Philipps-Universität Marburg

Fachbereich Mathematik und Informatik
Studiengang Informatik

Destroying Multicolored Paths and Cycles in Edge-colored Graphs

Bachelorarbeit

zur Erlangung des akademischen Grades eines
Bachelor of Science

vorgelegt von
Nils Jakob Eckstein
am 06.07.2020

Matrikelnummer:	3025817
Betreuer:	Prof. Dr. Christian Komusiewicz
	Frank Sommer, M. Sc.
	Niels Grüttemeier, M. Sc.

Zusammenfassung

Wir untersuchen die Berechnungskomplexität von c-Colored P_ℓ Deletion, c-Colored C_ℓ Deletion und Colored Path Deletion. Bei diesen Problemen will man in einem kanten-gefärbten Graphen durch höchstens k Kantenlöschungen alle induzierten c-farbigen Pfade, beziehungsweise Zyklen einer bestimmten Länge zerstören. Im Gegensatz zu den ersten beiden Problemen, ist beim dritten Problem die Anzahl c der Farben und die Länge der verbotenen Pfade, Teil der Eingabe. Zunächst analysieren wir die klassische Komplexität und beweisen, dass c-Colored P_ℓ Deletion und c-Colored C_ℓ Deletion für jede nicht-triviale Kombination von c und ℓ NP-schwer sind. Anschließend analysieren wir die parametrisierte Komplexität im Hinblick auf den natürlichen Parameter k und den dualen Parameter $m - k$. Wir beweisen, dass c-Colored P_ℓ Deletion und c-Colored C_ℓ Deletion parametrisiert mit k in FPT sind, während Colored Path Deletion parametrisiert mit k W[2]-schwer ist. Außerdem geben wir unter der Voraussetzung $c \geq 2$ einen linearen Problemkern für c-Colored P_ℓ Deletion und c-Colored C_ℓ Deletion parametrisiert mit $m - k$ an.

Abstract

We study the computational complexity of c-Colored P_ℓ Deletion, c-Colored C_ℓ Deletion and Colored Path Deletion. In these problems, one wants to destroy all induced c-colored paths or respectively cycles of a specified length from an edge-colored graph by deleting at most k edges. In contrast to the first two problems, the number c of colors and the length of the forbidden paths is part of the input in the third problem. First, we analyze the classical complexity and prove that c-Colored P_ℓ Deletion and c-Colored C_ℓ Deletion are NP-hard for each non-trivial combination of c and ℓ. Second, we analyze the parameterized complexity with regard to the natural parameter k and the dual parameter $m - k$. We prove that c-Colored P_ℓ Deletion and c-Colored C_ℓ Deletion are fixed-parameter tractable when parameterized by k, while Colored Path Deletion is W[2]-hard when parameterized by k. Furthermore, we give a linear size problem kernel for c-Colored P_ℓ Deletion and c-Colored C_ℓ Deletion for $c \geq 2$ when parameterized by $m - k$.

Contents

1 Introduction **1**

1.1 Related Work . . . 2

1.2 Our Results . . . 3

2 Preliminaries **5**

2.1 Graphs . . . 5

2.2 Computational Complexity . . . 6

2.3 Parameterized Complexity . . . 9

3 Classical Complexity **12**

3.1 c-colored P_ℓ Deletion . . . 12

3.2 c-colored C_ℓ Deletion . . . 23

4 Parameterized Complexity **26**

5 Conclusion **32**

6 Bibliography **34**

1 Introduction

Graphs provide a natural framework to model the interactions and relations between objects. Therefore, they play a central role in computer science. To get an understanding about the structure and properties of the modeled systems and to be able to categorize these systems one can classify graphs by different graph properties. Many interesting graph properties can be characterized by forbidden subgraphs, that is, a graph has property Π if and only if it does not contain a subgraph from some (possibly infinite) set $\mathcal{F}$ of graphs. For example, a graph is called a *tree* if it does not contain a cycle, and a graph is called a *cograph* if it does not contain an induced P_4, that is, a path of four vertices. Hence, trees are characterized by an infinite set of forbidden subgraphs, while cographs are characterized by a finite set of forbidden subgraphs.

A topic that arises naturally from this classification of graphs and has numerous applications are graph modification problems. In these problems, that have been studied intensively, one wants to modify a given graph such that it fulfills some graph property Π. And naturally, one often wants to apply the minimum number of modifications. Graph modification problems can be classified depending on the type of modification. A problem where only the edges of the graph are modified is called an edge modification problem. Three well-studied classes of edge modification problems are completion-problems, deletion-problems and editing-problems. In a completion-problem, the graph is modified by inserting edges, in a deletion-problem the graph is modified by deleting edges and in editing-problems both edge insertions and edge deletions are allowed. These problems arise in applications like data clustering [1], the analysis of social networks [2], numerical algebra [3] and the reassembly of fragmented DNA sequences [4]. Besides these applications, edge modification problems are an important topic in computer science: They are NP-complete for a wide range of well studied graph properties Π [5, 6] and hence, they can be used to prove NP-hardness results or to study algorithmic approaches to NP-hard problems.

A standard graph can model a network of objects with one type of relation. But many systems are too complex to be described by such a graph. In recent years, multilayer networks are getting increasingly important to describe such complex systems with multiple types of relations [7, 8]. A generalized graph that can formally describe such networks can be defined as an edge-colored (multi-)graph. In such a (multi-)graph, each edge-color represents one type of relation between the vertices. While edge modification problems are well-studied on uncolored graphs, there is not much work on edge-colored graphs. At the same time, one can expect an increasing number of applications for modification problems on these graphs with the increasing importance of edge-colored graphs. Therefore, we analyze the computational complexity of edge deletion problems on edge-colored graphs in this thesis.

For any set $\mathcal{F}$ of edge-colored graphs, a decision version of the edge deletion problem on edge-colored graphs can be formulated as follows.

> $\mathcal{F}$ DELETION
> **Input**: An edge-colored graph $G = (V, E = E_1 \sqcup \ldots \sqcup E_c)$ [1], an integer k.
> **Question**: Can we delete at most k edges from G such that the remaining graph has no induced subgraph that is isomorphic to any graph in $\mathcal{F}$?

Vertex deletion problems are NP-complete for hereditary graph properties on uncolored graphs [9], and there are similar results for edge-colored graphs [10]. A graph property Π is called *hereditary* if for any graph G with property Π the deletion of any vertex does not result in a graph violating Π. As far as we know, there is no such general NP-completeness result for edge deletion problems and it is suspected that the techniques leading to these results are not applicable to edge deletion problems [11].

In this thesis we analyze the computational complexity of three variations of $\mathcal{F}$ DELETION. The first problem we consider is the special case of $\mathcal{F}$ DELETION where $\mathcal{F}$ is the set of all c-colored paths of ℓ vertices for a fixed $c \in [\ell - 1]$. We refer to this problem by c-COLORED P_ℓ DELETION or cP_ℓD. The second problem we consider is the special case of $\mathcal{F}$ DELETION where $\mathcal{F}$ is the set of all c-colored cycles of ℓ vertices for a fixed $c \in [\ell]$. We refer to this problem by c-COLORED C_ℓ DELETION or cC_ℓD. The third problem we consider is a generalization of cP_ℓD where the number c of colors and the number ℓ of vertices of the forbidden subgraphs are part of the input. We refer to this problem by COLORED PATH DELETION or CPD. In Section 3 we will analyze classical complexity and in Section 4 we will analyze parameterized complexity.

1.1 Related Work

From a classical point of view, it is known for cP_ℓD and cC_ℓD that the uncolored case is NP-hard for any $\ell \geq 3$ [12, 5]. In fact, it is known that H DELETION is NP-complete for any graph H if and only if H has at least two edges [13]. There are some results for edge-colored graphs, too. A recent result shows that $2P_3$D is NP-hard in general [14]. However, there are some special cases of $2P_3$D, that are polynomial-time solvable. For example, $2P_3$D is polynomial-time solvable if the input is restricted to graphs where each subgraph that is a 2-colored paths containing three vertices, is an induced subgraph [14]. Furthermore, the problem of destroying

[1] The $\sqcup$-operator symbolizes the disjoint union. Hence, $E_i \cap E_j = \emptyset$ for $i, j \in [c]$ and $i \neq j$.

not only induced, but also non-induced 2-colored paths containing three vertices is polynomial time solvable on 2-colored graphs but NP-hard on 3-colored graphs [15]. As far as we know, there is not much work on CPD. However, it is easy to see that CPD is NP-hard, since $2P_3$D is an NP-hard special case [14].

From a parameterized point of view, there is a very general result for edge modification problems. It is known that edge deletion problems are fixed-parameter tractable for hereditary graph properties when parameterized by the natural parameter k [16]. This result implies that cP_ℓD and cC_ℓD are fixed-parameter tractable when parameterized by k. After showing that a problem is fixed-parameter tractable, the next question often is, whether or not it admits a polynomial kernelization. The uncolored case is well studied. In general, $1P_\ell$D does not admit a polynomial kernelization for any $\ell \geq 5$ and $1C_\ell$D does not admit a polynomial kernelization for any $\ell \geq 4$ when parameterized by k, unless NP $\subseteq$ coNP$_{/\text{poly}}$ [17]. However, if the input graph has maximum degree at most d, there is a kernelization with at most $\mathcal{O}(d^{2\cdot\ell-1} \cdot k^{p\cdot(\ell-1)+1})$ vertices for $1P_\ell$D and $1C_\ell$D where $p = \log_{\frac{2\ell-2}{2\ell-3}}(\ell-1)$ [18]. For edge colored graphs there are some results, too. For $2P_3$D, there is a kernelization comprising at most $\mathcal{O}(d \cdot k^2)$ vertices where d is the maximum degree of the input graph, and there is a linear kernelization comprising $\mathcal{O}(2 \cdot \delta)$ edges when parameterized by the dual parameter $\delta := m - k$ [14].

1.2 Our Results

First, we consider classical complexity of cP_ℓD and cC_ℓD in Section 3. We show that cP_ℓD is NP-hard for each $\ell \geq 3$ and each $c \in [\ell - 1]$ and that cC_ℓD is NP-hard for each $\ell \geq 3$ and each $c \in [\ell]$. Since cP_ℓD is trivially solvable if $\ell < 3$ or $c \geq \ell$, and cC_ℓD is not properly defined if $\ell < 3$ and trivially solvable if $c > \ell$, this implies that cP_ℓD and cC_ℓD are NP-hard for each non-trivial combination of c and ℓ.

From a classical point of view, one aspect that makes the analysis of $\mathcal{F}$ Deletion more complicated than the analysis of vertex deletion problems, is that the deletion of an edge can lead to a new induced forbidden subgraph, while the deletion of a vertex can not. Hence, we define the notion of *non-cascading* input graphs. For an instance (G, k) of $\mathcal{F}$ Deletion, we call the input graph G *non-cascading*, if each subgraph of G that is isomorphic to a graph in $\mathcal{F}$ is an induced subgraph. Recall that *$2P_3D$* is polynomial-time solvable on non-cascading graphs [14]. We show that *$cP_\ell D$* is NP-hard on non-cascading graphs for each $\ell \geq 4$ and each $c \in [2, \ell - 2]$ and that cC_ℓD is NP-hard on non-cascading graphs for each $\ell \geq 3$ and each $c \in [\ell]$.

Second, we consider parameterized complexity of cP_ℓD, cC_ℓD and CPD with regard to different parameters in Section 4. By giving a simple branching algorithm, we show that cP_ℓD and cC_ℓD are fixed parameter tractable for any $\ell \geq 1$ and any $c \in [\ell - 1]$ or $c \in [\ell]$ respectively when parameterized by k. We prove that the simple branching algorithm can not be enhanced to a running time where the exponential factor is independent of ℓ unless FPT=W[2], by showing that CPD is W[2]-hard when parameterized by k even if $c = 3$. Then, we show that there are polynomial size kernelizations for cP_ℓD and cC_ℓD when parameterized by k if the input graph is non-cascading. Furthermore, we show that cP_ℓD and cC_ℓD admits linear kernelizations with $\mathcal{O}(2 \cdot \delta)$ edges when parameterized by the dual parameter $\delta := m - k$ if $c \geq 2$.

2 Preliminaries

In this section, we define the basic concepts and notations that we use in this work. By $\mathbb{N}$ we denote the set of all positive integers excluding zero. For two integers $a \in \mathbb{N}$ and $b \in \mathbb{N}$ with $a \leq b$, we define $[a, b] := \{a, \ldots, b\}$ and if $a = 1$ we write $[b]$. Furthermore, for any set Σ we define Σ^* to be the set of all strings that can be build by concatenation of letters from Σ.

2.1 Graphs

We study the computational complexity of problems defined on undirected graphs. An undirected graph $G = (V, E)$ is a pair of sets. Usually G is considered as a set of *vertices* V that is connected by *edges* E. Conventionally, we define $n := |V|$ to be the number of vertices, $m := |E|$ to be the number of edges, and by $|G| := n + m$ we denote the *size* of G. Each edge connects exactly two vertices. If $\{u, v\} \in E$, we say that the vertices u and v are *adjacent* and we say the edge $\{u, v\}$ is *incident* with the vertices u and v. We only consider *simple* graphs. A graph G is called simple, if no vertex is adjacent to itself. Hence, if G is simple, then $\{v, v\} \notin E$ for each $v \in V$. For a given graph G, we denote its vertex set by $V(G)$ and its edge set by $E(G)$. In contrast to a graph, the edges of a *multigraph* are a multiset. Hence, in a multigraph two vertices can have multiple edges between them.

The problems studied here are defined on *edge-colored* graphs. In an edge-colored graph, the edge set is partitioned into $c \in \mathbb{N}$ disjoint, non-empty subsets $E_1, \ldots, E_c$. We call such a graph *c-colored*, and if $c = 1$ we call the graph *uncolored*. For sake of illustration we define $E_1 =: E_b$ as the set of *blue* edges, $E_2 =: E_r$ as the set of *red* edges, $E_3 =: E_y$ as the set of *yellow* edges, $E_4 =: E_g$ as the set of *green* edges and draw them in the figures accordingly. For a given graph G we denote the set of all edges with color $\alpha \in [c]$ by $E_\alpha(G)$.

For two sets of vertices $V' \subseteq V(G)$ and $V'' \subseteq V(G)$ we denote the set of edges between V' and V'' by $E_G(V', V'') := \{\{u, v\} \in E \mid u \in V' \text{ and } v \in V''\}$ and if $V' = V''$ we write $E_G(V')$. We call $G' = (V', E')$ a *subgraph* of G if $V' \subseteq V(G)$ and $E' \subseteq E_G(V')$. If $E' = E_G(V')$ we call it the *induced* subgraph and write $G[V']$. For a subset of edges $E' \subseteq E(G)$ we denote the graph formed by deleting the edges from E' in G by $G - E' := (V, E \setminus E')$. For a vertex $v \in V(G)$, we denote the *open neighborhood* of v in G by $N_G(v) := \{u \in V(G) \mid \{u, v\} \in E(G)\}$. We denote the *degree* of v in G by $\deg_G(v) := |N_G(v)|$.

We say that a vertex set $V' \subseteq V(G)$ is a *vertex cover* for G if at least one of u and v is in V' for each edge $\{u, v\} \in E(G)$. We say that a vertex set $V' \subseteq V(G)$ is an *independent set* if $\{u, v\} \notin E(G)$ for each pair of vertices $u, v \in V'$. A graph G

is called *tripartite* if $V(G)$ can be partitioned into 3 (possibly empty) independent sets.

Next, we will define some special graphs that we consider in this thesis. A graph G is called a *path* if it is possible to uniquely index its vertices with numbers from $[n]$, in such a way that there is an edge $\{v_i, v_j\} \in E(G)$ if and only if $i + 1 = j$. A path with one additional edge $\{v_n, v_1\}$ is called a *cycle*. We denote a path (cycle) consisting of ℓ vertices by P_ℓ (C_ℓ). The length of a path (cycle) G is defined as the number of edges $|E(G)|$. For any graph G, the 2-*subdivision* of G is the graph we get from inserting two new vertices on every edge, that is, from replacing each edge $\{u, v\} \in E(G)$ by a P_4 with vertices $\{u, x, y, v\}$. A graph G is a 2-*subdivision graph*, if it is the 2-subdivision of any graph H.

For a graph G that contains a cycle as a subgraph, the *girth* of G is defined as the length of a shortest cycle in G, and for acyclic graphs the girth is infinite. For two edge-colored graphs G and F, we say that G is *isomorphic* to F, and write $G \cong F$ if there is a bijective function $\varphi : V(G) \to V(F)$ such that $\{u, v\} \in E(G)$ is an edge with color c if and only if $\{\varphi(u), \varphi(v)\} \in E(F)$ is an edge with color c. Furthermore, we say that G is *F-free* if $G[V'] \ncong F$ for each vertex set $V' \in V(G)$. For any set $\mathcal{F}$ of graphs, we say that a graph G is $\mathcal{F}$-free if G is F-free for each $F \in \mathcal{F}$. We say that an edge set $S \subseteq E(G)$ is a *solution* for an instance (G, k) of $\mathcal{F}$ Deletion if $|S| \leq k$ and $G - S$ is $\mathcal{F}$-free. We say that $\mathcal{F}$ is *d-edge-bounded* for some integer d, if $|E(F)| \leq d$ for each $F \in \mathcal{F}$. Furthermore, we say that $\mathcal{F}$ is *polynomially enumerable* on G, if we can enumerate vertex sets $V' \subseteq V(G)$ such that $G[V'] \cong F$ for some $F \in \mathcal{F}$ in a running time that is polynomial in $|G|$. For further reading on graphs and graph-classes we refer the reader to some standard monographs [19, 20, 21] and the *Information System on Graph Classes and their Inclusions* (ISGCI) [22]

2.2 Computational Complexity

One of the most central questions in computer science is whether a given problem is efficiently computable, that is, whether one can find an algorithm that is able to compute an answer for every instance of the problem in an acceptable running time. Before we investigate this question, we have to define the term *problem*. In this thesis we consider so-called *decision problems*. A decision problem can formally be described as a function $f : \{0, 1\}^* \to \{0, 1\}$. Hereby $\{0, 1\}^*$ denotes the set of all binary strings. For example, $\mathcal{F}$ Deletion can be defined as a function $L_\mathcal{F} : \{0, 1\}^* \to \{0, 1\}$ that gets a binary input string $x \in \{0, 1\}^*$ encoding an edge-colored graph G and an integer k such that

$$L_{\mathcal{F}}(x) := \begin{cases} 1 & \text{if it is possible to make } G\ \mathcal{F}\text{-free by deleting at most } k \text{ edges.} \\ 0 & \text{otherwise.} \end{cases}$$

We call a problem instance $x \in \{0,1\}^*$ of a decision problem L a *yes-instance* if $L(x) = 1$. Otherwise, we call x a *no-instance.*

Next, we have to formalize the vague notion of "acceptable running time". The *running time* of an algorithm is the number of basic steps it performs. By definition, it depends on the machine model used for the computation. We will briefly describe some relevant machine models and refer to a standard monograph for further reading [23]. From a practical point of view, the machine model of the greatest importance is the commonly used random-access machine (RAM) model. In theoretical computer science though, one often prefers the model of Deterministic Turing Machines (DTM). A DTM M can be defined as a tuple (Γ, Q, δ) with

- A finite set Γ of symbols, called *alphabet* of M.
- A finite set Q of *states.*
- A *transition function* $\delta : Q \times \Gamma^k \to Q \times \Gamma^{k-1} \times \{L, S, R\}^k$ where $k \geq 2$ is the number of *tapes* of M.

Initially, one tape contains the input instance $x \in \Gamma^*$ and all other tapes are empty. Then, M carries out the computation according to the transition function δ. The input instance is *accepted* if M reaches a designated halting state $q_{\texttt{halt}} \in Q$.

Another machine model that is of great theoretical importance, is the Nondeterministic Turing Machine (NTM). In contrast to an DTM, an NTM has two transition functions δ_0, δ_1 and chooses one of them in each computational step. An NTM M accepts an input instance if there is a sequence of choices between the two transition functions such that M reaches the halting state $q_{\texttt{halt}} \in Q$.

For interesting algorithms the running time depends on the instance. In this thesis, we consider so-called *worst-case* running times. For an algorithm A the worst-case running time can be defined as a function $t : \mathbb{N} \to \mathbb{N}$ such that A performs at most $t(|x|)$ basic steps for each instance x. Note that the worst-case running time only depends on the size of an instance. Since this notion of running time can result in very complicated functions, we specify the worst case running time in *$\mathcal{O}$-notation.* For two functions $f, g : \mathbb{N} \to \mathbb{N}$ we say that $f \in \mathcal{O}(g)$ if there exists a constant c such that $f(n) \leq c \cdot g(n)$ for each sufficiently large n. We say that a problem is *polynomial time solvable* if there is an algorithm that is able to compute an answer for every instance x of the problem in $\mathcal{O}(poly(|x|))$ time where $poly(|x|)$ is a *polynomial.* For each machine model there is a set of problems, that are polynomial time solvable. Conventionally, we denote the set of all problems that are polynomial time solvable on a DTM by P and we denote the set of all problems that are polynomial time solvable on a NTM by NP. It is known that a problem is polynomial time solvable on the RAM model if and only if it is polynomial time solvable on the DTM model. Hence, it does not make a difference whether we use DTMs or RAMs for analyzing the question whether or not a problem is polynomial time solvable.

The question whether or not P = NP is one of the most important open problems in computer science. By definition, we know that P $\subseteq$ NP and it is reasonably to believe that this containment is proper. One reason to believe this is that there are so-called NP-complete problems. These problems are known to be in NP but there is no known polynomial time DTM algorithm despite an immense amount of research. To understand the notion of NP-completeness, one first has to understand the concept of polynomial time reducibility. We say that a problem L is *polynomial time reducible* to another problem L' and write $L \leq_p L'$ if there is a polynomial time computable function $g : \{0,1\}^* \to \{0,1\}^*$ such that $L(x) = 1$ if and only if $L'(g(x)) = 1$ for every $x \in \{0,1\}^*$. A problem L' is called NP-*hard* if $L \leq_p L'$ for each problem $L \in$ NP. A problem L' is called NP-*complete* if L' is NP-hard and $L' \in$ NP. If there was a polynomial time algorithm A for an NP-complete problems L', one could efficiently solve every problem $L \in$ NP by reducing L to L' in polynomial time and then using A to solve the instance. Hence, the NP-complete problems are considered the hardest problems in NP.

Cook and Levin independently proved that the SAT problem is NP-complete [24, 25]. It can be formulated as follows.

SAT
Input: A boolean formula Φ over a set $\mathcal{X} = \{x_1, \ldots, x_\eta\}$ of variables.
Question: Is there an assignment $\mathcal{A} : \mathcal{X} \to \{\texttt{true}, \texttt{false}\}$ that satisfies Φ?

There are numerous variants of SAT. We will use a special case of SAT called (3,B2)-SAT to prove the NP-hardness in Theorem 3.2. In (3,B2)-SAT Φ is a boolean formula in *conjunctive normal form.* This means that Φ is a conjunction of clauses $\mathcal{C} = \{c_1, \ldots, c_\mu\}$. Furthermore, in (3,B2)-SAT each clause $c_j \in \mathcal{C}$ contains *exactly* three literals and for each variable $x_i \in \mathcal{X}$ each literal x_i and $\neg x_i$ occurs *exactly twice* in Φ. It is known that (3,B2)-SAT is NP-complete [26].

One year after Cook had proven the NP-completeness of SAT, Karp proved NP-completeness for 20 further problems by polynomial time reduction [27]. We will use two of these problems in this thesis. The first one is commonly referred to as Vertex Cover and can be defined as follows.

Vertex Cover (VC)
Input: A graph $G = (V, E)$ and a positive integer k.

Question: Is there a vertex cover of size at most k for G?

The second problem is a generalization of VC to *hypergraphs*. A hypergraph has a finite vertex set $\mathcal{X} = \{x_1, \ldots, x_\eta\}$, called *universe* and a collection $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_\mu\}$ of *hyper-edges*. Each hyper-edge $\mathcal{C}_j \in \mathcal{C}$ is a subset of the universe. A *hitting set* for a hypergraph $(\mathcal{X}, \mathcal{C})$ is a subset $H \subseteq \mathcal{X}$ such that H contains at least one element from each hyper-edge $\mathcal{C}_j \in \mathcal{C}$. The VC-generalization to hypergraphs is commonly referred to as Hitting Set and can be defined as follows.

Hitting Set (HS)
Instance: A hypergraph $(\mathcal{X}, \mathcal{C})$ and a positive integer k.
Question: Is there a hitting set of size at most k for $(\mathcal{X}, \mathcal{C})$?

We may assume that each $\mathcal{C}_j$ is non-empty and that each $x_i \in \mathcal{X}$ occurs in at least one subset $\mathcal{C}_j \in \mathcal{C}$. If $|\mathcal{C}_i| \leq d$ for each subset $\mathcal{C}_i \in \mathcal{C}$ and some $d \in \mathbb{N}$, we call the problem d-Hitting Set (d-HS). We will use HS and d-HS in Section 4.

Furthermore, we will use the following NP-complete graph problem, that is closely related to VC.

Independent Set (IS)
Input: A graph $G = (V, E)$ and a positive integer k.
Question: Does G contain an independent set of size at least k?

For any graph G, a vertex subset $V' \subseteq V(G)$ is a maximum independent set if and only if $V(G) \setminus V'$ is a minimum vertex cover and, hence, (G, k) is a yes-instance for IS if and only if $(G, n - k)$ is a yes-instance for VC [28]. We will use this to prove the NP-hardness of VC on a special class of input graphs (see Lemma 3.9). For further reading we refer to the standard monographs [28, 23].

2.3 Parameterized Complexity

Despite an immense amount of research, there is no known algorithm that solves an NP-hard problem in polynomial time on a DTM or RAM respectively. But since many NP-hard problems have important applications, these problems have to be solved anyway. There are different approaches to do so efficiently. In some applications, one does not require an exact solution and an efficient algorithm that approximates a solution does the job. In other applications, the input data has special properties such that an NP-hard problem becomes efficiently solvable on all relevant input instances. For example, VC is efficiently solvable if the input graph is bipartite, since it is equivalent to computing a maximum matching in this case [19]. On a simplified level this example captures a core idea of parameterized complexity. Namely that the computational complexity of a problem may depend on other properties of the instance besides its size. This idea leads to the definition of parameterized problems and a more fine-grained complexity analysis.

Similar to a classic decision problem, a *parameterized decision problem* can be defined as a function $f : \{0,1\}^* \times \{1\}^* \to \{0,1\}$. An instance (x, κ) of a parameterized decision problem contains a binary coded input $x \in \{0,1\}^*$ and a *parameter* $\kappa \in \{1\}^*$ which is encoded in a unary alphabet. From each classical problem, we can define numerous parameterized problems, since there are many different way to define a parameter.

In this thesis, we study two common parameterizations for edge deletion problems. The *natural* parameter is the number k of edges that can be deleted, and the *dual* parameter is $m - k$. We say that a parameterized problem is *fixed-parameter tractable* if there is an algorithm A, a computable function g, and a constant c such that the running time of A is at most $g(\kappa) \cdot |x|^c$ and $L(x, \kappa) = A(x, \kappa)$ for each instance (x, κ). The class FPT is the set of all parameterized problems that are fixed-parameter tractable. A notion that is equivalent to fixed-parameter tractability, is the notion of *kernelization*. For a parameterized problem L, a kernelization replaces an instance (x, κ) by a reduced instance or *kernel* (x', κ') such that:

I. $\kappa' \leq \kappa$

II. $|x'| \leq h(\kappa)$ for some function h depending only on κ

III. $L(x, \kappa) = L(x', \kappa')$.

If the size of a kernel is polynomial in κ we call it a *polynomial kernel*.

Similar to the classical polynomial time reduction, we can define a *parameterized reduction*. For two parameterized problems L and L' we say that L can be reduced to L' by a parameterized reduction if there are functions $f, g : \mathbb{N} \to \mathbb{N}$ and $h : \{0,1\}^* \times \{1\}^* \to \{0,1\}^*$, and a constant c such that for each instance (x, κ):

I. $f(\kappa) = \kappa'$,

II. $h(x, \kappa) = x'$,

III. $L(x, \kappa) = L'(x', \kappa')$,

IV. $h(x, \kappa)$ is computable in time $g(\kappa) \cdot |(x, \kappa)|^c$.

Observe that if L can be reduced to L' by a parameterized reduction and $L' \in$ FPT, then $L \in$ FPT.

From a parameterized point of view, the class FPT corresponds to the classical class P, since they are both the class of problems that are tractable according to the respective definition of tractable problems. This poses the question about a parameterized class that corresponds to NP or more generally to any class of problems that is probably not tractable.

The W-*Hierarchy* is a hierarchy of classes of parameterized problems. For each $t \in \mathbb{N}$ it contains a class W[t]. A parameterized problem L' is W[t]-*hard* if each parameterized problem $L \in$ W[t] can be reduced to L' by a parameterized reduction and L' is called W[t]-*complete* if L' is W[t]-hard and $L' \in$ W[t]. For example IS parameterized by the natural parameter k is W[1]-complete and HS parameterized by the natural parameter k is W[2]-complete [29]. For a formal definition of the W-Hierarchy, we refer to a standart monograph [30]. It is known that FPT $\subseteq$ W[1] and W[t] $\subseteq$ W[t+1]. And it is conjectured that these containments are proper. Hence, no W[t]-hard problem is fixed-parameter tractable for any t. For further reading we refer to the standard monographs [29, 30].

3 Classical Complexity

In this section, we will prove the NP-hardness of cP_ℓD and cC_ℓD. To do so, we will give polynomial time reductions from some NP-hard problem. It is not hard to see that the reductions we give can be computed in polynomial time and hence, we will not further analyze the running time for the computations of the reduction.

3.1 c-colored P_ℓ Deletion

First, we consider cP_ℓD and show that it is NP-hard even if the input graph has a constant maximum degree and a large girth. But before we give the NP-hardness results, we prove the following Lemma that we will need in the first proof.

Lemma 3.1. *Let $G = (V = \{v_1, \dots, v_{d\cdot(\ell-1)}\}, E)$ be a path where any ℓ consecutive vertices form a c-colored P_ℓ.*

Then, $S \subseteq E$ is an edge-deletion set of size $d-1$ such that $G - S$ is c-colored P_ℓ-free if and only if $S = \{\{v_i, v_{i+1}\} \mid i \text{ mod}(\ell-1) = 0\}$.

Proof. Let $S = \{\{v_i, v_{i+1}\} \mid i \bmod(\ell-1) = 0\}$. Since $|V| = d\cdot(\ell-1)$ we conclude that $|S| = d-1$. We will show that $G' := G - S$ is P_ℓ-free and hence, G' is c-colored P_ℓ-free. Let $V' \subseteq V$ such that the induced subgraph $G[V']$ is a c-colored P_ℓ. Since G is a path, $V' = \{v_i, \dots, v_{i+\ell-1}\}$ for some $i \in [(d-1)\cdot(\ell-1)]$. Since $|[i, i+\ell-2]| = \ell-1$ there is a $\hat{i} \in [i, i+\ell-2]$ such that $\hat{i} \bmod(\ell-1) = 0$. Thus, we conclude that there is an edge $e \in E_G(V')$ such that $e \in S$. Hence, G' is c-colored P_ℓ-free.

Conversely, let S be an edge-deletion set of size $d-1$ such that $G - S$ is c-colored P_ℓ-free. Let $\{v_i, v_{i+1}\}$ be the j-th edge in S. We denote $V_1 := \{v_1, \dots, v_i\}$ and $V_2 := \{v_{i+1}, \dots, v_{d\cdot(\ell-1)}\}$. Since $\{v_i, v_{i+1}\}$ is the j-th edge in S, we know that $|S \cap E_G(V_1)| = j-1$.

First, assume towards a contradiction that $i \geq j\cdot(\ell-1)+1$. Then,

$$|V_1| = i \geq j\cdot(\ell-1)+1.$$

Since any ℓ consecutive vertices form a c-colored P_ℓ, we conclude that $G[V_1]$ contains at least j edge-disjoint c-colored P_ℓs. Since $|S \cap E_G(V_1)| = j-1$, this is a contradiction to the condition that $G - S$ is c-colored P_ℓ-free. Thus, $i \leq j\cdot(\ell-1)$.

Next, assume towards a contradiction that $i \leq j\cdot(\ell-1)-1$. Then,

$$\begin{aligned}|V_2| &= d\cdot(\ell-1) - i \\ &\geq (d-j)\cdot(\ell-1)+1.\end{aligned}$$

Since any ℓ consecutive vertices form a c-colored P_ℓ, we conclude that $G[V_2]$ contains at least $(d-j)$ edge-disjoint c-colored P_ℓs. Hence, $|S \cap E_G(V_2)| \geq (d-j)$. We conclude that $|S| \geq d$, since $|S \cap E_G(V_1)| = j - 1$ and $\{v_i, v_{i+1}\} \in S$. This is a contradiction to the condition that $|S| = d - 1$. Thus, $i \geq j \cdot (\ell - 1)$.

Hence, we know that $i = j \cdot (\ell - 1)$. Thus, the j-th edge in S is $\{v_{j\cdot(\ell-1)}, v_{j\cdot(\ell-1)+1}\}$ and therefor $S = \{\{v_i, v_{i+1}\} \mid i \bmod(\ell - 1) = 0\}$. □

Now we will show that cP_ℓD is NP-hard for $\ell \geq 4$ even if the input graph has a some how simple structure. From this result we will be able to prove that cP_ℓD remains NP-hard on non-cascading input graphs if $\ell \geq 4$ and $c \in [2, \ell - 2]$.

Theorem 3.2. *cP_ℓD is NP-hard for each $\ell \geq 4$ and each $c \in [2, \ell - 2]$ even if the maximum degree of G is three and the girth of G is greater than $2 \cdot d \cdot \ell$ for any constant $d \geq 1$.*

Proof. To show the NP-hardness of cP_ℓD for $\ell \geq 4$ and $c \in [2, \ell - 2]$ we give a polynomial time reduction from the NP-complete (3,B2)-SAT problem [26].

Construction: Let Φ be a (3,B2)-SAT formula with clauses $\mathcal{C} = \{c_1, \ldots, c_\mu\}$ and variables $\mathcal{X} = \{x_1, \ldots, x_\eta\}$. To construct an equivalent instance $(G = (V, E), k)$ of cP_ℓD from Φ we use the following *gadgets*.

For each clause $c_j \in \mathcal{C}$ we construct a *clause gadget* C_j as follows. The clause gadget C_j consists of three vertex sets U_j^1, U_j^2 and U_j^3 of $\ell - 1$ vertices each. For each $p \in [3]$, we denote the vertices in U_j^p by $u_j^{p,1}, \ldots, u_j^{p,\ell-1}$. For $s \in [\ell - 2]$ we add edges $\{u_j^{p,s}, u_j^{p,s+1}\}$. If $s < c$ we add an edge of color s. Else we add an edge of color c. In other words $\{u_j^{p,1}, u_j^{p,2}\}$ is blue, $\{u_j^{p,2}, u_j^{p,3}\}$ is red and the color of the next edges depends on c. Observe that $G[U_j^p]$ is a c-colored $P_{\ell-1}$. We connect the three $P_{\ell-1}$s by identifying $u_j^{1,1} = u_j^{2,1} = u_j^{3,1} =: u_j$ (see Figure 1 for an example).

For each variable $x_i \in \mathcal{X}$ we construct a *variable gadget* X_i as follows. First, let $z := d \cdot (\ell - 1) + 1$. Note that z is the minimum number of vertices on a path that contains d edge disjoint P_ℓs. The variable gadget X_i consists of four vertex sets of z vertices $T_i^1, T_i^2, F_i^1, F_i^2$ and a vertex set of $\ell - 4$ vertices $W_i := \{w_i^1, \ldots, w_i^{\ell-4}\}$. Note that $W_i = \emptyset$ for $\ell = 4$. For $q \in [2]$ the vertices in T_i^q are denoted by $t_i^{q,1}, \ldots, t_i^{q,z}$ and the vertices in F_i^q are denoted by $f_i^{q,1}, \ldots, f_i^{q,z}$. For $s \in [z - 1]$ we add edges $\{t_i^{q,s}, t_i^{q,s+1}\}$ and $\{f_i^{q,s}, f_i^{q,s+1}\}$. If $0 < s \bmod(\ell - 1) < c$ we add an edge with color $s \bmod(\ell - 1)$ and else we add an edge with color c. So if $s \bmod(\ell - 1) = 1$ we add an blue edge, if $s \bmod(\ell - 1) = 2$ we add a red edge and otherwise, the edge color depends on c.

We connect T_i^1 and T_i^2 by identifying $t_i^{1,1} = t_i^{2,1} =: t_i$ and analogously we connect F_i^1 and F_i^2 by identifying $f_i^{1,1} = f_i^{2,1} =: f_i$. If $\ell = 4$ we add a red edge $\{t_i, f_i\}$. If $\ell > 4$ we connect t_i and f_i by a $(c-1)$-colored path with vertices in $\{t_i\} \cup W_i \cup \{f_i\}$ that does not contain a blue edge (see Figure 2). Since $|\{t_i\} \cup W_i \cup \{f_i\}| = \ell - 2 \geq c$, this path always has at least $(c - 1)$ edges. Hence, we can always connect t_i and f_i by a $(c - 1)$-colored path.

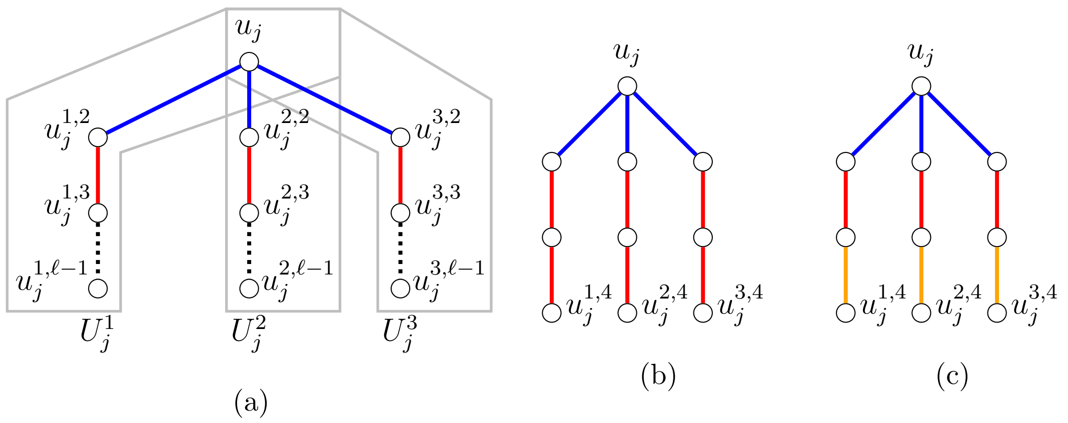

Figure 1: (a) General structure of a clause gadget C_j. The black dotted line represents a path containing $\ell - 3$ vertices in total. (b) Clause gadget for $\ell = 5$, $c = 2$. (c) Clause gadget for $\ell = 5$, $c = 3$.

Note that any ℓ consecutive vertices in $G[T_i^1]$, $G[T_i^2]$, $G[F_i^1]$ and $G[F_i^2]$ form a c-colored P_ℓ. Hence, $G[T_i^1]$, $G[T_i^2]$, $G[F_i^1]$ and $G[F_i^2]$ are four paths, each containing d edge disjoint c-colored P_ℓs. And since $\{t_i^{q,2}, t_i\} \in E_b(G)$ and t_i and f_i are connected by $(c-1)$-colored $P_{\ell-1}$ with no blue edge, the induced subgraphs $G[\{t_i^{q,2}, t_i, w_i^1, \ldots, w_i^{\ell-4}, f_i, f_i^{q',2}\}]$ are also c-colored P_ℓs for $q, q' \in [2]$.

Then, we denote the following edge sets (see Figure 2b).

$$T_i^b := \{\{t_i^{q,s}, t_i^{q,s+1}\} \in E \mid s \bmod(\ell-1) = 1, q \in [2]\}.$$

$$F_i^b := \{\{f_i^{q,s}, f_i^{q,s+1}\} \in E \mid s \bmod(\ell-1) = 1, q \in [2]\}.$$

$$T_i^r := \{\{t_i^{q,s}, t_i^{q,s+1}\} \in E \mid s \bmod(\ell-1) = 2, q \in [2]\}.$$

$$F_i^r := \{\{f_i^{q,s}, f_i^{q,s+1}\} \in E \mid s \bmod(\ell-1) = 2, q \in [2]\}.$$

Note that $|T_i^r| = |F_i^r| = |T_i^b| = |F_i^b| = 2 \cdot d$.

To connect the variable gadgets with the clause gadgets we identify vertices as follows (see Figure 3). For $p \in [3]$, $q \in [2]$, any variable $x_i \in \mathcal{X}$ and any clause $c_j \in \mathcal{C}$ we set

$$u_j^{p,2} = \begin{cases} t_i^{q,z} & \text{if the literal } x_i \text{ has its } q\text{-th occurence as the } p\text{-th literal in } c_j \\ f_i^{q,z} & \text{if the literal } \neg x_i \text{ has its } q\text{-th occurence as the } p\text{-th literal in } c_j. \end{cases}$$

Note that for each clause gadget C_j and $p \in [3]$ the vertex $u_j^{p,2}$ is identified with exactly one vertex from a variable gadget and for each variable gadget X_i and $q \in [2]$ the vertices $t_i^{q,z}$ and $f_i^{q,z}$ are each identified with exactly one vertex from a clause gadget, since each literal occurs exactly twice in Φ.

It is easy to see that the maximum degree of G is three. Furthermore, the girth of G is greater or equal to $2 \cdot d \cdot \ell$, since the smallest possible cycle in G contains

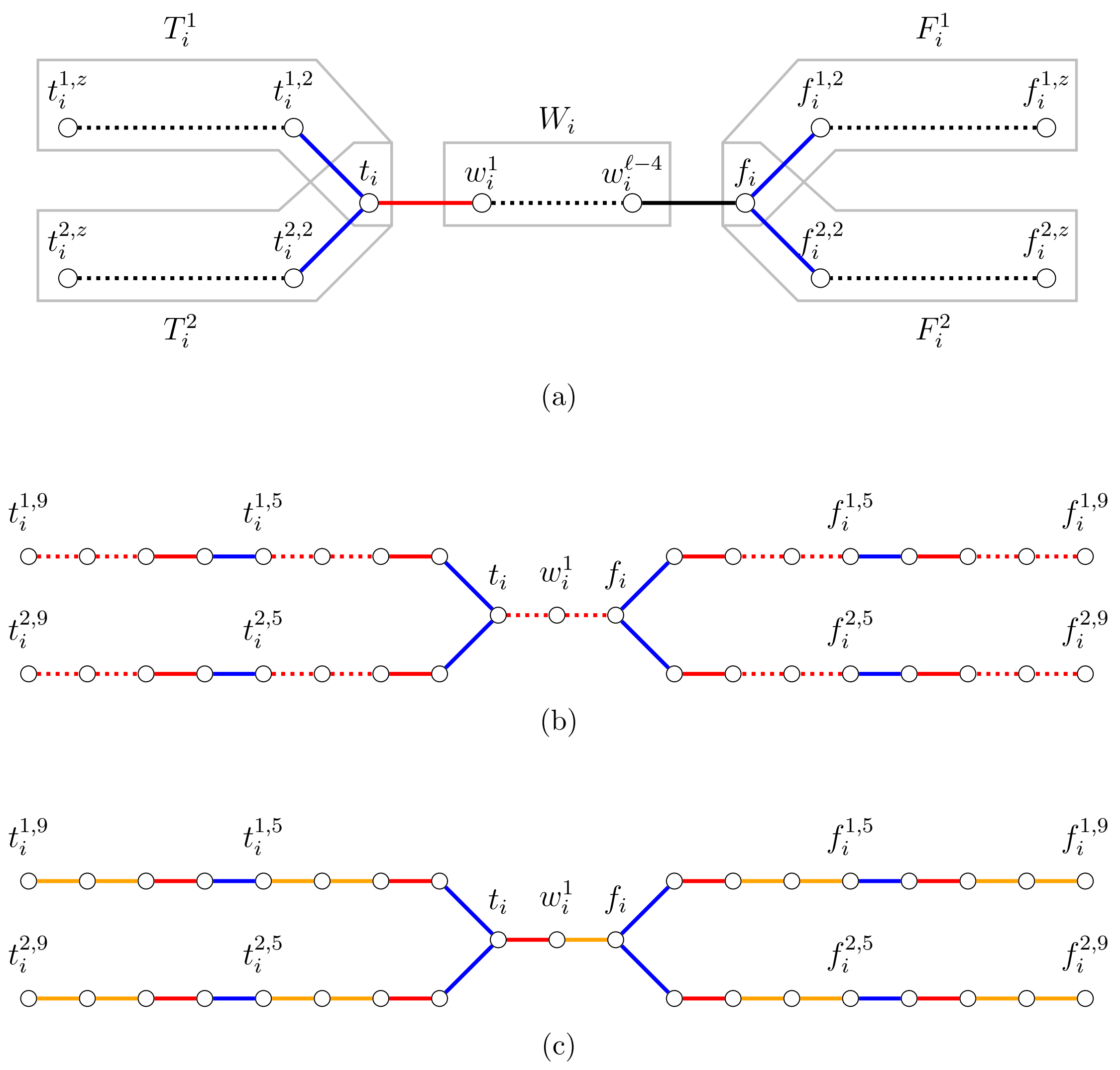

Figure 2: (a) The generalized structure of a variable gadget X_i. Note that $W_i = \emptyset$ if $\ell = 4$. The black line represents an edge of color c. (b) An exemplary variable gadget for $\ell = 5, c = 2, d = 2$. The red edges that are in T_i^r or F_i^r are represented as solid lines, while the other red edges are represented as dotted lines. (c) An exemplary variable gadget for $\ell = 5, c = 3, d = 2$.

the vertices from $T_i^1 \cup T_i^2 \cup \{u_j\}$ or $F_i^1 \cup F_i^2 \cup \{u_j\}$ respectively. Such a cycle will be constructed, when there is a clause $c_j = (x_i \vee x_i \vee \dots)$ or $c_j = (\neg x_i \vee \neg x_i \vee \dots)$ in Φ. We complete the construction by setting $k = 4 \cdot d \cdot \eta + 2 \cdot \mu$.

Intuition: Before we prove the correctness of the reduction, we informally describe its idea. Each variable gadget contains $4 \cdot d$ edge disjoint c-colored P_ℓs. So we have to delete at least $4 \cdot d$ edges per variable gadget. We can make a variable gadget c-colored P_ℓ-free with $4 \cdot d$ edge deletions by deleting the edges in T_i^r and F_i^b, or the edges in F_i^r and T_i^b. The former models the assignment $\mathcal{A}(x_i) = \texttt{true}$, and

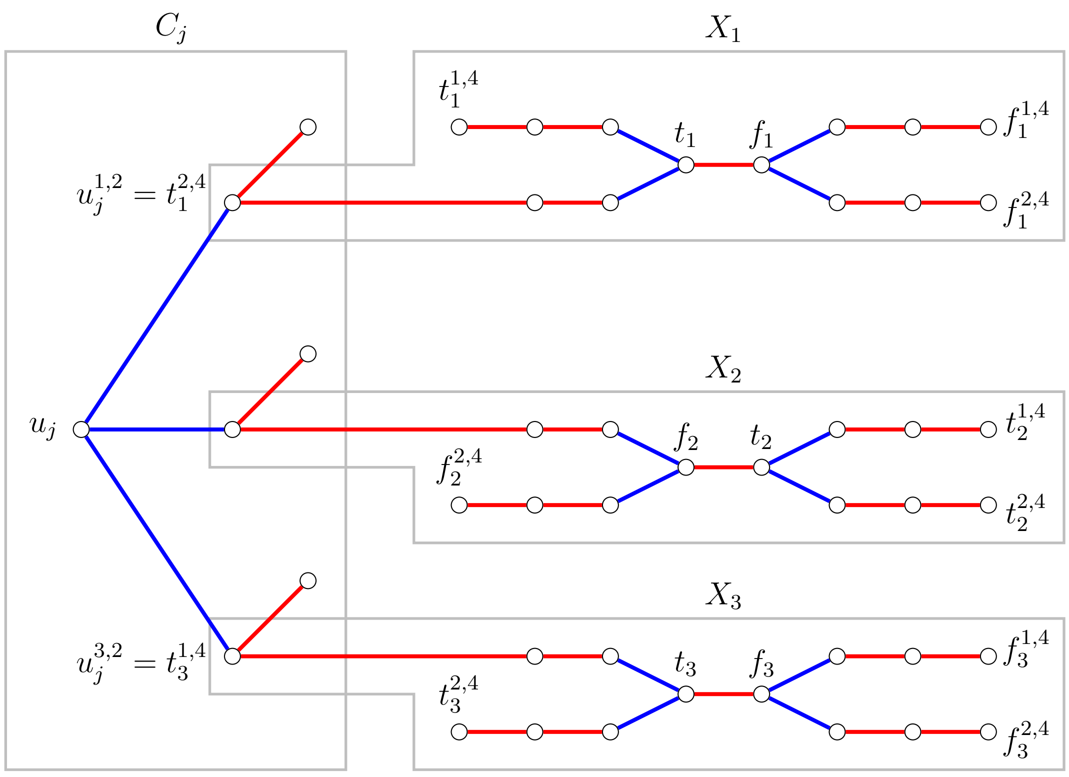

Figure 3: This figure shows a part of the constructed graph for $\ell = 4$, $d = 1$. The left side of the figure shows the clause gadget C_j for a clause $c_j = (x_1 \vee \neg x_2 \vee x_3)$. The right side of the figure shows the variable gadgets X_1, X_2 and X_3. Note that x_1 has its second occurrence as a positive literal in c_j, x_2 has its first occurrence as a negative literal in c_j and x_3 has its first occurrence as a positive literal in c_j. The clause gadgets where the variables have their other occurrences are not shown in the figure.

the later models the assignment $\mathcal{A}(x_i) = \texttt{false}$. If we delete T_i^r and F_i^b the vertices $f_i^{1,z}$ and $f_i^{2,z}$ are part of a $(c-1)$-colored $P_{\ell-1}$ with no blue edges, while the vertices $t_i^{1,z}$ and $t_i^{2,z}$ are part of a $(c-1)$-colored $P_{\ell-2}$. If we delete F_i^r and T_i^b, it is the other way around. We will be able to make a clause gadget c-colored P_ℓ-free with two edge deletions if and only if there is at least one vertex $u_j^{p,2}$ that is not part of a $(c-1)$-colored $P_{\ell-1}$ from the connected variable gadget. Thus, we will be able to make the constructed graph c-colored P_ℓ-free with exactly k edge deletions if and only if each clause is satisfied by at least one variable.

Before we give a proof for the correctness of the reduction, we formalize a part of the intuition in the following Claim.

Claim 3.3. *Let $G' := G - (F_i^b \cup T_i^r)$ and $G'' := G - (T_i^b \cup F_i^r)$ for any variable gadget X_i. Then:*

***I**. No blue edge $e_b \in E_{G'}(X_i)$ that is part of a P_ℓ in G'.*

II. *No blue edge $e_b \in E_{G''}(X_i)$ that is part of a P_ℓ in G''.*

Proof. We only prove ***I***, since the proof for ***II*** works analogously. Let $e_b \in E_{G'}(X_i)$ be a blue edge. Since $F_i^b \cap E(G') = \emptyset$, we conclude that $e_b \in T_i^b$. Hence, we know that $e_b = \{t_i^{q,s}, t_i^{q,s+1}\}$ for some s such that $s \bmod(\ell-1) = 1$ and $q \in [2]$. We consider two cases.
(Case 1) $s = 1$. By definition of T_i^r we know that $\{t_i^{q,2}, t_i^{q,3}\} \in T_i^r$. We can conclude that $\deg_{G'}(t_i^{q,2}) = 1$, since $T_i^r \cap E(G') = \emptyset$. Furthermore, since $F_i^b \cap E(G') = \emptyset$, we conclude that $\deg_{G'}(f_i) = 1$. Thus, the longest path in G' containing e_b is the induced subgraph $G'[\{t_i^{q,2}, t_i\} \cup W_i \cup \{f_i\}]$. By construction, it is easy to see that $|\{t_i^{q,2}, t_i\} \cup W_i \cup \{f_i\}| = \ell - 1$. Thus, e_b is not part of a P_ℓ in G'.
(Case 2) $s > 1$. It is easy to see that

$$s \bmod(\ell-1) = 1 \Leftrightarrow (s+1) \bmod(\ell-1) = 2 \quad (1)$$

$$\Leftrightarrow (s+2-\ell) \bmod(\ell-1) = 2 \quad (2)$$

From equation (1) we conclude that $\{t_i^{q,s+1}, t_i^{q,s+2}\} \in T_i^r$ and from equation (2) we conclude that $\{t_i^{q,s+2-\ell}, t_i^{q,s+3-\ell}\} \in T_i^r$. Since $T_i^r \cap E(G') = \emptyset$, we can conclude that $\deg_{G'}(t_i^{q,s+1}) = 1$ and $\deg_{G'}(t_i^{q,s+3-\ell}) = 1$. Thus, the longest path in G' containing e_b is the induced subgraph $G'[\{t_i^{q,s+3-\ell}, \dots, t_i^{q,s+1}\}]$. It is easy to see that $|\{t_i^{q,s+3-\ell}, \dots, t_i^{q,s+1}\}| = \ell - 1$. Thus, e_b is not part of a P_ℓ in G'.
Hence, no blue edge from $E_{G'}(X_i)$ is part of a P_ℓ in G'. △

Correctness: Now we will show the correctness of the reduction by proving that there is a satisfying assignment for Φ if and only if (G, k) is a yes-instance of cP_ℓD.

($\Rightarrow$) Let $\mathcal{A} : \mathcal{X} \to \{\texttt{true}, \texttt{false}\}$ be a satisfying assignment for Φ. We will prove that (G, k) is a yes-instance of cP_ℓD by constructing an edge-deletion set S of size k such that $G - S$ is c-colored P_ℓ-free.

For each variable $x_i \in \mathcal{X}$ we add $4 \cdot d$ edges to S. If $\mathcal{A}(x) = \texttt{true}$, then we add T_i^r and F_i^b to S. If $\mathcal{A}(x) = \texttt{false}$, then we add F_i^r and T_i^b to S. Since $\mathcal{A}$ satisfies Φ, there is at least one variable $x_i \in \mathcal{X}$ such that $\mathcal{A}(x_i)$ satisfies c_j for each clause $c_j \in \mathcal{C}$. Let $p \in [3]$ such that the p-th literal of c_j satisfies c_j. Let $\{\alpha, \beta\} := [3] \setminus \{p\}$. We add $\{u_j, u_j^{\alpha,2}\}$ and $\{u_j, u_j^{\beta,2}\}$ to S. Note that we added exactly two edges per clause. Hence, $|S| = 4 \cdot d \cdot \eta + 2 \cdot \mu = k$.

Next, we show that $G' := G - S$ is c-colored P_ℓ-free. Since G is a c-colored graph, it is sufficient to prove that no blue edge is part of a c-colored P_ℓ in G'. First, let $e_b \in E_{G'}(X_i)$ be a blue edge from any variable gadget X_i. Since either $T_i^r, F_i^b \subseteq S$ or $F_i^r, T_i^b \subseteq S$, we can conclude from Claim 3.3 that e_b is not part of a P_ℓ in G'. Note that the proof for Claim 3.3 shows that e_b can neither be part of a c-colored P_ℓ with edges from X_i, nor be part of a c-colored P_ℓ with edges from a connected clause gadget C_j. Next, let $\{u_j, u_j^{p,2}\} \in E'_G(C_j)$ be the blue edge in G' from any clause gadget C_j. By construction of S we know that $deg_{G'}(u_j) = 1$. This implies that $\{u_j, u_j^{p,2}\}$ is only part of a $P_{\ell-1}$ in $G'[C_j]$. Hence, we can conclude that any c-colored P_ℓ containing $\{u_j, u_j^{p,2}\}$ has to contain edges from a variable gadget.

We know by construction that $u_j^{p,2}$ is identified with a vertex from a variable gadget X_i such that $\mathcal{A}(x_i)$ satisfies c_j. Without loss of generality we assume $\mathcal{A}(x_i) = \texttt{true}$. Then, $u_j^{p,2}$ is identified with a vertex $t_i^{q,z}$ for some $q \in [2]$ and $T_i^r, F_i^b \subseteq S$. Assume towards a contradiction that there is a vertex set $V' \subseteq V$ such that $\{u_j, u_j^{p,2}\} \subseteq V'$ and $G'[V']$ is a c-colored P_ℓ. Since $deg_{G'}(u_j) = 1$, we conclude that $\{t_i^{q,z-\ell+2}, \ldots, t_i^{q,z}\} \subseteq V'$. Since $(z - \ell + 2) \bmod (\ell - 1) = 2$ we know by definition of T_i^r that $\{t_i^{q,z-\ell+2}, t_i^{q,z-\ell+3}\} \in T_i^r$. And since $T_i^r \subseteq S$, we conclude that $\{t_i^{q,z-\ell+2}, t_i^{q,z-\ell+3}\} \in S$. Hence, $G'[V']$ is not a c-colored P_ℓ This is a contradiction to the assumption. Hence, $\{u_j, u_j^{p,2}\}$ is not part of a c-colored P_ℓ in G'. Thus, no blue edge is part of a c-colored P_ℓ in G'. Hence, G' is c-colored P_ℓ-free.

($\Leftarrow$) Let S be an edge-deletion set with $|S| \leq k$ such that $G - S$ is c-colored P_ℓ-free. Before we define a satisfying assignment $\mathcal{A} : \mathcal{X} \to \{\texttt{true}, \texttt{false}\}$ for Φ, we show two Claims. First, we show how many edges from each variable gadget and clause gadget have to be in S.

Claim 3.4. *$|S \cap E_G(X_i)| = 4 \cdot d$ for any variable gadget X_i and $|S \cap E_G(C_j) \cap E_b| = 2$ for any clause gadget C_j.*

Proof. First, we will show that $|S \cap E_G(C_j)| \geq 2$. Assume towards a contradiction that $|S \cap E_G(C_j)| < 2$. First, consider the vertex sets $V_1 := U_j^1 \cup \{u_j^{2,2}\}$ and $V_2 := U_j^1 \cup \{u_j^{3,2}\}$. Note that $G[V_1]$ and $G[V_2]$ are two different c-colored P_ℓs and that $V_1 \cap V_2 = U_j^1$. Hence, we conclude that $|S \cap U_j^1| = 1$. Next, consider the vertex set $V_3 := U_j^2 \cup \{u_j^{3,2}\}$. The induced subgraph $G[V_3]$ is a c-colored P_ℓ and $V_3 \cap U_j^1 = \emptyset$. This is a contradiction, since $G - S$ is c-colored P_ℓ-free. Hence, $|S \cap E_G(C_j)| \geq 2$.

By construction $G[X_i]$ contains $4 \cdot d$ edge disjoint c-colored P_ℓs. Hence, we know that $|S \cap E_G(X_i)| \geq 4 \cdot d$. Since $|S| \leq k = 4 \cdot d \cdot \eta + 2 \cdot \mu$, we can conclude that $|S \cap E_G(X_i)| = 4 \cdot d$ and $|S \cap E_G(C_j)| = 2$.

Second, we will show that this implies that $|S \cap E_G(C_j) \cap E_b| = 2$. Assume towards a contradiction that $|S \cap E_G(C_j) \cap E_b| < 2$. Without loss of generality we can assume that $\{u_j, u_j^{1,2}\}, \{u_j, u_j^{2,2}\} \notin S$. Let $V_1 := U_j^1 \cup \{u_j^{2,2}\}, V_2 := U_j^2 \cup \{u_j^{1,2}\}$ and $V_3 := U_j^3 \cup \{u_j^{1,2}\}$. The induced subgraphs $G[V_1], G[V_2]$ and $G[V_3]$ are three c-colored P_ℓs. This is a contradiction, since $S \cap (E_G(V_\alpha) \cap E_G(V_\beta)) = \emptyset$ for $\alpha, \beta \in [3]$ with $\alpha \neq \beta$ and $|S \cap E_G(C_j)| = 2$. Hence, $|S \cap E_G(C_j) \cap E_b| = 2$. △

In the next Claim, we show more specifically which edges from a variable gadget X_i have to be in S.

Claim 3.5. *For each variable gadget X_i we have $S \cap E_G(T_i^1 \cup T_i^2) = T_i^b$ or $S \cap E_G(F_i^1 \cup F_i^2) = F_i^b$.*

Proof. First, we show that $E_G(\{t_i, t_i^{1,2}, t_i^{2,2}\}) \subseteq S$ or $E_G(\{f_i, f_i^{1,2}, f_i^{2,2}\}) \subseteq S$. Consider the four vertex sets in $\widetilde{X}_i := \{T_i^1, T_i^2, F_i^1, F_i^2\}$. By construction, for each set $A \in \widetilde{X}_i$ the induced subgraph $G[A]$ contains d edge disjoint c-colored P_ℓs, and for each $A, B \in \widetilde{X}_i$ with $A \neq B$ we know that $E_G(A) \cap E_G(B) = \emptyset$. From Claim 3.4 we know that $|S \cap E_G(X_i)| = 4 \cdot d$. Hence, we conclude that $|S \cap E_G(A)| = d$. Furthermore, we know by construction that $E_G(A) \cap E_G(W_i \cup \{t_i, f_i\}) = \emptyset$. Hence, we conclude that $S \cap E_G(W_i \cup \{t_i, f_i\}) = \emptyset$. This implies that $E_G(\{t_i, t_i^{1,2}, t_i^{2,2}\}) \subseteq S$ or $E_G(\{f_i, f_i^{1,2}, f_i^{2,2}\}) \subseteq S$, since the induced subgraphs $G[W_i \cup \{t_i, f_i, t_i^{q,2}, f_i^{q',2}\}]$ are c-colored P_ℓs for each $q, q' \in [2]$.

Without loss of generality we assume $E_G(\{t_i, t_i^{1,2}, t_i^{2,2}\}) \subseteq S$. We can conclude that $|S \cap (T_i^1 \setminus \{t_i\})| = d - 1 = |S \cap (T_i^2 \setminus \{t_i\})|$, since $|S \cap T_i^1| = d = |S \cap T_i^2|$, as we showed above. Furthermore, the induced subgraphs $G[T_i^1 \setminus \{t_i\}]$ and $G[T_i^2 \setminus \{t_i\}]$ are paths of $d \cdot (\ell - 1)$ vertices such that any ℓ consecutive vertices form a c-colored P_ℓ. Thus, from Lemma 3.1 we conclude $T_i^b \subseteq S$. Since $|T_i^b| = 2 \cdot d$ and the induced subgraph $G[F_i^1 \cup F_i^2]$ contains $2 \cdot d$ edge disjoint c-colored P_ℓs, we conclude that $S \cap E_G(T_i^1 \cup T_i^2) = T_i^b$ from Claim 3.4. Thus, $S \cap E_G(T_i^1 \cup T_i^2) = T_i^b$ or $S \cap E_G(F_i^1 \cup F_i^2) = F_i^b$. $\triangle$

Now, we will show how to construct an equivalent solution S' such that either $S' \cap E_G(X_i) = F_i^b \cup T_i^r$ or $S' \cap E_G(X_i) = T_i^b \cup F_i^r$ for each variable gadget X_i.

If $S \cap E_G(T_i^1 \cup T_i^2) = T_i^b$, then we can conclude that $S \cap E_G(X_i) = T_i^b \cup F$ such that $F \subseteq E_G(F_i^1 \cup F_i^2)$ from Claim 3.5. We will show that $S' := (S \setminus F) \cup F_i^r$ is an equivalent solution. Since $|T_i^b| = 2 \cdot d$, we can conclude that $|F| = 2 \cdot d$ from Claim 3.4. Hence, it is easy to see that $|S'| = |S|$. Now, we show that $G' := G - S'$ is c-colored P_ℓ-free. Since G is constructed from a (3,B2)-SAT formula, we know that for each clause gadget C_j and each $p \in [3]$, the vertex $u_j^{p,2}$ is identified with exactly one vertex from a variable gadget. Furthermore, we know that the three edges $\{u_j, u_j^{p,2}\}$ are the only blue edges in $E_G(C_j)$ and that G is a c-colored graph. Hence, it is sufficient to show that for any variable gadget X_i there is no blue edge $e_b \in E_{G'}(X_i)$ that is part of a c-colored P_ℓ in G' and for each clause gadget with $u_j^{p,2} = t_i^{q,z}$ or $u_j^{p,2} = f_i^{q,z}$ for some $p \in [3], q \in [2]$, the blue edge $\{u_j, u_j^{p,2}\}$ is not part of a c-colored P_ℓ in G'. Since $S' \cap E_G(X_i) = F_i^b \cup T_i^r$, it follows by Claim 3.3 that no blue edge from $E_{G'}(X_i)$ is part of a c-colored P_ℓ in G'. Let C_j be a clause gadget such that $u_j^{p,2} = t_i^{q,z}$ or $u_j^{p,2} = f_i^{q,z}$ for some $p \in [3], q \in [2]$. From Claim 3.4 we conclude that S' contains exactly two blue edges from $E_G(C_j)$. Hence, $\deg_{G'}(u_j) = 1$. We consider two cases.

(Case 1) $u_j^{p,2} = t_i^{q,z}$ for some $p \in [3], q \in [2]$. We will show that $\{u_j, u_j^{p,2}\} \in S'$. Assume towards a contradiction that $\{u_j, u_j^{p,2}\} \notin S'$. Since $S' \cap E_G(C_j) = S \cap E_G(C_j)$, we conclude that $\{u_j, u_j^{p,2}\} \notin S$. And since $S \cap E_G(T_i^1 \cup T_i^2) = T_i^b$, we conclude that the induced subgraph $G[\{u_j, t_i^{q,z}, \dots, t_i^{q,z-\ell+2}\}] - S$ is a c-colored P_ℓ. This is a contradiction, since $G - S$ is c-colored P_ℓ free. Hence, $\{u_j, u_j^{p,2}\} \in S'$ and thus, $\{u_j, u_j^{p,2}\}$ is not part of a c-colored P_ℓ in G'.
(Case 2) $u_j^{p,2} = f_i^{q,z}$ for some $p \in [3], q \in [2]$. Since it is obvious that $\{u_j, u_j^{p,2}\}$ is not part of a c-colored P_ℓ in G' if $\{u_j, u_j^{p,2}\} \in S'$, we only consider the case in which $\{u_j, u_j^{p,2}\} \notin S'$. Since $(z - \ell + 2) \bmod (\ell - 1) = 2$, we know by definition of F_i^r that $\{f_i^{q,z-\ell+2}, f_i^{q,z-\ell+3}\} \in F_i^r$. This implies that $\deg_{G'}(f_i^{q,z-\ell+3}) = 1$, since $F_i^r \subseteq S'$. Hence, we can conclude that the longest path in G' that contains $\{u_j, u_j^{p,2}\}$ is the induced subgraph $G'[V' := \{u_j, f_i^{q,z}, \dots, f_i^{q,z-\ell+3}\}]$. It is not hard to see that $|V'| = \ell - 1$. Thus, $\{u_j, u_j^{p,2}\}$ is not part of a c-colored P_ℓ in G'.

Hence, no blue edge is part of a c-colored P_ℓ in G' and therefor, S' is a solution such that $S' \cap E_G(X_i) = T_i^b \cup F_i^r$.

If $S \cap E_G(T_i^1 \cup T_i^2) \neq T_i^b$, then we conclude that $S \cap E_G(F_i^1 \cup F_i^2) = F_i^b$ from Claim 3.5. Hence, $S \cap E_G(X_i) = F_i^b \cup T$ such that $T \subseteq E_G(T_i^1 \cup T_i^2)$. With an analogous argument, we can show that $S' := (S \setminus T) \cup T_i^r$ is an equivalent solution. Thus, we have shown how to construct a solution S' from S such that either $S' \cap E_G(X_i) = F_i^b \cup T_i^r$ or $S' \cap E_G(X_i) = T_i^b \cup F_i^r$

Now we can define a satisfying assignment $\mathcal{A} : \mathcal{X} \to \{\texttt{true}, \texttt{false}\}$ for Φ as:

$$\mathcal{A}(x_i) := \begin{cases} \texttt{true} & \text{if } S' \cap E_G(X_i) = F_i^b \cup T_i^r \\ \texttt{false} & \text{if } S' \cap E_G(X_i) = T_i^b \cup F_i^r. \end{cases}$$

It remains to show that $\mathcal{A}$ satisfies Φ. Let $c_j \in \mathcal{C}$. We know that there is exactly one $p \in [3]$ such that $\{u_j, u_j^{p,2}\} \notin S$. Let $x_i \in \mathcal{X}$ be the variable that occurs as the p-th literal in c_j. If the p-th literal in c_j is a positive literal, we know that the vertex $u_j^{p,2}$ is identified with the vertex $t_i^{q,z}$ from the variable gadget X_i for $q \in [2]$. Since S' is a solution and $\{u_j, u_j^{p,2}\} \notin S'$, we conclude that $T_i^r \subseteq S'$. Hence, $S' \cap E_G(X_i) = F_i^b \cup T_i^r$. Thus, $\mathcal{A}(x_i) = \texttt{true}$ and therefore $\mathcal{A}(x_i)$ satisfies c_j. If the p-th literal in c_j is a negative literal, the argument works analogously. So each clause c_j is satisfied by the assignment $\mathcal{A}$ and therefore Φ is satisfied by $\mathcal{A}$. □

If the girth of a graph G is greater than ℓ it is not hard to see, that each subgraph that is isomorphic to a c-colored P_ℓ is an induced subgraph. Hence, we can conclude the following.

Corollary 3.6. *$c P_\ell$D is NP-hard for each $\ell \geq 4$ and $c \in [2, \ell - 2]$ on non-cascading graphs.*

Next, we consider the case $c = \ell - 1$. We will prove that $(\ell - 1)P_\ell$D is NP-hard.

Theorem 3.7. *$(\ell-1)P_\ell$D is NP-hard for any $\ell \geq 4$ even if the maximum degree of G is 16.*

Proof. We prove this Theorem by giving a polynomial time reduction from the NP-hard $2P_3$D Problem [14].

Construction: Let (G, k) be an instance of $2P_3$D. We will show how to construct an equivalent instance (H, k) of $(\ell-1)P_\ell$D for any $\ell \geq 4$. We use the instance (G, k) and add vertices and edges with new colors. Hence, $V(G) \subseteq V(H)$ and $E(G) \subseteq E(H)$. For each vertex $v \in V(G)$ we add $\deg_G(v) \cdot (\ell-3)$ new vertices v_i^j for $i \in [\deg_G(v)]$ and $j \in [\ell-3]$ to $V(H)$. Recall that yellow is the third color. We add yellow edges $\{v, v_i^1\}$ for each $i \in [\deg_G(v)]$. And for each $j \in [\ell-4]$ we add an edge $\{v_i^j, v_i^{j+1}\}$ with color $j+3$. Note that for each $i \in [\deg_G(v)]$ the induced subgraph $H[\{v, v_i^1, \dots, v_i^{\ell-3}\}]$ is an $(\ell-3)$-colored $P_{\ell-2}$. Hence, each vertex $v \in V(G)$ is part of $\deg_G(v)$ edge disjoint $(\ell-3)$-colored $P_{\ell-2}$ in H (see Figure 4). The budget k remains the same.

By construction, we know that $\deg_H(v) = 2 \cdot \deg_G(v)$ for each $v \in V(G)$, and that $\deg_H(v_i^j) \in [2]$ for the new vertices $v_i^j \in V(H) \setminus V(G)$. Since $2P_3$D is NP-hard even if the maximum degree of G is eigth [14], the correctness of the reduction will imply the NP-hardness even if the maximum degree of H is 16.

Correctness: We will now prove the correctness of the reduction by showing that (G, k) is a yes-instance of $2P_3$D if and only if (H, k) is a yes-instance of $(\ell-1)P_\ell$D.

($\Rightarrow$) Let S be an edge-deletion set of size at most k such that $G-S$ is 2-colored P_3-free. Since H is a $(\ell-1)$-colored graph, each $(\ell-1)$-colored P_ℓ in H has to include exactly one red edge and exactly one blue edge. By construction, we know that an edge $e \in E(H)$ is red or blue if and only if $e \in E(G)$. Since v_i^1 is the only vertex from $\{v_i^1, \dots, v_i^{\ell-3}\}$ that is adjacent to a vertex from $V(G)$ for each $i \in [\deg_G(v)]$, we conclude that each $(\ell-1)$-colored P_ℓ in H has to include an induced 2-colored P_3 from G. Since $G-S$ is 2-colored P_3-free, $H-S$ is $(\ell-1)$-colored P_ℓ-free. Thus, S is a solution for (H, k).

($\Leftarrow$) Let S be an edge-deletion set of size at most k such that $H-S$ is $(\ell-1)$-colored P_ℓ-free.

First, we will show how to construct an equivalent solution S' with $|S'| \leq |S|$ such that S' only contains blue, red and yellow edges. Note that this is only necessary if $\ell > 4$. Let $\{v_i^j, v_i^{j+1}\} \in S$ be an edge with color $c > 3$. Let $V' \subseteq V(H)$ be a vertex set such that $v_i^j, v_i^{j+1} \in V'$ and $H[V']$ is an $(\ell-1)$-colored P_ℓ. Since H is an $(\ell-1)$-colored graph, each $(\ell-1)$-colored P_ℓ has to include a red edge and a blue edge. By construction, the only vertices, to which blue and red edges can be incident, are the vertices from $V(G)$. Hence, we conclude that $v \in V'$.

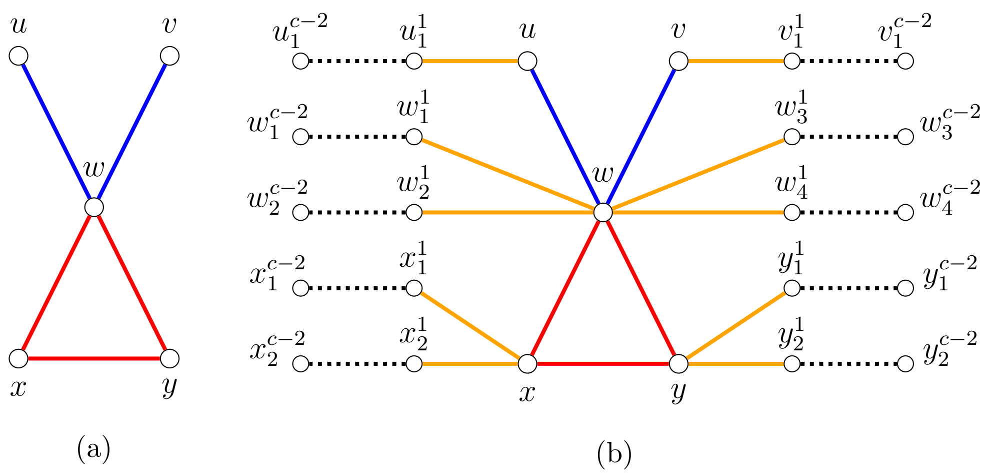

Figure 4: (a) A 2-colored Graph G. (b) The graph H with the new vertices and edges. The black dotted lines each represent a $(\ell-4)$-colored path containing $\ell-3$ with no blue, red or yellow edges.

Thus, $S' := (S \setminus \{v_i^j, vi^{j+1}\}) \cup \{\{v, v_i^1\}\}$ is an equivalent solution that only includes blue, red and yellow edges.

Second, we will show how to construct an equivalent solution S'' from S' that only contains blue and red edges. Recall that $E_H(\{v\}, \{v_i^1 \mid i \in [\deg_G(v)]\})$ denotes the set of all yellow edges that are incident to a vertex $v \in V(G)$. Let $\{v, v_i^1\} \in S'$ be a yellow edge. We have to consider two cases.

(Case1) $E_H(\{v\}, \{v_i^1 \mid i \in [\deg_G(v)]\}) \subseteq S'$. Since G is a $(\ell-1)$-colored graph, we know by construction that each $(\ell-1)$-colored P_ℓ that contains the vertex v, has to contain a red or blue edge that is incident to v. Thus, we can construct an equivalent solution by swapping the $\deg_G(v)$ yellow edges that are incident to v with the $\deg_G(v)$ blue or red edges that are incident to v.

We set $S'' := (S' \setminus E_H(\{v\}, \{v_i^1 \mid i \in [\deg_G(v)]\})) \cup E_G(\{v\}, N_G(v))$. Since there are no red or blue edges in $H - S''$ that are incident to v, we conclude that v is not part of a $(\ell-1)$-colored P_ℓ in $H - S''$ and therefor no edge in $E_H(\{v\}, \{v_i^1 \mid i \in [\deg_G(v)]\})$ is part of a $(\ell-1)$-colored P_ℓ in $H - S''$. Hence, S'' is an equivalent solution with no yellow edges.

(Case 2) $E_H(\{v\}, \{v_i^1 \mid i \in [\deg_G(v)]\}) \nsubseteq S'$. Then, for some $\alpha \in [\deg_G(v)]$ there is a yellow edge $\{v, v_\alpha^1\} \notin S'$. Let $\{v, v_\beta^1\} \in S'$ such that $\beta \neq \alpha$. Since S' is a solution and $H[\{v, v_\alpha^1, \ldots, v_\alpha^{c-2}\}] \cong H[\{v, v_\beta^1, \ldots, v_\beta^{c-2}\}]$, we conclude that $S' \setminus \{v, v_\beta^1\}$ is a solution. Hence, we set $S'' := (S' \setminus E_H(\{v\}, \{v_i^1 \mid i \in [\deg_G(v)]\}$ and get an equivalent solution with no yellow edges.

Thus, S'' only contains red and blue edges. It remains to show that S'' is a solution for (G, k). Assume towards a contradiction that there is an induced 2-colored P_3 in $G - S''$. Without loss of generality, this implies that there is an edge set $\{\{u,v\}\{v,w\}\} \subseteq (E(G) \setminus S'')$ so that $\{u,v\}$ is a blue edge, $\{v,w\}$ is a red edge and $\{u,w\} \notin (E(G) \setminus S'')$. Since S'' only contains blue or red edges, we can conclude that there is a vertex set $U_\alpha := \{u, u_\alpha^1, \dots, u_\alpha^{\ell-3}\}$ such that $H[U_\alpha] - S''$ is an induced $(\ell-3)$-colored $P_{\ell-2}$ with no blue or red edges. Hence, we conclude that $H[U_\alpha \cup \{v,w\}]$ is an induced $(\ell-1)$-colored P_ℓ. This is a contradiction, since S'' is an solution for (H,k). Hence, $G - S''$ is 2-colored P_3-free. Thus, S'' is a solution for (G,k). □

Since $2P_3$D is NP-hard [14] and $1P_\ell$D is NP-hard [12], we can conclude the following from Theorem 3.2 and Theorem 3.7.

Corollary 3.8. *cP_ℓD is NP-hard for each $\ell \geq 3$ and each $c \in [\ell-1]$.*

3.2 c-colored C_ℓ Deletion

Next, we analyze the computational complexity of cC_ℓD. The problem is known to be NP-hard for any $\ell \geq 3$ and $c = 1$ [5]. We will show that cC_ℓD is NP-hard for any $\ell \geq 3$ and any $c \in [\ell]$. But before we consider cC_ℓD, we establish an NP-hardness result for VC on C_3-free and tripartite graphs that we will use in our reduction.

Lemma 3.9. VC *is NP-hard even if G is C_3-free and tripartite.*

Proof. The IS problem is NP-hard on 2-subdivision graphs [31]. Since (G,k) is a yes-instance of IS if and only if $(G, n-k)$ is a yes-instance of VC [28], we conclude that VC is NP-hard on 2-subdivision graphs. Since each 2-subdivision graph is C_3-free and tripartite, VC is NP-hard on C_3-free and tripartite graphs. □

Now we can show the NP-hardness of cC_ℓD for $c = \ell$. With this result we will be able to prove the NP-hardness for all $c \in [\ell]$.

Theorem 3.10. *ℓC_ℓD is NP-hard for any $\ell \geq 3$ even if the girth of G is ℓ and every C_ℓ in G is ℓ-colored.*

Proof. We give a polynomial time reduction from the NP-hard VC problem on C_3-free and tripartite graphs (see Lemma 3.9). Note that this reduction is very similar to the one given by Yannakakis [5] to prove the NP-hardness for $c = 1$.

Let (H,k) be an instance of VC such that H is tripartite and C_3-free. Before we describe how to construct an equivalent instance (G,k) of ℓC_ℓD, we define two functions to color the vertices and edges of H. Since H is tripartite, there is a function $\varphi : V(H) \to [3]$ such that $\varphi(u) \neq \varphi(v)$ for each edge $\{u,v\} \in E(H)$. Hence, there is exactly one $\gamma \in [3] \setminus \{\varphi(u), \varphi(v)\}$. Thus, $\psi : E(H) \to [3]$ with $\psi(\{u,v\}) = \gamma$ is a well defined function.

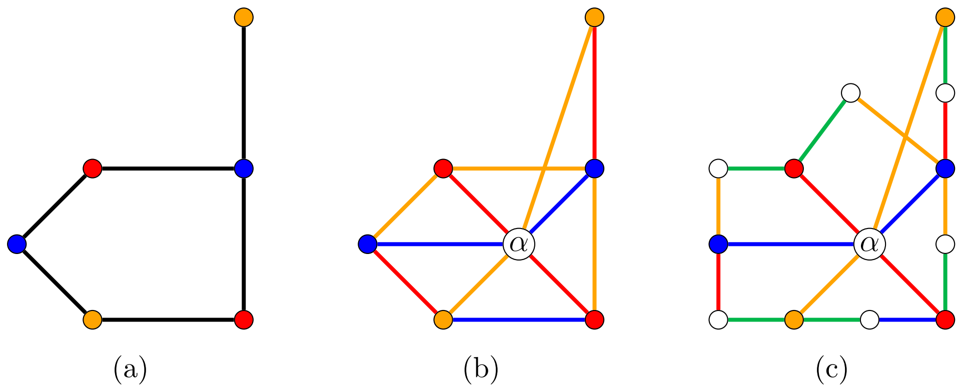

Figure 5: (a) A tripartite, C_3-free graph H. (b) The graph constructed on input (a) if $\ell = 3$. (c) The graph constructed on input (a) if $\ell = 4$.

Construction: Now we show how to construct (G, k). If $\ell = 3$, then we assign the color $\psi(\{u, v\})$ to each edge $\{u, v\} \in E(H)$. If $\ell > 3$, then we subdivide each edge $\{u, v\} \in E(H)$ with vertices $W^{uv} := \{w_1^{uv}, \dots, w_{\ell-3}^{uv}\}$. We then color the edges as follows. We assign color $\psi(\{u, v\})$ to the edge $\{u, w_1^{uv}\}$ and color ℓ to the edge $\{w_{\ell-3}^{uv}, v\}$. For $i \in \{1, \dots, \ell-4\}$, the edges $\{w_i^{uv}, w_{i+1}^{uv}\}$ are assigned color $i+3$. Hence, the induced subgraph $G[\{u, v\} \cup W^{uv}]$ is an $(\ell-2)$-colored $P_{\ell-1}$ with all colors except $\varphi(u)$ and $\varphi(v)$. Then, we add a vertex α. To complete the reduction we add an edge $\{v, \alpha\}$ with color $\varphi(v)$ for each $v \in V(H)$. The budget k remains the same.

Before we show the correctness of the reduction we prove the following Claim about the structure of G.

Claim 3.11. *The girth of G is ℓ and every C_ℓ in G is ℓ-colored.*

Proof. Obviously (G, k) is a trivial yes-instance if the girth of G is greater than ℓ. Since H is C_3-free and every edges in $E(H)$ corresponds to a induced $P_{\ell-1}$ in G, we conclude that the girth of $G[V \setminus \{\alpha\}]$ is greater than ℓ. Furthermore, we know by construction that for each edge $\{u, v\} \in E(H)$ the induced subgraph $G[\{\alpha, u, v\} \cup W^{uv}]$ is an ℓ-colored C_ℓ, since u and v are connected by an $(\ell-2)$-colored $P_{\ell-1}$ with all colors except $\varphi(u)$ and $\varphi(v)$, the edge $\{u, \alpha\}$ has color $\varphi(u)$ and the edge $\{v, \alpha\}$ has color $\varphi(v)$. Hence, we conclude that the girth of G is ℓ and every C_ℓ in G is ℓ-colored. $\triangle$

Correctness: To show the correctness of the reduction, we prove that (H, k) is a yes-instance of VC if and only if (G, k) is a yes-instance of ℓC_ℓD.

($\Rightarrow$) Let $V' \subseteq V(H)$ be a vertex cover of size at most k. Consider the edge-deletion set $S := \{\{v, \alpha\} \mid v \in V'\}$. Since $|V'| = |S|$, we know that $|S| \leq k$. To show that $G' = G - S$ is C_ℓ-free, it is sufficient to show that α is not part of a C_ℓ in G'. Let $\{\alpha, v\} \in E(G')$. By construction we know that $v \in V(H)$ and $v \notin V'$.

Assume towards a contradiction that $\{v, \alpha\}$ is part of a C_ℓ. Then, there is an edge $\{u, \alpha\} \in E(G')$ such that $u \neq v$ and u is connected to v by an induced $P_{\ell-1}$. This implies that $u \in V(H)$ and $u \notin V'$. Since $u, v \in V(H)$ and u is connected to v by induced $P_{\ell-1}$ in G, we can conclude that $\{u, v\} \in E(H)$. But that is a contradiction, since V' is a vertex cover of H and $v, u \notin V'$. Hence, G' is C_ℓ-free.

($\Leftarrow$) Let S be an edge-deletion set of size at most k such that $G' := G - S$ is ℓ-colored C_ℓ-free. Let $E' := \{\{u, v\} \in E(G) \mid u, v \neq \alpha\}$ be the set of edges from G that are not incident with α. We observe that each edge $e \in E'$ is part of at most one C_ℓ in G. Since α is part of every C_ℓ in G, we know that there is a vertex $\beta \in V(G)$ such that $S' = (S \setminus \{e\}) \cup \{\{\alpha, \beta\}\}$ is an equivalent solution. Since β is adjacent to α, we can conclude that $\beta \in V(H)$.

To finish the proof, we show that $V' := \{\beta \mid \{\alpha, \beta\} \in S'\}$ is a vertex cover for H. Let $\{v_1, v_2\} \in E(H)$ be an edge from H. Assume towards a contradiction that $v_1, v_2 \notin V'$. This implies that $\{\alpha, v_1\}, \{\alpha, v_2\} \notin S'$. By construction, we know that v_1 and v_2 are connected by an induced $P_{\ell-1}$ that consists of edges from E' and contains all colors except $\varphi(v_1)$ and $\varphi(v_2)$. Since $\{\alpha, v_1\}$ has color $\varphi(v_1)$, $\{\alpha, v_2\}$ has color $\varphi(v_2)$ and $S' \cap E' = \emptyset$, we conclude that v_1 and v_2 are part of a ℓ-colored C_ℓ in G'. That is a contradiction, since G' is ℓ-colored C_ℓ-free. So $v_1 \in V'$ or $v_2 \in V'$. Hence, V' is a vertex cover for H. □

Now we will use this result to prove the NP-hardness of cC_ℓD for all $c \in [\ell - 1]$.

Lemma 3.12. *cC_ℓD is NP-hard for each $c \in [\ell - 1]$ even if the girth of G is ℓ and each C_ℓ in G is c-colored.*

Proof. We give a polynomial time reduction from ℓC_ℓD on graphs where each C_ℓ is ℓ-colored.

Construction: Let (H, k) be an instance of ℓC_ℓD where the girth of H is ℓ and each C_ℓ in H is ℓ-colored. To construct an equivalent instance (G, k) of cC_ℓD we recolor the edges. For each $\alpha \in [c - 1]$ we set $E_\alpha(G) := E_\alpha(H)$. Next, we set $E_c(G) := E_c(H) \cup \ldots \cup E_\ell(H)$. Note that the vertex set and the budget k remains the same. Since we do not add new edges, the girth remains the same, and since each C_ℓ in H is ℓ-colored, we know by construction that each C_ℓ in G is c-colored.

Correctness: Since every C_ℓ in H is ℓ-colored, for each vertex set $V' \subseteq V(H)$ the induced subgraph $H[V']$ is an ℓ-colored C_ℓ if and only if the induced subgraph $G[V']$ is a c-colored C_ℓ. Hence, (H, k) is a yes-instance of ℓC_ℓD if and only if (G, k) is a yes-instance of cC_ℓD. □

It is not hard to see that each subgraph in G that is isomorphic to some c-colored C_ℓ is an induced subgraph if G has a girth of ℓ. Hence, we finish our analysis of NP-hardness by concluding the following from Theorem 3.10 and Lemma 3.12.

Corollary 3.13. *$c C_\ell$D is NP-hard for each $\ell \geq 3$ and each $c \in [\ell]$ even if the girth of G is ℓ and each C_ℓ in G is c-colored. Hence, $c C_\ell$D is NP-hard on non-cascading graphs.*

4 Parameterized Complexity

After proving the NP-hardness, we will now consider the parameterized complexity of cP_ℓD and cC_ℓD parameterized by the natural parameter k and the dual parameter $\delta := m - k$. Furthermore, we analyze the parameterized complexity of CPD. We will show most theorems for the more general $\mathcal{F}$ Deletion problem where $\mathcal{F}$ is a d-edge-bounded, finite set of graphs such that $\mathcal{F}$ is polynomial-enumerable on each input graph G. To show that these results are applicable to cP_ℓD and cC_ℓD, we first prove the following Lemma.

Lemma 4.1. ***I**. Each set of c-colored P_ℓs is $(\ell - 1)$-edge-bounded and each set of c-colored C_ℓs is ℓ-edge-bounded.*
***II**. If ℓ is constant, then the set of all c-colored P_ℓs and the set of all c-colored C_ℓs are polynomial-enumerable on any graph G.*

Proof. **I**. This is obvious by the definitions of P_ℓ and C_ℓ.
II. We can find all induced c-colored P_ℓs in $\mathcal{O}(n^\ell \cdot \ell^2)$ time by considering all vertex sub sets of ℓ vertices. Thus, if ℓ is constant, then the set of all c-colored P_ℓs and the set of all c-colored C_ℓs are polynomial-enumerable on any graph G. □

To prove that cP_ℓD and cC_ℓD are fixed-parameter tractable when parameterized by k we give a naive branching algorithm.

Theorem 4.2. *Let $\mathcal{F}$ be a d-edge-bounded, finite set of graphs such that $\mathcal{F}$ is polynomial-enumerable on G. Then, $\mathcal{F}$ Deletion can be decided in $\mathcal{O}(d^k \cdot poly(|G|))$ where $poly(|G|)$ is a polynomial in the size of G when parameterized by k and, hence, is fixed-parameter tractable.*

Proof. We give the following naive branching algorithm to prove this Theorem. First, we check if there is a vertex set $V' \subseteq V(G)$ such that $G[V'] \cong F$ for some $F \in \mathcal{F}$. Since $\mathcal{F}$ is polynomial-enumerable on G, this can be done in polynomial time. If there is no such vertex set, then (G, k) is a yes-instance. Else, if $k < 1$, then (G, k) is a no-instance. If $k \geq 1$, then we branch into the cases $(G - \{e\}, k - 1)$ for each $e \in E_G(V')$.

The running time of this algorithm is $\mathcal{O}(d^k \cdot poly(|G|))$ where $poly(|G|)$ is a polynomial in the size of G, such that we can find a vertex set $V' \subseteq V(G)$ with $G[V'] \cong F$ for some $F \in \mathcal{F}$ in $\mathcal{O}(poly(|G|))$ time if such a vertex set exists. □

Since the set of all c-colored P_ℓs is $(\ell-1)$-edge-bounded, the set of all c-colored C_ℓs is ℓ-edge-bounded and we can find an induced c-colored P_ℓ or C_ℓ respectively in $\mathcal{O}(n^\ell \cdot \ell^2)$ time by considering each vertex sub set of ℓ vertices, we conclude the following from Lemma 4.1 and Theorem 4.2.

Corollary 4.3. *cP_ℓD can be decided in $\mathcal{O}((\ell-1)^k \cdot n^\ell \cdot \ell^2)$ time and cC_ℓD can be decided in $\mathcal{O}(\ell^k \cdot n^\ell \cdot \ell^2)$ time. Hence, cP_ℓD and cC_ℓD are fixed-parameter tractable when parameterized by k, since ℓ is a constant.*

Next, we give a W[2]-hardness result that implies that CPD is not fixed-parameter tractable when parameterized by k, unless W[2]=FPT.

Theorem 4.4. CPD *is $W[2]$-hard when parameterized by k even if $c = 3$.*

Proof. We prove the $W[2]$-hardness by giving a parameterized reduction from the $W[2]$-hard problem HS parameterized with k [29]. Let $((\mathcal{X}, \mathcal{C}), k)$ be an instance of HS. Recall that $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_\mu\}$ is a collection of μ subsets of a finite set $\mathcal{X} = \{x_1, \ldots, x_\eta\}$. We can assume that each $\mathcal{C}_j$ is non-empty and that each $x_i \in \mathcal{X}$ occurs in at least one subset $\mathcal{C}_j \in \mathcal{C}$.

Construction: To construct an equivalent instance (G, c, ℓ, k) of CPD we first set $c = 3$ and $\ell = 1 + 3 \cdot \eta$. Then, we construct the following gadgets.

For each $x_i \in \mathcal{X}$ we construct an *element gadget* W_i as follows: We add two vertices $w_i, \widetilde{w}_i$ to W_i and connect w_i with $\widetilde{w}_i$ by a blue edge. By W we denote the set of all element gadgets.

Next, we construct a *subset gadget* C_j for each subset $\mathcal{C}_j \in \mathcal{C}$. We add a vertex v^j, and for each $i \in [\eta]$ a vertex u_i^j to C_j. Then, we add a yellow edge $\{v^j, u_1^j\}$. If $x_i \in \mathcal{C}_j$, then we connect the corresponding element gadget by adding a red edge $\{u_i^j, w_i\}$ and if $i < \eta$ a red edge $\{\widetilde{w}_i, u_{i+1}^j\}$. Else if $x_i \notin \mathcal{C}_j$, then we add two vertices $w_i^j, \widetilde{w}_i^j$ to C_j, and we add red edges $\{u_i^j, w_i^j\}, \{w_i^j, \widetilde{w}_i^j\}$ to G and if $i < \eta$ we add a red edge $\{\widetilde{w}_i^j, u_{i+1}^j\}$ to G. All edges that we have added so far are called *unfixed* edges.

Finally, we connect the subset gadgets as follows. Let $\mathcal{C}_p, \mathcal{C}_q$ be subsets such that $x_i \in \mathcal{C}_p$ and $x_i \in \mathcal{C}_q$ for an element $x_i \in \mathcal{X}$. We add red edges $\{u_i^p, u_i^q\}$, $\{v^p, u_i^q\}$, $\{v^q, u_i^p\}$, $\{u_1^p, u_i^q\}$, $\{u_1^q, u_i^p\}$ and if $i < \eta$ we add red edges $\{u_{i+1}^p, u_{i+1}^q\}$, $\{v^p, u_{i+1}^q\}$, $\{v^q, u_{i+1}^p\}$, $\{u_1^p, u_{i+1}^q\}$, $\{u_1^q, u_{i+1}^p\}$ (see Figure 6). We call these edges *fixed*. Observe that for any edge $e \in E$ we call e a fixed edge if e connects two vertices from different subset gadgets and otherwise, we call e an unfixed edge.

Intuition: Before we prove the correctness of the reduction, we describe its idea. We connected the subset gadgets to element gadgets such that for each subset gadget C_j, the induced subgraph $G[C_j \cup W]$ contains exactly one induced 3-colored P_ℓ. We then connected the subset gadgets such that there is no induced 3-colored P_ℓ in G that contains vertices from two different subset gadgets. So we can model a

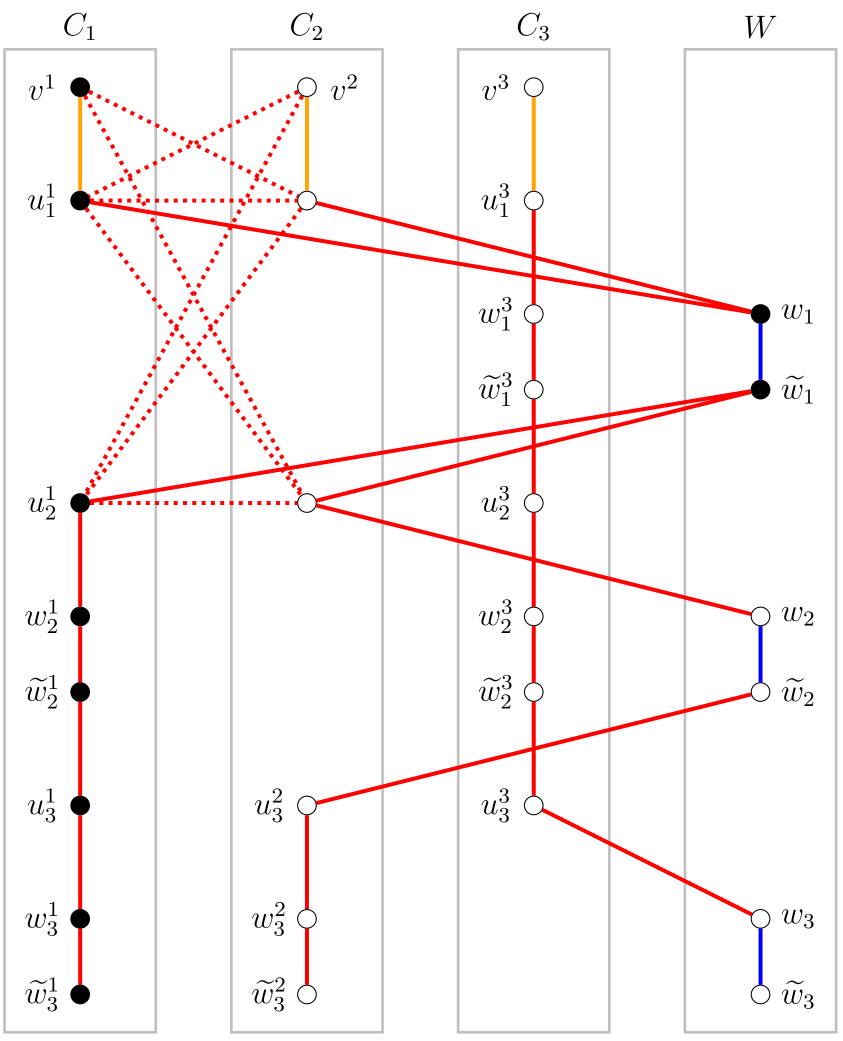

Figure 6: The constructed graph for $\mathcal{C} = \{\mathcal{C}_1 = \{1\}, \mathcal{C}_2 = \{1, 2\}, \mathcal{C}_3 = \{3\}\}$ and $\mathcal{X} = [3]$. The dotted lines represent fixed edges, while the solid lines represent unfixed edges. The filled vertices induce the 3-colored P_{10} in $G[\mathcal{C}_1 \cup W]$.

hitting set for a collection $\mathcal{C}$ by deleting the edges from the corresponding element gadgets.

Correctness: Before proving the correctness of the reduction, we show the following claims. First, we show that each 3-colored P_ℓ contains vertices of exactly one subset gadget if we do not delete fixed edges.

Claim 4.5. *Let S be an edge-deletion set that does not contain fixed edges. Furthermore, let $G' := G - S$ and $V' \subseteq V(G)$ such that the induced subgraph $G'[V']$ is a 3-colored P_ℓ. Then, V' contains vertices of at most one subset gadget C_j.*

Proof. Since each 3-colored P_ℓ has to include a yellow edge, we conclude that $v^p, u_1^p \in V'$ for some $p \in [\mu]$. Assume towards a contradiction that V' contains a vertex from a clause gadget C_q such that $q \neq p$. Without loss of generality there are adjacent vertices $\alpha, \beta \in V'$ such that $\alpha \in C_p$, $\beta \in C_q$ and $\{\alpha, \beta\} \in E(G)$, since we added the fixed edges. By construction, we know that $\{v^p, \beta\}, \{u_1^p, \beta\} \in E \setminus S$ since $\{v^p, \beta\}, \{u_1^p, \beta\}$ are fixed edges. This is a contradiction since the induced subgraph $G'[\{v^p, u_1^p, \beta\}]$ is a C_3. $\triangle$

Next, we show the following claim about 3-colored P_ℓs that are induced by one subset gadget.

Claim 4.6. *For each subset gadget C_j the induced subgraph $G[C_j \cup W]$ contains exactly one 3-colored P_ℓ. This 3-colored P_ℓ contains a blue edge $\{w_i, \widetilde{w}_i\}$ if and only if $x_i \in \mathcal{C}_j$.*

Proof. By construction, we know that $G[C_j \cup W]$ contains at least one 3-colored P_ℓ, and that $\{v^j, u_1^j\}$ is the only yellow edge in $E_G(C_j \cup W)$. Since $\deg_{G[C_j \cup W]}(v^j) = 1$ and $\deg_{G[C_j \cup W]}(\alpha) \leq 2$ for each $\alpha \in C_j \cup W$, we conclude that there is at most one 3-colored P_ℓ in $G[C_j \cup W]$.

By construction, this 3-colored P_ℓ contains a blue edge $\{w_i, \widetilde{w}_i\}$ if $x_i \in \mathcal{C}_j$. Otherwise, if $x_i \notin \mathcal{C}_j$, then the 3-colored P_ℓ contains a red edge $\{w_i^j, \widetilde{w}_i^j\}$ and does not contain the blue edge $\{w_i, \widetilde{w}_i\}$. $\triangle$

Now we prove the correctness of the reduction by showing that $(\mathcal{C}, k)$ is a yes-instance of HS if and only if (G, c, ℓ, k) is a yes-instance of CPD.

($\Rightarrow$) Let $H \subseteq \mathcal{X}$ be a hitting set of size at most k for $(\mathcal{X}, \mathcal{C})$. Consider the edge set $S := \{\{w_i, \widetilde{w}_i\} \mid x_i \in H\}$. We will show that $G' := G - S$ is 3-colored P_ℓ-free. Since S does not contain fixed edges, we know from Claim 4.5 that there is no induced 3-colored P_ℓ in G' that contains vertices from two different subset gadgets. Hence, it remains to show that no subgraph $G'[C_j \cup W]$ contains an induced 3-colored P_ℓ for any subset gadget C_j.

Let C_j be a subset gadget. By Claim 4.6 we know that there is exactly one 3-colored P_ℓ in the induced subgraph $G[C_j \cup X]$ and that this 3-colored P_ℓ includes exactly one blue edge from each element gadget W_i where $x_i \in \mathcal{C}_j$. Since H is a hitting set for $\mathcal{C}$, we conclude that S includes at least one blue edge from an element gadget W_i that is part of the 3-colored P_ℓ in $G[C_j \cup W]$. Hence, $G'[C_j \cup W]$ is 3-colored P_ℓ-free, which implies that G' is 3-colored P_ℓ-free.

($\Leftarrow$) Conversely, let S be an edge-deletion set of size at most k such that $G - S$ is 3-colored P_ℓ-free. First, we show that $S' := S \setminus \{e \in S \mid e \text{ is fixed}\}$ is an equivalent solution. Assume towards a contradiction that $G' := G - S'$ contains an induced 3-colored P_ℓ. Let $V' \subseteq V(G)$ such that $G'[V']$ is a 3-colored P_ℓ. Since S' does not contain fixed edges, we know by Claim 4.5 that V' contains vertices of at most one subset gadget C_j. Hence, $G'[V']$ only contains unfixed edges. This implies that $G'[V']$ does not contain an edge from S. Hence, the induced subgraph $G[V']$

is a 3-colored P_ℓ in G'. This is a contradiction, since $G - S$ is 3-colored P_ℓ-free. Hence, S' is an equivalent solution.

Next, we will show how to construct another equivalent solution $S'' \subseteq E_G(W)$ from S'. Let $\{\alpha, \beta\} \in S'$ such that $\{\alpha, \beta\} \notin E_G(W)$. We will show that $\{\alpha, \beta\}$ is part of at most one 3-colored P_ℓ. From Claim 4.5, we conclude that any 3-colored P_ℓ including $\{\alpha, \beta\}$ only includes vertices from at most one subset gadget, since S' only contains unfixed edges. Hence, $\{\alpha, \beta\}$ can only be part of a 3-colored P_ℓ in an induced subgraph $G[C_j \cup W]$ where $\alpha \in C_j$ or $\beta \in C_j$. From Claim 4.6 we know that there is exactly one 3-colored P_ℓ in $G[C_j \cup W]$. Hence, $\{\alpha, \beta\}$ is part of at most one 3-colored P_ℓ. This 3-colored P_ℓ has to include a blue edge $e \in E_G(W)$, since G is 3-colored and the edges in $E_G(W)$ are the only blue edges. Hence, $(S \setminus \{\{\alpha, \beta\}\}) \cup \{e\}$ is an equivalent solution. Thus, we can construct an equivalent solution S'' that only contains edges from $E_G(W)$.

To finish the proof, we will show that the set $H := \{x_i \mid \{w_i, \widetilde{w}_i\} \in S''\}$ is a hitting set for $\mathcal{C}$. Let $\mathcal{C}_j \in \mathcal{C}$. From Claim 4.6 we know that $G[C_j \cup W]$ includes a 3-colored P_ℓ. Hence, there is at least one edge from that 3-colored P_ℓ in every solution. Since S'' is a solution that only contains edges $\{w_i, \widetilde{w}_i\} \in E_G(W)$, we conclude that there is at least one $x_i \in H$ such that $x_i \in \mathcal{C}_j$. Hence, H is a hitting set for $\mathcal{C}$. □

We showed that CPD is probably not fixed-parameter tractable. But even if a problem is fixed-parameter tractable, it is unclear whether or not it admits a polynomial kernel. In uncolored graphs it is unlikely that one of the problems sudied here admit a polynomial kernel. More specifically, there is no polynomial kernel for $1P_\ell$D for any $\ell \geq 5$ and there is no polynomial kernel for $1C_\ell$D for any $\ell \geq 4$, unless NP $\subseteq$ coNP$_{/\text{poly}}$ [17]. We prove in the next theorem that there is a polynomial kernel for cP_ℓD and cC_ℓD if the input graph is non-cascading.

Theorem 4.7. *Let $\mathcal{F}$ be an d-edge-bounded set of graphs. If the input graph G is non-cascading and $\mathcal{F}$ is polynomial-enumerable on G, then $\mathcal{F}$ Deletion admits a polynomial-size kernel that can be computed in polynomial time when parameterized by k.*

Proof. We prove this by giving a parameterized reduction from $\mathcal{F}$ Deletion parameterized by k to d-HS parameterized by k.

Let (G, k) be an instance of $\mathcal{F}$ Deletion where G is a non-cascading graph. We will show how to construct an equivalent instance $(\mathcal{C}, k)$ of d-HS. Note that the parameter k stays the same. First, we set $\mathcal{X} := E(G)$. Then, we add a hyper-edge $\mathcal{C}_j := E_G(V')$ to $\mathcal{C}$ for each vertex set $V' \subseteq V(G)$ where $G[V'] \cong H$ for some $H \in \mathcal{F}$. Since $\mathcal{F}$ is polynomial-enumerable on G, this can be done in polynomial time, and since $\mathcal{F}$ is d-edge-bounded, the size of the hyper-edges in $\mathcal{C}$ is also bounded in d. Furthermore, we know that G is non-cascading. Hence, we conclude that for any edge-deletion set $S \subseteq E(G)$, the graph $G - S$ is $\mathcal{F}$-free if and only if S is a hitting set for $\mathcal{C}$. Thus, (G, k) is a yes-instance of $\mathcal{F}$ Deletion if and only if $(\mathcal{C}, k)$ of d-HS.

Since d-HS is NP-complete [28] and $\mathcal{F}$ Deletion is NP-hard, we know that there is a polynomial time reduction d-HS $\leq_p$ $\mathcal{F}$ Deletion. Furthermore, d-HS parameterized by k admits a kernel comprising at most $\mathcal{O}(k^{d-1})$ vertices that can be computed in polynomial time [32]. And since each edge in $E(G)$ corresponds to a vertex in $\mathcal{X}$, we conclude that for non-cascading input graphs, $\mathcal{F}$ Deletion admits a polynomial size kernel that can be computed in polynomial time when parameterized by k. □

From Lemma 4.1 and Theorem 4.7 we can conclude the following.

Corollary 4.8. *If the input graph G is non-cascading, cP_ℓD and cC_ℓD admit polynomial size kernels that can be computed in polynomial time when parameterized by k.*

We will finish this chapter by giving a linear size problem kernel for cP_ℓD and cC_ℓD parameterized by the dual parameter $\delta := m - k$. This kernel is a generalization of a kernel for $2P_3$D [14].

Theorem 4.9. *If G is a c-colored graph and $c \geq 2$, then cP_ℓD and cC_ℓD admit a kernel with $\mathcal{O}(2 \cdot \delta)$ edges which can be computed in $\mathcal{O}(m)$ time when parameterized by δ.*

Proof. We will show that instances with at least $2 \cdot \delta$ edges are trivial yes-instances. Since the proof cC_ℓD works analogously, we will only show the proof for cP_ℓD. Let (G, k) be an instance of cP_ℓD with $E(G) \geq 2 \cdot \delta$. Since each edge $e \in E(G)$ is either blue or not blue, we conclude that $|E_b(G)| \geq \delta$ or $|E(G) \setminus E_b(G)| \geq \delta$. Without loss of generality, we assume that $|E(G) \setminus E_b(G)| \geq \delta$. Since G is c-colored, each c-colored P_ℓ in G has to include a blue edge. Hence, $G - E_b(G)$ is c-colored P_ℓ-free. And since $|E_b(G)| = m - |E(G) \setminus E_b(G)| \leq m - \delta = k$, we conclude that $E_b(G)$ is a solution for (G, k). Thus, (G, k) is a trivial yes instance if $E(G) \geq 2 \cdot \delta$. □

5 Conclusion

In Section 3, we gave NP-hardness proves for cP_ℓD and cC_ℓD and thereby proved that these problems can not be solved efficiently for any $\ell \geq 3$ and any $c \in [\ell - 1]$ or respectively any $c \in [\ell]$, unless P = NP. An interesting question for further research is, if there is an interesting graph property Π such that cP_ℓD or cC_ℓD is non-trivially polynomial time solvable on graphs satisfying Π. Since we consider edge-colored graphs, these properties can either be defined on the entire structure of the input graph or on the graphs that are induced by different colors. For example, one could consider input graphs that does not contain an induce claw, that is, an graph with exactly three edges that have exactly one common vertex, since the graphs we construct in the proofs all contain induced claws. Another idea could be considering c-colored graphs where each edge color induces a graph with maximum degree two. Furthermore, an investigation about the structure and usefulness of c-colored P_ℓ-free or respectively c-colored C_ℓ-free graphs would be interesting. One could, for example, try to find graph modification problems that are NP-hard on general c-colored graphs but become polynomial time solvable when the input graph is restricted to c-colored P_ℓ-free or respectively c-colored C_ℓ-free graphs. Since the problem whether or not one can destroy all cycles with at most k edge deletions is known to be polynomial solvable on uncolored graphs, another interesting problem for further research would be whether or not it is possible to destroy all c-colored cycles with at most k edge deletions. And while we showed that cP_ℓD is NP-hard on non-cascading graphs for each $\ell \geq 4$ and each $c \in [\ell - 2]$, the question whether there is a polynomial time algorithm for $(\ell - 1)P_\ell$ D on non-cascading graphs for any $\ell \geq 4$ remains open. Furthermore, we showed that cP_ℓD is NP-hard on graphs with a constant maximum degree, while our proof that cC_ℓD is NP-hard does not give a graph of constant maximum degree. Hence, it remains an open question whether or not cC_ℓD is NP-hard on graphs with constant maximum degree. Finally, it would be interesting to analyze whether or not the corresponding completion-problems and editing-problems are NP-hard.

In Section 4, we proved that cP_ℓD and cC_ℓD are fixed parameter tractable, while CPD is W[2]-hard when parameterized by k even if $c = 3$. This suggests that there is no algorithm for cP_ℓD where the exponential factor is not dependent on ℓ. It would be interesting to analyze whether CPD is fixed-parameter tractable when parameterized by (ℓ, k). We furthermore showed that cP_ℓD and cC_ℓD admit polynomial size kernels if the input graph is non-cascading. It would be interesting, whether or not one can achieve similar results without the restriction to non-cascading input graphs. We finished the section by giving a kernel for cP_ℓD and cC_ℓD when restricted to c-colored input graphs with $c \geq 2$ comprising at most $\mathcal{O}(2 \cdot \delta)$ edges when parameterized by the dual parameter δ. Again, it would be interesting to analyze in what extend similar results can be achieved when the restrictions on the input graph are relaxed.

Furthermore, there are some interesting questions that we could not consider in this thesis. It would be interesting to analyze to what extend the results for uncolored graphs [17, 18] can be generalized to edge-colored graphs. And finally, while it makes no difference for the classical computational complexity whether we define the problems on edge-colored graphs or on edge-colored multigraphs, it would be interesting to analyze parameterized versions on multigraphs, since parameterized complexity analysis is more fine-grained and edge-colored multigraphs are a more general model for multilayer networks.

6 Bibliography

[1] Sebastian Böcker and Jan Baumbach. Cluster editing. In *The Nature of Computation. Logic, Algorithms, Applications - 9th Conference on Computability in Europe, CiE 2013, Milan, Italy, July 1-5, 2013. Proceedings*, volume 7921 of *Lecture Notes in Computer Science*, pages 33–44. Springer, 2013.

[2] Ulrik Brandes, Michael Hamann, Ben Strasser, and Dorothea Wagner. Fast quasi-threshold editing. In *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 251–262. Springer, 2015.

[3] Donald J. Rose. A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. In *Graph theory and computing*, pages 183–217. Elsevier, 1972.

[4] Hans L. Bodlaender and Babette van Antwerpen-de Fluiter. On intervalizing k-colored graphs for DNA physical mapping. *Discrete Applied Mathematics*, 71(1-3):55–77, 1996.

[5] Mihalis Yannakakis. Edge-deletion problems. *SIAM Journal on Computing*, 10(2):297–309, 1981.

[6] Pablo Burzyn, Flavia Bonomo, and Guillermo Durán. NP-completeness results for edge modification problems. *Discrete Applied Mathematics*, 154(13):1824–1844, 2006.

[7] Michele Berlingerio, Michele Coscia, Fosca Giannotti, Anna Monreale, and Dino Pedreschi. Foundations of multidimensional network analysis. In *International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2011, Kaohsiung, Taiwan, 25-27 July 2011*, pages 485–489. IEEE Computer Society, 2011.

[8] Stefano Boccaletti, Ginestra Bianconi, Regino Criado, Charo I Del Genio, Jesús Gómez-Gardenes, Miguel Romance, Irene Sendina-Nadal, Zhen Wang, and Massimiliano Zanin. The structure and dynamics of multilayer networks. *Physics Reports*, 544(1):1–122, 2014.

[9] John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980.

[10] Robert Bredereck, Christian Komusiewicz, Stefan Kratsch, Hendrik Molter, Rolf Niedermeier, and Manuel Sorge. Assessing the computational complexity of multi-layer subgraph detection. In *Algorithms and Complexity - 10th International Conference, CIAC 2017, Athens, Greece, May 24-26, 2017, Proceedings*, volume 10236 of *Lecture Notes in Computer Science*, pages 128–139, 2017.

[11] Mihalis Yannakakis. Node- and edge-deletion NP-complete problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 253–264. ACM, 1978.

[12] Ehab S. El-Mallah and Charles J. Colbourn. The complexity of some edge deletion problems. *IEEE Transactions on Circuits and Systems*, 35(3):354–362, 1988.

[13] N. R. Aravind, R. B. Sandeep, and Naveen Sivadasan. Dichotomy results on the hardness of H-free edge modification problems. *SIAM Journal on Discrete Mathematics*, 31(1):542–561, 2017.

[14] Niels Grüttemeier, Christian Komusiewicz, Jannik Schestag, and Frank Sommer. Destroying bicolored P_3s by deleting few edges. In *Computing with Foresight and Industry - 15th Conference on Computability in Europe, CiE 2019, Durham, UK, July 15-19, 2019, Proceedings*, volume 11558 of *Lecture Notes in Computer Science*, pages 193–204. Springer, 2019.

[15] Leizhen Cai and On Yin Leung. Alternating path and coloured clustering. *Computing Research Repository (CoRR)*, abs/1807.10531, 2018.

[16] Leizhen Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Information Processing Letters*, 58(4):171–176, 1996.

[17] Leizhen Cai and Yufei Cai. Incompressibility of H-free edge modification problems. *Algorithmica*, 71(3):731–757, 2015.

[18] N. R. Aravind, R. B. Sandeep, and Naveen Sivadasan. On polynomial kernelization of H-free edge deletion. *Algorithmica*, 79(3):654–666, 2017.

[19] Douglas Brent West et al. *Introduction to graph theory*, volume 2. Prentice hall Upper Saddle River, 2001.

[20] Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

[21] Béla Bollobás. *Modern Graph Theory*, volume 184 of *Graduate Texts in Mathematics*. Springer, 2002.

[22] H.N. de Ridder. Information system on graph classes and their inclusions, 2001. https://graphclasses.org/index.html.

[23] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[24] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing, May 3-5, 1971, Shaker Heights, Ohio, USA*, pages 151–158. ACM, 1971.

[25] Leonid A. Levin. Universal sequential search problems. *Problemy peredachi informatsii*, 9(3):115–116, 1973.

[26] Piotr Berman, Marek Karpinski, and Alex D. Scott. Approximation hardness of short symmetric instances of MAX-3SAT. *Electronic Colloquium on Computational Complexity (ECCC)*, (049), 2003.

[27] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

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

[29] Rolf Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[30] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.

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

[32] Faisal N. Abu-Khzam. A kernelization algorithm for d-hitting set. *Journal of Computer and System Sciences*, 76(7):524–531, 2010.

Declaration of Academic Integrity

Hereby, I declare that I have composed the presented thesis independently on my own and without any other resources than the ones indicated. All thoughts taken directly or indirectly from external sources are properly denoted as such. This thesis has neither been previously submitted to another authority nor has it been published yet.

Marburg, 06.07.2020

Jakob Eckstein