UNIVERZA NA PRIMORSKEM
FAKULTETA ZA MATEMATIKO, NARAVOSLOVJE IN
INFORMACIJSKE TEHNOLOGIJE

Master's thesis

(Magistrsko delo)

The maximum independent set problem and equistable graphs

(Problem največje neodvisne množice in ekvistabilni grafi)

Ime in priimek: Edin Husić
Študijski program: Matematične znanosti, 2. stopnja
Mentor: izr. prof. dr. Martin Milanič
Somentor: doc. dr. Ademir Hujdurović

Koper, avgust 2017

Ključna dokumentacijska informacija

Ime in PRIIMEK: Edin HUSIĆ

Naslov magistrskega dela: Problem največje neodvisne množice in ekvistabilni grafi

Kraj: Koper

Leto: 2017

Število listov: 63 Število slik: 8

Število referenc: 71

Mentor: izr. prof. dr. Martin Milanič

Somentor: doc. dr. Ademir Hujdurović

UDK: 519.17(043.2)

Ključne besede: neodvisna množica, polinomski algoritem, ekvistabilen graf

Math. Subj. Class. (2010): 05C69, 05C85, 05C22

Izvleček:

Glavni predmet magistrskega dela je problem neodvisne množice, optimizacijski problem, katerega cilj je v danem grafu G poiskati največjo množico paroma nesosednjih vozlišč. V magistrskem delu je ta NP-težek problem formalno definiran in motivirano je raziskovanje problema v posebnih grafovskih razredih. Predstavljene so številne znane tehnike za razvoj polinomskih algoritmov v posebnih primerih. Z uporabo tehnike povečujočih grafov izpeljemo nov rezultat, in sicer dokažemo, da je problem polinomsko rešljiv v razredu $\{P_{10}, C_4, C_6\}$-prostih grafov. Predstavljen je razred ekvistabilnih grafov in algoritem linearne časovne zahtevnosti za prepoznavanje k-ekvistabilnih grafov. Povzet je dokaz izreka, ki pravi, da je problem utežene neodvisne množice NP-težek za ekvistabilne grafe.

Key words documentation

Name and SURNAME: Edin HUSIĆ

Title of the thesis: The maximum independent set problem and equistable graphs

Place: Koper

Year: 2017

Number of pages: 63 Number of figures: 8

Number of references: 71

Mentor: Assoc. Prof. Martin Milanič, PhD

Co-Mentor: Assist. Prof. Ademir Hujdurović, PhD

Keywords: independent set, polynomial algorithm, equistable graph

Math. Subj. Class. (2010): 05C69, 05C85, 05C22

Abstract:

The main topic of the Master thesis is the independent set problem in graphs, an optimization problem that takes as input a graph G and asks about a maximum set of pairwise non-adjacent vertices. In the Master thesis we formally define this NP-hard problem and motivate its research on restricted graph classes. We present several known techniques for developing polynomial-time algorithms for the problem in special cases. Using the technique of augmenting graphs we develop a new result. Namely, we show that the independent set problem is solvable in polynomial time in the class of $\{P_{10}, C_4, C_6\}$-free graphs. We also present the class of equistable graphs, a linear-time recognition algorithm for the class of k-equistable graphs, and a proof of the fact that the weighted independent set problem is NP-hard in the class of equistable graphs.

Acknowledgement

I would like to thank my family for believing in me. I would like to thank UP Famnit for caring about its students and helping us with all they can. I would also like to thank prof. Milanič and prof. Hujdorović for all the work we did together during my Master's and for all the help and reviews of the thesis.

Contents

List of Figures

1 Introduction

An undirected graph $G = (V, E)$ is an ordered pair where V is a finite set of *vertices* of G and E is a set of unordered pairs of distinct elements of V. The set $V(G) = V$ is called the vertex set of G and the set $E(G) = E$ is called the edge set of G. We say that two vertices $u, v \in V$ are adjacent if there exists an edge $e \in E$ such that $e = \{u, v\}$, and non-adjacent otherwise. A set $I \subseteq V$ is an *independent set* (or a *stable* set) if every two vertices from I are non-adjacent. An independent set is *maximal* if it is not contained in any other independent set and *maximum* if it is of maximum possible size. An independent set, originally called *internal stable* set by Korshunov [38], is sometimes also called vertex packing. In the Independent Set problem we are given a graph G as input and the goal is to find an independent set in G of maximum cardinality. Independent Set was one of the first 21 problems that were proved NP-hard by Karp [36] in 1972. As such it is one of main problems in theoretical computer science and combinatorial optimization. The problem finds important applications in a wide range of practical problems arising in many aspects of human activities, including not only computer science, but also information theory, biology, transport management, telecommunications, and finance. More precisely in molecular biology, computer vision, railways dispatching, coding theory, scheduling in wireless networks, etc. [22]. Several examples will be described in Section 1.2.

Since solving an NP-hard optimization problem to optimality may be prohibitive in terms of computational time, we can instead try to obtain an approximate solution in reasonable time: this brings us to the area of approximation algorithms. Other approaches for solving NP-hard problem use heuristics, local optimization, more efficient exponential algorithms, approximation algorithms, randomized algorithms, fixed-parameter tractable algorithms, or polynomial algorithms for special cases of input, see [22] for a nice overview of these methods. It is known that the Independent Set problem:

- is NP-hard to approximate within a factor $O(n^{1-\epsilon})$ for $\epsilon > 0$ [5],
- is not fixed-parameter tractable unless $W[1] = \text{FPT}$ [21],
- admits no subexponential time algorithm under the Exponential Time Hypothesis [35, 44].

These complexity results motivate the study of INDEPENDENT SET problem on restricted graph classes. Even though the problem is hard in general, it is often possible to reveal restrictions on the input instances under which the problem can be solved efficiently, that is, in polynomial time. One goal towards understanding INDEPENDENT SET is to identify cases for which polynomial-time solutions are possible. For many restricted graph classes the problem becomes polynomial-time solvable. Such are the class of bipartite graphs [16] or the class of chordal graphs ($\{C_4, C_5, \dots\}$-free graphs) [25]. Polynomial time algorithms were also developed for graphs of bounded clique width [18], kK_2-free graphs [7], $2P_3$-free graphs [51], claw-free graphs [59, 68], $\{\text{claw} + K_2\}$-free graphs [50], and fork-free graphs [49]. It should be noted that results for $\{\text{claw} + K_2\}$-free and fork-free graphs are different generalizations of the result for claw-free graphs.

Not all restrictions and properties make the problem polynomially solvable. INDEPENDENT SET remains NP-hard even on planar graphs of maximum degree three [27], but it admits polynomial-time approximation schemes and as well as subexponential time exact algorithm and parametrized algorithms [6, 42] in the class of planar graphs. INDEPENDENT SET problem is NP-hard also for unit disk graphs [15] and for triangle-free graphs [60].

Many authors tried to classify the properties of graph classes for which the INDEPENDENT SET problem remains NP-hard. While a complete classification of the complexity for all hereditary classes of graphs seems out of reach, we can at least hope to classify the complexity status of the problem for all H-free classes, where H is a connected graph. For such cases, it was proved by Alekseev [1] that the INDEPENDENT SET problem remains NP-hard for H-free graphs unless H is a path or a subdivision of the claw ($K_{1,3}$). If H is a P_3 then the graphs that are H-free are disjoint unions of complete graphs and INDEPENDENT SET problem is trivially solvable in polynomial time in this graph class. The INDEPENDENT SET problem on P_4-free graphs (also known as cographs) was proven to be polynomially solvable by Corneil et al. [17]. For the class of P_5-free graphs a polynomial algorithm is known but is much more complicated and we will present the ideas of this algorithm in later chapters. Recently also a polynomial algorithm for P_6-free graphs was announced [32], improving a known quasipolynomial-time algorithm for P_6-free graphs [45]. Regarding other positive results for P_k-free graphs where $k \geq 7$, a subexponential algorithm for P_k-free graphs [13] is known.

Structure of the Master thesis: In the rest of this section we will introduce the notation and present a few examples of modeling using the independent set problem. After that we will survey some methods for solving the problem in particular graph classes. The largest part of the thesis is devoted to such a survey. More precisely, we will present the method of augmenting graphs and use it to develop a polynomial-time algorithm for the Independent Set problem for $\{P_{10}, C_4, C_6\}$-free graphs. This result is a new contribution and it is the main result of Section 2.1.

We will also present a recent technique that uses particular useful tree decomposition of minimal triangulations of a graph. We will mention how such a technique is used to obtain polynomial-time algorithms for Weighted Independent Set in P_5-free and P_6-free graphs.

Other important results such as modular decomposition, decomposition by clique separators, and graph transformations will be covered in Sections 2.3-2.6.

The last chapter is devoted to the class of equistable graphs. A linear-time algorithm for recognition of k-equistable graphs is presented [37]. We also show that Independent Set is NP-hard for equistable graphs [57].

1.1 Preliminaries

In the thesis we consider all graphs as finite, simple and undirected. For a graph $G = (V, E)$ we denote with $V(G) = V$ the set of vertices and with $E(G) = E$ the set of edges of G. We will often write $uv \in E$ for an edge $\{u, v\} \in E$. The *neighborhood* of a vertex v is the set of vertices $N_G(v) = \{u \in V \mid uv \in E\}$. The *closed neighborhood* of a vertex v is $N_G[v] = N_G(v) \cup \{v\}$. When it is clear from the context to which graph we are referring to we will write shortly $N(v)$ instead of $N_G(v)$ and similarly for $N[v]$. The neighborhood of a vertex set $W \subseteq V$ is $N_G[W] = \cup_{v \in W} N_G[v]$ and $N_G(W) = N_G[W] \setminus W$. We will refer to $N_G(v)$ and $N_G(W)$ as the *open neighborhood* of a vertex v and a set W, respectively. For simplicity we will write $N_G(u, v) = N_G(\{u, v\})$ and $N_G[u, v] = N_G[\{u, v\}]$, respectively. Define $\delta_G(v)$ as the subset of vertices of $N_G(v)$ with neighbors outside of $N_G(v)$ that is $\delta_G(v) = \{u \in N_G(v) \mid uw \in E \text{ for some } w \in V(G) \setminus N_G[v]\}$.

Let S, S' be subsets of vertices of a graph G. We say that S *dominates* S' if $S' \subseteq N[S]$ and we say that S is *anticomplete* to S' if there is no edge with one endpoint in $S \setminus S'$ and the other in $S' \setminus S$. To avoid confusion we will usually use terms "dominates" and "anticomplete to" when referring to disjoint subsets of vertices.

We say that a graph $H = (V', E')$ is a subgraph of a graph $G = (V, E)$ if $V' \subseteq V$ and $E \subseteq E'$. For a non-empty subset $W \subseteq V$, the *induced subgraph* $G[W]$ is defined as the graph $H = (W, E \cap \binom{W}{2})$, where $\binom{W}{2}$ is the set of all unordered pairs in W. The graph $G[V \setminus W]$ is denoted with $G \setminus W$. If the set $W = \{v\}$ we will write $G \setminus v$ for simplicity.

A *clique* $C \subseteq V(G)$ is a set of pairwise adjacent vertices, and an *independent* set $I \subseteq V(G)$ is a set of pairwise non-adjacent vertices. A clique C is maximal if there is no other clique C' such that C is contained in C', analogously we say that an independent set is maximal if there is no other independent set containing it. We denote with $\zeta(G)$ the family of all maximal cliques in a graph G.

The Independent Set problem is the problem of finding the maximum size of an independent set in a given graph. The *independence number* of a graph G is the maximum size of an independent set, denoted with $\alpha(G)$. A *weighted graph* is a pair (G, w) (also denoted by G_w) where G is a graph and $w : V(G) \to \mathbb{R}_+$ is a weight function on the vertex set. The Weighted Independent Set problem takes as input a weighted graph G_w and the task is to find an independent set in G of maximum weight, where the weight of an independent set I is defined as $w(I) = \sum_{v \in I} w(v)$. For a weighted graph G_w, we denote by $\alpha_w(G)$ the maximum weight of an independent set.

Independent Set:
Input: A graph G.
Task: Compute $\alpha(G)$.

Weighted Independent Set:
Input: A weighted graph G_w.
Task: Compute $\alpha_w(G)$.

A *path* in a graph G is a sequence of pairwise distinct vertices $(v_1, v_2 \dots, v_\ell)$ such that $v_i v_{i+1} \in E$ for every $i \in \{1, \dots, \ell - 1\}$. If those are the only edges in $G[\{v_1, v_2, \dots, v_\ell\}]$ then the path is said to be induced. If $v_1 v_\ell \in E$ for a path $(v_1, v_2 \dots, v_\ell)$ then this path is called a *cycle* and the cycle is induced if removing the edge $v_1 v_\ell$ from $G[\{v_1, v_2 \dots, v_\ell\}]$ makes the path $(v_1, v_2 \dots, v_\ell)$ induced. A cycle on at least four vertices is called a *hole*. A hole is even (resp. odd) if it contains even (resp. odd) number of vertices.

A graph G is *connected* if there exists a path connecting every two vertices of G. We say that G is *co-connected* if $\overline{G}$ is connected.

A vertex set $S \subseteq V(G)$ in a graph G is a *separator* in G if there exist vertices u, v in different connected components of $G \setminus S$ that belong to the same connected component of G, and in that case we say that S separates u and v. The separator is *minimal* for u, v if no proper subset of S separates u and v. A separator S of G is a *minimal* separator of G if it is minimal for some pair of vertices in G. A connected component C of $G \setminus S$ is a *full component* associated with S if $N(C) = S$. It is easy to see that a component C of $G \setminus S$ is full if and only if every vertex of S has a neighbor in C. Such a pair (C, S) is called a *block*.

Proposition 1.1 (see, e.g., Lokshtanov [43]). *S is a minimal separator if and only if there are at least two distinct full components associated with S.*

A *tree decomposition* of a graph $G = (V, E)$ is a pair (T, χ) where T is a tree and χ is a function $\chi : V(T) \to \mathcal{P}(V)$ satisfying the following:

- $\forall uv \in E$ there exists $w \in V(T)$ with $\{u, v\} \subseteq \chi(w)$,
- $\forall v \in V$ the set $\{w \in V(T) \mid v \in \chi(w)\}$ is a non-empty set inducing a connected subtree of T.

For all $w \in V(T)$ we call the image $\chi(w)$ a *bag* of T. We say that a tree decomposition (T, χ) is *rooted* if T is rooted and we denote the root vertex by $r(T)$.

1.1.1 Graph classes

For a graph G and a family of graphs $\mathcal{F}$ we say that G is $\mathcal{F}$-free if graph G does not contain any graph from $\mathcal{F}$ as an induced subgraph. If $\mathcal{F} = \{H\}$ we will write shortly that G is H-free. We define $Free(\mathcal{F})$ as the class of $\mathcal{F}$-free graphs. It is well known that a class is $Free(\mathcal{F})$ for some graph family $\mathcal{F}$ if and only if it is a hereditary class [2]. We say that a class of graphs X is *hereditary* if it is closed under vertex deletion, or equivalently, if it is closed under taking induced subgraphs, i.e., if the following implication holds:

$$G \in X, H \text{ is induced subgraph of } G \Rightarrow H \in X\,.$$

Many classes of theoretical or practical importance are hereditary, which includes among others

- forests, i.e., graphs without cycles,
- bipartite graphs,
- graphs of bounded vertex degree,
- planar graphs,
- graphs of bounded treewidth,
- graphs of bounded clique-width,
- chordal graphs,
- perfect graphs,
- interval graphs,
- circle graphs, and
- line graphs.

Hereditary graph classes are very well studied and many results are known. In the next chapters we will consider several hereditary graph classes and the Independent Set problem when input graphs are restricted to belong to one of those graph classes. For more details on hereditary graph classes we refer to [12].

Recall that a graph G is *perfect* if for every induced subgraph H of G it holds that $\omega(H) = \chi(H)$, where $\omega(H)$ is the maximum size of a clique in G and $\chi(H)$ is minimum number of independent sets that form a partition of the vertices of H.

Theorem 1.2 (Strong Perfect Graph Theorem)**.** *[Chudnovsky, Robertson, Seymour and Thomas [14]] A graph G is perfect if and only if G is $\{C_5, C_7, \overline{C_7}, C_9, \overline{C_9}, \dots\}$-free.*

We say that a graph is *even-hole-free* if it is $\{C_4, C_6, C_8, \dots\}$-free.

1.2 Examples

In the following chapter we show how several more or less (real- world) problems can be modeled as the Independent Set problem.

1.2.1 Rectangle intersection graphs

One of the well known applications of Independent Set is a problem of automatic label placement or simply map labeling. In this problem we are given a map together with names of regions and several possibilities for placement of each name close to its particular location. We need to place as many names as possible with the condition that no two names overlap. Formally we are given set of points $P = \{p_1, \dots p_n\}$ in $\mathbb{R}^2$ that correspond to actual places on a map. For each $p_i \in P$ we are given a set Q_i of rectangles with sides parallel to the two axes which correspond to the possible placements of the name of a place i on the map. For each such instance we can build a graph G with vertex set V, $V = \cup_{i=1}^{n} Q_i$ and for every two $p_u, p_v \in V$ it holds $p_u p_v \in E(G)$ if and only if the two rectangles corresponding to p_u and p_v intersect. Under the assumption that every two rectangles in some Q_i intersect, the solution to the starting problem exactly corresponds to finding a maximum independent set in graph G. We could extend the example further and add a weight to each possible placement of some regions name (for example, by the ratio of importance and/or the area of label needed to write the specific name). To solve such an extension we add a weight function to the vertices of graph G. In this case we want to solve the Weighted Independent Set problem. Such a problem motivates the study of the Independent Set problem in so-called rectangle intersection graphs. Such a class of graphs is trivially hereditary but the Independent Set remains NP-hard [39].

A similar problem appeared several times in the practice round of Google Hash Code competition.

1.2.2 Railways dispatching

During operations, railway dispatcher faces the challenging problem of rerouting and rescheduling trains in the presence of delays. Once a train is delayed, it might be in conflict with other trains that are planned to use the same track. The dispatcher then has to find a new feasible plan. Interestingly enough, these complicated decisions are carried out mostly by humans today, with only basic computer support such as graphical monitoring tools. Nevertheless, the dispatching decisions have a considerable impact on reliability and punctuality as experienced by passengers. Typically, a railway station is modeled as a graph with vertices representing points on the tracks, and edges representing track segments that connect such points. In many cases the resulting graphs are planar, which is the case for many junctions and stations. Hence, conflict free routes correspond to vertex disjoint paths. Not every route which is physically feasible is desirable in practice, though. Therefore, railway planners allow for each train only a small set of alternative paths for each train. For each pair of terminals (s_i, t_i) of some train i, there exists a set of feasible routes $\mathcal{P}_i$. Then the goal is to find a maximum number of vertex disjoint paths $P_{i_1}, P_{i_2}, \dots, P_{i_m}$, where $P_{i_j} \in \mathcal{P}_j$.

We construct a conflict graph $G = (V, E)$ as follows. We set $V = \cup_i P_i$. Two vertices u, v are adjacent if u, v belong to the same set $\mathcal{P}_i$ for some i or they are two routes sharing an inner point. Then the problem of finding the maximum number of vertex disjoint paths is the Independent Set problem in G.

2 Methods for solving Independent Set

2.1 Minimal augmenting graphs

For a graph $G = (V, E)$ we define its line graph $L(G)$ as the graph with vertex set $E(G)$ and two vertices of $L(G)$ being adjacent if and only if they share an endpoint. An example of a graph G and its line graph is depicted in Figure 1. One can easily see that an independent set in graph $L(G)$ corresponds to a matching in G. Hence, for the class of line graphs Independent Set and Weighted Independent Set are equivalent to finding a maximum matching and a maximum weight matching in the root graph, respectively. Due to Berge's idea of augmenting paths [9] and Edmond's algorithm for finding augmenting paths in "Paths, trees, and flowers" [23], we know that the problem of finding a maximum matching in a graph can be solved in polynomial time. Since we can find in linear time a root graph of a given line graph [40, 67], it follows that Independent Set is polynomially solvable in the class of line graphs.

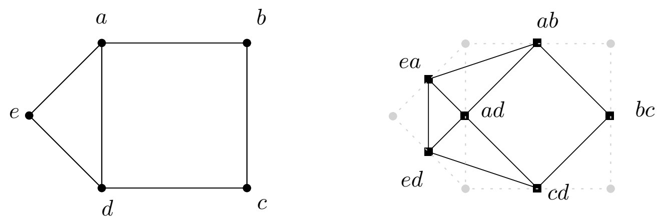

Figure 1: A graph G on the left, and its line graph $L(G)$ on the right.

Generalization of the notion of augmenting paths using augmenting graphs led to several new polynomial-time algorithms for the Independent Set problem. Augmenting graphs were first introduced by Sbihi, to show that the problem is polynomially solvable in the class of $K_{1,3}$-free graphs [68], where $K_{m,n}$ denotes the complete bipartite graph with parts of size m and n.

Definition 2.1. For an independent set $I \subseteq V(G)$, an induced bipartite graph $H = (W, B; E)$ in G is *augmenting* for I if the following holds:

- $W \subseteq I$,

- $B \subseteq V(G) \setminus I$,
- $|W| < |B|$ and
- $N(B) \cap I \subseteq W$.

The following theorem suggests a possible application of augmenting graphs.

Theorem 2.2 (Minty [59]). *An independent set I in a graph G is maximum if and only if there is no augmenting graph for I.*

Proof. Let I be an independent set in graph G. If I admits an augmenting graph $H = (W, B; E)$ then the set $I' = (I \setminus W) \cup B$ is an independent set and $|I'| > |I|$. Conversely, if I is not a maximum independent set and I' is a maximum independent set then $|I'| > |I|$. But then the subgraph induced by $(I \setminus I') \cup (I' \setminus I)$ is an augmenting graph for I. □

The above theorem suggests the following algorithmic approach for finding a maximum independent set. Start with any maximal independent set I, then find an augmenting graph for I if one exists. Since the size of an independent set is at most $n = |V(G)|$, we will repeat this step at most $n - 1$ times and obtain a maximum independent set. It is therefore trivial to conclude that testing whether an independent set admits an augmenting graph is NP-hard, yet such an approach could give efficient algorithm for particular classes of graphs. The same approach works even if we restrict the problem to finding a *minimal* augmenting graph. We say that an augmenting graph H is *minimal for I* if any proper induced subgraph of H is not an augmenting graph for I. We use the following lemma to characterize minimal augmenting graphs for an independent set I. Interestingly, the characterizing condition does not depend on I.

Lemma 2.3 (Lozin and Milanič [49]). *An augmenting graph $H = (W, B; E)$ is minimal if and only if*

i) $|W| = |B| - 1$, *and*

ii) for every non-empty subset $A \subseteq W$ it holds $|A| < |N(A)|$.

Proof. Let $H = (W, B; E)$ be a minimal augmenting graph for an independent set I. If $|W| < |B| - 1$ then the graph induced by $W \cup (B \setminus \{v\})$, for some $v \in B$, is an augmenting graph for I. Hence, *i)* holds. Suppose there exists a subset $A \subseteq W$ such that $A \geq |N(A)|$. Then the set of vertices $(W \setminus A) \cup (B \setminus N(A))$ induces a proper subgraph of H, which is furthermore augmenting for I. Hence, *ii)* holds.

To prove the other direction, let $H = (W, B; E)$ be an augmenting graph for I satisfying *i)* and *ii)*. Suppose there exists a proper induced subgraph $H' = (W', B'; E')$ of H that is also augmenting for I.

If $|W'| = |W|$, then $|B'| \geq |W'| + 1 = |W| + 1 = |B|$ implies that $H = H'$. We conclude $|W'| < |W|$. Hence, the set $A = W \setminus W'$ is non-empty. Since H' is augmenting it follows that $N(A) \subseteq (B \setminus B')$. This gives $|N(A)| \leq |B \setminus B'| \leq |W \setminus W'| = |A|$ which contradicts *ii*). □

Corollary 2.4 (Lozin and Milanič [49]). *An augmenting graph $H = (W, B; E)$ is minimal if and only if*

i) $|W| = |B| - 1$,

ii) *for every non-empty subset $A \subseteq W$ it holds $|A| < |N(A)|$, and*

iii) *H is connected.*

Proof. Using Lemma 2.3 it suffices to prove that *i*) and *ii*) imply *iii*). Let H be augmenting graph and suppose that *i*) and *ii*) hold. Suppose for the sake of contradiction that H is not connected. Let X be a subset of vertices that induces a connected component of H and let $Y = V(H) \setminus X$. Define $W_X = X \cap W$, $W_Y = Y \cap W$, $B_X = X \cap B$, and $B_Y = Y \cap B$. By *ii*) it follows that $|W_X| < |B_X|$ and $|W_Y| < |B_Y|$, but then $|W_X| + |W_Y| < |B_X| + |B_Y| - 1$. A contradiction with *i*). □

In the following sections we apply the method of augmenting graphs to obtain a polynomial-time algorithm for $\{P_{10}, C_4, C_6\}$-free graphs. We start by developing a polynomial-time algorithm for Independent Set in the class of $\{P_9, C_4, C_6\}$-free graphs.

2.1.1 Independent Set in $\{P_9, C_4, C_6\}$-free graphs

Let G be a graph and I a maximal independent set. If there exists an augmenting P_3, then by enumerating all P_3s, we can find an augmenting one in time $O(n^3)$. For the rest of the section, for any graph G and a maximal independent set I in G, we assume that I does not admit an augmenting P_3.

For a vertex $w \in I$ we define $K_I(w) = \{v \in V(G) : N(v) \cap I = \{w\}\}$ and $K_I[w] = K_I(w) \cup \{w\}$. When it is clear to which independent set I we are referring to we will write $K(w)$ and $K[w]$ for simplicity. We will often denote $R = V(G) \setminus I$.

Observation 1. Let I be a maximal independent set in a graph G such that I does not admit augmenting P_3. Then for any $w \in I$, $K_I[w]$ is a clique.

We will first describe minimal augmenting graphs for even-hole-free graphs and subsequently show how to detect such minimal augmenting graphs in subclasses of even-hole-free graphs. Let us introduce some necessary definitions.

Definition 2.5. We say that a tree T is a *black-white tree* if its vertex set can be partitioned into two sets of vertices B and W such that both sets are independent and every vertex in W has degree exactly 2. We say that vertices in W are white and vertices in B are black.

For better understanding we state the following easy proposition (without proof). Later, we show that the minimal augmenting graphs in the class of even-hole-free graphs are exactly the black-white trees.

Proposition 2.6. *A graph T is black-white tree if and only if it can be obtained by subdividing each edge of a non-trivial tree exactly once.*

Lemma 2.7. *Let G be a graph and I a maximal independent set. Let H be an augmenting tree for I. Then H is a minimal augmenting graph for I if and only if it is a black-white tree.*

Proof. Let $H = (W, B; E)$ be a minimal augmenting tree for I. We will show that every vertex in part W is of degree 2.

By Lemma 2.3 there exists no white vertex of degree 1. Also by Lemma 2.3 it holds $|B| = |W| + 1$ and hence $|B| + |W| = 2|W| + 1$. Since H is a tree, the latter implies that it has exactly $2|W|$ edges. Suppose that there exists a vertex $w \in W$ of degree 3 or more. Since every vertex in W has degree at least 2 and W is a part of bipartition (W, B) of H, we infer that H has strictly more than $2|W|$ edges, a contradiction.

Suppose that H is a black-white tree. Let us show that H is a minimal augmenting graph by Lemma 2.3. By definition of a black-white tree H has $|W|$ white vertices and $2|W|$ edges, hence it has $2|W| + 1$ vertices and it holds that $|B| = |W| + 1$, proving i). Since every white vertex has degree exactly 2 and every two white vertices have at most one common neighbor in H, condition ii) holds. □

Suppose that G is even-hole-free graph and I a maximal independent set in G. Then every bipartite induced subgraph of G is a forest. This means that every minimal augmenting graph for I is a tree by Corollary 2.4. Moreover, by the above lemma it follows that every minimal augmenting graph for G is a black-white tree. We will use this fact for developing a polynomial-time algorithm for Independent Set in the class of $\{P_{10}, C_4, C_6\}$-free graphs.

Definition 2.8. For numbers $r, s \in \mathbb{N} \setminus \{0\}$, we define a black-white rooted tree $T_{r,s}$ as follows. The root of the tree $T_{r,s}$ is white, it has two children, which are black, and they have exactly r and s white children, respectively. Recall that each white vertex has a unique child. All vertices are at distance at most 3 from the root.

Moreover, we define a black-white tree T_s as a rooted tree, having a black root vertex, and exactly s white children (and each white vertex has a unique child). All vertices are at distance at most 2 from the root.

For an example of $T_{r,s}$ and T_s we refer to Figure 2.

Observe that $T_{1,1}$ is a P_7 and T_2 is a P_5.

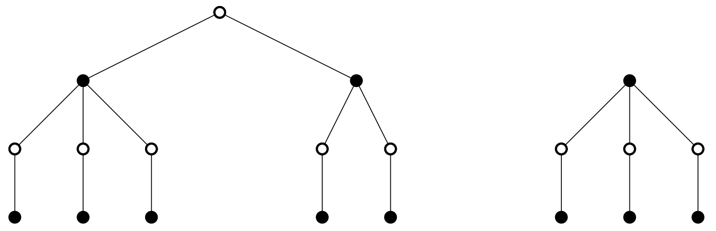

Figure 2: Example of trees $T_{3,2}$ and T_3.

Side note: Let $\mathcal{C}$ be a class of graphs. We define $\mathcal{C}^m$ to be the class of all graphs H for which there exists a graph $G \in \mathcal{C}$ and an independent set I of G such that H is a minimal augmenting graph for I. If for a class $\mathcal{C}$ it holds that $\mathcal{C}^m$ is finite, then we have a trivial polynomial-time algorithm for the INDEPENDENT SET problem in the class $\mathcal{C}$. Namely, we can exhaustively check all the subsets of $V(G)$ of size $O(1)$ to see whether one of them induces a graph augmenting for I.

The following are examples of infinite classes $\mathcal{C}^m$ for some hereditary class $\mathcal{C}$:

- the set $\mathcal{P}$ of all paths P_k for k odd integer,
- the set $\mathcal{T}$ of all trees T_s for $s \in \mathbb{N}$, and
- the set $\mathcal{K}$ of all complete bipartite graphs $K_{s,s+1}$ where $s \in \mathbb{N}$.

Moreover, it has been shown that these are the only minimal infinite classes $\mathcal{C}^m$.

Theorem 2.9 (Dabrowski et al. [19]). *Let $\mathcal{C}$ be a hereditary class of graphs and let $\mathcal{C}^m$ be the class of minimal augmenting graphs of $\mathcal{C}$. If $\mathcal{C}^m$ is infinite, then $\mathcal{C}^m$ contains $\mathcal{P}, \mathcal{T}$, or $\mathcal{K}$.*

When $\mathcal{C}$ is the class of even-hole-free graphs $\mathcal{C}^m$ contains both $\mathcal{P}$ and $\mathcal{T}$ since every graph in $\mathcal{P}$ and in $\mathcal{T}$ is a black-white tree.

Trees of type T_s have already appeared in the study of augmenting graphs, more precisely as one of the two classes of minimal augmenting graphs in the class of $\{P_6, C_4\}$-free graphs [61]. Subsequently they were considered as minimal augmenting graphs in the class of $\{\text{banner}, S_{1,2,5}\}$-free graphs [48], where *banner* is a graph on vertices a, b, c, d, e such that $\{a, b, c, d\}$ induce a C_4 and e is adjacent only to d.

The following is an easy consequence of Lemma 2.7.

Observation 2. Let G be an $\{P_9$, even-hole$\}$-free graph and I a maximal independent set in G. Let H be a minimal augmenting graph for I. Clearly, H is a P_9-free graph. Since every maximal induced path in H has black end vertices, it follows that H is P_8-free. If H contains P_7, then H is isomorphic to a $T_{r,s}$ for some integers r and s. On the other hand, if H is P_7-free, then H is isomorphic to a T_s.

In the remaining part of the section we show how to detect graphs T_s and $T_{r,s}$.

Lemma 2.10. *Let $\mathcal{F}$ be a set of graphs, let $k \geq 3$ and let $\mathcal{C} = \text{Free}(\{P_k\} \cup \mathcal{F})$. Let $G \in \mathcal{C}$ be a graph and I a maximal independent set in G. If the* Independent Set *problem can be solved in time $O(n^{t_k})$ in the class of $\{P_{k-1} \cup \mathcal{F}\}$-free graphs, then it can be tested in time $O(n^{t_k+1})$ whether I admits an augmenting graph of the form T_s.*

Proof. We show how to check whether there exists an augmenting T_s with root at x, for a vertex $x \in R$. Recall that $R = V(G) \setminus I$. Fix $x \in R$. We will try to build such a tree T. By definition of augmenting graph, all vertices in $N(x) \cap I$ are in T so add them to T. Set $s = |N(x) \cap I|$. Let $S = \big(\cup_{w \in N(x) \cap I} K(w)\big) \setminus N(x)$. Since we assumed that no augmenting P_3 exists we know that every $K(w)$ is a clique. Therefore, there exists an augmenting T_s with root at x if and only if there exists an independent set I^* in $G[S]$ such that $|I^*| = s$. Moreover, since S is a union of s cliques the set I^* is a maximum independent set in $G[S]$. The later can be checked in polynomial time by assumption if we show that the graph $G[S]$ is $\{P_{k-1} \cup \mathcal{F}\}$-free.

It is trivial to see that $G[S]$ is $\mathcal{F}$-free since $G[S] \in \mathcal{C}$. Suppose on the contrary that $G[S]$ contains an induced P_{k-1} with vertices $\{p_1, \ldots, p_{k-2}, p_{k-1}\}$. Denote w a vertex in $N(x) \cap I$ such that $p_{k-2} \in K(w) \setminus N(x)$ and a vertex $w' \in N(x) \cap I$ such that $p_{k-1} \in K(w') \setminus N(x)$. It may be possible that $w = w'$. If $w = w'$ then the set $\{p_1, \ldots, p_{k-2}, w, x\}$ induces a P_k in G. If $w \neq w'$ then the set $\{p_1, \ldots, p_{k-1}, w'\}$ induces a P_k in G. A contradiction. □

Lemma 2.11. *Let $\mathcal{F}$ be a set of graphs, let $k \geq 4$ and let $\mathcal{C} = \text{Free}(\{P_k\} \cup \mathcal{F})$. Let $G \in \mathcal{C}$ be a graph and I a maximal independent set in G. If the* Independent Set *problem can be solved in time $O(n^{t_{k-3}})$ in the class of $\{P_{k-3} \cup \mathcal{F}\}$-free graphs, then it can be tested, in time $O(n^{t_{k-3}+3})$, whether I admits an augmenting graph of the from $T_{r,s}$.*

Proof. We show how to test if there exists an augmenting $T_{r,s}$ with root at $v \in I$. Fix $v \in I$. If $N(v)$ is a clique, then such a tree with root at v clearly does not exist. Otherwise, choose $b', b'' \in N(v) \cap R$ such that b' and b'' are non-adjacent. We will check if there exists an augmenting tree T of type $T_{r,s}$ containing vertices v, b', b'' with root at v. If $N(b') \cap N(b'') \neq \{v\}$ then such a tree clearly does not exist. For the rest of the proof, suppose that $N(b') \cap N(b'') = \{v\}$. Since b' and b'' are black vertices, by definition of augmenting graph vertices in $N(b') \cap I$ and $N(b'') \cap I$ are in T. Denote $s = |N(b', b'') \cap (I \setminus \{v\})|$. Let $S = \left(\cup_{w \in N(b',b'') \cap I \setminus \{v\}} K(w)\right) \setminus N(b', b'')$. Similarly as before, since we assume there is no augmenting P_3 it follows that $K(w)$ is a clique. Therefore, there exists an augmenting $T_{r,s}$ with root at v containing $\{b', b''\}$ if and only if there exists an independent set I^* in $G[S]$ such that $|I^*| = s$. Moreover, since S is a union of s cliques, the set I^* is a maximum independent set in $G[S]$. The later can be checked in polynomial time by assumption if we show that the graph $G[S]$ is $\{P_{k-3} \cup \mathcal{F}\}$-free.

$G[S]$ is $\mathcal{F}$-free since $G[S] \in \mathcal{C}$. For the sake of contradiction suppose that there exists an induced P_{k-3} in $G[S]$. Denote its vertices as $\{p_1, \ldots, p_{k-3}\}$. Denote by w a vertex in $N(b', b'') \cap (I \setminus \{v\})$ such that $p_{k-4} \in K(w) \setminus N(b', b'')$ and a vertex $w' \in N(b', b'') \cap I$ such that $p_{k-3} \in K(w') \setminus N(b', b'')$. Then (depending on whether $w = w'$ or not) either the set $\{p_1, \ldots, p_{k-4}, w, b', v, b''\}$ or $\{p_1, \ldots, p_{k-3}, w', b', v\}$ induce a P_k in G. A contradiction. □

We obtain the following theorem.

Theorem 2.12. *The* Independent Set *problem is polynomially solvable in the class of* $\{P_9$, *even-hole*$\}$*-free graphs.*

Proof. By Observation 2, the minimal augmenting graphs for the class of $\{P_9$, even-hole$\}$-free are of type P_3, T_s, or $T_{r,s}$. On the other hand, Independent Set can be solved in polynomial time in the class of $\{P_8$, banner$\}$-free graphs [29], and hence also in the class of $\{P_8$, even-hole$\}$-free graphs (the banner contains a C_4, therefore every $\{P_8$, even-hole$\}$-free graph is also $\{P_8$, banner$\}$-free). By Lemmas 2.10 and 2.11 we conclude that the Independent Set problem is polynomial solvable in the class of $\{P_9$, even-hole$\}$-free graphs. □

It should be noted that we could devise faster and more direct algorithm for $\{P_9$, even-hole$\}$-free graphs by devising a faster algorithm for Independent Set in the class of $\{P_8$, even-hole$\}$-free graphs, but we focus just on polynomial-time solvability.

Recall that the $\{P_9$, even-hole$\}$-free graphs are exactly the $\{P_9, C_4, C_6, C_8\}$-free graphs. We strengthen the result to the class of $\{P_9, C_4, C_6\}$-free graph. We show that there are only finitely many minimal augmenting graphs in the class $\text{Free}(\{P_9, C_4, C_6\})$ that are not T_s and $T_{r,s}$. Observe that if a minimal augmenting graphs in the mentioned class is not a tree, then it contains a hole, more precisely a C_8.

Lemma 2.13. *Let G be a $\{P_9, C_4, C_6\}$-free graph and I a maximal independent set in G. Let $H = (W, B; E)$ be a minimal augmenting graph for I containing a C_8. Then $|V(H)| \leq 15$.*

Proof. Let C be an induced C_8 in H. Let $v \in N_H(C)$. Using the fact that H is bipartite, it is easy to see that $|N_H(v) \cap V(C)| = 1$.

Next, observe that C dominates H (that is, every vertex $v \in V(H) \backslash V(C)$ is adjacent to a vertex in C). Suppose to the contrary that there exists a vertex $w \in V(H) \setminus V(C)$ non-adjacent to a vertex in C. Take a shortest path P from w to C. Then set $V(P) \cup V(C)$ induces a graph that contains a P_9.

Denote the vertices of C as $v_1, \dots v_8$ such that $v_i v_{i+1 (\text{mod } 8)} \in E$ for $i \in \{1, \dots, 8\}$. Then, w.l.o.g., $v_1, v_3, v_5, v_7 \in W$ and $v_2, v_4, v_6, v_8 \in B$. Observe that $N_H(v_1) \setminus V(C)$ is a set containing only black vertices. Moreover, since H is a minimal augmenting graph if $|N_H(v_1) \setminus (V(C))| > 1$ we can find an augmenting P_3 (which contradicts an assumption we made in the beginning of the section). Hence, $|N_H(v_1) \setminus (V(C))| \leq 1$. Similarly $|N_H(v_i) \setminus (V(C))| \leq 1$ for $i \in \{3, 5, 7\}$.

Since $N_H(v_i) \subseteq B$ and every vertex $b \in B$ is adjacent to some v_i where $i \in \{1, 3, 5, 7\}$, we have $|B| \leq 4 + 4$. By Lemma 2.3 it follows $|W| \leq 7$. Hence, H is one of finitely many bipartite graphs on 15 vertices. □

Remark 2.14. In fact, it is easy to check that there are just a few possible graphs satisfying conditions of Lemma 2.13.

Theorem 2.15. *A maximum independent set in a given $\{P_9, C_4, C_6\}$-free graph G can be found in polynomial time.*

Proof. As we said before, every minimal augmenting graph in the class either contains a C_8 or is a tree. By Lemma 2.13 every minimal augmenting graph containing a C_8 has at most 15 vertices. For each subset $S \subseteq V(G), |S| \leq 15$ check if it induces an augmenting graph. This can be done in time $O(n^{15})$ using a brute force approach.

If there is no minimal augmenting graph containing a C_8, then the only possible minimal augmenting graphs are P_3, $T_{r,s}$ and T_s. Using Lemma 2.10, Lemma 2.11 and a polynomial-time algorithm for Independent Set in {banner, P_8}-free graphs [29] we can check if there exists a minimal augmenting tree in polynomial time.

If, in the above procedure, at any step we find an augmenting graph, we augment it and start again with the new maximal independent set. We do so, until we obtain a maximal independent set that admits no minimal augmenting graph. By Theorem 2.2 such an independent set is maximum.

The running time of checking if an independent set I admits a minimal augmenting graph is bounded by the running time of the first step. Since the size of independent set is bounded by n, we will consider at most n independent sets with increasing size at each step. Hence, the total running time is bounded by $O(n^{16})$. □

2.1.2 Independent set in $\{P_{10}, C_4, C_6\}$-free graphs

In this section we generalize the previous result on $\{P_9, C_4, C_6\}$-free graphs to the class of $\{C_4, C_6, P_{10}\}$-free graphs. First, we need to classify the minimal augmenting graphs. Next to the already mentioned trees P_3, T_s and $T_{r,s}$, a minimal augmenting tree can be a black-white tree that contains an induced P_9. It is easy to see that such trees are exactly black-white trees with a black root and at least two leaves at distance 4 from the root. An example of such a tree is depicted in Figure 3. Recall that we still keep the assumption that there is no augmenting P_3 for any graph and any independent set in that graph we consider in this section.

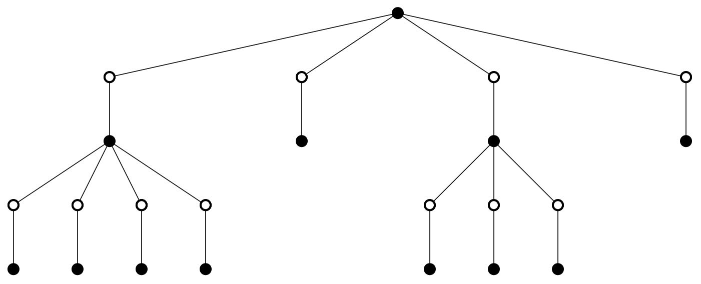

Figure 3: Example of a minimal augmenting graph containing a P_9.

Since we already know that Independent Set is polynomially solvable in the class of $\{P_9, C_4, C_6\}$-free graphs, Lemma 2.10 and Lemma 2.11 give us a polynomial-time algorithm for checking whether a maximal independent set I of G (where $G \in$ Free$(\{C_4, C_6, P_{10}\})$ admits a minimal augmenting graph T_s or $T_{r,s}$. Hence, we need to show how to test whether an independent set I admits a minimal augmenting tree containing P_9 or an augmenting graph containing a C_8 or a C_{10}. Let us start with the following lemma.

Lemma 2.16. *Let G be a $\{C_4, C_6\}$-free graph and I a maximal independent set in G. Let $x \in R$, $W_x = N(x) \cap I = \{w_1, \dots, w_k\}$. Define $B_i = N(w_i) \cap (R \setminus N[x])$. Then we can decide in polynomial time if there exists a set $I^* \subseteq \cup_{i=1}^k B_i$ such that*

- *I^* is an independent set, and*
- *$|I^* \cap B_i| = 1$ for every $i \in \{1, \dots, k\}$.*

Proof. Let $B = \cup_{i=1}^k B_i$. We use the following observation. Let $i \in \{1, \dots, k\}$. There exists no induced $P_3 = (\{a, b, c\}, \{ab, bc\})$ such that $b \in B_i$ and $a, c \notin B_i$. On the contrary, suppose that such a P_3 exists. If $a, c \in B_j$ for some $j \in \{1, \dots, k\} \setminus \{i\}$ then $\{a, b, c, w_j\}$ induces a C_4. Otherwise, $a \in B_j$ and $c \in B_t$ for different $j, t \in \{1, \dots, k\} \setminus \{i\}$. Then there exists an induced C_6 (on vertex set $\{x, w_j, a, b, c, w_t\}$), a contradiction.

Let Γ be a graph obtained from $G[B]$ by completing every B_i to a clique. Recall that since G is C_4-free the sets B_i are pairwise disjoint. Since the additional edges have both endpoints in a single B_i it follows that, in Γ, there exists no induced P_3 such that the middle vertex is contained is some B_j and the end vertices of P_3 are not in B_j.

The set I^* we want is exactly a maximum independent set in Γ of size k. We claim that Γ is a perfect graph. For the sake of contradiction suppose that Γ is not perfect. By Theorem 1.2 G contains either an odd hole or an odd antihole. Suppose that Γ contains an odd hole C. Since every B_i is a clique in Γ, every B_i contains at most two vertices of C. Since $|C|$ is odd follows that there exists a bag B_j such that $|B_j \cap V(C)| = 1$. Let $\{b\} = B_j \cap V(C)$. Then $N[b] \cap V(C)$ induces a P_3 such that b is contained in B_j and $a, c \notin B_j$, a contradiction.

Now, suppose that Γ contains an odd antihole A (on at least 7 vertices, since C_5 is self-complementary). Denote vertices in A as $v_1, \dots v_t$ such that every two consecutive vertices are non-adjacent. Let v_1, v_2 be two non-adjacent vertices in A, with $v_1 \in B_i$ and $v_2 \in B_j$. Since $|A| \geq 7$, it follows that $|N(v_1) \cap N(v_2) \cap A| \geq 3$. In particular, vertices v_4, v_5, v_6 are adjacent to v_1 and v_2. If one of them is a vertex in B_t such that $t \notin \{i, j\}$ we have reached a contradiction. It follows that, w.l.o.g, $v_4, v_6 \in B_i$ and $v_5 \in B_j$. Observe vertex v_3, it is not in B_i since it is non-adjacent to v_4 and it is not in B_j since it is non-adjacent to v_2. Hence, $\{v_5, v_3, v_6\}$ induces P_3 that intersects three different bags.

Since Γ is perfect, we can find a maximum independent set in Γ in polynomial time [30]. □

We say that two subsets of vertices S_1, S_2 of G are *complete* to each other if every vertex of S_1 is adjacent to every vertex S_2. We say that S_1 and S_2 are *anticomplete* to each other if there is no edge with one endpoint in $S_1 \setminus S_2$ and the other in $S_2 \setminus S_1$. In the rest of the section when using the term *anticomplete* we will always refer to disjoint sets. Now, we can show how to check if there exists an augmenting tree in $\{P_{10}, C_4, C_6\}$-free graphs.

Lemma 2.17. *Let G be a $\{P_{10}, C_4, C_6\}$-free graph and I a maximal independent set in G. It can be tested in polynomial time whether I admits a minimal augmenting tree $H = (W, B; E)$ containing a P_9.*

Proof. Let $x \in R$. We show how to check whether there exists an augmenting tree with root at x and at least two leaves at distance 4 from x, i.e., if x is a middle vertex of an induced P_9. We try to build such a tree T. Set x as the root of T. Define $W_x = N(x) \cap I$ and denote vertices in W_x as $w_1, \dots, w_k$. Add W_x to T ($W_x \subseteq V(T)$ by definition of an augmenting graph). For each $w_i \in W_x$, let $B_i = N(w_i) \setminus N[x]$. For every $w_i, w_j \in W_x, i \neq j$, it holds $B_i \cap B_j = \emptyset$ since G is C_4-free.

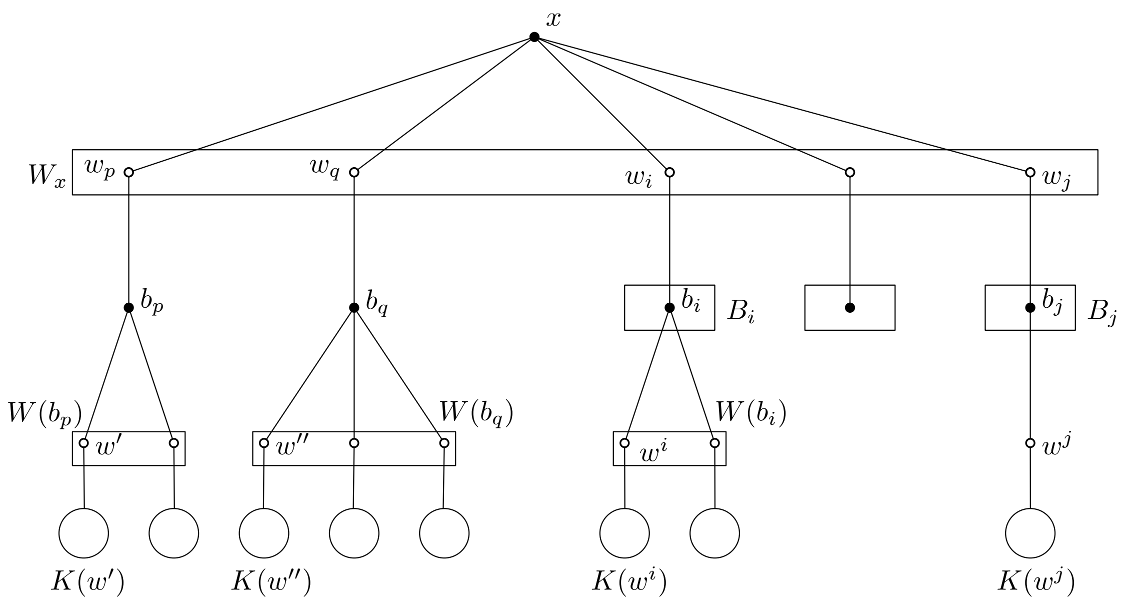

Figure 4: A scheme for the proof of Lemma 2.17.

Let $p, q \in \{1\dots,k\}$ such that $p \neq q$. Let $b_p \in B_p$ and $b_q \in B_q$ such that b_p and b_q are non adjacent. If such a p, q, b_p and b_q do not exist, then we conclude that augmenting tree with the root at x does not exist. We will test whether there exists an augmenting tree with root at x containing b_p and b_q such that b_p and b_q are not leaves of T. Add b_p and b_q to T.

For $b \in B_i$ let $W(b) = N(b) \cap (I \setminus \{w_i\})$. Note that $W(b) \cap N(x) = \emptyset$ since G is C_4-free. For every $b_i \in B_i$ and $b_j \in B_j$ such that $i \neq j$ it holds $W(b_i) \cap W(b_j) = \emptyset$ since G is $\{C_4, C_6\}$-free.

For every $w \in W(b)$ remove all vertices from $K(w)$ that are adjacent to x, b_p or b_q since they can not be in T. If for some $w \in W(b_p) \cup W(b_q)$ it holds that $K(w)$ is empty we conclude that desired tree does not exist. For the rest of the proof assume that $W(b_p) \neq \emptyset$, $W(b_q) \neq \emptyset$ and for every $w \in W(b_p) \cup W(b_q)$ it holds that $K(w) \neq \emptyset$. Obviously T has to contain vertices in $W(b_p)$ and $W(b_q)$ so add these to T. From each B_i, $i \in \{1, \dots, k\} \setminus \{p, q\}$, remove vertices that are adjacent to b_p or b_q since they can not be in T. Moreover, after the removal of such vertices each B_i has to be non-empty. We consider the following two cases based on the size of W_x.

Case 1: $|W_x| = 2$. Let $S = \cup_{w \in W(b_p) \cup W(b_q)} K(w)$. Define $s = |W(b_p) \cup W(b_q)|$. Similarly as in the proofs of Lemmas 2.10 and 2.11 we can finish building our augmenting tree if and only if there exists a maximum independent set I^* in $G[S]$ of size s. Moreover, we can find such an independent set since $G[S]$ is $\{P_5, C_4\}$-free and using, e.g., the algorithm given in [61].

Case 2: $|W_x| > 2$.

Claim 1: If $w' \in W(b_p)$ and $w'' \in W(b_q)$, then $K(w')$ is anticomplete to $K(w'')$.

Proof of Claim 1. We show how to remove some vertices that cannot possibly be in T so that the claim holds. Suppose that there exists $u \in K(w')$ adjacent to $v \in K(w'')$. If $K(w'') \subseteq N(u)$, we can remove u from $K(w')$ since in this case we can not complete tree T. Otherwise let $v' \in K(w'') \setminus N(u)$.

Let $w_i \in W_x \setminus \{w_p, w_q\}$ and $b_i \in B_i$. There exist such w_i and b_i since $|W_x| > 2$. Suppose that b_i is adjacent to u. Then the set $\{b_i, u, w', b_p, w_p, x, w_q, b_q, w'', v'\}$ where $v' \in K(w'') \setminus N(u)$ induces a P_{10}. Hence b_i is non-adjacent to u. Analogously, we conclude that b_i is non-adjacent to v. Then the set $\{b_p, w', u, v, w'', b_q, w_q, x, w_i, b_i\}$ induces P_{10}. A contradiction.

Claim 2: Let $w', w^* \in W(b_p), w' \neq w^*$ then $K(w')$ and $K(w^*)$ are anticomplete.

Proof of Claim 2. On the contrary, suppose that there exist $u \in K(w')$ and $v \in K(w^*)$ such that $uv \in E(G)$. Then the set $\{u, v, w^*, b_p, w_p, x, w_q, b_q\}$ together with a vertex $w'' \in W(b_q)$ and a vertex in $K(w'')$ induce a P_{10}.

A statement analogous to Claim 2 holds for every $w', w^* \in W(b_q), w' \neq w^*$. Let $B = \cup_{i \in \{1,\ldots,k\} \setminus \{p,q\}} B_i$. We show next that B is anticomplete to $K(w)$ for every $w \in W(b_p) \cup W(b_q)$. For the sake of contradiction suppose there exists $b_i \in B_i$ for some $i \in \{1, \ldots, k\} \setminus \{p, q\}$ and $w' \in W(b_p)$ such that $b_i u \in E(G)$ for some $u \in K(w')$. Then the set $\{b_i, u, w', b_p, w_p, x, w_q, b_q, w'', v\}$ induces a P_{10} for some $w'' \in W(b_q)$ and some $v \in K(w'') \setminus N(b_i)$. If $K(w'') \setminus N(b_i) = \emptyset$ then b_i can not be a vertex of T so we remove it from B_i.

Cleaning of B**.** Let $i \in \{1, \ldots, k\} \setminus \{p, q\}$. Let $b_i \in B_i$. If for some $w^i \in W(b_i)$ it holds that $K(w^i)$ is empty we can easily remove b_i from B_i since it can not be part of a minimal augmenting tree with root at x (we can not finish the tree with white vertices). Observe that for vertices $b_i \in B_i$ it is possible that $W(b_i) = \emptyset$ and we do not remove such vertices. Do the cleaning for every $i \in \{1, \ldots, k\} \setminus \{p, q\}$ and every $b \in B_i$.

By Lemma 2.16 it follows that we can check in polynomial time if there exists an independent set I^* of size $k-2$ such that $|I^* \cap B_i| = 1$ for every $i \in \{1, \ldots, k\} \setminus \{p, q\}$. If there is no such a set we conclude that the tree T does not exists.

Claim 3: If such a set I^* is found then we can complete T to a minimal augmenting tree rooted at x.

Proof of Claim 3: Let $b_i \in I^* \cap B_i$ and $b_j \in I^* \cap B_j$ for different $i, j \in \{1, \ldots, k\} \setminus \{p, q\}$. It is obvious that $W(b_i) \cap W(b_j) = \emptyset$ since starting graph is $\{C_4, C_6\}$-free. Moreover $W(b_i) \cup W(b_j) \subseteq I$ by definition. Let $w^i \in W(b_i)$ and let $u \in K(w^i)$. Similarly as before we can show that u is anticomplete to $K(w)$ where $w \in W(b_p) \cup W(b_q)$. If $ub_j \in E(G)$ then the set $\{b_j, u, w^i, b_i, w_i, x, w_p, b_p\}$ together with $w' \in W(b_p)$ and a vertex in $K(w')$ induce a P_{10}, hence $ub_j \notin E(G)$. The same holds if we swap i and j. Let $w^j \in W(b_j)$ and $v \in K(w^j)$ such that $uv \in E$, then again the set

$\{v, u, w^i, b_i, w_i, x, w_p, b_p\}$ together with $w' \in W(b_p)$ and a vertex in $K(w')$ induce a P_{10}, hence $K(w^j)$ is anticomplete to $K(w^i)$ for any $w^i \in W(b_i)$ and any $w^j \in W(b_j)$. The above gives us an obvious way to complete T. For each $b \in I^*$ add the set $W(b)$ to T and for every $w \in W(b)$ add any vertex of $K(w)$ to T (it can be checked that $K(w)$ is anticomplete to $K(w')$ for every $w' \in W(b')$, where $b' \in I^* \setminus \{b\}$). $\square$

Recall that every connected augmenting graph in the class of $\{P_{10}, C_4, C_6\}$-free graphs is either a tree, or it contains a C_8, or it contains a C_{10} (since these are only possible holes in a $\{P_{10}, C_4, C_6\}$-free bipartite graph). The following lemma will show that the last case never happens.

Lemma 2.18. *Let G be a graph and I a maximal independent set in G. Let $H = (W, B; E)$ be a minimal $\{P_{10}, C_4, C_6\}$-free augmenting graph for I. Then H is C_{10}-free.*

Proof. We will prove the lemma by contradiction. Let C be an induced C_{10} in H. Denote vertices of C as $v_1, \dots, v_{10}$ so that consecutive vertices are adjacent. Since H is augmenting, there exists a black vertex $b \in V(H)$ that is adjacent to some of $V(C) \cap W$. If b is adjacent to two vertices in $V(C) \cap W$ then we can easily find an induced C_4 or C_6. Hence, b is adjacent to exactly one vertex of C, say v_i. Then the set $\big(V(C) \setminus \{v_{i+1 (\mathrm{mod}\ 10)}\}\big) \cup \{b\}$ induces a P_{10}, a contradiction. $\square$

By the above lemma, we know that in the class of $\{P_{10}, C_4, C_6\}$-free graphs an augmenting graph H is either a black-white tree or H contains a C_8. We have already showed how to detect black-white augmenting trees so for the rest of the section we focus our attention to augmenting graphs that contain C_8. We start with necessary definitions.

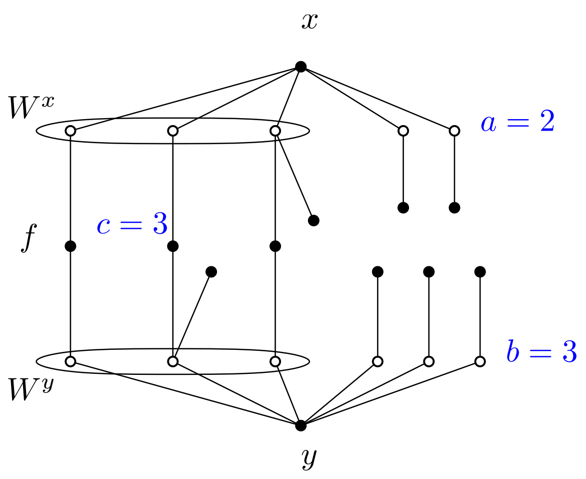

Figure 5: A $2, 3, 3$-satellite.

Definition 2.19. For non-negative integers a, b, c with $c \geq 2$ an a, b, c*-satellite* is a graph H that can be built from two disjoint trees of type T_r, say T and T', in the following way (see Figure 5):

- T is isomorphic to T_{a+c} and T' is isomorphic to T_{b+c}.
- Exactly c leaves of T are identified with c leaves of T'.
- We name the root of T as x and the root of T' as y. We say that x and y are *root* vertices of H. We denote by W^x the set of white vertices of T whose children are identified with some leaves of T'. Similarly for W^y.
- There are $c-1$ *additional vertices* in H, each of which is adjacent to exactly one vertex in $W^x \cup W^y$ and no vertex in $W^x \cup W^y$ is adjacent to two of the additional vertices.
- A set of $c-1$ indentified leaves (out of c) is chosen and for each such vertex, one of its two neighbors in $W^x \cup W^y$ is choosen and is adjacent to an additional vertex.

Lemma 2.20. *There exist two graphs F_1 and F_2 such that the following holds:*

Let G be a $\{P_{10}, C_4, C_6\}$-free graph and I a maximal independent set in G that admits nither an augmenting P_3 nor an augmenting P_5. Let $H = (W, B; E)$ be a minimal augmenting graph for I that contains a C_8. Then H is either F_1, F_2, or an a, b, c-satellite.

Proof. Let C be the set of vertices inducing a C_8 in H and denote vertices in C as $v_1, \ldots, v_8$ so that vertices with odd index are in W and consecutive vertices (modulo 8) are adjacent. Let $v \in N(C) \cap V(H)$. Then $|N(v) \cap C| = 1$ or otherwise we can find an induced C_4 or an induced C_6. Also observe that every vertex $v \in V(H) \setminus C$ is at distance at most 2 from C or there exists a P_{10}.

Claim: C dominates either W or B.

Proof of Claim: Suppose that C does not dominate W. Then there exists a white vertex w at distance 2 from C. Observe that by Lemma 2.3 we have $|N(C) \cap N(w)| \geq 2$. Let $b', b'' \in N(C) \cap N(w)$ be two different vertices. W.l.o.g. assume that $b'v_1 \in E(H)$. It is straightforward to check that $N(b'') \cap C = \{v_5\}$. For the sake of contradiction assume that there exists a vertex $b \in V(H)$ at distance 2 from C. Let $w' \in N(b) \cap N(C)$. W.l.o.g. $N(w') \cap C = \{v_8\}$. Then it is easy to check that the set $\{b, w', v_8, v_1, b', w, b'', v_5, v_4, v_3\}$ induces a P_{10}, a contradiction.

Case 1: C dominates both W and B. Let $v_i \in \{v_2, v_4, v_6, v_8\}$. We will show that $N(v_i) \cap (W \setminus \{v_{i-1}, v_{i+1}\}) = \emptyset$. Suppose on the contrary that there exists $w \in$

$N(v_i) \cap (W \setminus \{v_{i-1}, v_{i+1}\})$. By Lemma 2.3 there exists a vertex $b \in N(w) \cap B$ different from v_i. As we deduced in the beginning of the proof $b \notin C$. Hence, $b \in N(v_j)$ for some j, but this is a contradiction with the fact that H is $\{C_4, C_6\}$-free. Hence, $|W| = 4$ and $|B| = 5$. It is now easy to see that H is unique up to isomporphism and we can set $F_1 = H$.

Case 2: C dominates B but not W. Let $w \in W$ be a vertex at distance 2 from C. Let $b', b'' \in N(w) \cap N(C) \cap B$ be two different vertices. Such vertices exist by Lemma 2.3 and are non-adjacent since they are both black. Let v_i be unique vertex in $N(b') \cap C$ and v_j be a unique vertex in $N(b'') \cap C$. Trivially both v_i and v_j are in W. It is easy to check that it has to be $i \equiv j \pmod 4$. We may assume without loss of generality that $i = 1$ and $j = 5$. Now, suppose there exists $w^* \in N(b') \setminus \{v_1, w\}$. Similarly as for w, there exists a black vertex b^* adjacent to w^* and to C. As before it has to be $b^*v_5 \in E(H)$. Then $\{b', w^*, b^*, v_5, b'', w\}$ either contains a C_4 or induces a C_6. Hence, $N(b') = \{v_1, w\}$ and analogously $N(b'') = \{v_5, w\}$.

Suppose there exists a vertex $\tilde{w} \in N(C) \cap W$. We may assume without loss of generality that $\tilde{w}v_2 \in E(H)$. Moreover by Lemma 2.3 there exists a vertex $\tilde{b} \in B$ such that $\tilde{w}\tilde{b} \in E(G)$. Since every black vertex is dominated it holds $\tilde{b} \in N(C)$. Let $v_j \in N(\tilde{b}) \cap C$, then shortest path between v_j and v_2 in C together with $\{\tilde{b}, \tilde{w}\}$ induces either a C_4 or a C_6. Hence, all vertices in $W \setminus C$ are at distance 2 from C.

Let $w' \in W \setminus \{w\}$ be a vertex at distance 2 from C. Similarly as before there exist two distinct vertices $b_1, b_2 \in B$ such that $N(w') \cap N(C) = \{b_1, b_2\}$. Suppose $b_1 \notin \{b', b''\}$. If $b_1v_3 \in E(H)$, then $\{w', b_1, v_3, v_4, v_1, b', w, b'', w_5, w_6\}$ induces a P_{10}. On the other hand, if $b_1 \in \{b', b''\}$ then we can easily find either an induced C_4 or an induced C_6. Hence, w.l.o.g. $N(b_1) \cap C = \{v_1\}$ and $N(b_2) \cap C = \{v_5\}$.

Let $k = |W \setminus C|$. By the above we deduce that for every $z \in W \setminus C$ there exist two unique vertices b_1^z and b_2^z such that $zb_1^z, b_1^zv_1, zb_2^z, b_2^zv_5 \in E(G)$. Hence, $|B \setminus C| \geq 2|W \setminus C| = 2k$. By Lemma 2.3 $|B| = |W| + 1 = k + 4 + 1$, which gives us $k + 1 \geq 2k$. Hence $k = 1, |W| = 5$, and $|B| = 6$. Again, it that such a graph is unique up to isomorphism and we set $F_2 = H$.

Case 3: C dominates W but not B. W.l.o.g. $N(v_2) \setminus C \neq \emptyset$. For every $v_i \in \{v_2, v_4, v_6, v_8\}$ define $W(v_i) = N(v_i) \setminus C$. Since G is $\{C_4, C_6\}$-free, $N(v_i) \cap N(v_j) = \emptyset$ if $i \neq j$. Moreover since C dominates W, the vertices in $W \cap C$ and in $\cup_{i \in \{1,\dots,4\}} W(v_i)$ are all white vertices in H and we just need to describe the rest of black vertices of H. See Figure 6

Claim: $W(v_4) \cup W(v_8) = \emptyset$.

Proof of Claim: For the sake of contradiction assume that $W(v_4)$ (say) is non-empty. Let $\tilde{b} \in W(v_4)$. Let $w \in W(v_2)$. By Lemma 2.3 there exists $u \in N(w) \setminus$

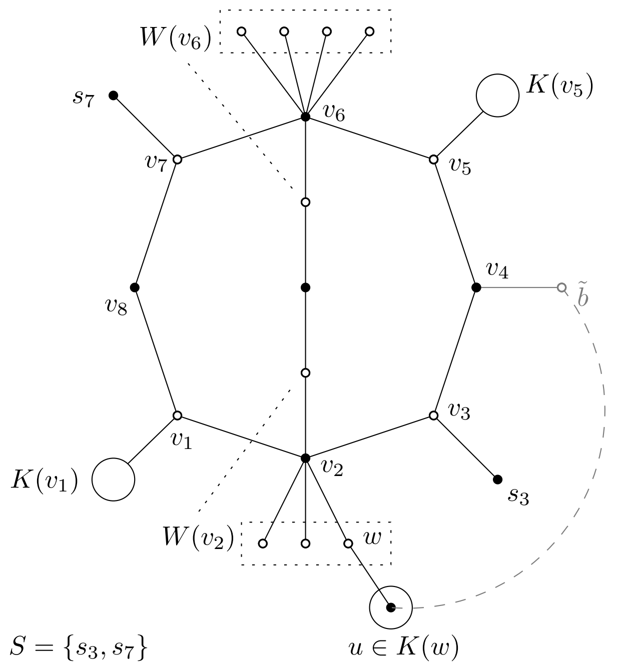

Figure 6: A scheme for the proof of Lemma 2.20, Case 3.

$\{v_2\}$. Due to $\{C_4, C_6\}$-freeness u is anticomplete to C. Then $\tilde{b}u \in E(H)$ or the set $\{\tilde{b}, v_4, v_5, v_6, v_7, v_8, v_1, v_2, w, u\}$ induces a P_{10}. But then $\{\tilde{b}, v_4, v_3, v_2, w, u\}$ induces a C_6. Thus $W(v_4) = \emptyset$; similarly, $W(v_8) = \emptyset$.

Next, we describe all black vertices dominated by C. Let $v_j \in \{v_1, v_3, v_5, v_7\}$. Clearly all black vertices not in C but dominated by C have to be in some $K(v_j)$. Recall that for a vertex w in an independent set I it holds $K(w) = \{b \in I : N(b) \cap I = \{w\}\}$. Let $S \subseteq K(v_1) \cup K(v_3) \cup K(v_5) \cup K(v_7)$ be an independent set such that $S \subseteq B$. Then $|S| \leq 2$. Suppose that there exist three different vertices in S, say $S = \{s_3, s_5, s_7\}$ such that $s_3 \in K(v_3)$, $s_5 \in K(v_5)$ and $s_7 \in K(v_7)$. Then $\{s_3, v_3, v_4, v_5, s_5\}$ induces an augmenting P_5. The same argument shows that there are at most two black vertices at distance 1 from C.

Let b be a black vertex of H at distance 2 from C. If $N(b) \cap N(C) = \{w\}$, then exactly one of the following holds: $w \in W(v_2)$ or $w \in W(v_6)$. Moreover, since I does not admit an augmenting P_3, we have $N(w) \setminus \{v_2, v_6\} = \{b\}$. Similarly, if $N(b) \cap N(C) = \{w', w''\}$ then it holds, w.l.o.g., $w' \in W(v_2)$ and $w'' \in W(v_6)$. Thus, in any case we have $1 \leq |N(b) \cap N(C)| \leq 2$.

Let b, b' be two different black vertices at distance 2 from C, each with exactly two white neighbors. Then $|N(b) \cap N(b')| \leq 1$. The latter follows since G is C_4-free graph.

Now, by counting and applying the claims analogous to the already proved ones it is not hard to conclude that H is an a, b, c-satellite. $\square$

Let H be an a, b, c-satellite. Then by definition there exists a unique vertex $f \in$

$N(W^x) \cap N(W^y)$ in H non-adjacent to any of the additional $c-1$ vertices. We say that the vertex f is the *flat* vertex of H.

Lemma 2.21. *Let G be a $\{P_{10}, C_4, C_6\}$-free graph and I a maximal independent set in G. Let a, b be non-negative integers and c an integer larger than 1. We can test in polynomial time if I admits an augmenting a, b, c-satellite.*

Proof. Let us try to construct such an augmenting graph H. First, since by definition of an a, b, c-satellite it holds that $c \geq 2$ it follows that every a, b, c-satellite contains an induced cycle on 8 vertices. Let C be such a cycle with $|C \cap I| = |C \cap R| = 4$, where $R = V(G) \setminus I$. Denote vertices of C as $\{v_1, \ldots, v_8\}$ such that $\{v_1, v_3, v_5, v_7\} \subseteq I$. W.l.o.g. fix v_2 and v_6 as root vertices of H. Also, without loss of generality, we can assume that v_8 is the flat vertex of H. Moreover, fix an additional vertex s_3 that has a unique neighbor v_3 in H. See Figure 7.

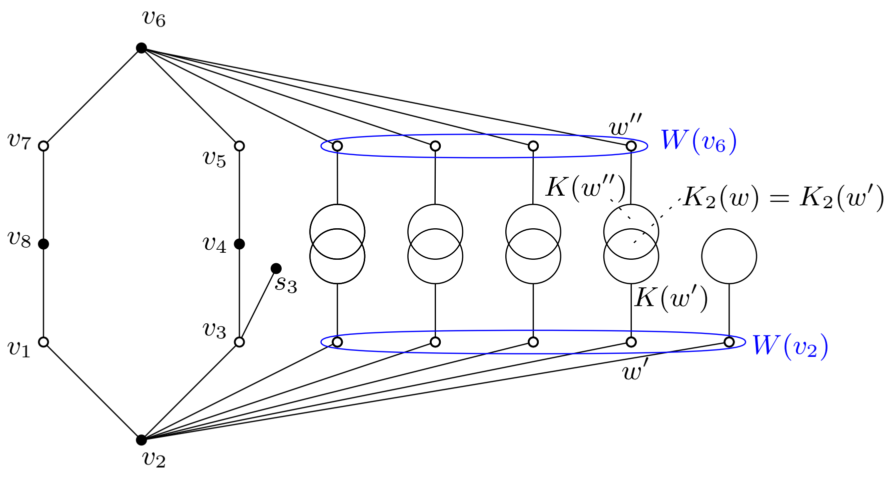

Figure 7: Testing for an augmenting a, b, c-satellite.

We will test if there exists an augmenting a, b, c-satellite that contains $C \cup \{v_3\}$ such that v_2 and v_6 are the root vertices and v_8 is the flat vertex of H. So in the rest of the proof it suffices to consider only the vertices that are non-neighbors of $\{v_1, v_3, v_4, v_5, v_7, v_8, s_3\}$ and the vertices adjacent to v_6 or v_2 that are contained in I.

Define $W(v_6) = N(v_6) \cap (I \setminus \{v_5, v_7\})$ and $W(v_2) = N(v_2) \cap (I \setminus \{v_1, v_3\})$. Observe that if for some $w' \in W(v_2)$ and some $w'' \in W(v_6)$ it holds that $N(w') \cap N(w'') \neq \emptyset$ then for all $\tilde{w} \in W(v_6) \setminus \{w''\}$ it holds that $N(w') \cap N(\tilde{w}) = \emptyset$ since G is C_4-free. Similarly if $\tilde{w} \in W(v_2) \setminus \{w'\}$.

For each $w \in W(v_2)$ define $K_2(w) = N(w) \cap N(W(v_6))$. Define analogously $K_2(w)$ for each $w \in W(v_6)$. By the above paragraph, for each $w' \in W(v_2)$ with $K_2(w') \neq \emptyset$ there exists a unique vertex $w'' \in W(v_6)$ such that $K_2(w') = K_2(w'')$ and for every other $\tilde{w} \in W(v_6)$ it holds $K_2(w') \cap K_2(\tilde{w}) = \emptyset$. We will say that such a w'' *corresponds* to w and vice versa.

Now, using the cycle C and the fact that G is P_{10}-free it is easy to conclude the following statements for every $w', w'' \in W(v_2) \cup W(v_6)$.

- $K_2(w')$ is anticomplete to $K_2(w'')$.
- $K_2(w')$ is anticomplete to $K(w'')$.
- $K(w')$ is anticomplete to $K(w'')$.

Using the above statements it follows that in order to find an augmenting a, b, c-satellite it suffices to check if there exist

- $c - 2$ non-empty sets of type K_2,
- a non-empty sets $K(w'')$ for $w'' \in W(v_6)$, and
- b non-empty sets $K(w')$ for $w' \in W(v_2)$.

The lemma follows. □

We obtain the following theorem.

Theorem 2.22. *The* Independent Set *problem is solvable in polynomial time for the class of* $\{P_{10}, C_4, C_6\}$*-free graphs.*

Proof. Let G be a $\{P_{10}, C_4, C_6\}$-free graph and I a maximal independent set. We need to find a minimal augmenting graph or certify that there is no such graph. By Lemma 2.18 we know that the only minimal augmenting graphs in the class are either black-white trees or an augmenting graph containing a C_8. First, we check if there exists an augmenting P_3 or an augmenting P_5. Then we test if there exists a minimal augmenting graph containing a C_8. By Lemma 2.20 we know that a minimal augmenting graph is either one of two specific graphs or an a, b, c-satellite. We can check in polynomial time if there exists an augmenting F_1 or F_2. By Lemma 2.21 we can check in polynomial time if there exists an augmenting a, b, c-satellite and hence we can check if there exists a minimal augmenting graph that contains a C_8. By Lemma 2.10, Lemma 2.11, and Theorem 2.15 it follows that we can test in polynomial time if there is augmenting tree of type T_s or $T_{r,s}$. So the only possible augmenting graph that we did not test is a black-white tree containing a P_9, but we can test efficiently if one exists by Lemma 2.17. Since we can test in polynomial time if there exists a minimal augmenting graph for a given independent set in a graph from the class the theorem is proved. □

2.2 Minimal triangulations

It is known that the Weighted Independent Set problem is efficiently solvable in the class of chordal graphs [25]. There are several algorithms for Weighted Independent Set and Independent Set for chordal graphs. We consider the following approach based on tree decomposition. Such an approach will allow us to generalize it and show that the independent set problem is polynomially solvable in the class of P_5-free graphs, based on special tree decomposition, of a triangulation of G. Before we start, let us introduce some definitions.

For clarity we first give an intuitive explanation of the approach. Later we show the simpler version of the approach for the class of chordal graphs and then extend it step by step to the class of P_5-free graphs.

The idea is to find a maximum-weight independent set using dynamic programming on a specific tree decomposition of a graph G. Let I be a maximal independent set of G unknown to the algorithm. Since I is independent, it is not hard to see that there exists a tree decomposition of the graph where every bag has at most one element of I (we will call such a tree decomposition I-sparse, see Definition 2.25). If such a tree decomposition was given to us, for some maximum-weight independent set I, then we are able to find I (or an independent set of the same weight) using dynamic programming on the tree decomposition, in polynomial time. Roughly, a state of the dynamic program is formed by a bag of the decomposition and at most one of its elements. The element of such a bag can be seen as the intended intersection of the bag with the already constructed independent set.

A problem of such an approach is that we are not given such a useful tree decomposition. To solve the issue, the authors show that it is possible to compute a sufficiently rich family of candidates for bags. The idea is that for some tree decomposition that can be used to find a maximum-weight solution, its bags will be included in the family. Then a similar dynamic programming procedure, which intuitively tries to assemble all possible tree decompositions using given candidate bags, computes a maximum-weight independent set. Such a family should be rich enough so that all bags of some useful decomposition are contained in order to compute a maximum weighted independent set. On the other hand, it should also be small enough so that the algorithm runs in polynomial time.

We proceed with the formal proof. A graph $H = (V, E \cup F)$ is a *triangulation* of graph a $G = (V, E)$ if every cycle of length at least four in H has a *chord*, i.e., H is a chordal graph such that G is a subgraph of H. The edges in F are called *fill edges*. A triangulation H is minimal if there is no triangulation $H' = (V, E \cup F')$ such that F' is a proper subset of F. A *potential maximal clique* of G is a set $\Omega \subset V(G)$ such that Ω is a maximal clique in some minimal triangulation of G. We denote by $\Omega(G)$ the set of potential maximal cliques of G. It is easy to see that if G is chordal then $\Omega(G)$ is exactly the set of all maximal cliques of G. For a vertex set $\Omega \subseteq V(G)$ we denote by $\mathcal{C}(\Omega) = \{C_1, \ldots, C_t\}$ the set of connected components of $G \setminus \Omega$. We denote $\Delta(\Omega) = \{N(C_i) : 1 \leq i \leq t\}$. Then for a subset of vertices Ω the set $\Delta(\Omega)$ is the set of all minimal separators of G that are subsets of Ω. Recall that we denote with $\zeta(G)$ the family of all maximal cliques in a graph G.

As said, we start from chordal graphs, and will use the following proposition.

Proposition 2.23 (Gavril [28]). *A graph G is chordal if and only if there exists a tree decomposition (T, χ) of G such that every bag is a maximal clique of G.*

Such a tree decomposition is called a *clique tree* and it is known that a clique tree can be found in time $O(m + n)$ for a given chordal graph with n vertices and m edges [10].

Proposition 2.24 (Gavril [28]). *Let (T, χ) be a clique tree of a chordal graph G and $S \subseteq V(G)$. Then S is minimal separator if and only if $S = \chi(u) \cap \chi(u')$ for some edge $uu' \in E(T)$.*

Let G be a chordal graph with a weight function w on vertices of G. Let us devise a polynomial-time algorithm for the Weighted Independent Set problem based on the above theorem using dynamic programming on a tree decomposition (T, χ) of G. Let I be an independent set in G. It is obvious that every bag $\chi(x)$, where $x \in V(T)$, can contain at most one vertex of I, i.e., $|I \cap \chi(x)| \leq 1$. This nice property of chordal graphs can be used for dynamic programming. To achieve our goal, we define the following recursive function f. The function f takes as input a vertex $x \in V(T)$, a vertex $v \in \chi(x)$ and a component $C \in \mathcal{C}(\chi(x))$ and returns the maximum weight of an independent set $I \subseteq \chi(x) \cup C$ such that $I \cap \chi(x) = \{v\}$. It can be verified that function f satisfies the following recurrence:

$$f(v, x, C) = w(v) + \max_{\substack{x' \in V(T) \setminus \{x\} \\ v' \in \chi(x')}} \left[w(v') + \sum_{\substack{C' \in \mathcal{C}(\chi(x')) \\ C' \subseteq C}} (f(v', x', C') - w(v')) \right],$$

where the maximum is taken over all x' and v' such that $\chi(x') \subseteq \chi(v) \cup C$ and $\{v, v'\}$ is independent. If such v' and x' do not exist we set $f(v', x', C) = -\infty$. In order to find a maximum independent set we could evaluate function f for every triple (v, x, C), but this is not necessary. We will show how to modify function f to obtain the result directly. Let $x \in V(T)$ and $X \subseteq \chi(x)$, $|X| \leq 1$. Then modified function f takes as input a vertex $x \in V(T)$, a subset $X \subseteq \chi(x)$ of size at most one, and a component

$C \in \mathcal{C}(\chi(x))$ and returns the maximum weight of an independent set $I \subseteq \chi(x) \cup C$ such that $I \cap \chi(x) = X$. The only difference is that function f now can take the empty set as input. The following recurrence holds:

$$f(x, X, C) = w(X) + \max_{\substack{x' \in V(T) \\ X' \subseteq \chi(x')}} \left[w(X' \setminus X) + \sum_{\substack{C' \in \mathcal{C}(\chi(x')) \\ C' \subseteq C}} (f(x', X', C') - w(X')) \right],$$

where the maximum is taken over all x' and X' such that $\chi(x') \subseteq C$, $|X'| \leq 1$, $\chi(x) \cap \chi(x') \cap X = \chi(x) \cap \chi(x') \cap X'$ and $X \cup X'$ is independent in G.

In the following steps we tend to extend the approach for a “nice” tree decomposition of any graph. Let us introduce necessary definitions.

Definition 2.25 (Lokshtanov et al. [43])**.** For an independent set $I \subseteq V(G)$, a tree decomposition (T, χ) of G is called *I-sparse* if for every bag B we have $|B \cap I| \leq 1$.

Definition 2.26 (Lokshtanov et al. [43])**.** Let G be a graph and (T, χ) a rooted tree decomposition of G. We say that (T, χ) is *simple* if

a) no bag B is subset of any other bag B', and

b) for every edge $uv \in V(T)$ where v is descendant of u, there exists a component $C \in \mathcal{C}(\chi(u))$ such that $\chi(v) \subseteq \chi(u) \cup C$.

Observe that for a chordal graph such a simple tree decomposition exists by Proposition 2.23. Moreover, the same proposition shows that for any minimal triangulation G' of G we can find a simple tree decomposition of G'. We proceed to our goal with the following lemma.

Proposition 2.27 (Bouchitté and Tondinca [11])**.** *Let G be a graph. If $\Omega \subseteq V(G)$ is a potential maximal clique of G, then for every connected component C of $G \setminus \Omega$, the set $N_G(C) \subseteq \Omega$ is a minimal separator of G.*

Lemma 2.28 (Lokshtanov et al. [43])**.** *Let G be a graph with n vertices and m edges and Π a list of vertex sets in G. There is an algorithm running in time $O(|\Pi|^2 n^4 m)$ that outputs the weight of the maximum weight independent set I for which there exists an I-sparse tree decomposition (T, χ) of G such that $\chi(v) \in \Pi$ for all $v \in V(T)$. If no such independent set exists, the algorithm returns $-\infty$.*

We will again modify the function f and use it to prove the above lemma. The function f takes as input a set $B \in \Pi$, a set $X \subseteq B$ such that $|X| \leq 1$ and a component $C \in \mathcal{C}(B)$. It returns the weight of a maximum weight independent set $I \subseteq B \cup C$ such that $I \cap B = X$ and such that there exists an I-sparse simple tree decomposition (T, χ) of $G[B \cup C]$ where all the bags of (T, χ) are in Π and for root vertex r of t it holds $\chi(r) = B$. If no such independent set exists, the function returns $-\infty$.

Lemma 2.29. *Function f satisfies the following recurrence*

$$f(B,X,C) = w(X) + \max_{\substack{B'\in\Pi\\ X'\subseteq B'}} \left[w(X'\setminus X) + \sum_{\substack{C'\in\mathcal{C}(B')\\ C'\subseteq C}} \left(f(B',X',C') - w(X')\right) \right]$$

where the maximum is taken over all sets $B' \in \Pi$ and $X' \subseteq B'$ such that

i) $B' \subseteq B \cup C$,

ii) $N(C) \subseteq B'$,

iii) $B' \cap C \neq \emptyset$,

iv) $|X'| \leq 1$,

v) $B \cap B' \cap X = B \cap B' \cap X'$, *and*

vi) $G[X \cup X']$ *is independent.*

If no such pair of sets exists we set $f(B,X,C) = -\infty$.

Proof. We prove the two inequalities. First, we show that

$$f(B,X,C) \geq w(X) + \max_{\substack{B'\in\Pi\\ X'\subseteq B'}} \left[w(X'\setminus X) + \sum_{\substack{C'\in\mathcal{C}(B')\\ C'\subseteq C}} \left(f(B',X',C') - w(X')\right) \right].$$

If the right hand side is equal to $-\infty$ then the inequality follows trivially. Consider a pair B', X' satisfying $i) - vi)$. For $C' \in \mathcal{C}(B')$ let $I_{C'}$ be an independent set of weight $f(B',X',C')$ such that $I_{C'} \cap B = X'$ and let $(T_{C'}, \chi_{C'})$ be an $I_{C'}$-sparse tree decomposition of $G[B' \cup C']$ such that for the root r' of $T_{C'}$ it holds $\chi_{C'}(r') = B'$. Define

$$I = \bigcup_{\substack{C'\in\mathcal{C}(B')\\ C'\subseteq C}} I_{C'} \cup X\,.$$

Then I is an independent set since the set B' is a separator of $C \setminus B'$ and $B \setminus B'$ whenever $C \setminus B', B \setminus B' \neq \emptyset$. Let us calculate the weight of I:

$$w(I) = w(X) + w(X'\setminus X) + \sum_{\substack{C'\in\mathcal{C}(B')\\ C'\subseteq C}} \left(f(B',X',C') - w(X')\right).$$

We still need to build an I-sparse tree decomposition (T, χ) of $B \cup C$. We do this by identifying roots r' of every tree decomposition $(T_{C'}, \chi_{C'})$ at a new vertex r and setting $\chi(r) = B$. For all other non-root vertices $v \in V(T)$ it holds that v is a vertex in some

$T_{C'}$ for some $C' \in \mathcal{C}(B')$ with $C' \subseteq C$ and we set $\chi(v) = \chi_{C'}(v)$. From construction it follows that (T, χ) is an I-sparse tree decomposition of $G[B \cup C]$ with all bags from Π. Hence,

$$f(B, X, C) \geq w(I)\,.$$

Let us show the $\leq$ inequality. If the left hand side is $-\infty$ then there is nothing to prove. Otherwise, let I be an independent set in $G[B \cup C]$ achieving $f(B, X, C)$ such that $I \cap B = X$. Let (T, χ) be an I-sparse tree decomposition of $G[B \cup C]$ with root r and $\chi(r) = B$. Let us show that r has a unique child. For the sake of contradiction assume the opposite. Let v_1 and v_2 be two children of r in T. Since (T, χ) is simple tree decomposition no bag is subset of another bag and since $\chi(r) = B$ it follows that $\chi(v_1) \cap C \neq \emptyset$ and $\chi(v_2) \cap C \neq \emptyset$. The latter contradicts the fact that the set $\{v \in V(T) : \chi(v) \cap C \neq \emptyset\}$ induces a connected subtree of T (since $\chi(r) \cap C = \emptyset$). Denote the unique child of r by r'. Define $B' = \chi(r')$ and $X' = I \cap B'$. It is straightforward to check that B' satisfies conditions $i) - vi)$. For a connected component $C' \in \mathcal{C}(B')$ define $I_{C'} = I \cap (B' \cup C')$. Observe that

$$w(I) = w(X) + w(X' \setminus X) + \sum_{\substack{C' \in \mathcal{C}(B') \\ C' \subseteq C}} \left(w(I_{C'}) - w(X')\right)\,.$$

Hence, to complete the proof it suffices to prove that for every connected component $C' \in \mathcal{C}(B')$ such that $C' \subseteq C$ it holds $f(B', X', C') \geq w(I_{C'})$. We define $V_{C'}$ as the set of vertices of T that contain some vertices of C', or formally $V_{C'} = \{v \in V(T) : \chi(v) \cap C' \neq \emptyset\}$. By definition of tree decomposition, $V_{C'}$ induces a connected subtree of T. Furthermore, since (T, χ) is a simple tree decomposition, it follows that for every vertex $v \in V_{C'}$ all vertices of $\chi(v)$ are in the same connected component of $G \setminus B'$, hence $\chi(v) \subseteq C'$. Then a tree decomposition $(T[V_{C'}], \chi)$ is an $(I_{C'} \cap C')$-sparse tree decomposition of $G[C']$. All bags are trivially in Π. Let r^* be a vertex in $V_{C'}$ closest to r in T. Let us show that $N(C') \subseteq \chi(r^*)$. Let $a \in N(C')$. Then, there exists a vertex $b \in C'$ such that $ab \in E(G)$. The bag closest to the root of T containing b is either $\chi(r^*)$ or $\chi(u)$ where u is descendant of r^*. By definition of C' it follows that $a \in B' = \chi(r')$. Since the set of vertices of T whose corresponding bags contain a induce a connected subtree and there exists a bag containing ab it follows that $a \in \chi(r^*)$. Hence $N(C') \subseteq \chi(r^*)$. Let $(T_{C'}, \chi_{C'})$ be a tree decomposition of $G[B' \cup C']$ obtained by attaching new root node $\hat{r}$ to tree decomposition $(T[V_{C'}], \chi)$ and making r' the unique child of $\hat{r}$. Moreover, define $\chi_{C'}(\hat{r}) = B'$ and $\chi_{C'}(v) = \chi(v)$ for all $v \in V_{C'}$. It is clear that $(T_{C'}, \chi_{C'})$ is an $I_{C'}$-sparse tree decomposition of $G[B' \cup C']$ using only bags from Π. Hence $f(B', X', C') \geq w(I_{C'})$. □

We can now prove Lemma 2.28.

Proof. Observe that the maximum weight of an independent set I for which there exists an I-sparse tree decomposition (T, χ) is

$$\max_{B,X} \left\{ w(X) + \sum_{C \in \mathcal{C}(B)} (f(B, X, C) - w(X)) \right\} \tag{2.1}$$

where the maximum is taken over all $B \in \Pi$ and $X \subseteq B$ with $|X| \leq 1$. Hence it suffices to compute $f(B, X, C)$ for every choice of B, X, and C as in Lemma 2.29. By recurrence equation for function f, we know that for computing the value $f(B, X, C)$ we only need to find the values of $f(B', X', C')$ for choices of B', X' and C' with $|C'| < |C|$. Thus it suffices to process the triples (B, X, C) sorted by the size of C. Once we compute the value $f(B, X, C)$ for each triple, we compute the maximum value of an independent set I for which there exists I-sparse tree decomposition of G by already mentioned recurrence. The correctness follows by Lemma 2.29.

Let us estimate the running time of the algorithm. The most time consuming step is to compute values $f(B, X, C)$ for each triple (B, X, C). There are exactly $|\Pi|$ possible choices for B, and for each B we have at most $|B| + 1 \leq n + 1$ choices for X and at most n choices for C. Hence there are at most $O(|\Pi|n^2)$ possible triples.

Let (B, X, C) be a triple. Then we can compute $f(B, X, C)$ in $O(|\Pi|nm)$ with $O(|\Pi|n^2)$ look-ups of the already computed values. Hence the total running time is bounded by $O(|\Pi|^2 n^4 m)$. □

Now by Lemma 2.28 the next theorem follows.

Theorem 2.30. *Let G be a graph and Π a list of potential maximal cliques in G. Then in time $O(|\Pi|^2 n^4 m)$ we can find the maximum weight of an independent set I for which there exists a minimal triangulation H of G, such that every maximal clique C of H is in Π and $|C \cap I| \leq 1$. If no such an independent set exists, the algorithm outputs $-\infty$.*

We have presented the approach as in [43], but the authors stated there that such a statement was previously implicitly proved in [24]. In fact Fomin and Villanger in [24] present the result where the list Π is exhaustive, but as we have seen this is not necessary. In the following section we will describe the idea of how Theorem 2.30 was used for developing a polynomial-time algorithm for the class of P_5-free graphs.

2.2.1 Weighted Independent Set in P_5-free and P_6-free graphs

As we already hinted, the algorithm for Weighted Independent Set in the class of P_5-free graphs relies on Theorem 2.30. Hence, the goal is to construct a suitable family of potential maximal cliques of G. Moreover, Π has to be constructed so that for every independent set I of G there is a minimal triangulation H such that every maximal clique of H intersects I in at most one vertex. In fact, it is known (Lemma 2.32) that for any maximal independent set I of G there exists some minimal triangulation H of G such that every maximal clique C of H satisfies $|C \cap I| \leq 1$.

This means when Π is the list of all potential maximal cliques the algorithm will return the maximum weight of independent set, but in this case the algorithm could run in exponential time. The goal is to find a list Π that is polynomially big in n and m and that for a maximum weight independent set I of G there is a minimal triangulation H of G such that every maximal clique of H is on the list Π and every maximal clique intersects I in at most one vertex.

Lokshtanov et al. [43] used the structure of P_5-free graphs and showed that in the case of P_5-free graph a desired list Π can be found in polynomial time. We will present in detail the construction of Π, omiting most proofs.

Definition 2.31. Let G be a graph and I an independent set of G. We say that a triangulation H is *I-good* if every vertex $v \in I$ satisfies $N_G[v] = N_H[v]$.

Lemma 2.32. *Let G be a graph and I an independent set of G. There exists an I-good minimal triangulation H of G.*

Proof. Let H' be a graph obtained from G by turning $V(G) \setminus I$ into a clique. Then graph H' is a *split graph*, i.e., $V(H')$ can be partitioned into a clique and an independent set. Since every split graph is also chordal it follows that H' is I-good triangulation of G. Now take a minimal subgraph H that is still a triangulation of G. Since H' is an I-good minimal triangulation, so is H. □

Observe now that if H is an I-good triangulation of a graph G for an independent set I of G, then every maximal clique of H intersects I in at most one vertex. The aim is to design a polynomial-time algorithm that for a given P_5-free graph G outputs a family Π such that $|\Pi|$ is polynomial in n and for a maximal independent set I there exists some I-good minimal triangulation of G such that $\zeta(H) \subseteq \Pi$. The family Π is built in two steps as $\Pi = \Pi_1 \cup \Pi_2$.

Π_1 : Define $\delta_G(v)$ as the subset of vertices of $N_G(v)$ with neighbors outside of $N_G(v)$ that is $\delta_G(v) = \{u \in N_G(v) \mid uw \in E \text{ for some } w \notin N_G[v]\}$. For a graph G and every pair $u, v \in V(G)$ of non-adjacent vertices define the graph G_{uv} as the graph obtained from G by turning $\delta_G(u)$ and $\delta_G(v)$ into cliques, and let H_{uv} be a $\{u, v\}$-good minimal triangulation of G_{uv}. In the rest of this section, unless stated differently, let $G = (V, E)$ be a P_5-free graph. Define the family Π_1 as follows:

$$\Pi_1 = \bigcup_{u,v \in V, uv \notin E} \zeta\left(H_{uv}[N_G[u, v]]\right) .$$

Π_2 : It has been shown that Π_1 contains every maximal clique Ω of an I-good minimal triangulation H of a graph G such that $\Omega \subseteq N_H[u,v]$ for some $u, v \in I$. Hence the family Π_2 has to contain every maximal clique Ω of an I-good minimal triangulation H of G such that Ω is not a subset of $N_G[u,v]$ for any $u, v \in I$.

Π_2 is defined as follows:

$$\Pi_2 = \{\Omega \in \Omega(G) \mid \Delta(\Omega) \subseteq \Delta_2\}\,.$$

Recall that $\Delta(\Omega) = \{N(C_i) : 1 \leq i \leq t\}$ and $\mathcal{C}(\Omega) = \{C_1, \ldots, C_t\}$ is the set of all connected components of $G \setminus \Omega$ for $\Omega \subseteq V(G)$. The set Δ_2 is defined as follows.

Definition 2.33. Let G be a graph. We define $\Delta_2 \subseteq \mathcal{P}(V(G))$ as the family of all sets of the form $N_G(\hat{C}_u)$ such that there exists a triple (u, v, w) where

- $\{u, v, w\}$ is an independent set in G,
- C_w is connected component of $G \setminus N_G[u,v]$ containing w,
- $\hat{C}_u$ is the connected component of $G \setminus N_G[C_w]$ containing u.

It is not hard to see that both Π_1 and Π_2 are of polynomial size. Moreover, it has been shown that they can be found in polynomial time and that for every maximal independent set I there exists an I-good minimal triangulation H such that every maximal clique is in Π.

Recently, Grzesik et al. used the same approach to solve the Weighted Independent Set problem in the class of P_6-free graphs. In fact, they prove the following theorem.

Theorem 2.34 (Grzesik et al. [32]). *Given a P_6-free graph on n vertices, one can in polynomial time compute a polynomial-size family Π of vertex subsets with the following property: for every maximal independent set I in G, there exists an I-good triangulation H of G such that Π contains all maximal cliques of H.*

The proof of the above theorem is the main contribution of the paper of Grzesik et al. [32]. Such a theorem easily gives a polynomial-time algorithm for P_6-free graphs.[1]

Let G be a P_6-free graph. By Lemma 2.32 we know that for any maximal independent set I of G there exists an I-good minimal triangulation. By Theorem 2.34 we can in polynomial time find a polynomially sized family Π that for each maximal independent set I contains all bags of some I-good triangulation H of G. Then by Theorem 2.30 it follows that the Weighted Independent Set problem is polynomialy solvable for P_6-free graphs.

Theorem 2.35. *[32] The* WEIGHTED INDEPENDENT SET *problem can be solved in polynomial time for P_6-free graphs.*

[1]It should be noted that at the time of this writing the result has not yet been peer reviewed.

2.3 Modular partition

One of the techniques for solving the Weighted Independent Set problem on a specific graph class is the so called *modular decomposition* or *modular partition.* The ideas of modular partition were used in obtaining a quasipolynomial-time algorithm for the Weighted Independent Set problem for P_6-free graphs, as well as in several other algorithms. More precisely, an algorithm running in time $n^{O(\log^2 n)}$ [45] was developed for the class of P_6-free graphs. A linear-time algorithm was achieved for the class of cographs graphs using this technique.

Given a graph G, a set of vertices $M \subseteq V(G)$ is called a *module* if every vertex not contained in M is either adjacent to all vertices of M or to none of them. A module M is trivial if M is the whole vertex set of G, $\{v\}$ for a vertex v, or the empty set. It is obvious that every graph contains trivial modules. If graph G does not contain any non-trivial module, then G is said to be a *prime* graph.

An important property of modules is the following.

Proposition 2.36 (Gallai [26]). *If G and its complement are both connected, then maximal modules are pairwise disjoint. Moreover, if both M and M' are maximal modules then either M dominates M' or M is anticomplete to M'.*

Using the above proposition we conclude that the set of all maximal non-trivial modules of a connected and co-connected graph G forms a partition of $V(G)$. This property can be used for a reduction rule for Weighted Independent Set on G. The reduction consists in finding maximal modules of G and contracting them to single vertices to obtain a graph G^0. We say that two sets of vertices X and Y of G are adjacent if there exists an edge xy with $x \in X$ and $y \in Y$.

It is important to observe that graph G^0 constructed at step 9 is an induced subgraph of G since each M_i is a module in G. Algorithm 1 is a recursive algorithm. The root of recursion tree is graph G. At each step we decompose graph G until we reach a graph on a single vertex. Observe that at the step 10 graph G^0 is either an edgeless graph, a complete graph, or a prime graph and we only solve Weighted Independent Set problem on such graphs. Since we can easily solve Weighted Independent Set problem on edgeless graphs and complete graphs, it follows that modular decomposition reduces the problem from a graph to its induced prime graphs. If we are able to solve the problem for the class of all prime induced subgraphs of the input graph, Algorithm 1 will give us a polynomial-time algorithm for the starting class also.

Algorithm 1: $\textsc{Alpha}(G, w)$

```
Input: A graph $G$ and a weight function $w : V(G) \to \mathbb{R}$.
Output: A maximum weight independent set $I$.
1  if $|V(G)| = 1$ then
2      set $I = \text{argmax}\{w(V(G)), w(\emptyset)\}$ and goto 11;
3  if $G$ is disconnected then
4      partition $V(G)$ into vertex sets of connected components $M_1, \dots M_k$;
5  if $\overline{G}$ is disconnected then
6      partition $V(G)$ into vertex sets of connected components $M_1, \dots, M_k$ of $\overline{G}$;
7  if $G$ and $\overline{G}$ are connected then
8      partition $V(G)$ into maximal modules $M_1, \dots, M_k$;
9  Construct a weighted graph $(G^0, w^0)$ such that $V(G^0) = \{M_1, \dots, M_k\}$ and two
     vertices $M, M'$ of $G^0$ are adjacent if and only if they are adjacent in $G$. Set
     $w^0(M) = w(\textsc{Alpha}(G[M], w))$ for every $M \in V(G^0)$;
10 Find a maximum weight independent set $I^0$ in $(G^0, w^0)$ and set
     $I = \cup_{M \in I^0} \textsc{Alpha}(G[M], w)$;
11 return $I$;
```

Theorem 2.37 (Lozin and Milanič [49]). *Let $\mathcal{C}$ be a class of graphs and let $\mathcal{C}^*$ be the class of all prime induced subgraphs of the graphs in $\mathcal{C}$. If for a constant $p \geq 1$, the maximum weight independent set problem in the class $\mathcal{C}^*$ can be solved in time $O(n^p)$, then the same problem is solvable in time $O(n^p + m)$ for the class $\mathcal{C}$.*

Proof. Let $G \in \mathcal{C}$. Denote $n = |V(G)|$ and $m = |E(G)|$. It has been shown by McConnell and Spinrad in [54] that the recursive decomposition of Algorithm 1 can be performed in time $O(n + m)$. Such a decomposition finds the recursion tree $T(G)$ of G. Every leaf of $T(G)$ corresponds to a vertex of G and every internal leaf corresponds to an induced subgraph of G on at least two vertices.

Let U be an internal node of $T(G)$ and let G_U be the induced subgraph of G corresponding to U. In this case, the children of U in $T(G)$ correspond to the graphs $G[M_1], \dots, G[M_k]$, where $\{M_1, \dots, M_k\}$ is the partition defined as in the steps 3-8 of the algorithm. If G_U or $\overline{G_U}$ is disconnected then we can trivially find a maximum weight set in G_U^0 (step 10) in time $O(|V(G_U^0)|)$. On the other hand, if both G_U and $\overline{G_U}$ are connected then G_U^0 is a prime graph, i.e., $G_U^0 \in \mathcal{C}^*$. Hence, by assumption we can do step 10 in time $O(|V(G_U^0)|^p)$. Summing over all vertices of $T(G)$ we infer that the total running time of the algorithm is bounded by $O(\sum_{U \in V(T(G))} |V(G_U^0)|^p)$. One can easily show by induction that the total number of vertices in all G_U^0 corresponding to the internal vertices of $T(G)$ is exactly equal to number of edges in $T(G)$. Since there are exactly n leaves in $T(G)$, there are at most $n-1$ internal vertices in $T(G)$ and $|V(T(G))| \leq 2n-1$. Hence

$$\sum_{U \in V(T(G))} |V(G_U^0)|^p \leq \left(\sum_{U \in V(T(G))} |V(G_U^0)| \right)^p \leq (2|V(T(G))| - 2)^p \leq (2n-2)^p = O(n^p).$$

Adding the term $O(n+m)$ necessary for the recursive decomposition we obtain the claimed time complexity. $\square$

Modular decomposition was used in the development of several polynomial-time algorithms for Weighted Independent Set problem in hereditary classes. An easy example of a linear-time algorithm for the maximum weight independent set can be obtained for the class of *cographs* (P_4-free graphs). It is known that for every cograph G on at least two vertices either G or $\overline{G}$ is disconnected graph [17]. Hence the only induced prime graphs of a cograph considered in Algorithm 1 are graphs on a single vertex. Therefore, Theorem 2.37 guarantees the existance of a linear-time algorithm for the problem.

Several other polynomial-time algorithms were developed for subclasses of P_5-free graphs and fork-free graphs. More generally, a polynomial-time algorithm for the whole class of fork-free graphs [49] and a quasipolynomial-time algorithm for the class of P_6-free graphs [45] rely on modular decomposition.

It is worth noting that modular decomposition is used for many other algorithmical and structural results. In fact the first algorithm for recognition of comparability graphs uses modular decomposition and this is the first appearance of modular decomposition in the literature [26]. Similarly, modules play an important role in the Lovász's proof of perfect graph theorem [46].

2.4 Decomposition by clique separators

We say that a subset $S \subseteq V(G)$ is a *separator* if $G \setminus S$ is disconnected graph. Moreover, if S is a clique we say that S is a clique separator. If a graph G does not admit a clique separator, we say that a graph G is an *atom*. It is well-known that the Weighted Independent Set problem can be reduced in polynomial time to graphs without a clique separator. Such an approach is known as the *decomposition by clique separators*. It was originally developed by Whitesides [71], and later generalized by Tarjan [69] and Alekseev [3] (for the unweighted case).

Let $\mathcal{C}$ be a hereditary class of graphs. If Weighted Independent Set can be solved in polynomial time for the graphs in $\mathcal{C}$ that have no clique separator, i.e., atoms, then the same problem is polynomially solvable for the class $\mathcal{C}$. We recall the following recursive method described by Tarjan.

Algorithm 2: DECOMPOSITION BY CLIQUE SEPARATORS

```
    Input: A graph $G$ and a weight function $w : V(G) \to \mathbb{R}$.
    Output: $\alpha(G)$.
1   Let $\{A, B, C\}$ be a partition of $V(G)$ such that $C$ is a clique separator and
      $G[A \cup C]$ is an atom.;
2   for $v \in C$ do
3       Find a maximum weight of an independent set $I(v)$ in $G[A \setminus N(v)]$;
4   Find a maximum weight independent set $I'$ in $G[A]$;
5   for $v \in C$ do
6       Redefine the weight of $v$ as $w(v) + w(I(v)) - w(I')$;
7   Find a maximum weight independent set $I''$ in $G[B \cup C]$ with respect to the new
      weights;
8   Define
      $$I = \begin{cases} I(v) \cup I'' & \text{if } v \in I'' \cap C\,; \\ I' \cup I'' & \text{if } I'' \cap C = \emptyset\,. \end{cases}$$
    return $I$;
```

Similarly, the decomposition by clique separators can be used for many other classical graph problems such as coloring and maximum clique problem. Moreover, clique separators are often used in structural description of hereditary graphs classes. As an example, we consider the following proposition without the proof.

Proposition 2.38 (Dirac [20]). *Let G be a chordal graph. Then G has a clique separator or G is complete.*

Such a proposition not only allows for the developlment of a linear-time algorithm for coloring chordal graphs, but also provides a more space efficient implementations of a coloring algorithm for chordal graphs than the one obtained using the so called perfect elimination scheme [33].

2.5 Graph transformations

In this chapter we present several simple graph transformations that can be used to decrease the size of an instance for the INDEPENDENT SET problem. Observe that such transformations could be used in combination with other techniques very often for development of new polynomial-time algorithms. Furthermore such transformations can be performed mostly in linear or quadratic time. As such, they are a nice tool for solving particular instances of a problem in real life, by repeatedly reducing the instance and then solving the problem.

For an exact polynomial-time algorithm in a particular graph class, usually the goal it to make several graph transformations and reduce an instance G to an instance G' for which we already know a polynomial-time algorithm. We present an overview following Lozin [47].

Let v be a vertex of degree one in a graph G and let u be its unique neighbor. Let I be any maximal independent set in G. If I contains u then $v \notin I$ and if that is the case, then we can define an independent set $I' = (I \setminus \{u\}) \cup \{v\}$. Obviously $|I| = |I'|$. Hence $\alpha(G) = \alpha(G \setminus \{u, v\}) + 1$. We can repeatedly make such a transformation until we obtain a graph without vertices of degree one. In particular, for trees, this reduction gives us an efficient algorithm for Independent Set. Reduction of vertices of degree one can be generalized in several ways.

2.5.1 Simplicial vertex reduction

A vertex v of G is said to be *simplicial* if $N(v)$ is a clique. Trivially, every vertex of degree one is simplicial. Now, let us design a similar reduction as before. Let I be an independent set and $v \in V(G)$ a simplicial vertex. If $N(v) \cap I \neq \emptyset$ then, there exists a unique vertex $u \in N(v)$ such that $u \in I$. Again, observe that for the independent set $I' = (I \cup \{v\}) \setminus \{u\}$ it holds that $|I| = |I'|$. Hence, $\alpha(G) = \alpha(G \setminus N(v)) + 1$.

Such a reduction can be done in polynomial time. Moreover, in any non-complete graph that contains a simplicial vertex there exists a clique cutset and in this case we could reduce the graph also by using the decomposition by clique separators.

2.5.2 Neighborhood reduction

Let uv be an edge of G. Define the following sets

- $C_u = N(u) \setminus N[v]$,
- $C_v = N(v) \setminus N[u]$,
- $C_{uv} = N(v) \cap N(u)$, and
- $C_0 = V(G) \setminus N[\{u, v\}]$.

Then $\{C_u, C_v, C_{u,v}, C_0\}$ is a partition of $V(G) \setminus \{u, v\}$. Suppose that $C_v = \emptyset$. Then for any independent set I containing u, the set $I' = (I \setminus \{u\}) \cup \{v\}$ is also independent set. Therefore, it holds $\alpha(G) = \alpha(G \setminus \{u\})$. Observe that in case when C_{uv} is a clique then this reduction is equivalent to a simplicial vertex reduction.

2.5.3 Removal of constantly many vertices

In the following, we show a technical lemma useful for removal of constantly many vertices. The idea of such an approach, is that for every graph G in a class $\mathcal{C}$ we can remove a set of constantly many vertices to obtain a graph G' such that $G' \in \mathcal{C}'$ and we already know a polynomial-time algorithm for maximum weight independent set in $\mathcal{C}'$.

Theorem 2.39. *Let $\mathcal{C}$ be a class of graphs such that there exists a constant p and a hereditary class $\mathcal{C}'$ such that*

- Weighted Independent Set *is solvable in polynomial time in class $\mathcal{C}'$,*
- *for each graph $G \in \mathcal{C}$ we can find, in polynomial time, a subset of vertices U such that $|U| < p$ and $G \setminus U \in \mathcal{C}'$.*

Then Weighted Independent Set *problem can be solved in polynomial time for graphs in $\mathcal{C}$.*

Proof. Let $G \in \mathcal{C}$. We show how to find a maximum weight independent set in G. First we find a subset U such that $|U| < p$ and $G \setminus U \in \mathcal{C}'$. Then, we enumerate in constant time all independent sets (not necessarily maximal) $\{I_1, \dots, I_t\}$ of $G[U]$. For each $i \in \{1, \dots t\}$ we find a maximum weight independent set I_i' in $G \setminus (U \cup N(I_i))$. Such a procedure can be done in polynomial time and clearly the solution of Weighted Independent Set problem is given by the maximum weight of $I_i \cup I_i'$ where $i \in \{1, \dots t\}$. □

Observe that if $\mathcal{C}'$ is a hereditary class for which the recognition problem is polynomially solvable, it suffices just that a subset U as above exists. Such a theorem was used in [56].

It should be noted that the removal of constantly many vertices is not a proper graph transformation, in a sense that we are allowed to perform it only once (or at most constantly many times), while other graph transformations can be done polynomially many times.

2.6 Other techniques

Graphs of bounded width parameters: Due to Courcelle's theorem, we know that every problem expressible in Monadic Second Order Logic (with quantifier over both vertex- and edge-subsets) can be solved in linear time on graphs of bounded treewidth [4]. This means that for a class of graphs $\mathcal{C}$ for which there exists $k \in \mathbb{N}$ such that for every $G \in \mathcal{C}$ it holds $tw(G) \leq k$, Independent Set is solvable in linear time when the input is restricted to $\mathcal{C}$.

On the other hand, a result due to Courcelle et al. [18] shows that every problem expressible in Monadic Second Order Logic with quantifiers only over vertex subsets, can be solved in linear time for graphs of bounded clique width. For a graph G the clique width $cw(G)$ is the minimum number of labels such that a vertex-labeled graph isomorphic to G can be constructed using the following four operations:

- Creation of a new vertex v with label i.
- Disjoint union of two labeled graphs H and H'.
- Joining by an edge every vertex labeled i to every vertex labeled j for $i \neq j$.
- Renaming label i to label j.

Since Independent Set is expressible in Monadic Second Order Logic with quantifies over vertex subsets only, we conclude that the problem is polynomially time solvable for any class of graphs of bounded clique width. Observe that for every graph such that $tw(G) \leq k$ we have $cw(G) \leq 3 \cdot 2^{k-1}$ [34]. Hence, every graph class of bounded treewidth has also bounded clique width, but it can be easier to prove that a certain graph class has bounded treewidth. Together with modular decomposition, this technique was used to solve efficiently the problem in some subclasses of fork-free and P_5-free graphs [22].

Another graph parameter that is often considered is so called rank-width. The notion of rank-width, denoted by $rwd(G)$, is defined by Oum and Seymour [62] so as to yield an approximation algorithm for clique-width. They also showed that rank-width and clique-width are in a sense equivalent; more precisely, for any graph G the following inequalities holds:

$$\log_2(cw(G)+1) - 1 \leq rwd(G) \leq cw(G)\,.$$

Thus, a set of graphs of bounded rank-width is also of bounded clique-width and vice versa. Hence, the same as before, Independent Set is polynomially solvable for any classs of graph of bounded rank-width.

Several other techniques were developed for Independent Set and Weighted Independent Set in special graph classes and we will just list a few of them here.

- Using network flow techniques Weighted Independent Set can be solved in the class of bipartite graphs [16].

- *Semidefinite programming* was used to solve Weighted Independent Set for perfect graphs [31].
- Special *dynamic programming* approaches have been designed for graphs in particular classes, based on their structural properties and characterizations. Example include interval graphs [66] and distance-hereditary graphs [8].

3 Equistable graphs

Closely related to the Independent Set and Weighted Independent Set problems is the class of equistable graphs. We say that a graph G is *equistable* if there exists a function $\varphi : V(G) \to \mathbb{R}_+$ such that for every subset of vertices I it holds that $\varphi(I) = 1$ if and only if I is maximal independent set. Equivalently, G is equistable if and only if there exists a positive integer t and a weight function $w : V(G) \to \mathbb{N}$ such that $I \subseteq V(G)$ is a maximal independent set if and only if $w(I) = t$. Such a function w is called an *equistable function* of G, while the pair (w, t), is called an *equistable structure*. The class was introduced by a French mathematician Charles Payan in 1980 [64].

Let us show that the above two defintions are indeed equivalent. It suffices to show that if a graph is equistable with respect to a function $\psi : V(G) \to \mathbb{R}_+$, then there exists an equistable structure for G.

Suppose that a graph $G = (V, E)$ with $V = \{1, \dots, n\}$ admits non-negative real weights $\varphi(x)$, for $x \in V$, such that a subset of vertices is of φ-weight 1 if and only if it is a maximal stable set. Partition the non-empty subsets of V that are not maximal stable sets into two parts: a set is either *light* or *heavy* depending on whether its φ-weight is smaller than or greater than 1.

The weight function φ can be seen as a vector in $\mathbb{R}^n$ that satisfies the following system of linear equations and inequalities (with variables $x_1, \dots, x_n$):

- $x_i \geq 0$ for all $i \in S$
- $\sum_{i \in S} x_i = 1$ for every maximal stable set S of G,
- $\sum_{i \in T} x_i < 1$ for every light set of vertices T,
- $\sum_{i \in T} x_i > 1$ for every heavy set of vertices T.

Let $\epsilon > 0$ be any positive rational number such that

$$\epsilon < \min \left\{ \sum_{i \in T} x_i - 1 \mid T \text{ is a heavy set} \right\}$$

and

$$\epsilon < \min \left\{ 1 - \sum_{i \in T} x_i \mid T \text{ is a light set} \right\}.$$

Notice that such an ϵ exists, since both minima above are positive. Then, the weight function φ satisfies the following system of linear equations and inequalities (with variables $x_1, \ldots, x_n$):

- $x_i \geq 0$ for all $i \in S$
- $\sum_{i \in S} x_i = 1$ for every maximal stable set S of G,
- $\sum_{i \in T} x_i \leq 1 - \epsilon$ for every light set T,
- $\sum_{i \in T} x_i \geq 1 + \epsilon$ for every heavy set T.

Considering ϵ as fixed, this system of linear equations and inequalities (over the variables $x_1, \ldots, x_n$) can be transformed into an equivalent system of a form $Az \leq b$, where A and b are a matrix and a vector of suitable dimensions have only rational coefficients. Since the original system has a feasible solution (namely, φ), so does the transformed system.

This system defines a non-empty bounded polyhedron P. Since all coefficients in the inequalities defining P are rational, each vertex of P has only rational coordinates. Let z^* be any vertex of P. We can transform z^* back to a feasible solution of the system

- $x_i \geq 0$ for all $i \in S$
- $\sum_{i \in S} x_i = 1$ for every maximal stable set S of G,
- $\sum_{i \in T} x_i \leq 1 - \epsilon$ for every light set of vertices T,
- $\sum_{i \in T} x_i \geq 1 + \epsilon$ for every heavy set of vertices T.

This gives an equistable weight function of G with rational weights, and by scaling we can obtain integer weights.

Payan defined the class of equistable graphs as a generalization of threshold graphs. Recall that a graph G is *threshold* if there exists a weight function $w : V(G) \to \mathbb{N}$ on the vertices of G and a *threshold* t such that a subset of vertices $S \subseteq V(G)$ is independent if and only if $w(S) \leq t$. Threshold graphs are a well-known and well-studied class of graphs [52].

Theorem 3.1 (Payan [64]). *Threshold graphs are equistable.*

Proof. Let G be a threshold graph. We prove the lemma by induction on $|V(G)|$.

It is known that every threshold graph contains a vertex v that is either an isolated vertex or a dominating vertex (i.e., a vertex adjacent to every other vertex in a graph) [52]. Recall also that the class of threshold graphs is hereditary. Denote with (w', t') an equistable structure of $G \setminus \{v\}$.

Case 1: v is an isolated vertex. Then a subset of vertices $S \subseteq V(G)$ is maximal independent set in G if and only if $v \in S$ and the set $S' = S \setminus \{v\}$ is a maximal independent set in $G \setminus \{v\}$. And conversly, a set $S' \subseteq V(G) \setminus \{v\}$ is a maximal independent set in $G \setminus \{v\}$ if and only if $S' \cup \{v\}$ is a maximal independent set in G. Then integer $t = 2t' + 1$ and the function w defined as $w(u) = 2w'(u)$ for every $u \in V(G) \setminus \{v\}$ and $w(v) = 1$ form an equistable structure of G.

Case 2: v is a dominating vertex. An independent set $S \subseteq V(G)$ is maximal in G if and only if either $S = \{v\}$, or $v \in S$ and S is a maximal independent set in $G \setminus \{v\}$. And conversly, a set $S' \subseteq V(G) \setminus \{v\}$ is a maximal independent set of $G \setminus \{v\}$ if and only if it is a maximal independnet set of Gx. Then function w defined as $w(u) = w'(u)$ for every $u \in V(G) \setminus \{v\}$ and $w(v) = t'$ and t' form an equistable structure of G. □

Besides threshold graphs, equistable graphs generalize the class of cographs and general partition graphs [55]. We say that a graph G is a *general partition* graph if there is some set U and an assignment of non-empty subsets $U_x \subseteq U$ to the vertices of G such that two vertices x and y are adjacent if and only if $U_x \cap U_y \neq \emptyset$ and for every maximal independent set I the set $\{U_i : i \in I\}$ is a partition of U.

A clique in a graph is a *strong clique* if it has non-empty intersection with each maximal independent set of the graph. Moreover, a collection of cliques is a *clique cover* of G if for every edge of G there is a clique in the collection that contains both its endpoints. Finally, a clique cover is a *strong* clique cover if all of its cliques are strong cliques.

Theorem 3.2 (McAvaney et al. [53]). *A graph G is a general partition graph if and only if there is a strong clique cover of G.*

It is worth to notice that the class of equistable graphs is not hereditary. The later follows trivially from the following proposition.

Proposition 3.3. *Let G be a graph. Then G is an induced subgraph of some equistable graph H.*

Proof. Let G be a graph. We construct a graph H from G as follows. For every maximal clique C of G we add a new vertex v_C. Furthermore we connect the new vertex with the vertices of G so that $N(v_C) = C$. We denote the graph obtained this way by H. Clearly G is an induced subgraph of H. Since every general partition graph is equistable, it suffices to prove that H is a general partition graph. We use Theorem 3.2. Let us show that H has a strong clique cover. It follows from construction that every maximal clique in H is a union of a maximal clique C in G and the singleton $\{v_C\}$. Furthermore, such a clique is strong since v_C is adjacent exactly to C. Hence the set of all maximal cliques in H is a strong clique cover of H. The proof follows. $\square$

In fact, the above proof shows that any graph is an induced subgraph of some general partition graph.

The complexity status of recognizing equistable graphs is still open. It is not even known if the problem is in NP. Furthermore, no combinatorial characterization of equistable graphs is known. Orlin conjectured in 2009 [55] that the class of equistable graphs is in fact the class of general partition graphs, but this was recently disproved by Milanič and Trotignon [58]. Given a graph G and a weight function $w : V(G) \to \mathbb{N}$ it is co-NP-complete to determine if w is an equistable function of G [57]. On the other hand, the recognition of k-equistable graphs can be done in linear time. For a positive integer k, a graph G is said to be *k-equistable* if it admits an equistable function $w : V \to \{1, \dots, k\}$. Such a function is called a k-equistable function, and the corresponding structure (w, t) is a k-equistable structure of G.

For a positive integer t, we say that a graph G is *target-t equistable* if it admits an equistable function $w : V(G) \to \mathbb{N}$ with equistable structure (w, t). Observe that every target-t equistable graph is also t-equistable, but the other direction does not hold. In Section 3.1 we will show that recognition of k-equistable graphs can be done in linear time.

It was proved by Levit et al. that for every fixed k there is a polynomial-time algorithm to test if a given graph is k-equistable.

Theorem 3.4 (Levit et al. [41])**.** *For every fixed k, there is an $O(n^{2k})$ algorithm to decide whether a given n-vertex graph G is k-equistable. Furthermore, in case of a positive instance, the algorithm also produces a k-equistable structure of G.*

The above result was further improved by Kim et al. [37] and we will present a part of their work here. In fact, they introduced the notion of target-t equistable graphs and using it obtained the mentioned linear-time algorithm for the recognition of k-equistable graphs.

We say that two vertices u and v of a graph G are *twins* if they have the same open neighborhood. Clearly, the relation of being twins in a graph is an equivalence relation.

Lemma 3.5 (Levit et al. [41])**.** *Let $G = (V, E)$ be a graph. The twin relation is an equivalence relation and every equivalence class is either a clique or an independent set.*

We refer to an equivalence class of the twin relation as a *twin class*. Moreover we will refer to twin classes that are cliques as *clique classes* and similarly for *independent set classes*. We will denote the set of all twin classes as $\Pi(G)$ and call it the *twin partition* of G. The number of twin classes of a graph G is denoted with $\pi(G)$. Since the twin relation is an equivalence relation, it follows that any two distinct twin classes are either complete or anticomplete to each other. The later allows us to define the *quotient graph* $\mathcal{Q}(G)$ having the vertex set $\Pi(G)$, two twin classes being adjacent if and only if they are complete to each other. Also, given a graph G, it is possible to find in linear time the twin partition $\Pi(G)$ and the quotient graph $\mathcal{Q}(G)$ [37].

The following lemmas show the importance of twin partition in the study of equistable graphs.

Lemma 3.6 (Levit et al. [41]). *For every equistable function w of G and for every $i \in \mathbb{N}$, the set of the form $V_i^w = \{x \in V : w(x) = i\}$ is a subset of a twin class of G. In particular, if G is a k-equistable graph, then $\pi(G) \leq k$.*

Lemma 3.7 (Levit et al. [41]). *For every equistable function w of an equistable graph G and for every clique class C there exists an i such that $V_i^w = C$.*

3.1 Recognizing k-equistable graphs

Parametrized Complexity. Rather than speaking about a linear-time algorithm for recognizing k-equistable graph, we will consider the problem of recognizing equistable graphs parametrized by k. We say that a problem is *fixed-parameter tractable* (FPT) when parametrized by solution size k, if there is an algorithm running in time $O(f(k)n^c)$ for some function f and some constant c. More generally, a problem is fixed-parameter tractable (FPT) with respect to parameter k (e.g., solution size, tree width, ...) if for any instance of size n it can be solved in time $O(f(k)n^c)$ for some fixed c. One of the main tools to design such algorithms is the so-called kernelization technique. A kernelization is a polynomial-time algorithm that transforms an instance (I, k) of a parameterized problem into an equivalent instance (I_0, k_0) of the same problem such that the size of I_0 is bounded by $g(k)$ for some computable function g and k_0 is bounded by a function of k. The instance I_0 is said to be a *kernel* of size $g(k)$. It is known that a parameterized problem is fixed-parameter tractable if and only if it admits a kernelization.

Usually we consider whether a problem is FPT if the problem is NP-hard. In that case the function f is not bounded by a polynomial, unless $P \neq NP$. In other words, for fixed-parameter tractable problems the difficulty of a problem is not in input size, but rather in the size of solution (parameter). In our case the parameter is a number

k and we are asking if a given graph is k-equistable. It is still not known if recognizing equistable graphs is NP-hard, but it seems plausible that this is the case.

In fact, we consider two different problems.

k-EQUISTABILITY:

Input: A graph G.
Parameter: k.
Output: TRUE if and only if G is k-equistable.

TARGET-t EQUISTABILITY:

Input: A graph G.
Parameter: t.
Output: TRUE if and only if G is target-t-equistable.

The main idea for fixed-parameter tractable algorithm for TARGET-t-EQUISTABILITY relies on the idea of r-clique reduction.

r-**Clique Reduction:** If a clique class C of the twin partition of G contains more than r vertices, we remove all but r vertices.

Lemma 3.8 (Kim, Milanič, and Schaudt [37]). *Let G be a graph, $T \subset \mathbb{N}$ a finite set, C a clique class of G with $|C| > r$ where $r = \max T$ and k a positive integer. Then, for every $t \in T$, graph G is target-t k-equistable if and only if G' is target-t k-equistable, where G' is graph obtained from G after the r-Clique Reduction has been applied to a clique class C of G.*

Proof. Let $t \in T$. Assume that G is target-t k-equistable. Let (w, t) be a k-equistable structure for G. It is immediate by definition of r-Clique Reduction that the restriction w' of w to $V(G')$ yields a k-equistable structure (w', t) of G'. Therefore G' is target-t k-equistable.

Let us show the other direction. Assume that G' is target-t k-equistable. Let (w', t) be a k-equistable structure of G'. We define a function $w : V(G) \to \{1, \dots, k\}$ by extending w' to the set $V(G)$. Indeed, we simply put $w(u) = w'(u)$ for all $u \in V(G')$, and $w(u) = w(v)$ for all $u \in C \backslash V(G')$ where $v \in C \cap V(G')$. The choice of $v \in C \cap V(G')$ is arbitrary, since w' is constant on $C \cap V(G')$ by Lemma 3.7. We claim that (w, t) is an equistable structure of G. To show this, pick an arbitrary maximal independent set I of G. Then $|I \cap C| \leq 1$, and so we may assume that $I \subseteq V(G')$. Clearly, I is a maximal independent set of G' , and so $w'(I) = t$. Therefore $w(I) = t$. Conversely, let $I \subseteq V(G)$ be a set with $w(I) = t$. Since $w(I \cap C) \leq w(I) = t$, we have $|I \cap C| \leq t$. As w is constant on C and $|C| > \max T \geq t$, we may assume w.l.o.g. that $I \subseteq V(G')$. Hence, $w'(I) = w(I) = t$, and so I is a maximal independent set of G' . Thus, I is a maximal independent set of G, which completes the proof. □

Using the above lemma we will show that Target-t-Equistability admits a kernel with at most t^2 vertices.

3.1.1 A kernel for the Target-t-Equistability

Theorem 3.9 (Kim, Milanič and Schaudt [37]). *The* Target-t-Equistability *problem admits a kernel of at most t^2 vertices, computable in linear time.*

Proof. Let G be a graph. As said before, we can find $\Pi(G)$ and consequently $\pi(G)$ in linear time. If $\pi(G) > t$, then we conclude that G is not target-t equistable, by Lemma 3.6. Similarly, if there exists an independent set class S with $|S| > t$, then we conclude that G is not target-t equistable since such an independent set has size more than t. Also, we can apply r-Clique Reduction rule with parameter t, to every clique class, in linear time. So the obtained graph is a graph with at most t twin classes each of which is an independent class on at most t vertices. Hence, the graph has at most t^2 vertices, which proves the theorem. □

Using the above theorem and the algorithm given by Theorem 3.4 we obtain an FPT algorithm for Target-t-Equistability, but it is possible to improve the running time of such an algorithm as shown in [37].

3.1.2 A kernel for k-Equistability

We will show the main proof idea of the following theorem.

Theorem 3.10 (Kim, Milanič and Schaudt [37]). *The* k-Equistability *problem admits an $O(k^2)$-vertex kernel, computable in linear time.*

Proof. We may assume that the input graph G satisfies $\pi(G) \leq k$, since otherwise G is not k-equistable, by Lemma 3.6. The following claim is the main step of kernelization and we use it without the proof.

Claim 1: If there exist two distinct twin classes X and Y such that one of them is a independent set and $\min\{|X|, |Y|\} \geq k(k + 1)$, then G is not k-equistable.

We consider the following two cases.

Case 1. Every twin class X with $|X| \geq k(k+1)$ is a clique class. In this case, every independent set class has less than $k(k+1)$ vertices, which implies that every maximal independent set of G contains at most $k(k+1)$ vertices from each twin class and is thus of total size at most $k^2(k + 1)$. Hence for every k-equistable structure (w, t) of G, we have $t \leq k^3(k + 1)$. We now perform r-Clique Reduction rule with $r = k^3(k + 1)$. By Lemma 3.8 applied with $T = \{1, \ldots, r\}$ and k, the application of r-Clique Reduction-rule is a valid operation. When the rule can no more be applied, we have a graph G' with at most k twin classes, each of size at most $k^3(k+1)$. We are done since $|V(G')| = O(k^5)$.

Case 2. There exists an independent set twin class X with $|X| \geq k(k+1)$. By Claim 1, we may assume that X is the unique twin class of size at least $k(k+1)$ (since otherwise G is not k-equistable). Note that $V(G)\setminus X$ contains at most $k-1$ twin classes, each containing less than $k(k+1)$ vertices, hence $|V(G) \setminus X| \leq (k-1)k(k+1) \leq k^3$. Suppose first that X corresponds to an isolated vertex in the quotient graph $\mathcal{Q}(G)$. If $|X| < k^5$, then $|V(G)| < k^5 + k^3 = O(k^5)$ and we are done. So suppose that $|X| \geq k^5$.

Claim 2: G is k-equistable if and only if it admits a k-equistable function that is constant on X.

We use Claim 2 without giving a proof. Using the claim, it suffices to test if G is k-equistable by considering all possible functions $w : V(G) \to \{1, \ldots, k\}$ that are constant on X, and test for each of them whether it is a k-equistable function. Before that, we reduce size of X. For this, we compute a graph G' from G by deleting all but k^4 many vertices from X. Note that, since X is a twin class, G' is unique up to isomorphism.

Claim 3: G is k-equistable if and only if G' is k-equistable.

Again, we omit the proof. By Claim 3, it suffices to check whether G' is k-equistable. Since $|V(G')| \leq k + k^3 = O(k^3)$, we are done.

Now, suppose that X corresponds to a non-isolated vertex in the quotient graph $\mathcal{Q}(G)$. Then, there exists a twin class Y that is complete to X. Let I be a maximal independent set of G containing a vertex of Y . Then, $I \subseteq V(G)\setminus X$. Since $|V(G)\setminus X| \leq k^3$, we have in particular that $|S| \leq k^3$. If $|X| > k|S|$, then for every k-equistable function w of G and every maximal independent set S' , such that $X \subseteq S'$, we have $w(S') \geq |X| > k|S| \geq w(S)$, hence G is not k-equistable. If $|X| \leq k|S|$, then $|V(G)| \leq (k+1)k^3 = O(k^4)$. The proof follows. □

3.2 Independent Set in equistable graphs

In this section we will mention several hardness results for the Independent Set and Weighted Independent Set problems in the class of equistable graphs. Recall that the class of equistable graphs is not hereditary. Moreover, the class of equistable graphs is not contained in any non-trivial hereditary class nor contains any hereditary class of graphs for which the problems are known to be NP-hard.

The following theorem shows that Weighted Independent Set is NP-hard in the class of equistable graphs. With some extra work in the same paper it has been also showed that Independent Set is also NP-hard. Recall that if the problem is APX-hard there is no algorithm running in time $O(f(\epsilon) \cdot p(n))$ where $p(n)$ is a polynomial that returns a solution that is not worse than $1 - \epsilon$ times the optimal value, unless P $\neq NP$ [63]. It is clear that if a problem is APX-hard that it is also NP-hard.

Theorem 3.11 (Milanič, Orlin, and Rudolf [57])**.** *Finding a maximum weight independent set in a vertex weighted equistable graph is APX-hard, even if the graph is given together with an equistable structure.*

We will present the proof as given by Milanič et al. [57].

Proof. The results is proved using a reduction from the Independent Set problem in general graphs to the same problem in an equistable graphs. Recall that the Independent Set problem is APX-hard for general graphs [5]. Let $G = (V, E)$ be a graph. Denote its vertex set as $\{1, \ldots, n\}$. An equistable graph $G' = (V', E')$ is formed as follows:

- $V' = \{v_1, \ldots, v_n\} \cup \{w_1, \ldots, w_n\} \cup \{u_e : e \in E\}$,
- for each $j \in \{1, \ldots, n\}$ there is an edge $v_j w_j$ in E', and
- for every edge $e = ij$ in G, there are edges $v_i v_j$, $v_i u_e$ and $v_j u_e$ in G'.

For an example of such a construction see Figure 8.

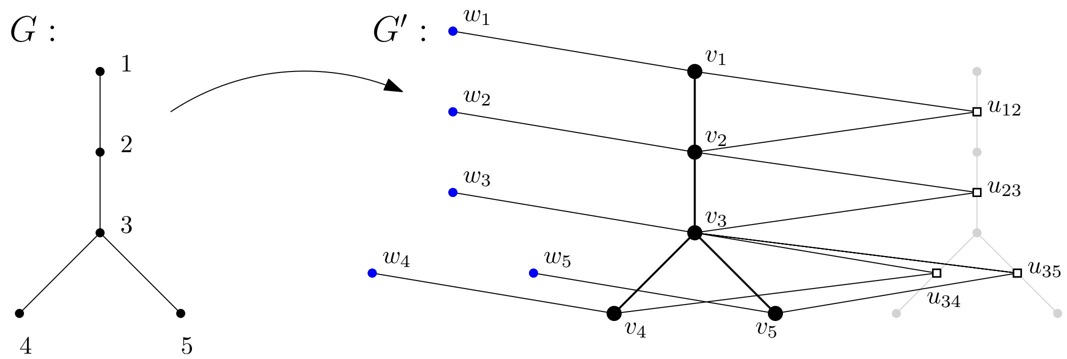

Figure 8: An example construction for the proof of Theorem 3.11.

Property 1: *A set* $I \subseteq \{1, \ldots, n\}$ is an independent set in G if and only if the set $V' = \{v_j : j \in I\} \cup \{w_j : j \notin S\} \cup \{u_{ij} : i \notin I \land j \notin S\}$ is a *maximal* independent set in G'.

The above property states that there is a one-to-one correspondence between independent sets of G and *maximal* independent sets of G'.

Let us now assign a weight to each vertex in G' in order to obtain an equistable function for G' and show that G' is an equistable graph.

Let $b_1, \dots, b_n$ and $\{a_e : e \in E\}$ be integers. For a vertex v_j we define $w(v_j) = b_j + 3\sum_{ij\in E} a_{ij}$. We say that b_j is the V-cost of v_j and $3\sum_{ij\in E} a_{ij}$ is the E-cost of v_j.

For a vertex w_j we define $w(w_j) = b_j + 2\sum_{ij\in E} a_{ij}$. We say that b_j is the V-cost of w_j and $2\sum_{ij\in E} a_{ij}$ is the E-cost of w_j.

For u_{ij} we define $w(u_{ij}) = a_{ij}$ and we call a_{ij} the E-cost of u_{ij}.

We set $t = \sum_{i=1}^{n} b_i + 5\sum_{e\in E} a_e$.

Claim 1: For a maximal independent set I' in G' we have $w(I') = t$.

Observe that I' contains either v_i or w_i but not both for every i. Thus the sum of the V-costs of all vertices in I' is exactly $\sum_{i=1}^{n} b_i$. Similarly, for each $ij \in E$ it follows that I' will contain exactly one of the following:

- v_i, w_j, or
- w_i, v_j, or
- w_i, w_j, u_{ij}.

Now, it is easy to see that E-cost of any option is exactly $5a_{ij}$. Thus the total E-cost of I' is $\sum_{ij\in E} a_{ij}$ and by definition of t it follows that $w(S') = t$.

We will now explicitly set the values b_i and a_{ij}. Denote $m = |E|$. We have to define $n + m$ values. For $i \in \{1, \dots, n\}$ let $b_i = 8^i$. Order the integers corresponding to the edges arbitraril y. To the j-th integer (edge) assign the value 8^{n+j}. Let us prove the following claim.

Claim 2: A subset $I' \subseteq V'$ has weight t if and only if it is a maximal independent set in G'.

By Claim 1, it suffices to prove that every set of vertices that has weight t is a maximal independent set. We say that a subset of vertices $I \subseteq V'$ is *vertex maximal* if for every j, it contains v_j or w_j but not both. We say that I is *edge maximal* if for every edge $ij \in E$, S contains one of the following:

- v_i and w_j,
- w_i and v_j,
- w_i, w_j and u_{ij}.

Similarly as before we conclude that a set I is maximal independent set if and only if it is both vertex maximal and edge maximal.

Let I be a subset of V'. Suppose that $w(I) = t$. By considering values modulo 8^{j+1} , one can show that I must contain exactly one of v_j and w_j; therefore, I is vertex maximal. Now, consider an edge $ij \in E$, and suppose it is the k-th edge. The contribution due to edge ij in any vertex maximal subset S is either $4a_{ij}$, $5a_{ij}$, $6a_{ij}$ or $7a_{ij}$. By considering values modulo 8^{n+k+1} , one can show that the contribution of the edge ij must be $5a_{ij}$, and thus I has edge maximality with respect to the j-th edge. Hence I is edge maximal and thus also a maximal independent set.

Therefore G' is an equistable graph. An equistable structure for G' is (w, t) defined as above.

Let us complete the proof. Consider the following weight function w' for G'. For each v_j we set $w'(v_j) = 1$ and for the other vertices we set the weight to zero. As we said in the beginning of the proof, finding a maximum weight independent set in G' (with respect to w') is equivalent to the finding a maximum independent set in G. The later problem is APX-hard. The proof is thus complete. □

4 Conclusion

The Independent Set problem is NP-hard, but when the input is restricted to a particular class of graphs there is hope that the problem could become polynomially solvable. A big goal would be to classify all hereditary graph classes with respect to whether the Independent Set is NP-hard or not. Such a goal might be too ambitious. A good starting point are the methods for developing polynomial-time algorithms for Independent Set in special graph classes. We presented some of them in Chapter 2. Moreover, using the techinques of augmenting graphs we showed that the problem is polynomially solvable if the input is restricted to $\{P_{10}, C_4, C_6\}$-free graphs. This is a novel result. Regarding $\{P_k, C_4, C_6\}$-free graphs, it was known that the problem is polynomially solvable in the class of $\{P_8, C_4, C_6\}$-free graphs, since the problem is polynomially solvable in the class of $\{P_8, \text{banner}\}$-free graphs [29]. It follows from a result of Alekseev [1] we know that Independent Set is NP-hard in the class of C_4-free graphs. The problem also remains NP-hard if we forbid $\{C_4, C_6\}$, that is, the condition on P_{10}-freeness is needed. Another way to see that the problem is NP-hard in the class of $\{C_4, C_6\}$-free graphs is to use an observation by Poljak [65] that $\alpha(G') = \alpha(G) + 1$, where a graph G' is obtained from G by replacing a single edge with a P_4 (subdividing it twice). On the other hand, we still do not know if the Independent Set is polynomially solvable in the class of $\{P_9, C_4\}$-free graphs.

A trivial corollary of the results for $\{P_{10}, C_4, C_6\}$-free graphs is that the Indepednent Set problem is polynomially solvable in the class of $\{P_{10}, \text{even-hole}\}$-free graphs. The question is whether a similar approach could be used to devise a polynomial-time algorithm for even-hole-free graphs. Such a question remains open and is very interesting due to its connections with perfect graphs. For more on even-hole-free graph we refer to [70] and references therein.

A second important question is to determine the complexity of the problem in the classes of P_k-free graphs. As we mentioned before, the problem is polynomially solvable for classes of P_k-free graphs where $k \leq 6$ [32]. The results for P_5- and P_6-free graphs are both obtained using minimal triangulations as presented in Section 2.2.

We also presented the class of equistable graphs in Chapter 3. We showed that the problem of recognizing k-equistable graphs is solvable in linear time, while it remains an open question whether the problem of recognizing equistable graphs is efficiently solvable. Related to the previous problem is the problem of finding a combinatorial characterization of equistable graphs.

5 Povzetek naloge v slovenskem jeziku

Naj bo $G = (V, E)$ graf. Za množico vozlišč I pravimo, da je *neodvisna*, če nobeni dve vozlišči v množici nista sosednji. Problem neodvisne množice je poiskati neodvisno množicno največje velikosti v danem grafu. To je eden izmed prvih problemov, za katere je bilo pokazano, da je NP-težek [36]. Problem neodvisne množice se uporablja za modeliranje problemov z različnih področij kot so: računalništvo, teorija informacij, upravljanje prometa, telekomunikacije in finance. Natačneje, problem se pojavi v molekularni biologiji, računalniškem vidu, teoriji kodiranja in pri razporejanju v brezžičnih omrežjih [22].

Razširjeno je prepričanje, da NP-težkih optimizacijskih problemov ni moč učinkovito rešiti do optimalnosti. Verjame se torej, da ne obstaja polinomski algoritem za problem neodvisne množice v grafih. Še več, pri podobnih predpostavkah za problem ne obstaja polinomski algoritem, ki bi aproksimiral rešitev znotraj faktorja $n^{1-\epsilon}$ za poljuben $\epsilon \in (0, 1)$, kjer je n število vozlišč danega grafa. Tovrstni rezultati o računski zahtevnosti problema neodvisne množice motivirajo študij problema na grafih z dodatnimi omejitvami. Problem tako pogosto študiramo v kontekstu hereditarnih razredov grafov. V številnih posebnih primerih postane problem polinomsko rešljiv. Najbolj znani razredi grafov, za katere je problem rešljiv v polinomskem času, so dvodelni grafi [16], tetivni grafi [25], popolni grafi [30] in grafi omejenih širinskih parametrov [4, 18].

Problem seveda ne postane polinomsko rešljiv za poljuben netrivialen hereditaren razred grafov. Znano je, da je problem neodvisne množice NP-težek za ravninske grafe največje stopnje kvečjemu tri [27], za grafe enotskih diskov [15] in za grafe brez trikotnikov [60].

V poglavju 2 predstavimo nekatere najpogosteje uporabljene metode za razvoj polinomskih algoritmov za problem neodvisne množice v posebnih razredih. Najprej je predstavljena metoda povečujočih grafov. Ideja metode izhaja iz znane metode povečujočih poti, s katero je bil rešen problem največjega prirejanja. Podobno kot pri prirejanjih se lahko dokaže, da je dana neodvisa množica največja možna če in samo če graf nima nobenega povečujočega grafa za to neodvisno množico. Če torej dana neodvisna množica ni največja, potem zanjo obstaja nek povečujoč graf. Na ta način problem prevedemo na problem iskanja povečujočega grafa za dano neodvisno množico. Od tod zlahka izpeljemo, da je problem iskanja povečujočega grafa NP-težek. Kljub temu pa v posebnih primerih ta pristop vodi do polinomskih algoritmov. Metoda povečujočih grafov je uporabljenja v poglavjih 2.1.1 in 2.1.2 v razvoju polinomskega algoritma za problem neodvisne množice v razredu $\{P_{10}, C_4, C_6\}$-prostih grafov, kar je nov rezultat.

Nadalje je predstavljena novejša metoda, ki je zasnovana na "dobrih" triangulacijah danega grafa. Ideja je zasnovana na enem od pristopov za reševanje problema v razredu tetivnih grafov (triangulacija tetivnega grafa je kar isti graf). Idejo se nato posploši in namesto ene same triangulacije upoštevamo več "dobrih" triangulacij. Ta metoda je uporabljena za razvoj polinomskega algoritma v razredu P_5-prostih grafov [43] in kasneje v razredu P_6-prostih grafov [32].

V preostanku poglavja 2 so predstavljene bolj klasične metode, kot so modularna dekompozicija, dekompozicija glede na prerezne klike in transformacije grafov.

Tesno povezan s problemom neodvisne množice je razred ekvistablnih grafov. Pravimo, da je graf G *ekvistabilen*, če obstaja taka funkcija $\varphi : V(G) \to \mathbb{R}_+$, da za vsako podmnožico vozlišč I velja $\varphi(I) = 1$ natanko tedaj, ko je I maksimalna neodvisna množica. Ekvivalento, graf G je ekvistabilen natanko tedaj, ko obstajata tako naravno število t in taka funkcija $w : V(G) \to \mathbb{N}$, da je $w(I) = t$ če in samo če je I neodvisna množica. Tako funkciji w rečemo ekvistabilna funkcija, paru (w, t) pa ekvistabilna struktura grafa G.

Določitev računske zahtevnosti problema prepoznavanja ekvistabilnih grafov je še vedno odprt problem. Prav tako ni znano, ali je problem v razredu NP. Vemo pa, da je problem prepoznavanja k-ekvistabilnih grafov rešljiv v linearnem času. Za dani graf G rečemo, da je *k-ekvistabilen* natanko tedaj ko obstaja ekvistablina funckija $w : V(G) \to \{1, \dots, k\}$. Algoritem linearne časovne zahtevnosti za prepoznavanje k-ekvistabinih grafov je predstavljen v poglavju 3.1 kot FPT algoritem s parametrom k. Dokaz NP-težkosti problema utežene neodvisne množice v razredu ekvistabilnih grafov je obravnavan v poglavju 3.2.

6 Bibliography

[1] V Alekseev. The effect of local constraints on the complexity of determination of the graph independence number. *Combinatorial-algebraic methods in applied mathematics*, pages 3–13, 1982. *(Cited on pages 2 and 53.)*

[2] VE Alekseev. Hereditary classes and coding of graphs. *Problemy Kibernet*, 39:151–164, 1982. *(Cited on page 5.)*

[3] Vladimir E Alekseev. On easy and hard hereditary classes of graphs with respect to the independent set problem. *Discrete Applied Mathematics*, 132(1):17–26, 2003. *(Cited on page 36.)*

[4] Stefan Arnborg and Andrzej Proskurowski. Linear time algorithms for np-hard problems restricted to partial k-trees. *Discrete Applied Mathematics*, 23(1):11–24, 1989. *(Cited on pages 40 and 55.)*

[5] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 2009. *(Cited on pages 1 and 50.)*

[6] Brenda S Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM (JACM)*, 41(1):153–180, 1994. *(Cited on page 2.)*

[7] Egon Balas and Chang Sung Yu. On graphs with polynomially solvable maximum-weight clique problem. *Networks*, 19(2):247–253, 1989. *(Cited on page 2.)*

[8] Hans-Jürgen Bandelt and Henry Martyn Mulder. Distance-hereditary graphs. *Journal of Combinatorial Theory, Series B*, 41(2):182–208, 1986. *(Cited on page 41.)*

[9] Claude Berge. Two theorems in graph theory. *Proceedings of the National Academy of Sciences*, 43(9):842–844, 1957. *(Cited on page 8.)*

[10] Jean RS Blair and Barry Peyton. An introduction to chordal graphs and clique trees. In *Graph theory and sparse matrix computation*, pages 1–29. Springer, 1993. *(Cited on page 27.)*

[11] Vincent Bouchitté and Ioan Todinca. Treewidth and minimum fill-in: Grouping the minimal separators. *SIAM Journal on Computing*, 31(1):212–232, 2001. *(Cited on page 28.)*

[12] Andreas Brandstädt, Van Bang Le, and Jeremy P Spinrad. *Graph classes: a survey.* SIAM, 1999. *(Cited on page 6.)*

[13] Christoph Brause. A subexponential-time algorithm for the maximum independent set problem in P_t-free graphs. *Discrete Applied Mathematics*, 2016. *(Cited on page 2.)*

[14] Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph theorem. *Annals of mathematics*, pages 51–229, 2006. *(Cited on page 6.)*

[15] Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete mathematics*, 86(1-3):165–177, 1990. *(Cited on pages 2 and 55.)*

[16] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, 3rd Edition.* MIT Press, 2009. *(Cited on pages 2, 40, and 55.)*

[17] Derek G Corneil, H Lerchs, and L Stewart Burlingham. Complement reducible graphs. *Discrete Applied Mathematics*, 3(3):163–174, 1981. *(Cited on pages 2 and 36.)*

[18] Bruno Courcelle, Johann A Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000. *(Cited on pages 2, 40, and 55.)*

[19] Konrad K. Dabrowski, Vadim V. Lozin, Dominique de Werra, and Viktor Zamaraev. Combinatorics and algorithms for augmenting graphs. *Graphs and Combinatorics*, 32(4):1339–1352, 2016. *(Cited on page 12.)*

[20] Gabriel Andrew Dirac. On rigid circuit graphs. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 25, pages 71–76. Springer, 1961. *(Cited on page 37.)*

[21] Rodney G. Downey and Michael Ralph Fellows. *Parameterized Complexity.* Springer Science & Business Media, 2012. *(Cited on page 1.)*

[22] Ngoc C. Lê. *Algorithms for the Maximum Independent Set Problem.* PhD thesis, Technische Universität Bergakademie Freiberg, 2014. *(Cited on pages 1, 40, and 55.)*

[23] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17(3):449–467, 1965. *(Cited on page 8.)*

[24] Fedor V Fomin and Yngve Villanger. Finding induced subgraphs via minimal triangulations. In *27th International Symposium on Theoretical Aspects of Computer Science-STACS 2010*, pages 383–394, 2010. *(Cited on page 31.)*

[25] András Frank. Some polynomial algorithms for certain graphs and hypergraphs. In *Proceedings of the Fifth British Combinatorial Conference*, pages 211–226, 1975. *(Cited on pages 2, 26, and 55.)*

[26] Tibor Gallai. Transitiv orientierbare Graphen. *Acta Mathematica Hungarica*, 18(1-2):25–66, 1967. *(Cited on pages 34 and 36.)*

[27] Michael R Garey and David S Johnson. *Computers and Intractability*, volume 29. wh freeman New York, 2002. *(Cited on pages 2 and 55.)*

[28] Fănică Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16(1):47–56, 1974. *(Cited on page 27.)*

[29] Michael U Gerber, Alain Hertz, and Vadim V Lozin. Stable sets in two subclasses of banner-free graphs. *Discrete Applied Mathematics*, 132(1):121–136, 2003. *(Cited on pages 14, 15, and 53.)*

[30] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981. *(Cited on pages 17 and 55.)*

[31] Martin Grötschel, László Lovász, and Alexander Schrijver. Polynomial algorithms for perfect graphs. *North-Holland Mathematics Studies*, 88:325–356, 1984. *(Cited on page 41.)*

[32] Andrzej Grzesik, Tereza Klimošová, Marcin Pilipczuk, and Michał Pilipczuk. Polynomial-time algorithm for maximum weight independent set on P_6-free graphs, 20/07/2017. arXiv:1707.05491. *(Cited on pages 2, 33, 53, and 56.)*

[33] Rajiv Gupta, Mary Lou Soffa, and Denise Ombres. Efficient register allocation via coloring using clique separators. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 16(3):370–386, 1994. *(Cited on page 37.)*

[34] Frank Gurski and Egon Wanke. The tree-width of clique-width bounded graphs without $K_{n,n}$. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 196–205. Springer, 2000. *(Cited on page 40.)*

[35] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 653–662. IEEE, 1998. *(Cited on page 1.)*

[36] Richard M Karp. *Reducibility among combinatorial problems.* Springer, 1972. *(Cited on pages 1 and 55.)*

[37] Eun Jung Kim, Martin Milanic, and Oliver Schaudt. Recognizing k-equistable graphs in FPT time. *arXiv preprint arXiv:1503.01098*, 2015. *(Cited on pages 3, 45, 46, 47, and 48.)*

[38] AD Korshunov. Coefficient of internal stability of graphs. *Cybernetics and Systems Analysis*, 10(1):19–33, 1974. *(Cited on page 1.)*

[39] Jan Kratochvíl and Jaroslav Nešetřil. Independent set and clique problems in intersection-defined classes of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 31(1):85–93, 1990. *(Cited on page 6.)*

[40] Philippe GH Lehot. An optimal algorithm to detect a line graph and output its root graph. *Journal of the ACM (JACM)*, 21(4):569–575, 1974. *(Cited on page 8.)*

[41] Vadim Levit, Martin Milanič, and David Tankus. On the recognition of k-equistable graphs. In *Graph-Theoretic Concepts in Computer Science*, pages 286–296. Springer, 2012. *(Cited on pages 45 and 46.)*

[42] Richard J Lipton and Robert Endre Tarjan. Applications of a planar separator theorem. *SIAM Journal on Computing*, 9(3):615–627, 1980. *(Cited on page 2.)*

[43] Daniel Lokshantov, Martin Vatshelle, and Yngve Villanger. Independent set in P_5-free graphs in polynomial time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 570–581, 2014. *(Cited on pages 4, 28, 31, 32, and 56.)*

[44] Daniel Lokshtanov, Dániel Marx, Saket Saurabh, et al. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, (105):41–72, 2011. *(Cited on page 1.)*

[45] Daniel Lokshtanov, Marcin Pilipczuk, and Erik Jan van Leeuwen. Independence and efficient domination on P_6-free graphs. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington,*

VA, USA, January 10-12, 2016, pages 1784–1803, 2016. *(Cited on pages 2, 34, and 36.)*

[46] László Lovász. Normal hypergraphs and the perfect graph conjecture. *Discrete Mathematics*, 306(10):867–875, 2006. *(Cited on page 36.)*

[47] Vadim V Lozin. Stability preserving transformations of graphs. *Annals of Operations Research*, 188(1):331–341, 2011. *(Cited on page 38.)*

[48] Vadim V Lozin and Martin Milanič. On finding augmenting graphs. *Discrete Applied Mathematics*, 156(13):2517–2529, 2008. *(Cited on page 12.)*

[49] Vadim V Lozin and Martin Milanič. A polynomial algorithm to find an independent set of maximum weight in a fork-free graph. *Journal of Discrete Algorithms*, 6(4):595–604, 2008. *(Cited on pages 2, 9, 10, 35, and 36.)*

[50] Vadim V Lozin and Raffaele Mosca. Independent sets in extensions of $2K_2$-free graphs. *Discrete Applied Mathematics*, 146(1):74–80, 2005. *(Cited on page 2.)*

[51] Vadim V. Lozin and Raffaele Mosca. Maximum regular induced subgraphs in $2P_3$-free graphs. *Theoretical Computer Science*, 460:26 – 33, 2012. *(Cited on page 2.)*

[52] Nadimpalli VR Mahadev and Uri N Peled. *Threshold Graphs and Related Topics*, volume 56. Elsevier, 1995. *(Cited on pages 43 and 44.)*

[53] Kevin McAvaney, Jack Robertson, and Duane DeTemple. A characterization and hereditary properties for partition graphs. *Discrete Mathematics*, 113(1-3):131–142, 1993. *(Cited on page 44.)*

[54] Ross M McConnell and Jeremy P Spinrad. Modular decomposition and transitive orientation. *Discrete Mathematics*, 201(1-3):189–241, 1999. *(Cited on page 35.)*

[55] Štefko Miklavič and Martin Milanič. Equistable graphs, general partition graphs, triangle graphs, and graph products. *Discrete Applied Mathematics*, 159(11):1148–1159, 2011. *(Cited on pages 44 and 45.)*

[56] Martin Milanič. *Algorithmic developments and complexity results for finding maximum and exact independent sets in graphs.* Rutgers University, 2007. *(Cited on page 39.)*

[57] Martin Milanič, James Orlin, and Gábor Rudolf. Complexity results for equistable graphs and related classes. *Annals of Operations Research*, 188(1):359–370, 2011. *(Cited on pages 3, 45, and 50.)*

[58] Martin Milanič and Nicolas Trotignon. Equistarable graphs and counterexamples to three conjectures on equistable graphs. *Journal of Graph Theory*, 84(4):536–551, 2017. *(Cited on page 45.)*

[59] George J Minty. On maximal independent sets of vertices in claw-free graphs. *Journal of Combinatorial Theory, Series B*, 28(3):284–304, 1980. *(Cited on pages 2 and 9.)*

[60] Bojan Mohar and Svatopluk Poljak. *Eigenvalues in Combinatorial Optimization*. Springer, 1993. *(Cited on pages 2 and 55.)*

[61] Raffaele Mosca. Stable sets in certain P_6-free graphs. *Discrete Applied Mathematics*, 92(2):177–191, 1999. *(Cited on pages 12 and 18.)*

[62] Sang-il Oum and Paul Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006. *(Cited on page 40.)*

[63] Christos H Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *Journal of computer and system sciences*, 43(3):425–440, 1991. *(Cited on page 50.)*

[64] Charles Payan. A class of threshold and domishold graphs: equistable and equidominating graphs. *Discrete Mathematics*, 29(1):47–52, 1980. *(Cited on pages 42 and 43.)*

[65] Svatopluk Poljak. A note on stable sets and colorings of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 15(2):307–309, 1974. *(Cited on page 53.)*

[66] Ganessan Ramalingam and C Pandu Rangan. A unified approach to domination problems on interval graphs. *Information Processing Letters*, 27(5):271–274, 1988. *(Cited on page 41.)*

[67] Nicholas D Roussopoulos. A max $\{m, n\}$ algorithm for determining the graph H from its line graph G. *Information Processing Letters*, 2(4):108–112, 1973. *(Cited on page 8.)*

[68] Najiba Sbihi. Algorithme de recherche d'un stable de cardinalité maximum dans un graphe sans étoile. *Discrete Mathematics*, 29(1):53–76, 1980. *(Cited on pages 2 and 8.)*

[69] Robert E Tarjan. Decomposition by clique separators. *Discrete mathematics*, 55(2):221–232, 1985. *(Cited on page 36.)*

[70] Kristina Vušković. Even-hole-free graphs: a survey. *Applicable Analysis and Discrete Mathematics*, pages 219–240, 2010. *(Cited on page 53.)*

[71] Sue Hays Whitesides. An algorithm for finding clique cut-sets. *Information Processing Letters*, 12(1):31–32, 1981. *(Cited on page 36.)*