$. This implies that $\mathcal{C}_E$ is an edge-identifying code. □

Hypercubes, being the natural ground of code-like structures, have been a center of focus for determining the smallest size of their identifying codes. The problem has proved to be a challenging one from both theoretical and computational points of view. Today the precise identifying code number is known for only seven hypercubes [22]. In contrast, using Corollary 4.26, we show here that finding the edge-identifying code number of a hypercube is not so difficult. For this purpose, we recall that as mentioned in the previous section of preliminaries, Corollary 4.26 can be extended to graphs with girth 4 which have a perfect matching with no pair of edges of the matching that induces a C_4. This is the case for the hypercube of dimension $d \geq 4$.

Proposition 4.29. For $d \geq 4$, we have $\gamma^{\text{EID}}(\mathcal{H}_d) = 2^{d-1}$.

Proof. By Theorem 4.25, we have $\gamma^{\text{EID}}(\mathcal{H}_d) \geq 2^{d-1}$. We will construct by induction a perfect matching M_d of $\mathcal{H}_d$ such that no pair of edges induces a C_4, for $d \geq 4$. By Lemma 4.23, M_d will be an edge-identifying code of $\mathcal{H}_d$, proving the result. Two such matchings of $\mathcal{H}_4$, which are also disjoint, are presented in Figure 4.5. The matching M_5 can now be built using each of these two matchings of $\mathcal{H}_4$ — one matching per copy of $\mathcal{H}_4$ in $\mathcal{H}_5$. It is easily verified that M_5 has the required property. Furthermore, M_5 has the extra property that for each edge uv of M_5, u and v do not differ on the first coordinate (recall that we build $\mathcal{H}_5$ from $\mathcal{H}_4$ by adding a new coordinate on the left, hence the first coordinate is a the new one). We now build the matching M_d of $\mathcal{H}_d$ ($d \geq 6$) from M_{d-1} in such a way that no two edges of M_d belong to a 4-cycle in $\mathcal{H}_d$ and that for each edge uv of M_d, u and v do not differ on the first coordinate. To do this, let $\mathcal{H}_1'$ be the copy of $\mathcal{H}_{d-1}$ in $\mathcal{H}_d$ induced by the set of vertices whose first coordinate is 0. Similarly, let $\mathcal{H}_2'$ be the copy of $\mathcal{H}_{d-1}$ in $\mathcal{H}_d$ induced by the other vertices. Let $\mathcal{M}_1'$ be a copy of M_{d-1} in $\mathcal{H}_1'$ and let $\mathcal{M}_2'$ be a matching in $\mathcal{H}_2'$ obtained from $\mathcal{M}_1'$ by the following transformation: for $e = uv \in \mathcal{M}_1'$, define $\psi(e) = \sigma(u)\sigma(v)$ where $\sigma(x) = x + (1, 0, 0, \ldots, 0)$. It is now easy to check that the new matching $M_d = \mathcal{M}_1' \cup \mathcal{M}_2'$ has both properties we need. □

We note that the formula of Proposition 4.29 does not hold for $d = 2$ and $d = 3$. For $d = 2$ the hypercube $\mathcal{H}_2$ is isomorphic to C_4 and thus $\gamma^{\text{EID}}(\mathcal{H}_2) = 3$. For $d = 3$, we note that an identifying code of size 4, if it exists, must be a matching with no pair of edges belonging to a 4-cycle. But this is not possible. An identifying code of size 5 is shown in Figure 4.6, therefore $\gamma^{\text{EID}}(\mathcal{H}_3) = 5$.

4.2.3 Lower Bounds

Recall from Theorem 4.2 that $\gamma^{\text{ID}}(G)$ is bounded below by a function of the order of G. As mentioned before, this bound is tight. Let $\mathcal{C}$ be a set of c isolated vertices. We can build a graph G of order $2^c - 1$ such that $\mathcal{C}$ is an identifying code of G. To do this, for every subset X of $\mathcal{C}$

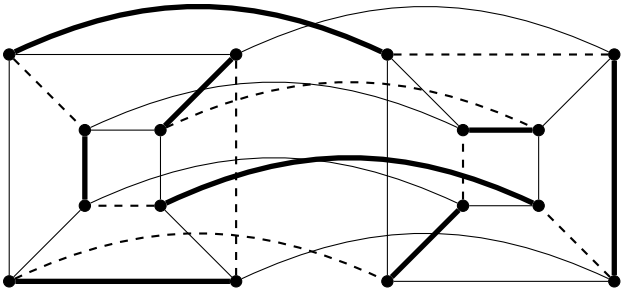


Figure 4.5: Two disjoint edge-identifying codes of $\mathcal{H}_4$.

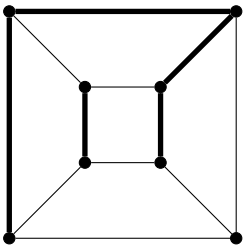


Figure 4.6: An optimal edge-identifying code of $\mathcal{H}_3$.

with $|X| \geq 2$, we associate a new vertex which is joined to all vertices in X and only to those vertices. Then, it is easily seen that $\mathcal{C}$ is an identifying code of this graph. However, the graph built in this way is far from being a line graph as it contains $K_{1,t}$ for even large values of t. In fact this lower bound turns out to be far from being tight for the family of line graphs. In this section we give a tight lower bound on the size of an edge-identifying code of a graph in terms of the number of its edges. Equivalently we have a lower bound for the size of an identifying code in a line graph in terms of its order. This lower bound is of the order $\Theta(\sqrt{n})$ and thus is a much improved lower bound with respect to the general bound of Theorem 4.2.

Let G be an edge-identifiable graph and let $\mathcal{C}_E$ be an edge-identifying code of G. To avoid trivialities such as having isolated vertices we may assume that G is connected. We note that this does not mean that the subgraph induced by $\mathcal{C}_E$ is also connected, in fact we observe almost the contrary, i.e. in most cases, an edge-identifying code of a minimum size will induce a disconnected subgraph of G. We first prove a lower bound for the case when an edge-identifying code induces a connected subgraph.

Theorem 4.30. If an edge-identifying code $\mathcal{C}_E$ of a non-trivial graph G induces a connected subgraph of G which is not isomorphic to K_2, then G has at most $\binom{|\mathcal{C}_E|+2}{2} - 4$ edges. Furthermore, inequality can only hold if $\mathcal{C}_E$ induces a path.

Proof. Let G' be the subgraph induced by $\mathcal{C}_E$. Since we assumed G' is connected, and since G' is edge-twin-free, it cannot have three vertices. Since we assumed $G' \ncong K_2$, we conclude that G' has at least four vertices. For each vertex x of G', let $\mathcal{C}_E^x$ be the set of all edges incident to x in G'. Let $e = uv$ be an edge of G, then one or both of u and v must be in $V(G')$. Therefore, depending on which of these vertices belong to $\mathcal{C}_E$, e is uniquely determined by either $\mathcal{C}_E^u$ (if $u \in V(G')$ and $v \notin V(G')$), or $\mathcal{C}_E^v$ (if $u \notin V(G')$ and $v \in V(G')$), or $\mathcal{C}_E^u \cup \mathcal{C}_E^v$ (if both $u, v \in V(G')$). The total number of sets of this form can be at most $|V(G')| + \binom{|V(G')|}{2} = \binom{|V(G')|+1}{2}$, thus if $|V(G')| \leq |\mathcal{C}_E|$ we are done. Otherwise, since G' is connected, $|V(G')| = |\mathcal{C}_E| + 1$ and G' is an edge-twin-free tree on at least 4 vertices. If v is a vertex of degree 1 adjacent to u, then we have $\mathcal{C}_E^v = \{uv\}$ but $uv \in \mathcal{C}_E^u$ and, therefore, $\mathcal{C}_E^u = \mathcal{C}_E^u \cup \mathcal{C}_E^v$. On the other hand, by Lemma 4.24, there are two vertices of degree 2 that have neighbours of degree 1. Let u be such a vertex, let v be its neighbour

of degree 1 and x be its other neighbour. Then $\mathcal{C}_E^v = \{uv\}$ and $\mathcal{C}_E^u = \{uv, ux\}$ and, therefore, $\mathcal{C}_E^u \cup \mathcal{C}_E^x = \mathcal{C}_E^v \cup \mathcal{C}_E^x$. Thus the total number of distinct sets of the form $\mathcal{C}_E^y$ or $\mathcal{C}_E^y \cup \mathcal{C}_E^z$ is at most $\binom{|\mathcal{C}_E|+2}{2} - 4$. But if equality holds there can only be two vertices of degree 1 in G' and hence $\mathcal{C}_E$ is a path. □

We note that if this bound is tight, then G' is a path. Furthermore, for each path P_{k+1} one can build many graphs which have P_{k+1} as an edge-identifying code and have $\binom{k+2}{2} - 4$ edges. The set of all these graphs will be denoted by $\mathcal{J}_k$. An example of such a graph is obtained from K_{k+2} by removing a certain set of four edges as shown in Figure 4.7. Note that every other member of $\mathcal{J}_k$ is obtained from the previous example by splitting the vertex that does not belong to P_{k+1} (but without adding any new edge).

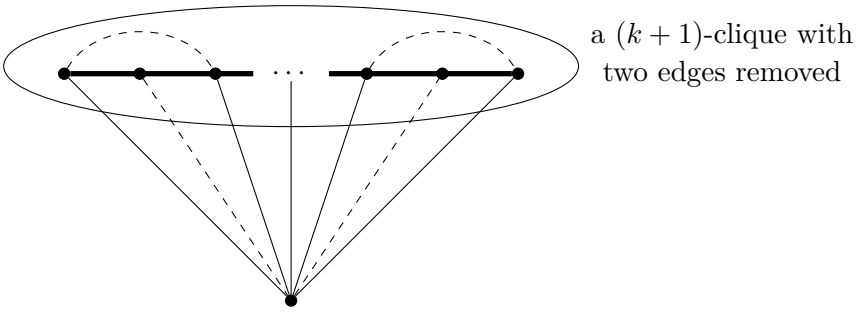


Figure 4.7: An extremal graph of $\mathcal{J}_k$ with its connected edge-identifying code

Next we consider the case when the subgraph induced by $\mathcal{C}_E$ is not necessarily connected.

Theorem 4.31. Let G be an edge-identifiable graph and let $\mathcal{C}_E$ be an edge-identifying code of G with $|\mathcal{C}_E| = k$. Then we have:

$$|E(G)| \leq \begin{cases} \binom{\frac{4}{3}k}{2}, & \text{if } k \equiv 0 \bmod 3 \\ \binom{\frac{4}{3}(k-1)+1}{2} + 1, & \text{if } k \equiv 1 \bmod 3 \\ \binom{\frac{4}{3}(k-2)+2}{2} + 2, & \text{if } k \equiv 2 \bmod 3 \end{cases}$$

Proof. Let G be a graph with maximum number of edges among all graphs with $\gamma^{\mathrm{EID}}(G) = k$. It can be easily checked that for $k = 1, 2$ or 3, the maximum number of edges of G is 1, 3 or 6 respectively. For $k \geq 4$, we prove a slightly stronger statement: given an edge-identifying code $\mathcal{C}_E$ of G of size k, all but at most two of the connected components of the subgraph induced by $\mathcal{C}_E$ must be isomorphic to P_4. When there is only one component not isomorphic to P_4, it must be isomorphic to a P_2, a P_5 or a P_6. If there are two such components, then they can be two copies of P_2, a P_2 with a P_5, or just two copies of P_5. This depends on the value of k mod 3.

To prove our claim let G be a graph as defined above, let $\mathcal{C}_E$ be an edge-identifying code of size k of G and let G' be the subgraph induced by $\mathcal{C}_E$. For each vertex $u \in V(G) \setminus V(G')$, we can assume that u has degree 1: if u has degree $d > 1$, with neighbours $v_1, \ldots, v_d$ necessarily in $V(G')$, then replace u by d vertices of degree 1: $u_1, \ldots, u_d$, connecting u_i to v_i. Then the number of edges does not change, and the code $\mathcal{C}_E$ remains an edge-identifying code of size k, thus it suffices to prove our claim for this new graph. Let $G'_1, G'_2, \ldots, G'_r$ be the connected components of G' with $|V(G'_i)| = n'_i$. For each $i \in \{1, \ldots, r\}$, let G_i be the graph induced by the vertices of G'_i and the vertices connected to G'_i only. To each vertex x of G' we assign the set $\mathcal{C}_E^x$ of edges in G' incident to x.

We first note that no G'_i can be of order 3, because there is no connected edge-twin-free graph on three vertices. If u and v are vertices from two disjoint components of G' with each component being of order at least 4, then the pair u, v is uniquely determined by $\mathcal{C}_E^u \cup \mathcal{C}_E^v$, thus by maximality

of G, uv is an edge of G. If a component of G' is isomorphic to K_2, assuming u and u' are vertices of this component, then for any other vertex v of G' exactly one of uv or $u'v$ is an edge of G.

We now claim that each G'_i with $n'_i \geq 4$ is a path. By contradiction, if a G'_i is not a path, we replace G_i by a member $J_{n'_i-1}$ of $\mathcal{J}_{n'_i-1}$ with $P_{n'_i}$ being its edge-identifying code. Then we join each vertex of $P_{n'_i}$ to each vertex of each G'_j (with $j \neq i$ and $n'_j \geq 4$) and to exactly one vertex of each G_j with $n'_j = 2$. We note that the new graph still admits an edge-identifying code of size k. However, it has more edges than G. Indeed, while the number of edges connecting G'_i and the G'_j's ($j \neq i$) is not decreased, the number of edges in G_i is increased when we replace G_i by $J_{n'_i-1}$. This can be seen by applying Theorem 4.30 on G_i.

We now show that none of the G'_i's can have more than six vertices. By contradiction, suppose G'_1 is a component with $n'_1 \geq 7$ vertices (thus $n'_1 - 1$ edges). We build a new graph G^*_1 from G as follows. We take disjoint copies of $J_3 \in \mathcal{J}_3$ and $J_{n'_1-4} \in \mathcal{J}_{n'_1-4}$ with P_4 and $P_{n'_1-3}$ being, respectively, their edge-identifying codes. We now let $V(G^*_1) = V(J_3) \cup V(J_{n'_1-4}) \cup (V(G) \setminus V(G_1))$. The edges of J_3, $J_{n'_1-4}$ and $G - G_1$ are also edges of G^*_1. We then add edges between these three parts as follows. We join every vertex of P_4 to each vertex of $P_{n'_1-3}$. For $i = 2, 3, \ldots, r$ if $n'_i \geq 4$, join every vertex of G'_i to each vertex of $P_4 \cup P_{n'_1-3}$. If $n'_i = 2$, we choose exactly one vertex of G'_i and join it to each vertex of $P_4 \cup P_{n'_1-3}$. The construction of G^*_1 ensures that it still admits an edge-identifying code of size k, but it has more edges than G. In fact, the number of edges is increased in two ways. First, because $P_4 \cup P_{n'_1-3}$ has one more vertex than G'_1, the number of edges connecting $P_4 \cup P_{n'_1-3}$ to $G - G_1$ has increased (unless $r = 1$). More importantly, the number of edges induced by $J_3 \cup J_{n'_1-4}$ is $6 + \binom{n'_1-2}{2} - 4 + 4 \times (n'_1 - 3) = \frac{{n'_1}^2}{2} + \frac{3n'_1}{2} - 7$ which is strictly more than $|E(G'_1)| = \frac{{n'_1}^2}{2} + \frac{n'_1}{2} - 4$ for $n'_1 \geq 3$. Since $n'_1 \geq 7$, this contradicts the maximality of G.

With a similar method, the following transformations strictly increase the number of edges while the new graph still admits an edge-identifying code of size k:

1. Two components of G' each on six vertices transform into two graphs of $\mathcal{J}_3$ and a graph of $\mathcal{J}_4$.

2. One component of G' on six vertices and another component on five vertices transform into three graphs of $\mathcal{J}_3$.

3. One component of G' on six vertices and one on two vertices transform into two graphs of $\mathcal{J}_3$.

4. Three components of G' each on five vertices transform into four graphs of $\mathcal{J}_3$.

5. Two components of G' on five vertices and one on two vertices transform into three graphs of $\mathcal{J}_3$.

6. A component of G' on five vertices and two on two vertices transform into two graphs of $\mathcal{J}_4$.

7. Three components of G' each isomorphic to P_2 transform into a graph of $\mathcal{J}_3$.

For the proof of case 7, we observe that the number of edges identified by the three P_2's would be the same as the number of edges identified by the P_4. However, since $k \geq 4$, there must be some other component in G'. Moreover, the number of vertices of the three P_2's, which are joined to the vertices of the other components of G', is three, whereas the number of these vertices of the P_4, is four. Hence the maximality of G is contradicted.

We note that cases 1, 2 and 3 imply that if a component of G' is isomorphic to P_6, every other component is isomorphic to P_4. Then cases 4, 5 and 6 imply that if a component is isomorphic to P_5, then at most one other component is not isomorphic to P_4 and such component is necessarily

either a P_2 or a P_5. Finally, case 7 shows that there can be at most two components both isomorphic to P_2.

We conclude that each of the components of G' is isomorphic to P_4 except for possibly two of them. These exceptions are dependent on the value of k mod 3 as we described. The formulas of the theorem can be derived using these structural properties of G. For instance, in the case $k \equiv 0 \bmod 3$, each component of G' is isomorphic to P_4. There are $\frac{k}{3}$ such components. For each component G'_i, there are six edges in the graph G_i. That gives $2k$ edges. The other edges of G are edges between two components of G'. By maximality of G, between two components of G', there are exactly 16 edges. There are $\binom{\frac{k}{3}}{2}$ pairs of components of G'. Hence, the number of edges in G is:

$$2k + 16\binom{\frac{k}{3}}{2} = \binom{\frac{4}{3}k}{2}.$$

The other cases can be proved with the same method. □

We note that this bound is tight and the examples were in fact built inside the proof. More precisely, for $k \equiv 0 \bmod 3$ we take $\frac{k}{3}$ disjoint copies of elements of $\mathcal{J}_3$ each having a P_4 as an edge-identifying code. We then add an edge between each pair of vertices coming from two distinct such P_4's. We note that the union of these P_4's is a minimum edge-identifying code of the graph. If $k \not\equiv 0 \bmod 3$, then we build a similar construction. This time we use elements from $\mathcal{J}_3$ with at most two exceptions that are elements of $\mathcal{J}_4$ or $\mathcal{J}_5$.

The above theorem can be restated in the language of line graphs as follows.

Corollary 4.32. Let G be a line graph. Then we have $\gamma^{\mathrm{ID}}(G) \geq \frac{3}{8}(1 + \sqrt{1 + 8|V(G)|}) > \frac{3\sqrt{2}}{4}\sqrt{|V(G)|}$.

Proof. Suppose G is the line graph of an edge-twin-free graph H ($L(H) = G$). Let $k = \gamma^{\mathrm{ID}}(G) = \gamma^{\mathrm{EID}}(H)$, and let n be the number of vertices of G ($n = |E(H)|$). Then, after solving the quadratic inequalities of Theorem 4.31 for k, we have:

$$\begin{aligned}
k \geq& \frac{3}{8} + \frac{3\sqrt{8n+1}}{8}, \text{ for } k = 0 \bmod 3\\
k \geq& \frac{5}{8} + \frac{3\sqrt{8n-7}}{8}, \text{ for } k = 1 \bmod 3\\
k \geq& \frac{3}{8} + \frac{3\sqrt{8n-15}}{8}, \text{ for } k = 2 \bmod 3
\end{aligned}$$

It is then easy to check that the right-hand side of each of the three inequalities is at least as $\frac{3\sqrt{2}}{4}\sqrt{n}$ for $n \geq 3$. □

Remark. Note that the lower bound of $\gamma^{\mathrm{ID}}(G) \geq \Theta(\sqrt{|V(G)|})$, which holds for the class of line graphs, is also implied by Theorem 4.25. However, the bound of 4.32 is more precise. In [10], Beineke characterized line graphs by a list of nine forbidden induced subgraphs. Considering Beineke's characterization, the lower bound of Corollary 4.32 can be restated as follows: $\gamma^{\mathrm{ID}}(G) \geq \Theta(\sqrt{|V(G)|})$ holds if G has no induced subgraph from Beineke's list. It is then natural to ask what is a minimal list of forbidden induced subgraphs for which a similar claim would hold. Note that the claw graph, $K_{1,3}$, belongs to Beineke's list of forbidden subgraphs. However, we remark that the bound $\gamma^{\mathrm{ID}}(G) \geq \Theta(\sqrt{|V(G)|})$ does not hold for the class of claw-free graphs. Examples can be built as follows: let A be a set of size k and let B be the set of nonempty subsets of A. Let G be the graph built on $A \cup B$, where A and B each induce a complete graph and a vertex a of A is joined to a vertex b of B if $a \in b$. This graph is claw-free and it is easy to find an identifying code of size at most $2k = \Theta(\log |V(G)|)$ in G.

4.2.4 Upper bounds

In Theorem 4.15 of the previous section we gave the characterization of graphs having $\gamma^{\text{ID}} \leq |V(G)|-1$. It is easy to check that only six of these graphs are line graphs, namely $P_3, P_4, C_4, P_4 \bowtie K_1, C_4 \bowtie K_1, L(K_4)$. Thus we have the following corollary:

Corollary 4.33. If G is a line graph with $G \notin \{P_3, P_4, C_4, P_4 \bowtie K_1, C_4 \bowtie K_1, \mathcal{L}(K_4)\}$, then we have $\gamma^{\text{ID}}(G) \leq |V(G)| - 2$.

Since $\gamma^{\text{EID}}(K_{2,n}) = 2n-2$, $\gamma^{\text{ID}}(\mathcal{L}(K_{2,n})) = |V(L(K_{2,n}))|-2$ and the bound of Corollary 4.33 is tight for an infinite family of graphs. Conjecture 4.4 proposes a better bound in terms of both the number of vertices and the maximum degree of a graph. As pointed out in Proposition 4.28, most of the known extremal graphs for Conjecture 4.4 are line graphs. In this section, after proving some general bound for the edge-identifying code number of an edge-twin-free graph we will show that Conjecture 4.4 holds for the class of line graphs of high enough density.

A graph on n vertices is *2-degenerate* if its vertices can be ordered $v_1, v_2, \ldots, v_n$ such that each v_i is joined to at most two vertices in $\{v_1, v_2, \ldots, v_{i-1}\}$. Our main idea for proving upper bounds is to show that given an edge-twin-free graph G, any (inclusionwise) minimal edge-identifying code $\mathcal{C}_E$ induces a 2-degenerated subgraph of G and hence $|\mathcal{C}_E| \leq 2|V(G)| - 3$. Our proofs are constructive and one could build such small edge-identifying codes.

Theorem 4.34. Let G be an edge-twin-free graph and let $\mathcal{C}_E$ be a minimal edge-identifying code of G. Then G', the subgraph induced by $\mathcal{C}_E$, is 2-degenerated.

Proof. Let uv be an edge of G' with $d_{G'}(u), d_{G'}(v) \geq 3$. By minimality of $\mathcal{C}_E$ the subset $\mathcal{C}' = \mathcal{C}_E - uv$ of $E(G)$ is not an edge-identifying code of G. By the choice of uv, $\mathcal{C}'$ is still an edge-dominating set, thus there must be two edges, e_1 and e_2, that are not separated by $\mathcal{C}'$. Hence one of them, say e_1, is incident either to u or to v (possibly to both) and the other one (e_2) is incident to neither one.

We consider two cases: either $e_1 = uv$ or e_1 is incident to only one of u and v. In the first case, e_2 is adjacent to every edge of $\mathcal{C}'$ which uv is adjacent to. Since for each vertex of uv there are at least two edges in $\mathcal{C}'$ incident to this vertex, the subgraph induced by u, v and the vertices of e_2 must be isomorphic to K_4 and there should be no other edge of $\mathcal{C}'$ incident to any vertex of this K_4 (see Figure 4.8a).

In the other case, suppose e_1 is adjacent to uv at u. Let x and y be two neighbours of u in G' other than v. Then it follows that $e_2 = xy$ and, therefore, $d_{G'}(u) = 3$. Let z be the other end of e_1. We consider two subcases: either $z \notin \{x, y\}$, or, without loss of generality, $z = x$. Suppose $z \notin \{x, y\}$. Recall that uv is the only edge separating e_1 and e_2, but e_1 must be separated from ux. Thus $zy \in \mathcal{C}_E$. Similarly, e_1 must be separated from uy, so $zx \in \mathcal{C}_E$. Furthermore, $d_{G'}(x) = d_{G'}(y) = d_{G'}(z) = 2$ and $\{x, y, z, u\}$ induces a C_4 in G' (see Figure 4.8b). Now suppose $e_1 = ux$, since uv is the only edge separating e_1 and e_2, then uy and possibly xy are the only edges in G' incident to y, so $d_{G'}(y) \leq 2$ and $d_{G'}(u) = 3$ (see Figures 4.8c and 4.8d).

To summarize, we proved that given an edge uv, in a minimal edge-identifying code $\mathcal{C}_E$, we have one of the following cases.

- One of u or v is of degree at most 2 in G'.
- Edge uv is an edge of a connected component of G' isomorphic to K_4^- (that is K_4 with an edge removed), see Figure 4.8a.
- $d_{G'}(u) = 3$ (considering the symmetry between u and v) in which case either u is incident to a C_4 whose other vertices are of degree 2 in G' (Figure 4.8b), or to a vertex of degree 1 in G' (Figure 4.8c) or to a triangle with one vertex y of degree 2 in G' and y is not adjacent to v (Figure 4.8d).

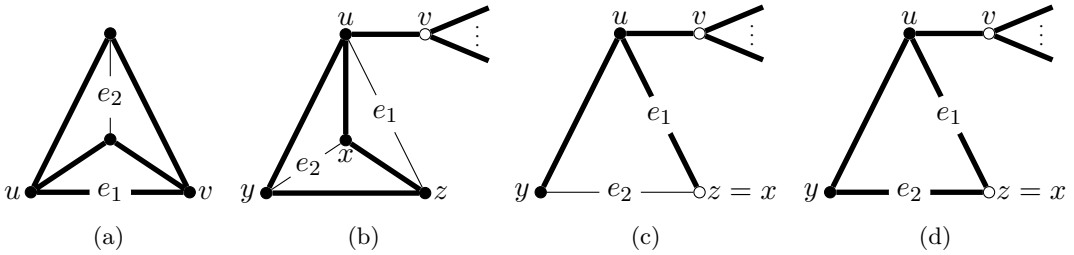


Figure 4.8: Case distinctions in the proof of Theorem 4.34. Black vertices have fixed degree in G'. Thick edges belong to $\mathcal{C}_E$.

In either case, there exists a vertex x of degree at most 2 in G' such that when x is removed, at least one of the vertices u, v has degree at most 2 in the remaining subgraph of G'. In this way we can define an order of elimination of the vertices of G' showing that G' is 2-degenerated. □

By further analysis of our proof we derive the following corollary, the upper bound $2n-3$ being much easier to prove:

Corollary 4.35. If G is an edge-twin-free graph on n vertices not isomorphic to K_4 or K_4^-, then $\gamma^{\text{EID}}(G) \leq 2n-5$.

Proof. We first prove that if G is an edge-twin-free graph on n vertices not isomorphic to K_4, then $\gamma^{\text{EID}}(G) \leq 2n-4$. Let $\mathcal{C}_E$ be a minimal edge-identifying code and let G' be the subgraph induced by $\mathcal{C}_E$. Then, by Theorem 4.34, G' is 2-degenerated. Let $v_n, v_{n-1}, \ldots, v_1$ be a sequence of vertices of G' obtained by a process of eliminating vertices of degree at most 2. Since v_1 and v_2 can induce at most a K_2, we notice that there could only be at most $2n-3$ edges in G'. Furthermore, if there are exactly $2n-3$ edges in G', then $v_1v_2 \in \mathcal{C}_E$ and each vertex v_i, $3 \leq i \leq n$, has exactly two neighbours in $\{v_1, \ldots, v_{i-1}\}$. Hence, the subgraph induced by $\{v_1, v_2, v_3, v_4\}$ is isomorphic to K_4^-. Considering symmetries, there are three possibilities for the subgraph induced by $\{v_1, \ldots, v_5\}$ (recall that v_5 is of degree 2 in this subgraph): see Figure 4.9 . In each of these three cases, the edge uv has both ends of degree at least 3. Thus, we can apply the argument used in the proof of Theorem 4.34 on G' and uv, showing that we have one of the four configurations of Figure 4.8. But none of them matches with the configurations of Figure 4.9, a contradiction.

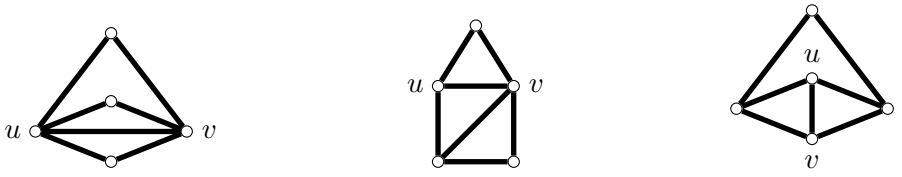


Figure 4.9: The three maximal 2-degenerated graphs on five vertices.

Now we show that if $\gamma^{\text{EID}}(G) = 2n-4$, then $G \cong K_4^-$. This can be easily checked if G has at most four vertices, so we may assume $n \geq 5$. Let G'' be the subgraph of G' induced by $\{v_1, v_2, v_3, v_4, v_5\}$. If G'' has seven edges, then it is isomorphic to one of the graphs of Figure 4.9, and we are done just like in the last case. Therefore, we can assume that G'' has exactly six edges and, since it is 2-degenerated, by an easy case analysis, it must be isomorphic to one of the graphs of Figure 4.10.

If G'' is a graph in part (i), (ii) or (iii) of Figure 4.10, then again one could repeat the arguments of the proof of Theorem 4.34 with G' and the edge uv of the corresponding figure, to obtain a contradiction.

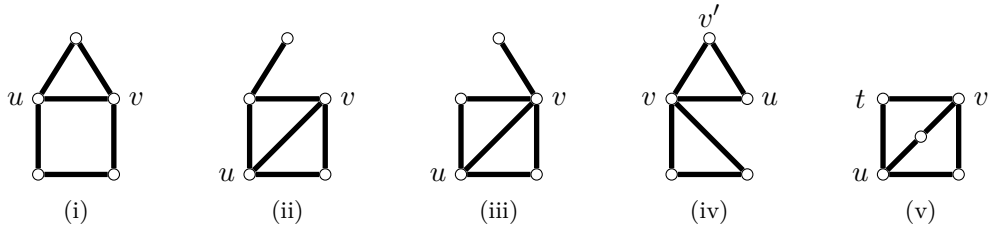


Figure 4.10: The five possibilities of 2-degenerated graphs on five vertices with six edges.

Suppose G'' is isomorphic to the graph of Figure 4.10(iv). Since G'' is not edge-twin-free, there must be at least one more vertex in G'. Let v_6 be as in the sequence obtained by 2-degeneracy of G'. Since G' has exactly $2n-4$ edges, v_6 must have exactly two neighbours in G''. By the symmetry of the four vertices of degree 2 in G'', we may assume $uv_6 \in \mathcal{C}_E$. Then u and v are both of degree at least 3 in G'. Therefore, we could again repeat the argument of Theorem 4.34 with G' and uv, where only one of the configurations of this theorem, namely 4.8d, matches G''. Furthermore, if this happens then $v'v_6$ should also be an edge of G'. Now u and v' are both of degree at least 3 and we apply the argument of Theorem 4.34 with G' and uv' to obtain a contradiction.

Finally, let G'' be isomorphic to the graph of Figure 4.10(v). We claim that every other vertex v_i ($i \geq 6$) is adjacent , in G', only to u and v. By contradiction suppose v_6 is adjacent to t. Then using the technique of Theorem 4.34 applied on G' and tu (respectively tv), we conclude that v_6 is adjacent to u (respectively v).

Since $|E(G')| = |\mathcal{C}_E| = 2n-4$, G' is a spanning subgraph of G. But then it is easy to verify that $\mathcal{C}_E \setminus \{xu, xv\}$ is an edge-identifying code of G — a contradiction. □

We note that $\gamma^{\text{EID}}(K_{2,n}) = 2n-2 = 2|V(K_{2,n})|-6$ thus this bound cannot be improved much.

Corollary 4.35 implies that Conjecture 4.4 holds for a large subclass of line graphs:

Corollary 4.36. If G is an edge-twin-free graph on n vertices and with average degree $\bar{d}(G) \geq 5$, then we have $\gamma^{\text{ID}}(\mathcal{L}(G)) \leq n - \frac{n}{\Delta(\mathcal{L}(G))}$.

Proof. Let u be a vertex of degree $d(u) \geq \bar{d}(G) \geq 5$. Since G is edge-twin-free there is at least one neighbour v of u that is of degree at least 2. Thus there is an edge uv in G with $d(u)+d(v) \geq \bar{d}(G)+2$ and, therefore, $\Delta(\mathcal{L}(G)) \geq \bar{d}(G)$. Hence, considering Corollary 4.35, it is enough to show that $2|V(G)| - 5 \leq |E(G)| - \frac{|E(G)|}{\bar{d}(G)}$.

To this end, since $\bar{d}(G) \geq 5$, we have $4|V(G)| \leq (\bar{d}(G)-1)|V(G)|$, therefore,

$$4|V(G)| - 10 \leq (\bar{d}(G)-1)|V(G)|$$

Mutiplying both sides by $\frac{\bar{d}(G)}{2}$ we have:

$$(2|V(G)|-5)\bar{d}(G) \leq (\bar{d}(G)-1)\frac{\bar{d}(G)}{2}|V(G)| = (\bar{d}(G)-1)|E(G)|$$

□

4.3 Complexity

This section is devoted to the study of the decision problem associated to the concept of identifying codes. We study it for both vertex- and edge-identifying codes.

4.3.1 Vertex-identification for split graphs

The IDCODE problem is defined as follows:

INSTANCE: A graph G and an integer k.

QUESTION: Does G have an identifying code of size at most k?

The 3-SAT problem is defined as follows:

INSTANCE: A collection $\mathcal{Q}$ of clauses over a set X of boolean variables, where each clause contains at most three distinct literals (a variable x_i or its negation $\overline{x_i}$).

QUESTION: Can $\mathcal{Q}$ be satisfied, *i.e.* is there a truth assignment of the variables of X such that each clause contains at least one true literal?

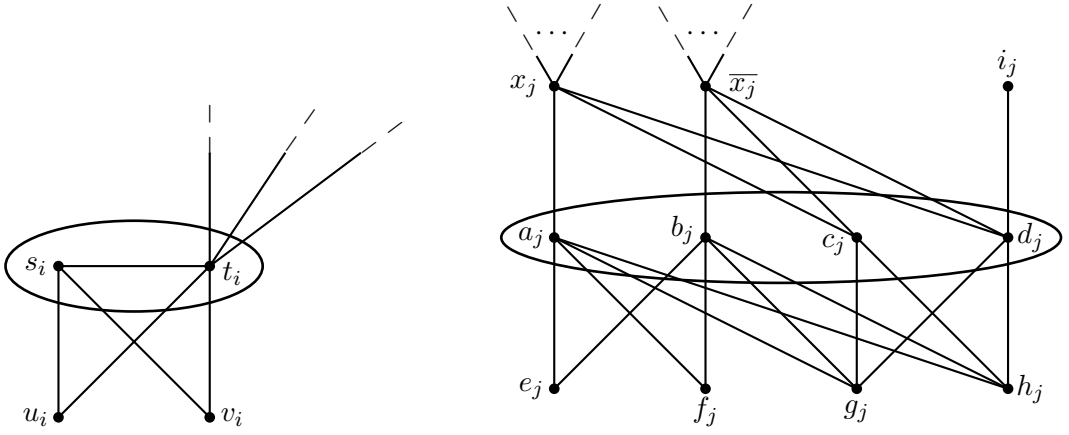
Theorem 4.37. IDCODE is NP-complete even when restricted to split graphs (and therefore to chordal graphs and complements of chordal graphs as well).

Proof. The problem is clearly in NP: given a subset of vertices of G, one can check in polynomial time whether it is an identifying code of G by computing and comparing the identifying sets of all the vertices of G.

We now reduce 3-SAT, which is known to be NP-hard, to IDCODE for split graphs.

Given an instance $\mathcal{Q} = \{C_1, \ldots, C_m\}$ of 3-SAT over the set of boolean variables $X = \{x_1, \ldots, x_n\}$, we build the split graph $G_\mathcal{Q}$ as follows:

For each variable x_j and clause C_i we build the subgraphs G_{x_j} and G_{C_i} respectively, as shown in Figure 4.11. For each clause $C_i = \{l_{i_1}, l_{i_2}, l_{i_3}\}$ we add the three edges $\{t_i, l_{i_1}\}, \{t_i, l_{i_2}\}, \{t_i, l_{i_3}\}$.



(a) Clause gadget (b) Variable gadget

Figure 4.11: Reduction gadgets for clause C_i and variable x_j. Circled vertices belong to the clique of the split graph.

The constructed graph $G_\mathcal{Q} = (V, E)$ is clearly split and twin-free. $|V| = 4m + 11n$ and therefore the construction is polynomial in $n + m$. Moreover, we set $k = 2m + 6n$ and we claim that $\mathcal{Q}$ is satisfiable if and only if $G_\mathcal{Q}$ has an identifying code of size k.

On the one hand, suppose $\mathcal{Q}$ is satisfiable. We build the identifying code $\mathcal{I}$ as follows: for each clause C_i let $s_i, u_i \in \mathcal{I}$. For each variable x_j let $c_j, f_j, h_j, i_j \in \mathcal{I}$. If x_j is true then $a_j, x_j \in \mathcal{I}$. Otherwise, $\overline{x_j}, b_j \in \mathcal{I}$. Now, using the fact that $\mathcal{Q}$ is satisfiable and assuming that $\mathcal{Q}$ contains at least one clause and one variable, one can check that $\mathcal{I}$ is an identifying code of $G_\mathcal{Q}$ of size k.

On the other hand, suppose $\mathcal{I}$ is an identifying code of $G_\mathcal{Q}$ of size at most k. Note that for each subgraph G_{C_i}, $|V(G_{C_i}) \cap \mathcal{I}| \geq 2$. Indeed, to separate u_i from v_i, one of these two vertices, say u_i, must belong to $\mathcal{I}$. Now, if no other vertex of $V(G_{C_i})$ is in the code, v_i is not dominated. Note also that for each subgraph G_{x_j}, in order to separate c_j from d_j, i_j must belong to $\mathcal{I}$.

Moreover, necessarily $|\{a_j, b_j, c_j, d_j, e_j, f_j, g_j, h_j\} \cap \mathcal{I}| \geq 4$. Indeed, similarly to the case of G_{C_i}, one of $\{e_j, f_j\}$ (say e_j) as well as one of $\{g_j, h_j\}$ (say g_j) must belong to $\mathcal{I}$. In order to dominate and separate f_j and h_j, one can see that two additional vertices of $\{a_j, b_j, c_j, d_j, e_j, f_j, g_j, h_j\}$ are required. Now, in order to separate a_j from b_j, either x_j or $\overline{x_j}$ must belong to $\mathcal{I}$. Hence we already fixed $k - n$ vertices in $\mathcal{I}$, *i.e.* $|(\bigcup_{i=1}^{m} V(G_{C_i}) \cup \bigcup_{j=1}^{n} (V(G_{x_j}) \setminus \{x_j, \overline{x_j}\})) \cap \mathcal{I}| \geq 2m + 5n$. Therefore, by the "pigeonhole principle", for each subgraph G_{x_j}, exactly one of $\{x_j, \overline{x_j}\}$ belongs to $\mathcal{I}$. Let us build the following truth assignment of the variables of X: for each variable x_j, if vertex $x_j \in \mathcal{I}$, $x_j \leftarrow$ "true"; otherwise $x_j \leftarrow$ "false". Finally, since $\mathcal{I}$ is an identifying code, for each subgraph G_{C_i}, s_i and t_i are separated in $G_{\mathcal{Q}}$ by at least one of the three neighbours of t_i. Therefore this neighbour belongs to $\mathcal{I}$. Hence, at least one literal of C_i has been set to "true" and $\mathcal{Q}$ is satisfiable. □

Note that 3-SAT is known to remain NP-hard even when each variable appears at most three times (as itself or its negation) and each literal appears at most two times in the formula [99]. Therefore, in the previous proof, one can do the reduction from this restricted version of 3-SAT. Let G be the constructed split graph with $V(G) = K \cup S$, $K \cup S$ being the decomposition of $V(G)$ into a clique K and a stable set S. It follows from the construction that a vertex from K has at most five neighbours in S, and a vertex of S has at most five neighbours in K. Hence we can strengthen the theorem as follows:

Corollary 4.38. IDCODE is NP-complete even when restricted to split graphs whose vertex set V can be partitioned into a clique K and a stable set S such that a vertex of K has at most five neighbours in S, and a vertex of S has at most five neighbours in V.

4.3.2 Edge-identification

The EDGE-IDCODE problem is defined as follows:
INSTANCE: A graph G and an integer k.
QUESTION: Does G have an edge-identifying code of size at most k?

We will prove that EDGE-IDCODE is NP-hard in some restricted class of graphs by reduction from PLANAR $(\leq 3, 3)$-SAT, which is a variant of the SAT problem and is defined as follows [32]:

PLANAR $(\leq 3, 3)$-SAT
INSTANCE: A collection $\mathcal{Q}$ of clauses over a set X of boolean variables, where each clause contains at least two and at most three distinct literals (a variable x or its negation $\overline{x}$). Moreover, each variable appears in exactly three clauses: twice in its non-negated form, and once in its negated form. Finally, the bipartite incidence graph of $\mathcal{Q}$, denoted $B(\mathcal{Q})$, is planar ($B(\mathcal{Q})$ has vertex set $\mathcal{Q} \cup X$ and $Q \in \mathcal{Q}$ is adjacent to $x \in X$ if x or $\overline{x}$ appears in clause Q).
QUESTION: Can $\mathcal{Q}$ be satisfied, i.e. is there a truth assignment of the variables of X such that each clause contains at least one true literal?

PLANAR $(\leq 3, 3)$-SAT is known to be NP-complete [32]. We are now ready to prove the main result of this subsection.

Theorem 4.39. EDGE-IDCODE is NP-complete even when restricted to bipartite planar graphs of maximum degree 3 and arbitrarily large girth.

Proof. The problem is clearly in NP: given a subset $\mathcal{C}$ of edges of G, one can check in polynomial time whether it is an edge-identifying code of G by computing the sets $\mathcal{C} \cap N[e]$ for each edge e and comparing them pairwise.

We now reduce PLANAR $(\leq 3, 3)$-SAT to EDGE-IDCODE. We first give the proof for the case of girth 8 and show that it can be easily extended to an arbitrarily large girth.

We first need to define a generic sub-gadget (denoted P-gadget) that will be needed for the reduction. In order to have more compact figures, we will use the representation of this gadget as drawn in Figure 4.12. We will say that a P-gadget is *attached* at some vertex x if x is incident to edge a of the gadget as depicted in the figure.

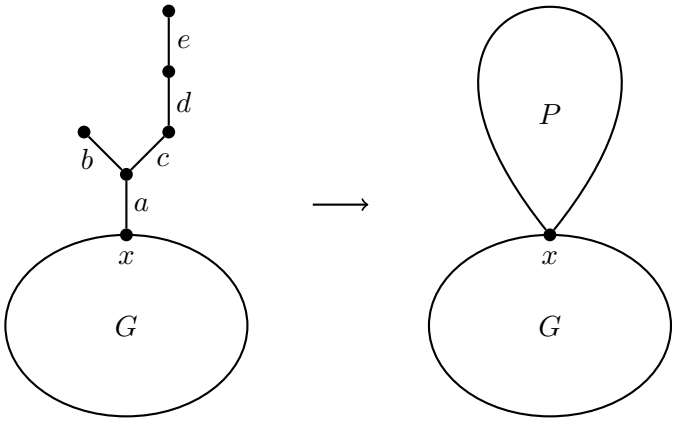


Figure 4.12: The generic P-gadget

We make the following claims.

Claim 4.40. In any graph containing a P-gadget, at least three edges of this gadget must belong to any edge-identifying code.

Claim 4.40 is true because d is the only edge separating b and c. Similarly c is the only edge separating d and e. Finally, in order to separate d and c, one has to take at least one of a, b or e.

Claim 4.41. If G is an edge-twin-free graph obtained from a graph H with a P-gadget attached at a vertex x of H, then any edge-identifying code of G must contain an edge of H incident to x.

Claim 4.41 follows from the fact that edge a must be separated from edge b.

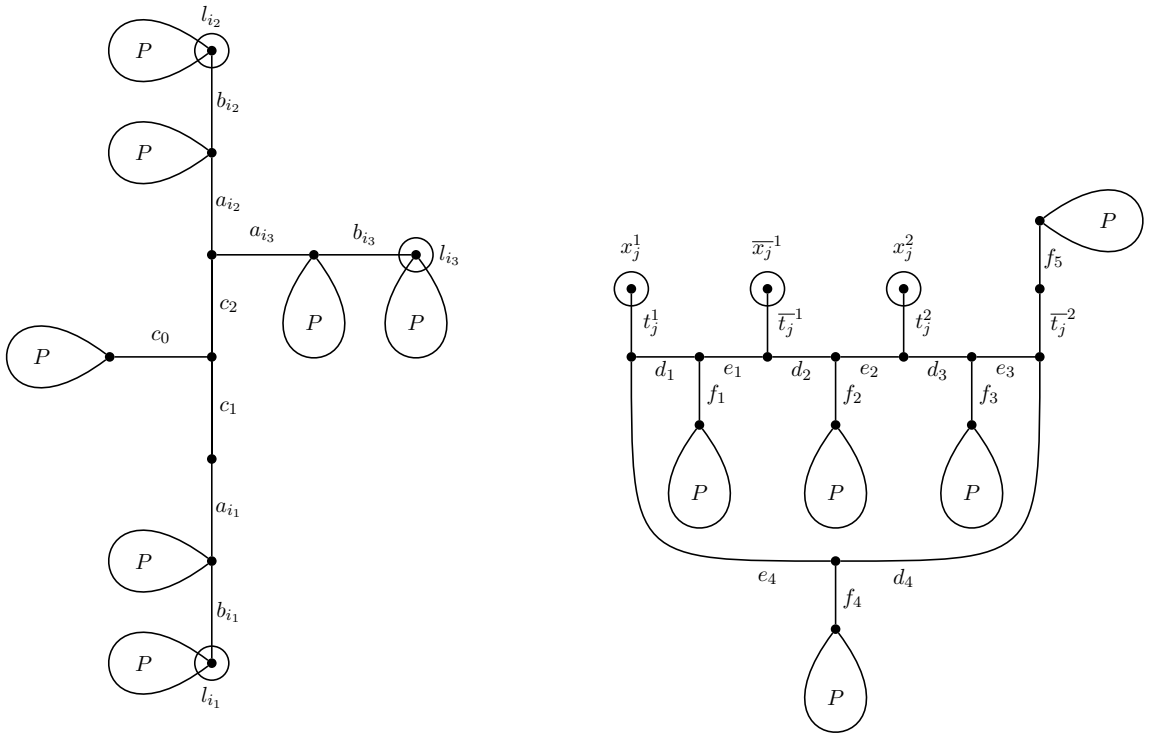
We are now ready to describe the reduction.

Given an instance $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ of PLANAR $(\leq 3, 3)$-SAT over the set of boolean variables $X = \{x_1, \ldots, x_n\}$ together with an embedding of its bipartite incidence graph $B(\mathcal{Q})$ in the plane, we build the graph $G_{\mathcal{Q}}$ as follows.

For each variable x_j and clause Q_i we build the subgraphs G_{x_j} and G_{Q_i} respectively, as shown in Figure 4.13. We recall that a given variable x_j appears in positive form in exactly two clauses, say Q_p, Q_q, and in negative form in exactly one clause, say Q_r. We then unify vertex x_j^1 of G_{x_j} with vertex l_{p_k} of G_{Q_p} which corresponds to x_j . We do a similar unification for vertices x_j^2 and $\overline{x_j}^1$ with corresponding vertices from G_{Q_q} and G_{Q_r}. This can be done while ensuring the planarity of $G_{\mathcal{Q}}$, using the given planar embedding of $B(\mathcal{Q})$. Moreover, $G_{\mathcal{Q}}$ is bipartite since $B(\mathcal{Q})$ is bipartite and there are no odd cycles in the variable and clause gadgets. Finally, it is easy to see that $G_{\mathcal{Q}}$ has maximum degree 3 and girth 8. Since a clause gadget has fourty-five vertices and a variable gadget, fourty-two vertices, $G_{\mathcal{Q}}$ has $45m + 42n$ vertices and, therefore, the construction has polynomial size in terms of the size of $\mathcal{Q}$.

We will need two additional claims in order to complete the proof.

Claim 4.42. In a variable gadget G_{x_j}, in order to separate the four pairs of edges $\{d_i, e_i\}$ for $1 \leq i \leq 4$, at least two edges of $A = \{d_i, e_i \mid 1 \leq i \leq 4\} \cup \{t_j^1, \overline{t_j}^1, t_j^2, \overline{t_j}^2\}$ belong to any edge-identifying code $\mathcal{C}$. Moreover, if $|\mathcal{C} \cap A| = 2$, then either $\mathcal{C} \cap A = \{t_j^1, t_j^2\}$ or $\mathcal{C} \cap A = \{\overline{t_j}^1, \overline{t_j}^2\}$.



(a) Clause gadget G_{Q_i}

(b) Variable gadget G_{x_j}

Figure 4.13: Reduction gadgets for clause Q_i and variable x_j.

The first part of Claim 4.42 follows from the fact that the two edges of each of the pairs $\{d_1, e_1\}$ and $\{d_3, e_3\}$ must be separated. The second follows from an easy case analysis.

The following claim follows directly from Claim 4.41.

Claim 4.43. Let $v_1v_2v_3v_4$ be a path of four vertices where each of the vertices v_2 and v_3 has its own P-gadget attached and both v_2 and v_3 have degree 3. Then, at least one of the three edges of the path belong to any identifying code of the graph. If exactly one belongs to a code, it must be v_2v_3.

We now claim that $\mathcal{Q}$ is satisfiable if and only if $G_{\mathcal{Q}}$ has an edge-identifying code of size at most $k = 25m + 22n$.

For the sufficient side, given a truth assignment of the variables satisfying $\mathcal{Q}$, we build an edge-identifying code $\mathcal{C}$ as follows. For each P-gadget, edges a, c, d are in $\mathcal{C}$. For each clause gadget G_{Q_i}, edge c_0 belongs to $\mathcal{C}$. For each literal l_{i_k} of Q_i, $1 \leq k \leq 3$, if l_{i_k} is true, edge a_{i_k} belongs to $\mathcal{C}$; otherwise, edge b_{i_k} belongs to $\mathcal{C}$. If Q_i has only two literals and vertex l_{i_k} is the vertex not corresponding to a literal of Q_i, then edge b_{i_k} belongs to $\mathcal{C}$. Now, one can see that all edges of G_{Q_i} are dominated. Furthermore, all pairs of edges of G_{Q_i} are separated. This can be easily seen for all pairs besides $\{c_1, c_2\}$. For this pair, since we are considering a satisfying assignment of $\mathcal{Q}$, in every clause Q_i of $\mathcal{Q}$, there exists a true literal. Hence, for each clause Q_i, at least one edge a_{i_j} with $1 \leq j \leq 3$, must be in the code and, therefore, the pair $\{c_1, c_2\}$ is separated. Next, in each variable gadget G_{x_j}, if x_j is true, edges t_j^1 and t_j^2 belong to $\mathcal{C}$. Otherwise, edges $\overline{t_j}^1$ and $\overline{t_j}^2$ belong to $\mathcal{C}$. Edges f_1, f_2, f_3, f_4 and f_5 also belong to $\mathcal{C}$. Because of this choice, all edges of $G_{x_j} \setminus \{t_j^1, t_j^2, \overline{t_j}^1\}$ are dominated. Since each of the three edges $t_j^1, t_j^2, \overline{t_j}^1$ is incident to a vertex of a P-gadget of some clause gadget, they are also dominated. Moreover, all pairs of edges containing at least one edge of $G_{x_j} \setminus \{t_j^1, t_j^2, \overline{t_j}^1\}$ are clearly separated. Now, since for each

P-gadget of the clause gadgets, edge a is in $\mathcal{C}$, $t_j^1, t_j^2, \overline{t_j}^1$ are separated from all edges in $G_\mathcal{Q}$. It remains to prove that $\mathcal{C}$ separates the pair $\{c_1, c_2\}$ of each clause gadget. Since we are considering a satisfying assignment of $\mathcal{Q}$, in every clause Q_i of $\mathcal{Q}$, there exists a true literal. Hence, for each clause Q_i, at least one edge a_{i_j} with $1 \leq j \leq 3$, must be in the code and, therefore, the pair $\{c_1, c_2\}$ is separated. We conclude that $\mathcal{C}$ is an edge-identifying code of size k.

For the necessary side, let $\mathcal{C}'$ be an edge-identifying code of $G_\mathcal{Q}$ with $|\mathcal{C}'| \leq k$. It follows from Claim 4.40 that three edges of each of the seven P-gadgets of a clause gadget G_{Q_i} must belong to $\mathcal{C}'$. Moreover, by Claim 4.41, edge c_0 is forced to be in any code. Finally, by Claim 4.41, for each vertex l_{i_k} $(1 \leq k \leq 3)$ of G_{Q_i}, one of the edges a_{i_k} and b_{i_k} is in $\mathcal{C}'$.

Note that this is a total of at least twenty-five edges per clause gadget.

Similarly, it follows from Claim 4.40 that in each variable gadget G_{x_j}, at least fifteen edges of $\mathcal{C}'$ are contained in the P-gadgets of G_{x_j}. Following Claim 4.41, all edges f_i $(1 \leq i \leq 5)$ belong to $\mathcal{C}'$. Note that this is a total of at least twenty edges in each variable gadget. We have considered $25m + 20n$ edges of $\mathcal{C}'$ so far. Hence $2n$ edges remain to be considered. It follows from Claim 4.42 that for each variable gadget, at least two additional edges belong to $\mathcal{C}'$ (in order to separate the pairs $\{d_i, e_i\}$, for $1 \leq i \leq 4$). Therefore, in each variable gadget, exactly two of these edges belong to $\mathcal{C}'$. Hence, following the second part of Claim 4.42, either $\{t_j^1, t_j^2\}$ or $\{\overline{t_j}^1, \overline{t_j}^2\}$ is a subset of $\mathcal{C}'$.

Remark that we have now considered all $k = 25m + 22n$ edges of $\mathcal{C}'$. Therefore, in each clause gadget G_{Q_i}, *exactly* one of the edges a_{i_k} and b_{i_k} of G_{Q_i} belongs to $\mathcal{C}'$.

We can now build the following truth assignment: for each variable gadget, if $\{t_j^1, t_j^2\}$ is a subset of $\mathcal{C}'$, x_j is set to TRUE. Otherwise, $\{\overline{t_j}^1, \overline{t_j}^2\}$ is a subset of $\mathcal{C}'$ and x_j is set to FALSE. Let us prove that this assignment satisfies $\mathcal{Q}$.

In each clause gadget G_{Q_i}, note that edges c_1 and c_2 must be separated by $\mathcal{C}'$; this means that one edge a_{i_k} from $\{a_{i_1}, a_{i_2}, a_{i_3}\}$ belongs to $\mathcal{C}'$. Hence, as noted in the previous paragraph, $b_{i_k} \notin \mathcal{C}'$ and by Claim 4.43, in the path formed by edges $\{a_{i_k}, b_{i_k}, t_j^1\}$, t_j^1 belongs to the code (without loss of generality, we suppose that $l_{i_k} = x_j$ and t_j^1 is the edge of G_{x_j} incident to vertex l_{i_k} of G_{Q_i}). Therefore, in the constructed truth assignment, literal l_{i_k} has value TRUE, and the clause is satisfied. Repeating this argument for each clause shows that the formula is satisfied.

Now, it remains to show that similar arguments can be used to prove the final statement of the theorem for larger girth. Consider some integers $\lambda \geq 1$ and $\mu \geq 2$. We build the graph $G_\mathcal{Q}(\lambda, \mu)$ using modified variable gadgets $G_{x_j}(\mu)$ and modified clause gadgets $G_{Q_i}(\lambda)$, which are depicted in Figure 4.14. The construction is the same as in the previous proof and $G_\mathcal{Q}(\lambda, \mu)$ has $(36\lambda + 9)m + (30\mu - 18)n$ vertices. We claim that the girth of $G_\mathcal{Q}(\lambda, \mu)$ is now at least $\min\{4\mu, 8(\lambda+1)\}$. Indeed, $G_{x_j}(\mu)$ has a cycle of size exactly 4μ and since the girth of $B(\mathcal{Q})$ is at least 4, it follows that the minimum length of a cycle between some clause gadgets (at least two) and some variable gadgets (at least two) is at least $4(2\lambda + 1) + 2 + 2 = 8(\lambda + 1)$.

Now, using a similar proof as the proof for girth 8, it can be shown that $\mathcal{Q}$ is satisfiable if and only if $G_\mathcal{Q}(\lambda, \mu)$ has an identifying code of size at most $k = (21\lambda + 4)m + (17\mu - 12)n$. □

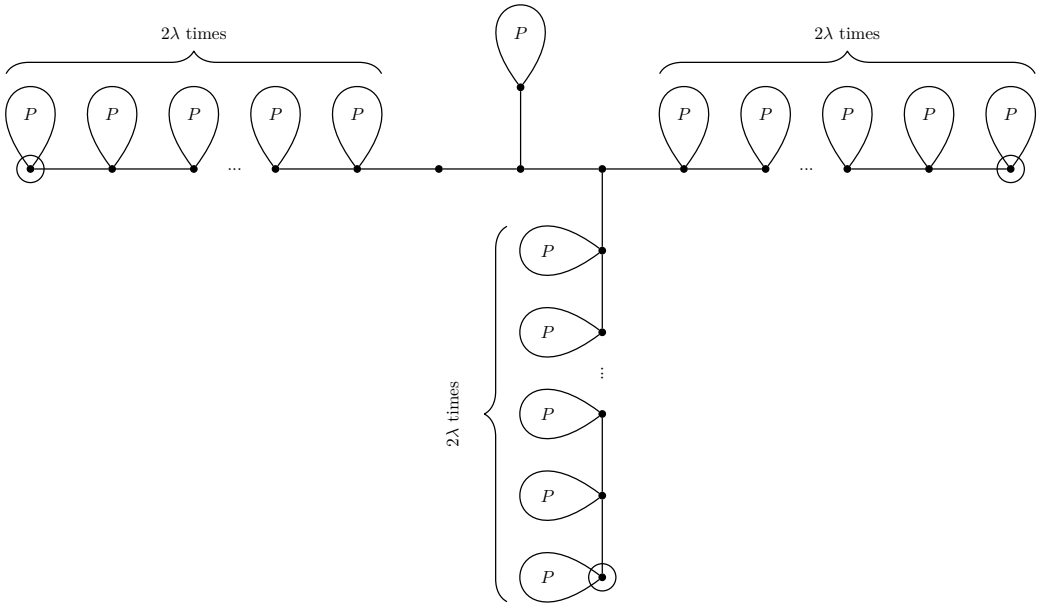
It is known that a line graph $L(G)$ is perfect if and only if G has no odd cycles of length more than 3, see [97]. Moreover, one can check that the line graphs of the graphs constructed in the previous proof are planar, have maximum degree 4 and clique number 3. Therefore, the following corollary follows:

Corollary 4.44. IDCODE is NP-complete even when restricted to perfect 3-colourable planar line graphs of maximum degree 4.

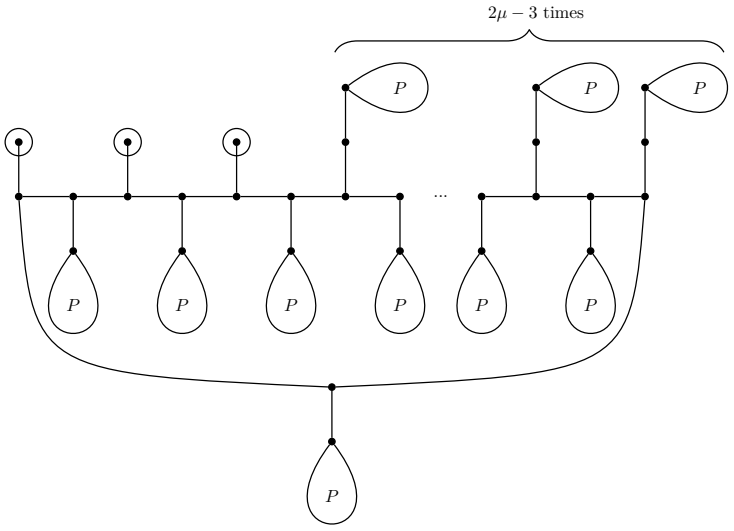
In the following, by slightly restricting the class of graphs considered in Theorem 4.39, we show that EDGE-IDCODE becomes linear-time solvable in this restricted class.

Let us first introduce some necessary concepts.

A graph property $\mathcal{P}$ is expressable in *counting monadic second-order logic*, CMSOL for short (see [30] for further reference), if $\mathcal{P}$ can be defined using:



(a) Clause gadget $G_{Q_i}(\lambda)$



(b) Variable gadget $G_{x_j}(\mu)$

Figure 4.14: Reduction gadgets for clause Q_i and variable x_j for arbitrarily large girth.

- vertices, edges, sets of vertices and sets of edges of a graph
- the binary adjacency relation *adj* where $adj(u, v)$ holds if and only if u, v are two adjacent vertices
- the binary incidence relation *inc*, where $inc(v, e)$ holds if and only if edge e is incident to vertex v
- the equality operator $=$ for vertices and edges
- the membership relation $\in$, to check whether an element belongs to a set
- the unary cardinality operator *card* for sets of vertices

- the logical operators OR, AND, NOT (denoted by $\vee$, $\wedge$, $\neg$)
- the logical quantifiers $\exists$ and $\forall$ over vertices, edges, sets of vertices or sets of edges

It has been shown that CMSOL is particularly useful when combined with the concept of the graph parameter *tree-width* (we refer the reader to [30] for a definition). Some important classes of graphs have bounded tree-width. For example, trees have tree-width 1, series-parallel graphs have tree-width 2 and outerplanar graphs have tree-width 3.

The following result shows that many graph properties can be checked in linear time for graphs of bounded tree-width.

Theorem 4.45 (Courcelle [30])**.** Let $\mathcal{P}$ be a graph property expressible in CMSOL and let c be a constant. Then, for any graph G of tree-width at most c, it can be checked in linear time whether G has property $\mathcal{P}$.

We now show that CMSOL can be used in the context of edge-identifying codes:

Proposition 4.46. Given a graph G and an integer k, let $\mathcal{EID}(G,k)$ be the property that $\gamma^{\text{EID}}(G) \leq k$. Property $\mathcal{EID}(G,k)$ can be expressed in CMSOL.

Proof. Let $V = V(G)$ and $E = E(G)$. We define the CMSOL relation $dom(e,f)$ which holds if and only if e, f are edges of E and e, f dominate each other, i.e. e and f are incident to the same vertex. We have $dom(e,f) := \exists x \in V, (inc(x,e) \wedge inc(x,f))$.

Now we define $\mathcal{EID}(G,k)$ as follows:

$$\begin{aligned}\mathcal{EID}(G,k) := \exists \mathcal{C}, \mathcal{C} \subseteq E, card(\mathcal{C}) \leq k, (\forall e \in E, \exists f \in \mathcal{C}, dom(e,f)) \wedge \\ \Big(\forall e \in E, \forall f \in E, e \neq f, \exists g \in \mathcal{C}, ((dom(e,g) \wedge \neg dom(f,g)) \vee \\ (dom(f,g) \wedge \neg dom(e,g)))\Big)\end{aligned}$$

□

This together with Theorem 4.45 implies the following corollary.

Corollary 4.47. EDGE-IDCODE can be solved in linear time for all classes of graphs having their tree-width bounded by a constant.

This result implies, in particular, that one can find the edge-identifying code number of a tree in linear time. Note that a similar approach has been used for the case of vertex-identifying codes in [81]. The proof of Theorem 4.45 is constructive and gives a linear-time algorithm, but it is very technical. Therefore, it would be interesting to give a simpler linear-time algorithm for EDGE-IDCODE in trees. Observe that this has been done in [3] for the case of vertex-identifying codes.

4.4 Open problems

In this chapter we studied identifying codes in different classes of graphs. In particular, in Section 4.1 we gave a characterization of graphs having the maximum identifying code number at most $|V(G)| - 1$. Proving the upper bound of Conjecture 4.4 remains a challenging problem.

As we mentioned in the introduction, the identifying code problem is NP-hard in the class of planar graphs [3]. Thus finding some bounds on the identifying code number becomes even more motivating. In terms of the upper bounds, since the star $K_{1,t}$ is a planar graph which achieves the bound $|V(G)| - 1$, this bound cannot be improved in the case of planar graphs. But what about lower bounds? In [94] it is proved that for every planar graph $\gamma^{\text{LD}}(G) \geq \frac{n+10}{7}$ and thus

if G is twin-free, then $\gamma^{\text{ID}}(G) \geq \frac{n+10}{7}$. Is it possible to improve this lower bound in the class of planar graphs?

From the computational point of view, we proved that the problem of finding the minimum identifying code of some restricted families of perfect graphs is NP-hard. One of the intriguing open questions remains the following:

Question 4.48. What is the complexity of the problem for the class of interval graphs?

It is known that the dominating set problem is linear time solvable when the input is an interval graph [13] and even when considered different versions of weighted dominating sets [84]. However, for identifying codes no complexity result has been found until now. Naturally, a next step in the study of the algorithmic part of the identifying code problem, could be the class of interval graphs.

As regards the weighted version of the problem, one could investigate the polyhedral approach. We note that there is some ongoing work on this [2].

Chapter 5

Conclusion

In this thesis, we focused on the study of three problems: orthogonal packing (Chapter 2), strong edge-colouring (Chapter 3) and identifying codes (Chapter 4). The core subject of our work includes the algorithmic and complexity issues of these problems. For OPP we designed an exact algorithm exploiting the nice properties of MPQ-trees, whereas in the case of strong edge-colouring and identifying codes we addressed the corresponding decision problems: are they polynomial or NP-hard for various classes of graphs?

We proved that STRONG k-EDGE-COLOURING remains NP-hard even within very restricted subclasses of planar graphs. Similarly, we established that IDCODE remains NP-hard in various restricted subclasses of perfect graphs. However, the hardness of these two problems does not seem to be of the same nature. For example, when the input graph is a chordal graph, STRONG k-EDGE-COLOURING is polynomial whereas IDCODE is NP-hard (c.f. page 73). This asymmetry deserves to be investigated.

As of chordal graphs, split graphs and interval graphs are two of its well-known subclasses. For split graphs, we have shown that IDCODE is still NP-hard. Interestingly, for the case of interval graphs the IDCODE complexity is not known till date. Moreover, we do not even know a tight lower bound of the identifying code number for this family of graphs while many graphs having $\gamma^{\text{ID}}(G) = |V(G)| - 1$ described in Section 4.1.2 of Chapter 4, are interval graphs.

Given that we have studied interval graphs in the context of orthogonal packing using the MPQ-tree data structure, a natural line of work would be to look at identifying codes in interval graphs using MPQ-trees. One could arguably say that MPQ-trees encode interval graphs efficiently by capturing all possible orders of the maximal cliques. Intuitively, the fact that each branch of an MPQ-tree is distinct from the others somehow models the symmetric differences between the vertices. This makes us think that one "efficient" technique for computing the minimum identifying code could be obtained by exploiting this information. Additionally, one could check, if it is possible to derive lower and upper bounds for this parameter using the MPQ-tree data-structure.

Similar to STRONG k-EDGE-COLOURING, deciding whether a graph can be edge-coloured using k colours is known to be NP-hard [60]. But a striking fact is that the *fractional* version of the edge-colouring problem can be solved in polynomial time [83]. It would therefore be interesting to examine the fractional version of strong edge-colouring too. In this version, instead of assigning one colour per edge, sets of colours of a given size are assigned. The constraint is that edges at distance at most 2 must have disjoint sets of colours. Given a graph G, let us denote by b the size of the sets of colours of each edge and by $\chi'_{sb}(G)$ the minimum number of colours needed to obtain a fractional strong edge-colouring with sets of size b. The question to study would be then: what is the smallest value of the *fractional strong chromatic index* $\chi'_{sf}(G) = \lim_{b\to\infty} \frac{\chi'_{sb}(G)}{b}$? Equivalently, fractional strong edge-colouring is just a linear relaxation of the strong edge-colouring integer

linear program given in the introduction chapter.

At last, we would like to mention that by identifying two opposite boundaries of the container, the orthogonal packing problem would be turned into a "circular" orthogonal packing problem, suitable to modelize periodic scheduling problems. Intuitively, the feasible solutions could be characterized by circular interval graphs. However, these graphs are in general not perfect, and therefore one of the basic arguments used by Fekete & Schepers would not be available any more. We believe that the techniques we used in our algorithm could be adapted with a limited overhead.

References

[1] L. D. Andersen. The strong chromatic index of a cubic graph is at most 10. *Discrete Mathematics*, 108(1–3):231–252, 1992.

[2] G. Argiroffo, S. Bianchi, and A. Wagler. The identifying code polyhedron of cycles. To be presented at ISMP - International Symposium on Mathematical Programming, Berlin, 2012.

[3] D. Auger. Minimal identifying codes in trees and planar graphs with large girth. *European Journal of Combinatorics*, 31(5):1372–1384, 2010.

[4] D. Auger, I. Charon, O. Hudry, and A. Lobstein. Complexity results for identifying codes in planar graphs. *International Transactions in Operational Research*, 17(6):691–710, 2010.

[5] R. Baldacci and M. A. Boschetti. A cutting plane approach for the two-dimensional orthogonal non-guillotine cutting stock problem. *European Journal of Operational Research*, 183(3):1136–1149, 2007.

[6] C. L. Barrett, G. Istrate, V. S. A. Kumar, M. V. Marathe, S. Thite, and S. Thulasidasan. Strong edge coloring for channel assignment in wireless radio networks. In *Pervasive Computing and Communications Workshops, 2006. PerCom Workshops 2006. Fourth Annual IEEE International Conference on*, pages 5–110, March 2006.

[7] J. Beasley and A. Mingozzi. A new formulation for the two-dimensional orthogonal cutting stock problem. Technical report, University of Bologna, 2003.

[8] J. E. Beasley. Algorithms for unconstrained two-dimensional guillotine cutting. *Journal of the Operational Research Society*, 36(4):297–306, 1985.

[9] J. E. Beasley. An exact two-dimensional non-guillotine cutting tree search procedure. *Operations Research*, 33(1):49–64, 1985.

[10] L. W. Beineke. Characterizations of derived graphs. *Journal of Combinatorial Theory*, 9(2):129–135, 1970.

[11] N. Bertrand. Codes identifiants et codes localisateurs-dominateurs sur certains graphes. Master's thesis, ENST, Paris, France, June 2001.

[12] J. A. Bondy. Induced subsets. *Journal of Combinatorial Theory, Series B*, 12(2):201–202, 1972.

[13] K. S. Booth and J. H. Johnson. Dominating sets in chordal graphs. *SIAM Journal on Computing*, 11(1):191–199, 1982.

[14] K. S. Booth and G. S. Lueker. Linear algorithms to recognize interval graphs and test for the consecutive ones property. In *Proceedings of the seventh Annual ACM Symposium on Theory of Computing (STOC'75)*, pages 255–265, 1975.

[15] K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976.

[16] M. A. Boschetti, A. Mingozzi, and E. Hadjiconstantinou. New upper bounds for the two-dimensional orthogonal non-guillotine cutting stock problem. *IMA Journal of Management Mathematics*, 13(2):95–119, 2002.

[17] K. Cameron. Induced matchings. *Discrete Applied Mathematics*, 24(1-3):97–102, 1989.

[18] A. Caprara and M. Monaci. On the two-dimensional knapsack problem. *Operations Research Letters*, 32(1):5–14, 2004.

[19] G. J. Chang, H.-B. Chen, M. Montassier, A. Pêcher, and A. Raspaud. Strong chromatic index of planar graphs with large girth. private communication, 2012.

[20] G. J. Chang and D. D.-F. Liu. Strong edge-coloring for cubic halin graphs. *Discrete Mathematics*, 312(8):1468–1475, 2012.

[21] E. Charbit, I. Charon, G. Cohen, O. Hudry, and A. Lobstein. Discriminating codes in bipartite graphs: bounds, extremal cardinalities, complexity. *Advances in Mathematics of Communications*, 2(4):403–420, 2008.

[22] I. Charon, G. Cohen, O. Hudry, and A. Lobstein. New identifying codes in the binary hamming space. *European Journal of Combinatorics*, 31(2):491–501, 2010.

[23] I. Charon, I. Honkala, A. Lobstein, and G. Zémor. On identifying codes. *Proceedings of the DIMACS Workshop on Codes and Association Schemes '99*, 56:97–109, 2001.

[24] I. Charon, O. Hudry, and A. Lobstein. Minimizing the size of an identifying or locating-dominating code in a graph is NP-hard. *Theoretical Computer Science*, 290(3):2109–2120, 2003.

[25] I. Charon, O. Hudry, and A. Lobstein. Extremal cardinalities for identifying and locating-dominating codes in graphs. *Discrete Mathematics*, 307(3-5):356–366, 2007.

[26] N. Christofides and E. Hadjiconstantinou. An exact algorithm for orthogonal 2-d cutting problems using guillotine cuts. *European Journal of Operational Research*, 83(1):21–38, 1995.

[27] F. R. K. Chung, A. Gyárfás, Z. Tuza, and W. T. Trotter. The maximum number of edges in $2K_2$-free graphs of bounded degree. *Discrete Mathematics*, 81(2):129–135, 1990.

[28] F. Clautiaux, J. Carlier, and A. Moukrim. A new exact method for the orthogonal packing problem. *European Journal of Operational Research*, 183(3):1196–1211, 2007.

[29] N. Cohen. *Three years of graphs and music : some results in graph theory and its applications.* Thesis, Université de Nice Sophia-Antipolis, October 2011.

[30] B. Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.

[31] D. W. Cranston. Strong edge-coloring of graphs with maximum degree 4 using 22 colors. *Discrete Mathematics*, 306(21):2772–2778, 2006.

[32] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

[33] G. A. Dirac. On rigid circuit graphs. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 25:71–76, 1961.

[34] W. Duckworth, D. F. Manlove, and M. Zito. On the approximability of the maximum induced matching problem. *Journal of Discrete Algorithms*, 3(1):79–91, 2005.

[35] P. Erdős. Problems and results in combinatorial analysis and graph theory. *Discrete Mathematics*, 72(1–3):81–92, 1988.

[36] J. Erickson, S. Thite, and D. P. Bunde. Distance-2 edge coloring is NP-complete.

[37] R. J. Faudree, A. Gyárfas, R. H. Schelp, and Zs. Tuza. Induced matchings in bipartite graphs. *Discrete Mathematics*, 78(1-2):83–87, 1989.

[38] R. J. Faudree, A. Gyárfas, R. H. Schelp, and Zs. Tuza. The strong chromatic index of graphs. *Ars Combinatoria*, 29(B):205–211, 1990.

[39] S. P. Fekete and J. Schepers. On more-dimensional packing I: Modeling. Technical report, University of Köln, Germany, 1997.

[40] S. P. Fekete and J. Schepers. A combinatorial characterization of higher-dimensional orthogonal packing. *Mathematics of Operations Research*, 29(2):353–368, 2004.

[41] S. P. Fekete, J. Schepers, and J. van der Veen. An exact algorithm for higher-dimensional orthogonal packing. *Operations Research*, 55(3):569–587, 2007.

[42] E. P. Ferreira and J. F. Oliveira. Fekete and Schepers' graph-based algorithm for the two-dimensional orthogonal packing problem revisited. In *Intelligent Decision Support - Current Challenges and Approaches*, pages 15–31. Gabler Edition Wissenschaft, 2008.

[43] F. Foucaud, S. Gravier, R. Naserasr, A. Parreau, and P. Valicov. Identifying codes in line graphs. *To appear in Journal of Graph Theory*, 2012.

[44] F. Foucaud, E. Guerrini, M. Kovše, R. Naserasr, A. Parreau, and P. Valicov. Extremal graphs for the identifying code problem. *European Journal of Combinatorics*, 32(4):628–638, 2011.

[45] F. Foucaud, R. Klasing, A. Kosowski, and A. Raspaud. On the size of identifying codes in triangle-free graphs. *Discrete Applied Mathematics*, 160(10-11):1532–1546, 2012.

[46] F. Foucaud, R. Naserasr, and A. Parreau. Characterizing extremal digraphs for identifying codes and extremal cases of Bondy's theorem on induced subsets. *Graphs and Combinatorics*, pages 1–11, 2012. 10.1007/s00373-012-1136-4.

[47] J. L. Fouquet and J. L. Jolivet. Strong edge-coloring of graphs and applications to multi-k-gons. *Ars Combinatoria*, 16A:141–150, 1983.

[48] J. L. Fouquet and J. L. Jolivet. Strong edge-coloring of cubic planar graphs. *Progr. Graph Theory (Waterloo 1982)*, pages 247–264, 1984.

[49] A. M. Frieze, M. Krivelevich, and B. Sudakov. The strong chromatic index of random graphs. *SIAM Journal on Discrete Mathematics*, 19(3):719–727, 2005.

[50] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15:835–855, 1965.

[51] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976.

[52] A. Ghouila-Houri. Caractérisation des graphes non orientes dont on peut orienter les aretes de maniere à obtenir le graphe d'une rélation d'ordre. *Comptes Rendus Mathématique. Académie des Sciences. Paris*, 254:1370–1371, 1962.

[53] P. C. Gilmore and A. J. Hoffman. A characterization of comparability graphs and of interval graphs. *Canadian Journal of Mathematics*, 16:539–548, 1964.

[54] S. Gravier, R. Klasing, and J. Moncel. Hardness results and approximation algorithms for identifying codes and locating-dominating codes in graphs. *Algorithmic Operations Research*, 3(1):43–50, 2008.

[55] S. Gravier and J. Moncel. On graphs having a $v \setminus \{x\}$ set as an identifying code. *Discrete Mathematics*, 307(3-5):432–434, 2007.

[56] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[57] T. W. Haynes, D. J. Knisley, E. Seier, and Y. Zou. A quantitative analysis of secondary rna structure using domination based parameters on trees. *BMC Bioinformatics*, 7:108, 2006.

[58] H. Hocquard, P. Ochem, and P. Valicov. Strong edge coloring and induced matchings. April 2011.

[59] H. Hocquard and P. Valicov. Strong edge colouring of subcubic graphs. *Discrete Applied Mathematics*, 159(15):1650–1657, 2011.

[60] I. Holyer. The NP-completeness of edge-coloring. *SIAM Journal on Computing*, 10(4):718–720, 1981.

[61] P. Horák, H. Qing, and W. T. Trotter. Induced matchings in cubic graphs. *Journal of Graph Theory*, 17(2):151–160, 1993.

[62] T. R. Jensen and B. Toft. Choosability versus chromaticity. *Geombinatorics*, 5(2):45–64, October 1995.

[63] C. Joncour. *Problèmes de placement 2D et application à l'ordonnancement : modélisation par la théorie des graphes et approches de programmation mathématique*. PhD thesis, Université de Bordeaux 1, December 2010.

[64] C. Joncour and A. Pêcher. Consecutive ones matrices for multi-dimensional orthogonal packing problems. *Journal of Mathematical Modelling and Algorithms*, 11:23–44, 2012.

[65] C. Joncour, A. Pêcher, and P. Valicov. MPQ-trees for the orthogonal packing problem. *Journal of Mathematical Modelling and Algorithms*, 11:3–22, 2012.

[66] M. G. Karpovsky, K. Chakrabarty, and L. B. Levitin. On a new class of codes for identifying vertices in graphs. *Information Theory, IEEE Transactions on*, 44(2):599–611, March 1998.

[67] N. Korte and R. H. Möhring. Transitive orientation of graphs with side constraints. In *Graphtheoretic Concepts in Computer Science, Proceedings WG '85*, pages 143–160. Trauner, Linz, 1986.

[68] N. Korte and R. H. Möhring. An incremental linear-time algorithm for recognizing interval graphs. *SIAM Journal on Computing*, 18(1):68–81, 1989.

[69] H.-H. Lai, K.-W. Lih, and P.-Y. Tsai. The strong chromatic index of halin graphs. *Discrete Mathematics*, 312(9):1536–1541, 2012.

[70] M. Laifenfeld and A. Trachtenberg. Identifying codes and covering problems. *IEEE Transactions on Information Theory*, 54(9):3929–3950, September 2008.

[71] M. Laifenfeld, A. Trachtenberg, R. Cohen, and D. Starobinski. Joint monitoring and routing in wireless sensor networks using robust identifying codes. In *Broadband Communications, Networks and Systems, 2007. BROADNETS 2007. Fourth International Conference on*, pages 197–206, September 2007.

[72] R. Laskar, J. Pfaff, S. M. Hedetniemi, and S. T. Hedetniemi. On the algorithmic complexity of total domination. *SIAM Journal on Algebraic and Discrete Methods*, 5(3):420–425, 1984.

[73] J. Y-T. Leung, T. W. Tam, C.S. Wong, G. H. Young, and F. Y.L. Chin. Packing squares into a square. *Journal of Parallel and Distributed Computing*, 10(3):271–275, 1990.

[74] K.-W. Lih and D. D.-F. Liu. On the strong chromatic index of cubic halin graphs. *Applied Mathematics Letters*, 25(5):898–901, 2012.

[75] E. L. Lloyd and S. Ramanathan. On the complexity of distance-2 coloring. In *Proceedings of the Fourth International Conference on Computing and Information: Computing and Information*, ICCI '92, pages 71–74. IEEE Computer Society, 1992.

[76] V. V. Lozin. On maximum induced matchings in bipartite graphs. *Information Processing Letters*, 81(1):7–11, 2002.

[77] M. Mahdian. The strong chromatic index of graphs. Master's thesis, Department of Computer Science, University of Toronto, 2000.

[78] M. Mahdian. On the computational complexity of strong edge coloring. *Discrete Applied Mathematics*, 118(3):239–248, 2002.

[79] M. Molloy and B. Reed. A bound on the strong chromatic index of a graph. *J. Combin. Theory Ser. B*, 69(2):103–109, 1997.

[80] J. Moncel. On graphs on n vertices having an identifying code of cardinality $\log_2(n+1)$. *Discrete Applied Mathematics*, 154(14):2032–2039, 2006.

[81] J. Moncel. *Codes Identifiants dans les Graphes.* PhD thesis, Université Joseph-Fourier - Grenoble I, June 2005.

[82] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan. Achieving mac layer fairness in wireless packet networks. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, pages 87–98, New York, NY, USA, 2000. ACM.

[83] M. Padberg and L. Wolsey. Fractional covers for forests and matchings. *Mathematical Programming*, 29:1–14, 1984.

[84] G. Ramalingam and C. P. Rangan. A unified approach to domination problems on interval graphs. *Information Processing Letters*, 27(5):271–274, 1988.

[85] S. Ramanathan. A unified framework and algorithm for (t/f/c)dma channel assignment in wireless networks. In *INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 2, pages 900–907 vol.2, April 1997.

[86] S. Ray, R. Ungrangsi, F. Pellegrini, A. Trachtenberg, and D. Starobinski. Robust location detection in emergency sensor networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 1044–1053, 2003.

[87] D. J. Rose. Triangulated graphs and the elimination process. *Journal of Mathematical Analysis and Applications*, 32(3):597–609, 1970.

[88] D. J. Rose, R. E. Tarjan, and G. S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2):266–283, 1976.

[89] M. R. Salavatipour. A polynomial time algorithm for strong edge coloring of partial k-trees. *Discrete Applied Mathematics*, 143(1–3):285–291, 2004.

[90] W. C. Shiu, P. C. B. Lam, and W. K. Tam. On strong chromatic index of halin graph. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 57:211–222, May 2006.

[91] W. C. Shiu and W. K. Tam. The strong chromatic index of complete cubic halin graphs. *Applied Mathematics Letters*, 22(5):754–758, 2009.

[92] R. D. Skaggs. *Identifying vertices in graphs and digraphs.* PhD thesis, University of South Africa, South Africa, February 2007.

[93] P. J. Slater. Dominating and reference sets in a graph. *Journal of Mathematical and Physical Sciences*, 22(4):445–455, 1988.

[94] P. J. Slater and D. F. Rall. On location-domination numbers for certain classes of graphs. *Congressus Numerantium*, 45:97–106, 1984.

[95] L. J. Stockmeyer and V. V. Vazirani. Np-completeness of some generalizations of the maximum matching problem. *Information Processing Letters*, 15(1):14–19, 1982.

[96] J. Suomela. Approximability of identifying codes and locating-dominating codes. *Information Processing Letters*, 103(1):28–33, 2007.

[97] M. E. Trotter. Line perfect graphs. *Mathematical Programming*, 12(1):255–259, 1997.

[98] G. Wegner. Graphs with given diameter and a coloring problem. Technical report, University of Dortmund, 1977.

[99] M. Yannakakis. Edge-deletion problems. *SIAM Journal on Computing*, 10(2):297–309, 1981.