Some Results On Graph Contraction Problems

By

Prafullkumar Prabhakar Tale

MATH10201305002

The Institute of Mathematical Sciences, Chennai

A thesis submitted to the

Board of Studies in Mathematical Sciences

In partial fulfillment of requirements

for the Degree of

DOCTOR OF PHILOSOPHY

of

HOMI BHABHA NATIONAL INSTITUTE

May, 2020

Homi Bhabha National Institute

Recommendations of the Viva Voce Committee

As members of the Viva Voce Committee, we certify that we have read the dissertation prepared by Prafullkumar Prabhakar Tale entitled "Some Results On Graph Contraction Problems" and recommend that it may be accepted as fulfilling the thesis requirement for the award of Degree of Doctor of Philosophy.

_____ Date: May 25, 2020

Chairman - Prof. Venkatesh Raman

_____ Date: May 25, 2020

Guide/Convenor - Prof. Saket Saurabh

_____ Date: May 25, 2020

Examiner - Prof. N R Aravind

_____ Date: May 25, 2020

Member 1 - Prof. V. Arvind

_____ Date: May 25, 2020

Member 2 - Prof. Geevarghese Philip

_____ Date: May 25, 2020

Member 3 - Prof. Vikram Sharma

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to HBNI.

I hereby certify that I have read this thesis prepared under my direction and recommend that it may be accepted as fulfilling the thesis requirement.

Date: May 25, 2020

Place: Chennai Prof. Saket Saurabh (Guide)

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at Homi Bhabha National Institute (HBNI) and is deposited in the Library to be made available to borrowers under rules of the HBNI.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgement of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the Competent Authority of HBNI when in his or her judgement the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

Prafullkumar Prabhakar Tale

DECLARATION

I hereby declare that the investigation presented in the thesis has been carried out by me. The work is original and has not been submitted earlier as a whole or in part for a degree / diploma at this or any other Institution / University.

Prafullkumar Prabhakar Tale

List of Publications arising from the thesis

Journal

1. (Published) **On the Parameterized Complexity of Contraction to Generalization of Trees** : A. Agrawal, S. Saurabh, and P. Tale. Published in *Theory of Computing Systems, Nov, 2018*. Volume 63. p587-614.

2. (Under Review) **Lossy Kernels for Graph Contraction Problems.** : R. Krithika, P. Misra, A. Rai, and P. Tale. Under review in *SIAM Journal on Discrete Mathematics (SIDMA)*.

Conferences

1. **Path Contraction Faster than** 2^n : A. Agrawal, F. Fomin, D. Lokshtanov, S. Saurabh and P. Tale. Accepted in 46^{th} *International Colloquium on Automata, Languages, and Programming, ICALP 2019*.

2. **An FPT Algorithm for Contraction to Cactus** : R. Krithika, P. Misra, and P. Tale. Appered in Proc. of *Annual International Computing and Combinatorics Conference, COCOON 2018*.

3. **Paths to Trees and Cacti.** : A. Agrawal, L. Kanesh, S. Saurabh, and P. Tale. Appered in Proc. of *Algorithms and Complexity* - 10^{th} *International Conference, CIAC 2017*.

4. **On the Parameterized Complexity of Contraction to Generalization of Trees.** : A. Agrawal, S. Saurabh, and P. Tale. Appered in Proc. of *International Symposium on Parameterized and Exact Computation, IPEC 2017*.

5. **Lossy Kernels for Graph Contraction Problems.** : R. Krithika, P. Misra, A. Rai, and P. Tale. Appered in Proc. of *Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016*.

Prafullkumar Prabhakar Tale

Dedicated to

Vara Prasad,

my *brother-in-forest* who dreamt of doing research in Physics while climbing in remote Himalayas. He never rested in peace, I doubt his soul will either.

Acknowledgements

In past few years, many people believed in me more than I believed myself. The foremost among them is my adviser, Saket Saurabh. I am grateful to him for all the time and energy he has spent over last five years educating me. I thank him for going out of his way to nourish me. His care goes beyond professional boundaries. He worked hard to create platforms where I can flurish. I will forever appreciate all the discussions, academics and otherwise, we had. He showed an extraordinary level of patience towards my academic flaws. His encouragements helped me to reach this stage. If I can cultivate one tenth of his work-ethics and ability for hard work, I will feel confident about my future.

I am in debt of G Philip and V Raman for mentoring me on various projects. I am grateful to Philip for all the support he has provided in last couple of years. I would also like to thank Abhishek, Akanksha, Ashutosh, Daniel, Diptapriyo, Fahad, Fedor, Krithika, Lawqueen, Meesum, Pallavi, Pranabendu, Roohani, Sanjukta, Spoorthy, and Sudeshna for all the technical discussions we had. Those thoughtful discussions cemented my understanding of various topics and opened new directions for my work.

I also like to thank other faculties at IMSc - V Arvind, Kamal Lodha, Meena Mahajan, R Ramanujam, Vikram Sharma, and C R Subramanian. My sincere thanks to Administrative, Library and Technical Staff for their help.

I was fortunate to find supporting friends during my graduate studies. I will forever cherish moments shared with Aashita, Akanksha, Anantha, Anuj, Deeksha, Lawqueen, Krithika, Rajesh, Roohani, Sanjukta, Saumia, and Swaroop. My special thanks to Roohani for being there as a support system when I needed one. I would also like to thank all my other friends from school, college, and Chennai Trekking Club.

Words fail me to describe my gratitude towards my parents and other family members. They had always been blind supporters of my decisions. Their faith in me is the immense source of inspiration to excel in the work I do.

Contents

Summary

For a family of graphs $\mathscr{F}$, the $\mathscr{F}$-EDITING problem takes as an input a graph G and an integer k, and the objective is to decide if at most k edit operations on G can result in a graph that belongs to $\mathscr{F}$. Various graph editing problems have been considered in the literature. These graph editing problems generalize many NP-Hard problems. Most of the studies regarding $\mathscr{F}$-EDITING have been restricted to combination of vertex deletion, edge deletion or edge addition. Only recently, *edge contraction* as an edit operation has started to gain attention.

The *contraction* of edge uv in graph G deletes vertices u and v from G, and replaces them by a new vertex, which is made adjacent to vertices that were adjacent to either u or v. In this thesis, we explore $\mathscr{F}$-CONTRACTION for various graph classes from the viewpoints of *parameterized complexity, lossy kernelization* and *exact algorithms*. We extend the known boundaries about graph contraction problems in several ways. We consider $\mathscr{F}$-CONTRACTION problems which do not have a polynomial kernels when parameterized by solution size. We compliment this negative result in two ways. Firstly, we present a polynomial kernel when parameterized by solution size and an additional parameter. In other words, we identify new graph classes for which there is a polynomial kernel. We also prove that these kernels are optimal under certain complexity conjecture. Secondly, we present a lossy kernel of polynomial size for all these problems. We present two FPT algorithms to append the list of graph classes $\mathscr{F}$ for which $\mathscr{F}$-CONTRACTION parameterized by solution size is FPT. We end this thesis with a non-trivial exact algorithm

to determine what is the largest size of graph in a specific $\mathscr{F}$ to which an input graph can be contracted. To best of our knowledge, this is first such kind of algorithm in case of graph contraction problems.

List of Figures

Chapter 1

Introduction

1.1 Preamble

Graph editing problems are one of the central problems in graph theory that have received lot of attention in theoretical computer science. Some of the important graph editing operations are vertex deletion, edge deletion, edge addition and edge contraction. For a family of graphs $\mathscr{F}$, the $\mathscr{F}$-EDITING problem takes as an input a graph G and an integer k, and the objective is to decide if at most k edit operations on G can result in a graph that belongs to $\mathscr{F}$. The *contraction* of edge uv in graph G deletes vertices u and v from G, and replaces them by a new vertex, which is made adjacent to vertices that were adjacent to either u or v. In this thesis, we explore $\mathscr{F}$-EDITING problem when edit operation is restricted to edge contraction for various graph classes from the viewpoints of *parameterized complexity, lossy kernelization* and *exact algorithms*.

Various graph editing problems have been considered in the literature with restriction on allowed edit operations and it generalizes many NP-Hard problems. For instance, the $\mathscr{F}$-EDITING problems encompasses problems such as VERTEX COVER [22], FEEDBACK VERTEX SET [19, 66], PLANAR $\mathscr{F}$-DELETION [41, 65], INTERVAL EDITING [17, 21, 18, 11], CHORDAL EDITING [42, 78, 20], ODD CYCLE TRANSVERSAL [86], CLUSTER

EDITING [40], TREE CONTRACTION [55], SPLIT EDITING [45], PERFECT GRAPHS EDITING [54], TRIVIALLY PERFECT GRAPH EDITING [34, 36], PROPER INTERVAL COMPLETION [10], PLANAR EDITING [59], THRESHOLD EDITING [35], etc. Most of the studies regarding $\mathscr{F}$-EDITING have been restricted to combination of vertex deletion, edge deletion or edge addition. Only recently, edge contraction as an edit operation has started to gain attention.

When we restrict the operations to only vertex/edge deletion then the corresponding problem is called $\mathscr{F}$-VERTEX/EDGE DELETION problem. On the other hand if we only allow edge contraction then the corresponding problem is called $\mathscr{F}$-CONTRACTION. Edge contraction problems generally turn out to be more difficult compared to their vertex/edge deletion/addition counterparts. For instance, the problem of determining whether one can delete at most k edges from a connected input graph to obtain a tree, also known as FEEDBACK EDGE SET, is polynomial time solvable. Whereas, the problem of determining whether one can contract at most k edges from a connected input graph to obtain a tree, also known as TREE CONTRACTION, is NP-Hard [6]. In fact, even determining whether a given graph can be contracted to a path on four vertices turns out to be NP-Hard [14]. Early papers showed that $\mathscr{F}$-CONTRACTION is NP-Hard even for several simple and well structured graph classes such as paths, stars, trees, etc. [6, 14, 93, 94].

In parameterized complexity, each instance of problem Π is accompanied by a parameter, usually denoted by k. A central notion in this field is *fixed parameter tractable* (FPT) problems. A parameterized problem Π is said to be FPT if for a given instance (I,k), one can decide whether or not it is a YES instance of Π in time $f(k)|I|^{\mathcal{O}(1)}$ where f is some computable function of k. Every parameterized problem need not be fixed-parameter tractable for given parameter. For now, consider following subset relation among classes of problems: $\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \cdots$. Each class is believed to be properly contained in its superclass.

In the framework of FPT algorithms, edge contraction problems exhibit properties that are

quite different than those of problems where we only delete or add vertices and edges. A well-known result by Cai [15] states that in case $\mathscr{F}$ is a hereditary family of graphs with a finite set of forbidden induced subgraphs, then the graph editing problem defined by $\mathscr{F}$ and the edit operations restricted to vertex deletion, edge deletion and edge addition admits an FPT algorithm. Results of such favor does not exist in the case of edge contraction. Consider an example of *split graphs*. A graph is called split graph if it can be partitioned into two sets, one of which induces a clique and another one is an independent set. A graph is split graph if and only if it does not contain an induced graph in $\{C_4, C_5, 2K_2\}$, leading to a finite forbidden characterization of this graph class. Note that SPLIT VERTEX/EDGE DELETION/ADDITION admits an FPT algorithm running in time $5^k \cdot n^{\mathcal{O}(1)}$ (See [46] for improved algorithms) but SPLIT CONTRACTION is W[1]-Hard [2].

Other important notion in parameterized complexity is *kernelization*, which captures the efficiency of data reduction techniques. A parameterized problem Π admits a kernel of size $g(k)$ (or $g(k)$-kernel) if there is a polynomial time algorithm (called *kernelization algorithm*) which takes as an input (I,k), and returns an instance (I',k') of Π such that: (i) (I,k) is a YES instance if and only if (I',k') is a YES instance; and (ii) $|I'|+k' \leq g(k)$, where $g(\cdot)$ is a computable function whose value depends only on k. Depending on whether the function $g(\cdot)$ is *linear, polynomial* or *exponential*, the problem is said to admit a *linear, polynomial* or *exponential kernel*, respectively. It is easy to see that any problem that admits a kernel is also FPT. The converse also turned out to be true. Any problem that is fixed-parameter tractable admits an exponential kernel [25]. This makes linear and polynomial kernels more interesting from the kernelization perspective. Researchers have developmented the framework for ruling out existence of certain types of kernel under some complexity theoretic assumptions [12, 28, 43, 68]. With this results, a new direction of research in the recent years have been proving optimality of the kernel sizes and ruling out existence of kernels of some types for a parameterized problem at hand.

Not surprisingly, edge contraction problems exhibit different behavior as compare to their

counter part when it comes to admitting a kernel. Consider a case when target graph class is set of acyclic graphs. If the edit operation is deletion of vertex then the problem is known as FEEDBACK VERTEX SET which admits a kernel with $\mathcal{O}(k^2)$ vertices [58, 91]. On the other hand, TREE CONTRACTION is known not to have a polynomial kernel under a widely believed complexity theoretical conjecture [55].

The notion of polynomial kernels turns out to be a bit stringent, and it has been discovered that many problems do not admit a polynomial kernel under well-known complexity theoretic conjectures. On the other hand this notion turns out to be too lax as the instances (I,k) and (I',k') are not as *tightly-coupled* as one would like them to be. For example, in general, it may not be possible to translate an approximate solution to the instance (I',k'), into an approximate solution to the original instance (I,k). Given anything but an optimal solution (or a solution of size k') to (I',k'), it is impossible to conclude anything about the original instance (I,k). These issues, among others, have led to the development of a framework for *approximation preserving kernelization* or *lossy kernelization*. Informally, an α-approximate kernelization algorithm ensures that given any c-approximate solution to the kernel (I',k'), it can be converted into a $(c \cdot \alpha)$-approximate solution to the original instance (I,k) in polynomial time. This notion was formally introduced by Lokshtanov et al. [75].

Almost all combinatorial problems are solvable in finite time by examining all of its candidate solutions i.e. by brute-force search method. For NP-Hard problems, the number of candidate solutions is exponential in the size of input. One of the most important question in theoretical computer sciences is to find whether enumeration of solutions is the only approach to solve NP-Hard problems *in general*. While this long last problem remains difficult to tackle, there has been interest in developing *exact exponential algorithms* which are specific to a problem at hands. First question while designing exact exponential algorithms for a particular problem is: *can be avoid brute-force search?*

1.2 Known Results about Graph Contraction

The complexity of edge contraction problems has been studied in the literature, but it has not received as much attention as other graph editing problems. In the limited body of work, $\mathscr{F}$-CONTRACTION has been analyzed in various dimension. It has been studied for various graph classes. There are attempts to understand the complexity of $\mathscr{F}$-CONTRACTION problems depending on finite forbidden characterization of $\mathscr{F}$. The problem turned out to be hard even when $\mathscr{F}$ is finite or even if it contains one graph. Another line of research is to study $\mathscr{F}$-CONTRACTION with restrictions on input graph. In recent time, there is new line of research where $\mathscr{F}$ is defined in *parameterized way* with respect to input graph. In some graph contraction problems, the task is to determine size of largest graph in $\mathscr{F}$ to which an input graph can be contracted.

Watanabe et al. [93, 94] showed that $\mathscr{F}$-CONTRACTION is NP-Hard if $\mathscr{F}$ is finitely characterizable by 3-connected graphs. Their result was generalized by Asano and Hirata [6] who showed that $\mathscr{F}$-CONTRACTION is NP-Hard whenever $\mathscr{F}$ is a graph class that fulfills the following three conditions. (i) $\mathscr{F}$ is be closed under contractions, which is to say, if a graph G is in $\mathscr{F}$ then any graph obtained from G by edge contractions is also in $\mathscr{F}$. (ii) $\mathscr{F}$ is not a trivial graph class. There are infinitely many graphs which are contained in $\mathscr{F}$ and there are infinitely many graphs which are not. (iii) A graph belongs to $\mathscr{F}$ if and only if each of its 2-connected components belong to $\mathscr{F}$. This result implies that $\mathscr{F}$-CONTRACTION is NP-Hard when $\mathscr{F}$ is family of planar graphs, outerplanar graphs, series-parallel graphs, forests, chordal graphs, or more generally, graphs with no cycles of length at least ℓ for some fixed integer $\ell \geq 3$. Martin and Paulusma showed that $\mathscr{F}$-CONTRACTION is NP-Hard when $\mathscr{F}$ is the class of bicliques $K_{p,q}$ with $p,q \geq 2$ [76].

In the realm of parameterized complexity, $\mathscr{F}$-CONTRACTION has been studied with parameter being the size of solution. A well-known result by Cai [15] states that in case $\mathscr{F}$ is a hereditary family of graphs with a finite set of forbidden induced subgraphs, the problem of modifying an input graph into a graph in $\mathscr{F}$ when the allowed edit operations

are vertex deletion, edge deletion and edge addition admits an FPT algorithm. Central idea in Cai's argument is: to destroy a structure which forbids the input graph from being in $\mathscr{F}$, one needs to include at least one vertex (or edge) from that structure into a solution. This is not necessarily true in the case of contractions. A forbidden structure may be destroyed by contracting edges which are not contained in the structure. Hence the classical *branching technique* does not work even for graph classes that have a finite forbidden structure characterization. There are concrete examples for the fact that results of similar flavor as that of Cai [15] do not hold when the edit operation is edge contraction. Lokshtanov et al. [73] and Cai and Guo [16] independently showed that if $\mathscr{F}$ is either the family of $P_{\ell+1}$-free graphs or the family of C_ℓ-free graphs for some $\ell \geq 4$, then $\mathscr{F}$-CONTRACTION is W[1]-Hard.

In rest of this section, we use n and m to denote the number of vertices and edges, respectively, in an input graph. The size of solution, i.e. the maximum number of edges we are allowed to contracted in input graph to obtain graph in target graph class, is denoted by k. Unless otherwise specified, in all the problems mentioned below, the parameter is the size of solution. Whenever we mention a problem does not have a polynomial kernel, it is under the assumption that NP $\not\subseteq$ coNP/poly.

To best of our knowledge, Hergerners et al. [55] were the first to explicitly study edge contraction problems in the realm of parameterized complexity. They presented a $4^k n^{\mathscr{O}(1)}$ algorithm for TREE CONTRACTION based on a variant of the color coding technique of Alon et al. [5] and using an algorithm for CONNECTED VERTEX COVER [24] as subroutine. They also presented an algorithm running in time $\mathscr{O}(2^{k+o(k)}+m)$ for PATH CONTRACTION. The authors presented a parameter preserving reduction from an instance of RED BLUE DOMINATING SET (defined later) problem to an instance of TREE CONTRACTION to rule out polynomial kernel. They presented a kernel with $5k+3$ vertices for PATH CONTRACTION. This kernel was later improved to $3k+4$ by Li et al. [72]. A subset of others (from [55]) proved that if the input graph is chordal then TREE CONTRACTION and

PATH CONTRACTION can be solved in time $\mathcal{O}(n+m)$ and $\mathcal{O}(nm)$, respectively [53].

Golovach et al. [50] proved that PLANAR CONTRACTION is FPT. Their algorithm starts by finding a set S of at most k vertices whose deletion transforms G into a planar graph [63, 80]. This is a recurring theme in designing an FPT algorithm for $\mathscr{F}$-CONTRACTION problems. We first solve $\mathscr{F}$-VERTEX DELETION problem to get structural insight of input graph and exploit it to obtain an FPT algorithm for contraction version of the problem. The authors showed that if the input graph has large *treewidth* then one can find an edge which can safely be contracted. This yields a smaller equivalent instance. They use the *irrelevant vertex technique* developed in the graph minors project of Robertson and Seymour [88, 87] to find such edge. After repeatedly contracting such irrelevant edges, which results in graph of bounded treewidth, authors used Courcelle's Theorem [23] to solve the remaining instance in linear time.

Heggernes et al. [56] proved that BIPARTITE CONTRACTION is FPT using, first of its kind, a combination of the irrelevant vertex technique and *important sets or separators*. Important sets and the closely related notion of important separators were introduced in [77] to prove the fixed-parameter tractability of multiway cut problems. The algorithm starts by finding treewidth of input graph. If the treewidth is small then it solves instance using Courcelle's Theorem. If the treewidth is large, then it identify an irrelevant edge that can be deleted without affecting the outcome. The algorithm crucially deviates from most of the work in which finding irrelevant edge is crucial. While most works has rely on large minor models as obstructions to small treewidth, this algorithm uses the fact that any graph of high treewidth contains a large p-connected set X [30]. A vertex set X is p-connected if, for any two subsets X_1 and X_2 of X with $|X_1| = |X_2| \leq p$, there are $|X_1|$ vertex-disjoint paths with one endpoint in X_1 and the other in X_2.

Marx et al. [79] observed that a simple corollary of their result immediately proved that BIPARTITE CONTRACTION is almost linear time FPT. Guillemot and Marx [51] presented a new FPT algorithm for BIPARTITE CONTRACTION, which is both conceptually

simpler and faster then the one mentioned previous paragraph. Their algorithm reduces an instance of BIPARTITE CONTRACTION to several instances of an auxiliary cut problem. These instance are then solved using the notion of important separators together with the randomized coloring technique [5]. They presented a randomized FPT algorithm with running time $2^{\mathcal{O}(k^2)}nm$ and a deterministic algorithm with running time $2^{\mathcal{O}(k^2)}n^{\mathcal{O}(1)}$.

Cai and Gua [16] and Lokshtanov et al. [74], independently, initiated the study to determine the parameterized complexity of $\mathscr{F}$-CONTRACTION in terms of forbidden induced subgraphs characterization of $\mathscr{F}$. We say $\mathscr{F}$ is F-FREE if F is one of the forbidden induced subgraphs of $\mathscr{F}$. In other words, a graph G is contained in $\mathscr{F}$ if and only if G does not contain F as an induced subgraph. Let K_t denote a complete graph on t vertices. If $\mathscr{F}$ is K_t-FREE then $\mathscr{F}$-CONTRACTION is FPT as the only way to destroy a copy of K_t is to contract some edges in the copy. This implies an FPT algorithm by the branching technique. This need not be the case for any other forbidden induced subgraphs. They proved that if $\mathscr{F}$ is P_ℓ-FREE for $\ell \leq 4$ then $\mathscr{F}$-CONTRACTION is FPT but admits no polynomial kernel. They complimented this result by showing that P_ℓ-FREE CONTRACTION is W[2]-Hard for every fixed path P_ℓ with $\ell \geq 5$. They also proved that C_3-FREE CONTRACTION is FPT but admits no polynomial kernel and C_ℓ-FREE CONTRACTION is W[2]-Hard for every fixed cycle C_ℓ with $\ell \geq 4$. Last result implies that CHORDAL CONTRACTION is W[2]-Hard. Cai and Gua gave a complete charaterization of F-FREE GRAPHS when F is 3-connected. They proved that, apart from being a 3-connected graph, if F is a complete graph then F-FREE CONTRACTION is FPT but admits no polynomial kernel otherwise F-FREE CONTRACTION is W[2]-Hard.

If input graph is connected then P_3-FREE CONTRACTION problem is same as that that of CLIQUE CONTRACTION. Cai and Gua presented an algorithm running in time $\mathcal{O}(2^{7k}k^{2k+5}+m)$ to solve this problem [16]. Their algorithm first finds a large seed clique in the input graph using an algorithm for VERTEX COVER [22], and then uses a branch-and-search algorithm to contract other edges into the clique.

Belmonte et al. [8] studied $\mathscr{F}$-CONTRACTION problem when $\mathscr{F}$ is the family of degree constrained graphs like bounded degree, regular graphs and degenerate graphs. For any integer $d \geq 0$, let $\mathscr{F}_{\leq d}$ denote the class of graphs that have maximum degree at most d; $\mathscr{F}_{=d}$ denote the class of d-regular graphs and $\mathscr{F}_{d\text{-deg}}$ denote the class of d-degenerate graphs. Belmonte et al. completely characterized the parameterized complexity for $\mathscr{F}$-CONTRACTION problems with respect to the parameters k, d, and $d+k$, where $\mathscr{F} \in \{\mathscr{F}_{\leq d}, \mathscr{F}_{=d}, \mathscr{F}_{d\text{-deg}}\}$. They proved that $\mathscr{F}_{\leq d}$-CONTRACTION and $\mathscr{F}_{=d}$-CONTRACTION can be solved in time $\mathscr{O}((d+k)^{2k}(n+m))$. When parameterized by only k, they showed that $\mathscr{F}_{=d}$-CONTRACTION is W[1]-Hard, while $\mathscr{F}_{\leq d}$-CONTRACTION is W[2]-Hard even when input is restricted to split graphs. In case of $\mathscr{F}_{d\text{-DEG}}$-CONTRACTION, they proved that this problem is not fixed-parameter tractable when parameterized by $d+k$. When $d=2$, authors showed that $\mathscr{F}_{\leq d}$-CONTRACTION and $\mathscr{F}_{=d}$-CONTRACTION admit $O(k)$ vertex kernels on connected graphs and hence quadratic vertex kernels on general graphs. In other words, they proved that the $\mathscr{F}$-CONTRACTION problem admits a linear vertex kernel when $\mathscr{F}$ is the class of cycles or when $\mathscr{F}$ is the class of paths and cycles. This complements the fact that PATH CONTRACTION admits a linear vertex kernel [55].

Let $\mathscr{F}_{\geq d}$ be the family of graphs whose minimum degree is at least d. Golovach et al. [49] proved that $\mathscr{F}_{\geq d}$-CONTRACTION is NP-complete even when $d=14$. They proved that this problem is FPT when parameterized by both k and d but it is W[1]-Hard when parameterized by k alone.

Agarwal et al. [2] studied SPLIT CONTRACTION under various parameters. They proved that SPLIT CONTRACTION is W[1]-Hard parameterized by the size of the solution. They also studied this problem when parameter is the size of a minimum vertex cover (ℓ) of the input graph. To the best of our knowledge, this is the only study regarding graph contraction problems when parameter is not a solution size. Gua and Cai's work [52] implied that there exists an algorithm running in time $2^{\mathscr{O}(\ell^2)} \cdot n^{\mathscr{O}(1)}$ to solve SPLIT CONTRACTION. Agarwal et al. proved that unless the Exponential Time Hypothesis (ETH) [57] fails, SPLIT

CONTRACTION can not have an algorithm running in time $2^{o(\ell^2)} \cdot n^{\mathcal{O}(1)}$. This is the first tight lower bound of this form for problems parameterized by the vertex cover number of the input graph.

$\mathscr{F}$-CONTRACTION problems has be been proved to be NP-Hard even for finite graph classes. When $\mathscr{F}$ contains only one graph, say F, we call $\mathscr{F}$-CONTRACTION as F-CONTRACTIBILITY. Brouwer and Veldman proved that P_4-CONTRACTIBLITY and C_4-CONTRACTIBLITY is NP-Hard [14]. They also proved that if F is a connected graph other than a star which does not contain a triangle then F-CONTRACTIBLITY is NP-Hard. Levin et al. [71] followed by showing that for every fixed graph F on at most 5 vertices, F-CONTRACTIBLITY can be solved in polynomial time if F has a dominating vertex, and it is NP-Hard otherwise. In addition, Hof et al. [92] presented an infinite family of graphs with a dominating vertex, the smallest having 69 vertices, such that for any graph F in this family F-CONTRACTIBLITY is NP-Hard.

F-CONTRACTIBLITY problems has been studied with restriction on input graphs. Kaminski et al. [61] showed that for every fixed graph F, there exists a polynomial-time algorithm for deciding whether a given planar graph can be contracted to F. Kaminski and Thilikos [62] improved this result by showing that given a graph F and a planar graph G, the problem of deciding whether G can be contracted to F is fixed-parameter tractable when parameterized by $|V(F)|$. Belmonte et al. [9] showed that for any fixed graph F, the F-CONTRACTIBLITY problem is polynomial solvable in the input graph is a split graph. Golovach et al. [48] proved that if F is a split graph or a tree then F-CONTRACTIBLITY is polynomial solvable if the input graph is chordal. Belmonte et al. [7] generalized this result by showing that F-CONTRACTIBLITY on chordal graphs is polynomial solvable for any fixed F.

In the results mentioned until now, graph class $\mathscr{F}$ is specified either by some property or by finite forbidden characterization or by explicitly stating it. Instead of specifying a target graph class we can specify a graph parameter that needs to be reduced by certain

threshold using edge contractions. For example, for a specified graph parameter π and integer q, given a graph G, and an integer k, one can ask whether G can be transformed into a graph G' by using at most k edge contractions such that $\pi(G') \leq \pi(G) - q$? Such problems are called BLOCKER PROBLEMS. In general, blocker problems can have other graph modification operations apart from edge contraction. Blocker problems when edit operation is edge contraction have been studied in the recent literature [31, 84]. Several problems mentioned so far can be thought of blocker problems for appropriate parameter π and threshold ℓ. We mention two new problems. In HADWIGER NUMBER problem, the input is graph G and integer ℓ and the question is to determine whether G can be contracted to K_ℓ, a clique on ℓ vertices. This problem is *parametric dual* of CLIQUE CONTRACTION. Consider diameter as our parameter, this problem can be thought of as: can we contract at most k $(= n - \ell)$ edges to reduce the diameter of input graph by q $(= \text{diam(G)} - 1)$. With this definition, it is easy to generalize HADWIGER NUMBER to s-CLUB CONTRACTION. In this problem, we determine whether one can contract at most k edges in input graph G to reduce its diameter by $\text{diam}(G) - s$.

We end this section with few known results regarding P_ℓ-CONTRACTIBILITY for a fixed ℓ. There has been interest in strengthening the result of Brouwer and Veldman which prove that P_4-CONTRACTIBILITY is NP-Hard. This problem was proved to be NP-Hard even for P_6-FREE graphs [85]. Heggernes et al. [53] showed that P_6-CONTRACTIBILITY is NP-Hard for bipartite graphs. This result was improved to $k = 5$ in [27]. Moreover, P_7-CONTRACTIBILITY is NP-Hard for line graphs [38]. On the positive side, if input is P_5-FREE graphs then we can decide the length of longest path it can be contracted in polynomial time [85]. In very recent paper, authors [64] prove that P_k-CONTRACTIBILITY, for some suitable value of k, is NP-Hard for bipartite graphs of large girth strengthening the result of [53]. Cygan et al. [26] gave an algorithm running in time $\mathcal{O}(1.933^n \cdot n^{\mathcal{O}(1)})$ to solved P_4-CONTRACTIBILITY. Telle and Villanger [90] presented an improved algorithm to solve the same problem in time $\mathcal{O}(1.7804^n \cdot n^{\mathcal{O}(1)})$.

1.3 Scope of this thesis

In this thesis, we extend the known boundaries about graph contraction problems in several ways. We consider $\mathscr{F}$-CONTRACTION problems which do not have a polynomial kernels when parameterized by solution size. We compliment this negative result in two ways. Firstly, we present a polynomial kernel when parameterized by solution size and an additional parameter. In other words, we identify new graph classes for which there is a polynomial kernel. We also prove that these kernels are optimal under certain complexity conjecture. Secondly, we present a lossy kernel of polynomial size for all these problems. We present two FPT algorithms to append the list of graph classes $\mathscr{F}$ for which $\mathscr{F}$-CONTRACTION parameterized by solution size is FPT. We end this thesis with a non-trivial exact algorithm to determine what is the largest size of graph in a specific $\mathscr{F}$ to which an input graph can be contracted. To best of our knowledge, this is first such kind of algorithm in case of graph contraction problems.

Starting point of this thesis is the result of Heggernes et al. [55] who studied $\mathscr{F}$-CONTRACTION when $\mathscr{F}$ is the family of paths and trees. They showed that PATH CONTRACTION admits a polynomial kernel but TREE CONTRACTION does not. The natural question here is to identify properties of paths that separate it from trees and allows PATH CONTRACTION to have a polynomial kernel. This question can be formulated in the following way.

– *What additional parameter we can associate with* TREE CONTRACTION *to make sure that it admits a polynomial kernel?*

One of the possible candidates is the number of leaves in resulting graphs. In Chapter 3, we prove that this is indeed is the case by designing a polynomial kernel TREE-CONTRACTION when parameters are solution size and number of leaves in resultant graph. We also prove that this kernel is optimal under certain complexity assumption.

From the point of view of lossy kernelization, another natural question regarding TREE CONTRACTION is:

– If we are allowed to have small loss in accuracy then does TREE CONTRACTION *admits a polynomial kernel?*

In the same chapter, we address this question by presenting a lossy kernel of polynomial size for this problem.

At this point, for TREE CONTRACTION, we know how to get a polynomial kernel with two parameters; a lossy kernel and an FPT algorithm (due to [55]). We want to understand how far can we generalize methods used to obtain these results. We consider following two characterization of trees.

– A tree is a connected graph in which every edge is a part of zero cycles.

– A tree is a connected graph which can be made acyclic by deleting zero edges.

A connected graph is called a *cactus* if every edge in the graph is part of at most one cycle. In Chapter 4, we study CACTUS CONTRACTION. We show that this problem does not admit a polynomial kernel when parameterized by solution size but does admit a lossy kernel of polynomial size. We define notion of *cactus with bounded leaves* and present a polynomial kernel for this problem when parameterized by solution size and number of leaves in resultant cactus. We prove that this kernel is optimal under certain complexity assumption. We also present an FPT algorithm for this problem running in time $c^k \cdot n^{\mathcal{O}(1)}$.

Let $\mathbb{T}_\ell$ is a set of connected graphs which can be made acyclic by deleting at most ℓ edges or, equivalently, any graph in $\mathbb{T}_\ell$ have a *feedback edge set* of size at most ℓ. In Chapter 5, we study $\mathbb{T}_\ell$-CONTRACTION problem. We prove that this problem does not have a polynomial kernel when parameterized by solution size for any fixed ℓ. In other words, solution size with ℓ as an additional parameter do not give us polynomial kernel. However, this additional parameter is crucial in getting a lossy kernel of polynomial size and an FPT algorithm for this problem. This FPT algorithm can be seen as generalization of the FPT algorithm for TREE CONTRACTION.

Heggernes et al. [55] showed that TREE CONTRACTION is FPT which implies an FPT

algorithm for $\mathscr{F}_3$-CONTRACTION where $\mathscr{F}_3$ is a collection of C_3-free graphs. Lokshtanov et al. [73] and Cai and Guo [16] independently showed that $\mathscr{F}_4$-CONTRACTION is not FPT where $\mathscr{F}_4$ is a collection of C_4-free graphs. As the number of leaves in resultant graph is crucial to understand the gap between existence of polynomial kernel for PATH CONTRACTION and non-existence for TREE CONTRACTION; feedback edge set of resultant graph is crucial to understand the gap between existence of FPT algorithm for $\mathscr{F}_3$-CONTRACTION and non-existence for $\mathscr{F}_4$-CONTRACTION.

In Chapter 6, we study OUT-TREE CONTRACTION. A digraph is called an *out-tree* if its underlying undirected graph is a tree and every vertex in digraph has at most one in-neighbor. We address this problem to illustrate the fact that with some modification, techniques developed to obtain lossy kernelization for undirected graph contraction problems can be applied to directed graph contractions. We show that this problem does not admit a polynomial when parameterized by solution size. We are able to get similar results as in case of TREE CONTRACTION with slightly bigger size of kernel. We present an optimal polynomial kernel for OUT-TREE CONTRACTION when parameterized by solution size and number of leaves in resultant out-tree. We also describe a lossy kernel of polynomial size for this problem.

Effectiveness of lossy kernelization is not limited to a case when target graph class is close to trees. In Chapter 7, we present a lossy kernel for CLIQUE CONTRACTION. Recall that CLIQUE CONTRACTION parameterized by solution size does not admit a polynomial kernel under certain complexity assumption [16, 73]. We generalize this problem to s-CLUB CONTRACTION in which the objective to obtain a graph of diameter at most s. We prove that even when $s = 2$, there is no lossy kernel of polynomial size for this problem unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

In Chapter 8, we present an exact algorithm for PATH CONTRACTION. Any connected graph can be contracted to a path on two vertices. In this chapter, we address the question of determining the highest integer ℓ such that an input graph can be contracted to a path on

ℓ vertices. We observe that there is a simple brute force algorithm to find such integer. Our main contribution is the fact that this brute force search can be avoided.

Chapter 2

Preliminaries

In this chapter, we define notations which we use in rest of the thesis. We start with graph theoretical notations with separate section for graph contraction operations and related observations. We then present definitions and main results from Parameterized Complexity theory. We devote Section 2.4 for Lossy Kernelization.

We denote the set of natural numbers by $\mathbb{N}$ (including 0). For $k \in \mathbb{N}$, by $[k]$ we denote the set $\{1,2,\ldots,k\}$. Let X,Y be two sets. For a function $\varphi : X \to Y$ and $y \in Y$, by $\varphi^{-1}(y)$ we denote the set $\{x \in X \mid \varphi(x) = y\}$.

2.1 Graph Theory

In this thesis, we consider simple graphs with finite number of vertices. We use standard notation from graph theory [29]. For an undirected graph G, sets $V(G)$ and $E(G)$ denote the set of vertices and edges respectively. Two vertices u,v in $V(G)$ are said to be *adjacent* if there is an edge uv in $E(G)$. The neighborhood of a vertex v, denoted by $N_G(v)$, is the set of vertices adjacent to v and its degree $d_G(v)$ is $|N_G(v)|$. The subscript in the notation for neighborhood and degree is omitted if the graph under consideration is clear. For a set

of edges F, set $V(F)$ denotes the collection of endpoints of edges in F. For a subset S of $V(G)$, we denote the graph obtained by deleting S from G by $G-S$ and the subgraph of G induced on set S by $G[S]$. For two subsets S_1, S_2 of $V(G)$, we say S_1, S_2 are adjacent if there exists an edge with one end point in S_1 and other in S_2.

Two non-adjacent vertices u and v are called *false twins* of each other if $N(u) = N(v)$. A *path* $P = (v_1, \ldots, v_l)$ is a sequence of distinct vertices where every consecutive pair of vertices is adjacent. The vertices of P is the set $\{v_1, \ldots, v_l\}$ and is denoted by $V(P)$. A path P in a graph G is called a *simple path* of G, if every internal vertex of P has degree exactly equal to two in G. For a path P in G, let $N(P)$ denote the neighborhood of P, i.e. the set of vertices in $V(G) \setminus V(P)$ that are adjacent to a vertex in P. The endpoints of the path P are the only vertices with a neighbor in $G \setminus P$. A *cycle* is a sequence $(v_1, \ldots, v_l, v_1)$ of vertices such that $(v_1, \ldots, v_l)$ is a path and $v_l v_1$ is an edge.

A graph is called *connected* if there is a path between every pair of distinct vertices. It is called *disconnected* otherwise. A *component* of a graph is a maximal connected subgraph. A *cut-vertex* in G is a vertex v such that the number of components in $G - \{v\}$ is strictly more than the number of connected components in G. A graph that has no cut-vertex is called a *2-connected* graph. An edge uv of a graph G is called a *cut-edge* if the number of connected components in $G - \{uv\}$ is more than the number of connected components in G. We note that the number of connected components after removal of an edge can increase by at most one.

A vertex of degree one is called as *pendant vertex*. A graph is called a *forest* or an *acyclic graph* if it does not contain any cycle. A *tree* is a connected acyclic graph. A pendant vertex in a tree is called *leaf*. The vertices in a tree which are not leaves are *internal* vertices. A *star* is a tree in which there is a path of length at most two between any two vertices. A vertex that is adjacent to every other vertex in a star is called *center*. A connected graph is called a *cactus* if every edge is a part of at most one cycle. We use following result to bound the summation of degrees of vertices with degree 3 or more in a tree. Following

proposition also implies that in a tree, the number of vertices with degree at least 3 is upper bounded by number of vertices with degree 1.

Proposition 2.1.1 (Lemma 3 [67]). *For a tree T on at least two vertices, if V_1, V_2, V_3 are the set of vertices of degree 1, degree 2 and at least 3 respectively, then $\sum_{v \in V_3} deg_T(v) \leq 3|V_1|$.*

Proof. By definition, $|V(T)| = |V_1| + |V_2| + |V_3|$. Since there are no isolated vertices, $\sum_{v \in V(T)} \deg_T(v) = 2|E(T)|$. Since T is a tree, $|E(T)| < |V(T)|$. This implies $\sum_{v \in V(T)} \deg_T(v) < 2(|V_1| + |V_2| + |V_3|)$. Substituting lower bounds of degrees for each set, we get $|V_1| + 2|V_2| + 3|V_3| \leq \sum_{v \in V_1} \deg_T(v) + \sum_{v \in V_2} \deg_T(v) + \sum_{v \in V_3} \deg_T(v) = \sum_{v \in V(T)} \deg_T(v)$. Using the two equations we get $|V_1| + 2|V_2| + 3|V_3| \leq 2(|V_1| + |V_2| + |V_3|)$ which implies $|V_3| \leq |V_1|$. Adding the degree of vertices only in V_3 we get $\sum_{v \in V_3} \deg_T(v) = 2|V(T)| - (\sum_{v \in V_1} \deg_T(v) + \sum_{v \in V_2} \deg_T(v)) = 2(|V_1| + |V_2| + |V_3|) - (|V_1| + 2|V_2|) \leq |V_1| + 2|V_3|$. Using the bound of $|V_3|$, $\sum_{v \in V_3} \deg_T(v) \leq 3|V_1|$. □

A vertex subset $S \subseteq V(G)$ is said to *cover* an edge $uv \in E(G)$ if $S \cap \{u, v\} \neq \emptyset$. A vertex subset $S \subseteq V(G)$ is called a *vertex cover* in G if it covers all the edges in G. A *minimum vertex cover* is a set $S \subseteq V(G)$ such that S is a *vertex cover* and for all $S' \subseteq V(G)$ such that S' is a *vertex cover*, we have $|S| \leq |S'|$. A *vertex cover* S in G is said to be a *connected-vertex cover* if $G[S]$ is a *connected* graph. A set $I \subseteq V(G)$ of pairwise non-adjacent vertices is called an *independent set*. A set S of vertices is said to *dominate* another set S' of vertices if for every vertex v in S', $N(v) \cap S \neq \emptyset$.

We mention an FPT algorithm and a 2-factor approximation algorithm to compute connected vertex cover of given graph.

Proposition 2.1.2 ([24]). *Given a graph on n vertices and an integer k, there exists an algorithm which runs in time $2^k n^{\mathcal{O}(1)}$ and either outputs a connected vertex cover of size at most k or correctly concluded that no such connected vertex cover exists.*

Proposition 2.1.3 ([89]). *Let T be a depth-first search spanning tree of G with $NL(T)$ being set of non leaves in T and $vc(G)$ (resp. $cvc(G)$) be minimum (resp. connected) vertex*

cover of G. Then $NT(L)$ is a (connected) vertex cover of G and $|NL(T)| \leq 2 \cdot vc(G) \leq 2 \cdot cvc(G)$.

2.2 Graph Contraction

The *contraction* of edge uv in G deletes vertices u and v from G, and adds a new vertex, which is made adjacent to vertices that were adjacent to either u or v. Any parallel edges added in the process are deleted so that the graph remains simple. The resulting graph is denoted by G/e. For a given graph G and edge $e = uv$, we define G/e in the following way.

$$V(G/e) = (V(G) \cup \{w\}) \setminus \{u,v\}$$

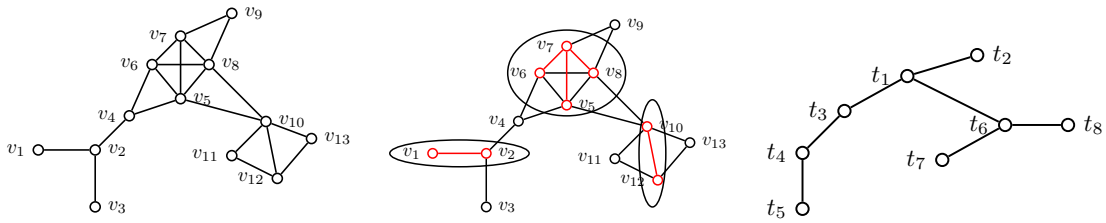
$$E(G/e) = \{xy \mid x,y \in V(G) \setminus \{u,v\}, xy \in E(G)\} \cup \{wx \mid x \in N_G(u) \cup N_G(v)\}$$

Every edge contraction reduces the number of vertices in graph by exactly one. Several edges might disappear due to one edge contraction. For a subset of edges F in G, graph G/F denotes the graph obtained from G by repeatedly contracting edge in F until no such edges remains.

We say graph G is *contractible* to graph H if there exists an onto function $\psi : V(G) \to V(H)$ such that following properties hold.

- For any vertex h in $V(H)$, graph $G[W(h)]$ is connected, where set $W(h) := \{v \in V(G) \mid \psi(v) = h\}$.
- For any two vertices h, h' in $V(H)$, edge hh' is present in H if and only if there exists an edge in G with one end point in $W(h)$ and another in $W(h')$.

For example, see Figure 2.1. For a vertex h in H, set $W(h)$ is called a *witness set* associated with h. We define *H-witness structure* of G, denoted by $\mathscr{W}$, as collection of all witness set. Formally, $\mathscr{W} = \{W(h) \mid h \in V(H)\}$. Witness structure $\mathscr{W}$ is a partition of vertices in G. If



(a) Input Graph (b) Graph Partition (c) Resultant graph

Figure 2.1: Graph contraction operation

a *witness set* contains more than one vertex then we call it *big* witness-set, otherwise it is *small/singleton* witness set.

If graph G has a H-witness structure then graph H can be obtained from G by series of edge contractions. For a fixed H-witness structure, let F be union of spanning trees of all witness sets. By convention, spanning tree of a singleton set is an empty set. To obtain graph H from G, it is sufficient to contract edges in F. If such witness structure exists then we say graph G is contractible to H. We say graph G is *k-contractible* to H if cardinality of F is at most k. In other words, H can be obtained from G by at most k edge contractions. Following observation are immediate consequences of definitions.

Observation 2.2.1. *If graph G is k-contractible to graph H then following statements are true.*

- $|V(G)| \leq |V(H)| + k$.
- *For any witness set W in a H-witness structure of G, cardinality of W is at most $k+1$.*
- *Any H-witness structure of G has at most k big witness sets.*
- *For a fixed H-witness structure, the number of vertices in G which are contained in big witness sets is at most $2k$.*

2.3 Parameterized Complexity

Many computational problems arising from real-world problems are NP-Hard, and we do not expect any efficient algorithms for solving them optimally. Parameterized complexity is an algorithm paradigm to tackle NP-Hard problems. Central notions in this paradigm are *fixed parameter tractability* and *kernelization*. The former notion is developed to identify NP-Hard problems which can be solved by restricting unavoidable exponential factor in running time to a *parameter* which is expected to be smaller then entire input. Kernelization has been developed as a mathematical framework to study data reduction rules and to quantify their efficacy.

A *parameterized problem* is a classical problem with an additional integer associated with each (classical) instance of the problem. Formally, it is defined as follows.

Definition 2.3.1 (Parameterized Problem)**.** *A parameterized problem is a language* $\Pi \subseteq \Sigma^* \times \mathbb{N}$*, where* Σ *is a fixed, finite alphabet. For an instance* $(I,k) \in \Sigma^* \times \mathbb{N}$*, integer* k *is called the* parameter.

For a parameterized problem $\Pi \subseteq \Sigma^* \times \mathbb{N}$, the derived classical problem Π^c is the set $\{I1^k | (I,k) \in \Pi\}$, where $1 \notin \Sigma$. Typically, parameter k reflects some structural property of the instance. A common parameter is a bound on the size of an optimum solution to the problem instance.

Definition 2.3.2 (Fixed Parameter Tractable (FPT))**.** *A parameterized problem* $\Pi \subseteq \Sigma^* \times \mathbb{N}$ *is called fixed parameter tractable if there exists an algorithm* $\mathscr{A}$ *(called a* fixed parameter algorithm*), a computable function* $f : \mathbb{N} \to \mathbb{N}$*, and a constant* c *such that, given* $(I,k) \in \Sigma^* \times \mathbb{N}$*, the algorithm* $\mathscr{A}$ *correctly decides whether* $(I,k) \in \Pi$ *in time bounded by* $f(k) \cdot (|I|+k)^c$.

The complexity class containing all fixed-parameter tractable problems is called FPT. Every parameterized problem need not be fixed-parameter tractable for given parameter.

Downey and Fellows introduced W-hierarchy in an attempt to classify parameterized problems according to their hardness (See [33]). We restrain from specifying exact definition of these classes. To understand results in this thesis it is sufficient to know following containment relationship among these classes: $\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \cdots$. Each class is believed to be properly contained in its superset. We mention that problems CLIQUE and DOMINATING SET parameterized by solution size are $\mathsf{W}[1]$-Complete and $\mathsf{W}[2]$-Complete, respectively.

One can obtain finer classification of FPT problems by examining the efficiency with which instances of these problems can be reduced to smaller instances without changing the answer. To quantify efficiency of such reductions, we define kernelization algorithm.

Definition 2.3.3 (Kernelization Algorithm)**.** *A kernelization algorithm, or simply kernel, of a parameterized language $\Pi \subseteq \Sigma^* \times \mathbb{N}$ is an algorithm that takes as input an instance $(I,k) \in \Sigma^* \times \mathbb{N}$, and in time polynomial in $|I|+k$ returns another instance (I',k') such that:*

- *$|I'|+k' \leq g(k)$ for some computable function $g(\cdot)$, and*
- *$(I,k) \in \Pi$ if and only if $(I',k') \in \Pi$.*

If the function $g(\cdot)$ is *linear, polynomial* or *exponential*, the problem is said to admit a *linear, polynomial* or *exponential kernel*, respectively. If there is a kernel for given problem then it is clearly FPT but, interestingly, the converse is also true. Any FPT problem admits an exponential kernel (See [25, Lemma 2.2]). This makes linear and polynomial kernels more interesting from the kernelization perspective. Every problem which is FPT by specified parameter may not have a polynomial kernel. An interesting line of research was started to rule out existence of polynomial kernels under reasonable complexity theoretic assumptions [12, 43]. Before stating results regarding non-existence of polynomial kernel, we mention definition of *polynomial compression* which generalizes the notion of kernels.

Definition 2.3.4 (Polynomial Compression)**.** *A polynomial compression of a parameterized language $\Pi \subseteq \Sigma^* \times \mathbb{N}$ into a language $L \subseteq \Sigma^*$ is an algorithm that takes as input an instance $(I,k) \in \Sigma^* \times \mathbb{N}$, and in time polynomial in $|I|+k$ returns a string y such that:*

- $|y| \leq p(k)$ *for some polynomial* $p(\cdot)$*, and*
- $y \in L$ *if and only if* $(I,k) \in \Pi$.

If $|\Sigma| = 2$, the polynomial $p(\cdot)$ is called *bit-size* of the compression. Note that a polynomial kernel is also a polynomial compression by treating the output kernel as the instance of un-parameterized version of Π.

Let Π_1 be a problem for which we already know that it does not admit a polynomial compression. To be able to *transfer* this hardness to other problems, we need following notion of reduction.

Definition 2.3.5 (Polynomial Parameter Transformation)**.** *Let* $\Pi_1, \Pi_2 \subseteq \Sigma^* \times \mathbb{N}$ *be two parameterized problems. An algorithm* $\mathscr{A}$ *is called a polynomial parameter transformation from* Π_1 *to* Π_2 *if given an instance* (I_1,k_1) *of* Π_1*,* $\mathscr{A}$ *works in polynomial time and outputs an instance* (I_2,k_2) *of* Π_2 *such that:*

- $|k_2| \leq p(k_1)$ *for some polynomial* $p(\cdot)$*, and*
- $(I_1,k_1) \in \Pi_1$ *if and only if* $(I_2,k_2) \in \Pi_2$.

In the following theorem, we formalize the notation of transfer of hardness.

Theorem 2.3.1 ([25] Theorem 15.15)**.** *Let* $\Pi_1, \Pi_2 \subseteq \Sigma^* \times \mathbb{N}$ *be two parameterized problems and assume that there exits a polynomial parameter transformation from* Π_1 *to* Π_2*. Then, if* Π_1 *does not admit a polynomial compression, neither does* Π_2.

Note that Theorem 2.3.1 uses the notion of compression instead of kernelization. If we liked to prove an analogous statement for polynomial kernelization, we would need to provide a way to reduce *back* Π_2^c to Π_1^c. This requires some additional assumptions on complexity of Π_1^c and Π_2^2. We mention following result by Bodlaender et al. which we use to rule out polynomial kernels.

Proposition 2.3.1 ([13])**.** *Let* Π_1 *and* Π_2 *be parameterized problems such that* Π_1^c *is* NP*-complete and* Π_2^c *is in* NP*; if there is a polynomial parameter transformation from* Π_1 *to*

Π_2 *and* Π_2 *has a polynomial kernel, then* Π_1 *has a polynomial kernel.*

Until now, we mentioned results that can be used to establish that for a particular problem, there is no polynomial sized compression. These results, with more involved treatments, can be used to argue that for a particular problem a certain sized compression is optimal. We mention one problem for which compression lower bound is known to be optimal under standard complexity assumptions. The problem DOMINATING SET takes as an input a graph and an integer k, and the goal is to decide whether the input graph contains a dominating set of size at most k. Any instance can be encoded with $\mathcal{O}(n^2)$ bits where n is the number of vertices in the input graph. Jansen and Pieterse proved that DOMINATING SET does not admit a compression of bit-size $\mathcal{O}(n^{2-\varepsilon})$, for any $\varepsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$ [60]. We use this result to obtain compression lower bound for another problems which is more useful in reduction. The input instance for RED-BLUE DOMINATING SET (RBDS) is a bipartite graph G with bi-partition (R,B) and an integer t. The question is whether R has a subset of at most t vertices that dominates B.

Proposition 2.3.2. RED-BLUE DOMINATING SET *does not admit a polynomial compression of bit size* $\mathcal{O}(n^{2-\varepsilon})$*, for any* $\varepsilon > 0$ *unless* $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. *Here, n is the number of vertices in the input graph.*

Proof. Assuming a contradiction, suppose RBDS admits a compression into $L \subseteq \Sigma^*$ with bit-size in $\mathcal{O}(n^{2-\varepsilon})$ for some $\varepsilon > 0$, where n is the number of vertices in the input graph for RBDS. This implies that there exists an algorithm $\mathscr{A}$ which takes an instance $I = (G,R,B,k)$ of RBDS and in time $n^{\mathcal{O}(1)}$ returns an equivalent instance I' of L with $|I'| \in \mathcal{O}(n^{2-\varepsilon})$.

Let (G,k) be an instance of DOMINATING SET and $n = |V(G)|$. We construct as instance (G',R,B,k') of RBDS as the following. For each $v \in V(G)$, we add vertices v_R and v_B to R and B, respectively. Further, for each $v_R \in R$ we make it adjacent to the corresponding copies in B of vertices in $N_G[v]$. Finally, we set $k' = k$. It is easy to see that (G,k) is

a YES instance of DOMINATING SET if and only if (G', R, B, k') is a YES instance of RBDS. Furthermore, the reduction takes polynomial time and $|V(G')| \in \mathcal{O}(n)$. But then DOMINATING SET admits a compression into Π with bit-size $\mathcal{O}(n^{2-\varepsilon})$, a contradiction. □

For more details on parameterized complexity, we refer the reader to the books of Downey and Fellows [33], Flum and Grohe [39], Niedermeier [83], and the more recent book by Cygan et al. [25].

2.4 Lossy Kernelization

As the goal in parameterized algorithms is to eventually solve the given instance of a problem, the application of a kernelization algorithm is typically followed by an exact or approximation algorithm that finds a solution to a reduced instance. However, the definition of kernel mentioned in Section 2.3 provides no insight into how this solution relates to a solution to the original instance. For instance, consider a parameterized problem Π, an instance (I,k) of Π, and a kernelization algorithm $\mathscr{A}$ of Π. Let (I',k') be the instance returned by $\mathscr{A}$ on the input (I,k). Given an approximate solution to (optimization version of) Π in (I',k'), using the (classical) notion of polynomial kernels we can say nothing about a solution to the instance (I,k). Many state of the art approximation algorithms used to tackle NP-Hard problems are extremely sophisticated and it is infeasible to apply them to large problem instances. It is far more practical to reduce a large instance to a small kernel, then obtain a good approximate solution to this kernel, and finally transform it into an approximate solution to the original instance. Lokshtanov et al. [75] introduced the notion of *lossy kernelization*, which provides a framework for "*approximation preserving kernelization*". Building on the notion of classical kernelization, this framework combines well with approximation algorithms and heuristics.

In lossy kernelization, we work with optimization analogue of parameterized problem.

Along with an instance and a parameter, optimization analogue of the problem also has a string called *solution*. We start with a definition of a *parameterized optimization problem*, which is the parameterized analogue of an optimization problem in the theory of approximation algorithms.

Definition 2.4.1 (Parameterized Optimization Problem)**.** *A parameterized Optimization problem is a computable function* $\Pi^\circ : \Sigma^* \times \mathbb{N} \times \Sigma^* \mapsto \mathbb{R} \cup \{\pm\infty\}$. *The instances of* Π° *are pairs* $(I,k) \in \Sigma^* \times \mathbb{N}$ *and a solution to* (I,k) *is simply a string* $S \in \Sigma^*$ *such that* $|S| \leq |I| + k$.

In this thesis, all optimization problems are minimization problems. We present rest of section with respect to parameterized *minimization* problem as parameterized *maximization* problem can be defined in a similar way. We treat decision version of problem (Π) and optimization version of problem (Π°) in separate sections. In chapters, we denote decision version and optimzation version of a problem by same notation.

The *value* of a solution S is $\Pi^\circ(I,k,S)$. The *optimum value* of (I,k) is defined as: $\mathrm{OPT}_\Pi(I,k) = \min_{S \in \Sigma^*, |S| \leq |I|+k} \Pi^\circ(I,k,S)$, and an *optimum solution* for (I,k) is a solution S such that $\Pi^\circ(I,k,S) = \mathrm{OPT}_{\Pi^\circ}(I,k)$. For a constant $c > 1$, S is *c-factor approximate* solution for (I,k) if $\frac{\Pi^\circ(I,k,S)}{OPT_{\Pi^\circ}(I,k)} \leq c$. We omit the subscript Π° in the notation for optimum value if the problem under consideration is clear from the context. We define an α*-approximate polynomial-time preprocessing algorithm* for a parameterized minimization problem Π°.

Definition 2.4.2 (α-Approximate Polynomial-time Preprocessing Algorithm)**.** *Let* $\alpha \geq 1$ *be a real number and* Π° *be a parameterized minimization problem. An* α*-approximate polynomial-time preprocessing algorithm is defined as a pair of polynomial-time algorithms, called the* reduction algorithm *and the* solution lifting algorithm*, that satisfy the following properties.*

- *Given an instance* (I,k) *of* Π°*, the reduction algorithm computes an instance* (I',k') *of* Π°.

- *Given the instances (I,k) and (I',k') of Π°, and a solution S' to (I',k'), the solution lifting algorithm computes a solution S to (I,k) such that $\frac{\Pi^\circ(I,k,S)}{\mathrm{OPT}(I,k)} \leq \alpha \cdot \frac{\Pi^\circ(I',k',S')}{\mathrm{OPT}(I',k')}$.*

We sometimes refer α-approximate polynomial-time preprocessing algorithm kernel as α-*lossy rule* or α-*reduction rule*. A *reduction rule* is the reduction algorithm of a polynomial time preprocessing algorithm. We say that a reduction rule is *applicable* on an instance (I,k) if the result (I',k') obtained by applying reduction rule on it is different from (I,k). It is *applicable* on an instance if the output is different from the input instance.

Definition 2.4.3 (α-Approximate Kernel)**.** *An α-approximate kernelization (or α-approximate kernel) for Π° is an α-approximate polynomial-time preprocessing algorithm $\mathscr{A}$ such that $size_{\mathscr{A}}(k) = \sup\{|I'| + k' : (I',k') = R_{\mathscr{A}}(I,k), I \in \Sigma^*\}$, is upper bounded by a computable function $g : N \to N$, where $R_{\mathscr{A}}$ is the reduction algorithm in $\mathscr{A}$.*

We sometimes refer α-approximate kernel as α-*lossy kernel*. In classical kernelization, often we apply reduction rules several times to reduce the given instance. This however breaks down in lossy kernelization. Each application of a reduction rule could potentially increase the "gap" between (i) the approximation quality of the solution to the kernel on the one hand, and (ii) the approximation quality of solution to the original instance that is computed by the solution lifting algorithm, on the other. Let $(G,k) \to (G',k') \to (G'',k'')$ be a series of instance obtained after applying α-reduction rules. In other words, $(G',k'),(G'',k'')$ are instances obtained by applying α-reduction rule on $(G,k),(G',k')$ respectively. Given a c-factor approximate solution S'' for (G'',k''), solution lifting algorithm can obtain $(\alpha^2 \cdot c)$-factor approximate solution S for (G,k). To remedy this shortcoming, we require the notion of α-strict kernelization and α-safe reduction rules.

Definition 2.4.4 (Strict Kernel)**.** *An α-approximate kernelization is said to be* strict *if $\frac{\Pi^\circ(I,k,s)}{\mathrm{OPT}(I,k)} \leq \max\{\frac{\Pi^\circ(I',k',s')}{\mathrm{OPT}(I',k')}, \alpha\}$.*

Definition 2.4.5 (Safe reduction rule)**.** *A reduction rule is said to be α-safe for Π° if there is a solution lifting algorithm, such that the rule together with this algorithm constitutes a strict α-approximate polynomial-time preprocessing algorithm for Π°.*

In the above example, if reduction rules are α-safe, we can obtain $\max\{\alpha, c\}$-factor (instead of $\alpha^2 c$-factor) approximate solution for (G,k) from c-factor approximate solution for (G'',k''). A (lossy) reduction rule being 1-safe is more strict than (classical) reduction rule begin safe. To prove the correctness of reduction rule in classical kernelization, we prove that for an input instance is a YES instance if and only if output instance is a YES instance. In case of lossy kernelization, we need to argue that for any c-factor approximate solution for an output instance, one can obtain c-factor approximate solution for input instance.

We mentioned α-approximate kernelization algorithms for given problem Π°. We now define family of algorithms, one for every α, to compute approximate kernel.

Definition 2.4.6 (PSAKS)**.** *A polynomial-size approximate kernelization scheme (PSAKS) for* Π° *is a family of* α*-approximate polynomial kernelization algorithms for each* $\alpha > 1$.

The size of an output instance of a PSAKS, when run on (I,k) with approximation parameter α, is upper bounded by $f(\alpha) \cdot k^{g(\alpha)}$ for some functions f and g independent of $|I|$ and k.

We briefly discuss the importance of the parameter k in this framework. For the sake of simplicity imagine that we are considering a parameterized problem with solution size as parameter. In classical settings, the question is to determine whether there exists a solution of size at most k. Assuming that there always exists a trivial solution of large size, the question is to differentiate solutions of size at most k from solutions of size at least $k+1$. To reflect this, parameterized minimization problem is defined in the following way.

$$\Pi^\circ(I,k,S) = \begin{cases} \infty & \text{if S is not a solution} \\ \min\{|S|, k+1\} & \text{otherwise} \end{cases}$$

Above definition allows us to design solution lifting algorithms which consider all of solutions of value more than $k+1$ are *equally bad*. Consider a case when input of solution

lifting algorithm is (I,k,I',k',S') where S' is a solution for (I',k'). Let $k'=k$; $k>\alpha$; $k>2$ and $\Pi^{\circ}(I',k',S')=k^{100}$. Since the solution lifting algorithm runs in polynomial time and we are dealing with NP-hard problem, it is unlikely that we find solution S for (I,k) of size at most k. By definition of solution lifting algorithm, it needs to find a solution for (G,k) which is at least $\max\{k^{99},\alpha\}=k^{99}$ times the size of optimum solution. It is futile to compel a solution lifting algorithm to find a k^{99}-factor approximate solution. Present definition of Π° allows solution lifting algorithm to return a feasible solution S of any cardinality. Notice that above definition ensures both $\Pi^{\circ}(I',k',S')$ and $\Pi^{\circ}(I,k,S)$ are equal.

We define a notion of a polynomial time reduction appropriate for obtaining lower bounds for α-approximate kernels. This is very similar to the definition of α-approximate polynomial time pre-processing algorithm (Definition 2.4.2).

Definition 2.4.7. *Let $\alpha \geq 1$ be a real number. Let Π and Π' be two parameterized minimization problems. An α-approximate polynomial parameter transformation (α-`appt` for short) $\mathscr{A}$ from Π to Π' is a pair of polynomial time algorithms, called reduction algorithm $R_{\mathscr{A}}$ and solution lifting algorithm. Given as input an instance (I,k) of Π the reduction algorithm outputs an instance (I',k') of Π'. The solution lifting algorithm takes as input an instance (I,k) of Π, the output instance $(I',k')=R_{\mathscr{A}}(I,k)$ of Π', and a solution s' to the instance I' and outputs a solution s' to (I,k) such that*

$$\frac{\Pi(I,k,s)}{\mathrm{OPT}_{\Pi}(I,k)} \leq \frac{\Pi(I',k',s')}{\mathrm{OPT}_{\Pi'}(I',k')}$$

In the standard kernelization setting lower bounds machinery rules out existence of compression algorithms. Similar to this, lower bound machinery in lossy kernelization rules out existence of compression algorithms. Towards that we need to generalize the definition of α-approximate kernel to α-approximate compression. The only difference is that in the later case the reduced instance can be an instance of any parameterized optimization problem.

Definition 2.4.8. *Let $\alpha \geq 1$ be a real number. Let Π and Π' be two parameterized optimization problems. An α-approximate compression from Π to Π' is an α-appt $\mathscr{A}$ from Π to Π' such that $size_{\mathscr{A}}(k) = \sup\{|I'| + k' : (I',k') = R_{\mathscr{A}}(I,k), I \in \Sigma^*\}$, is upper bounded by a computable function $g : N \to N$, where $R_{\mathscr{A}}$ is the reduction algorithm in $\mathscr{A}$.*

In [75], authors proved that parameterized optimization version of SET COVER parameterized by universe size does not admit an α-approximate compression of polynomial size for any $\alpha \geq 1$ unless NP$\subseteq$ coNP/poly. The input of SET COVER is a family $\mathscr{S}$ of subsets of a universe U and the objective is to choose a minimum sized subfamily $\mathscr{F}$ of S such that $\bigcup_{S \in \mathscr{F}} S = U$. Such a set $\mathscr{F}$ is called a set cover of $(\mathscr{S},U)$. Since the parameter used here is a structural parameter, its parameterized version SET COVER$/n$ (SC$/n$) can be defined as follows.

$$\text{SC}/n((\mathscr{S},U),|U|,\mathscr{F}) = \begin{cases} \infty & \text{if F is a set cover} \\ |\mathscr{F}| & \text{otherwise} \end{cases}$$

Theorem 2.4.1. SET COVER$/n$ *does not have a polynomial size α-approximate compression for any $\alpha \geq 1$, unless* NP $\subseteq$ *coNP/poly.*

Parameterized version of SET COVER is generally associated with parameter k which is size of solution. We can define it in the following way.

$$\text{SC}/k((\mathscr{S},U),k,\mathscr{F}) = \begin{cases} \min\{|\mathscr{F}|,k+1\} & \text{if } \mathscr{F} \text{ is a set cover} \\ \infty & \text{otherwise} \end{cases}$$

Without loss of generality, we can assume that we are working with an instance in which $k \leq n$. If $k \geq n$ then we can select a private set to cover each element in universe and hence the instance can be solved in polynomial time. Suppose there exists a polynomial size α-approximate compression of SET COVER$/k$ for some α. We can use this compression algorithm to get a lossy compression for SET COVER$/n$ by substituting $k = n$. This is a

contradiction to Theorem 2.4.1. This leads to following corollary.

Corollary 2.4.1. SET COVER/k *does not have a polynomial size* α*-approximate compression for any* $\alpha \geq 1$*, unless* NP $\subseteq$ *coNP/poly.*

We encourage the reader to see [75] for more comprehensive discussion of these ideas and definitions. The authors presented lossy kernels for several problems which do not admit a classical kernelization, such as CONNECTED VERTEX COVER, DISJOINT CYCLE PACKING, DISJOINT FACTORS, etc. They also develop a lower bound framework for lossy kernels, by extending the lower bound framework of classical kernelization. They show that LONGEST PATH does not admit a lossy kernel of polynomial size unless NP $\subseteq$ coNP/poly.

In this thesis, we investigate several graph contraction problems in the framework of lossy kernelization. We design lossy polynomial kernels for some graph contraction problems which do not admit classical polynomial kernels NP $\subseteq$ coNP/poly.

Chapter 3

Tree Contraction

3.1 Introduction

In this chapter, we study problem of contracting given graph to a tree. PATH CONTRACTION and TREE CONTRACTION were first problems studied in parameterized setting by Heggernes et al. [55]. They proved that when parameterized by solution size, PATH CONTRACTION admits a linear vertex kernel but TREE CONTRACTION does not admit a polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. Difference in sizes of kernels for these two closely related problems is starting point of work presented in this chapter. We formally define TREE CONTRACTION.

TREE CONTRACTION	**Parameter:** k
Input: A graph G and an integer k	
Question: Is it possible to obtain a tree from G with at most k edge contractions?	

Other problems mentioned in this section are defined in similar way. Heggernes et al. presented a parameter preserving reduction from an instance of RED BLUE DOMINATION problem to an instance of TREE CONTRACTION [55, Theorem 2]. This reduction also holds for STAR CONTRACTION. It is interesting that when parameterized by solution size PATH CONTRACTION admits a polynomial kernel but STAR CONTRACTION does not. One

of the structural difference between path and star is number of leaves. While path has at most two leaves, a star graph can have unbounded number of leaves. This hints that number of leaves can be a additional parameter one needs to consider to change the parameterized complexity of STAR CONTRACTION problem. To formalize the question: *does the bound on the number of leaves makes the difference in kernelization complexity of these two problems?* We prove that indeed this is the case. We show that when parameterized by addition parameter l, number of leaves, we do get polynomial kernels. In fact, what we get is an uniform kernel i.e. the kernel which is polynomial both in terms of k and l. Formally, the problem is defined below.

BOUNDED TREE CONTRACTION (BOUNDED TC) **Parameter:** $k+\ell$
Input: A graph G and integers k, ℓ
Question: Is it possible to obtain a tree with at most ℓ leaves from G with at most k edge contractions?

We prove that there exists a polynomial kernel with $\mathcal{O}(k\ell)$ vertices and $\mathcal{O}(k^2+k\ell)$ edges in Section 3.3. In Section 3.4, we prove that these kernels are optimal unless NP $\subseteq$ coNP/poly.

We know that TREE CONTRACTION does not have a polynomial kernel when parameterized by solution size. Next natural question is: *does it have a lossy kernel of polynomial size when parameterized by solution size?*. We prove that given a graph G on n vertices, an integer k and an approximation parameter $\alpha > 1$, there is an algorithm that runs in $n^{\mathcal{O}(1)}$ time and outputs a graph G' on $\mathcal{O}(k^{d+2}+k^3)$ vertices and an integer k' such that for every $c > 1$, a c-approximate solution for (G',k') can be turned into a $(c\alpha)$-approximate solution for (G,k) in $n^{\mathcal{O}(1)}$. Here, $d = \lceil \frac{\alpha}{\alpha-1} \rceil$.

Results presented in Section 3.3 and 3.4 are from [1]. Lossy kernel for TREE CONTRACTION is based on [69].

3.2 Preliminaries

Let G be a connected graph and F is a set of edges in G such that $G/F = T$ is a tree. Let $\mathscr{W}$ be a T-witness structure of G. We start with following observation on neighbors of vertices which are contained in $W(t)$ for some leaf t in T.

Observation 3.2.1. *Let t be a leaf in T and t' be its unique neighbor. Then, for every vertex v in $W(t)$, its neighborhood is contained in $W(t') \cup W(t)$.*

Proof. For the sake of contradiction, assume that there exists a vertex v in $W(t)$ which has neighbors outside $W(t)$ and $W(t')$. Let $W(t'')$ be a witness set such that $N(v)$ intersects with $W(t'')$ and t'' is not equal to t or t'. Since G is contractible to T, there exists an edge between t and t''. This implies that t has at least two neighbors in T contradicting the fact that it is a leaf. □

For every integer $\ell \geq 2$, consider a set of trees which has at most ℓ leaves. For $\ell = 2$, this set is a collection of all paths. Following observation states that this set of graphs is closed under edge contraction.

Observation 3.2.2. *Let T be a tree and T' be the graph obtained from T by contracting an edge v_1v_2 in $E(T)$. If T has at most ℓ leaves then T' is a tree with at most ℓ leaves.*

This set is also closed under an operation of *uncontracting* an edge with some additional conditions. We first formally define such operation. Consider a tree T and one of its internal vertex, say v. Let L, R be a partition of $N(v)$ such that none of them is an empty set. We define operation $\textsc{Split}(T, v, L, R)$ as follows. See Figure 3.1 for illustration.

$\textsc{Split}(T, v, L, R)$: Remove vertex v and add two vertices v_1 and v_2. Make v_1 adjacent with every vertex in L and v_2 adjacent with every vertex in R. Add edge v_1v_2. If T' is the graph obtained from T by this operation then $V(T') = (V(T) \setminus \{v\}) \cup \{v_1, v_2\}$ and $E(T') = (E(T) \setminus (\{vu \mid u \in N(v)\})) \cup \{v_1u \mid u \in L\} \cup \{v_2u \mid u \in R\} \cup \{v_1v_2\}$.

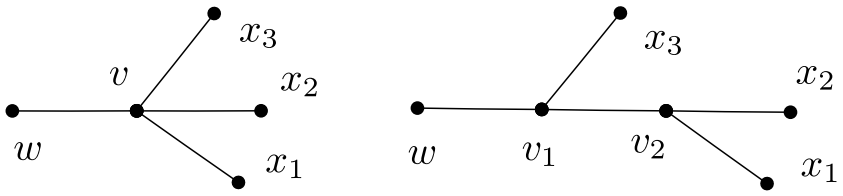


Figure 3.1: Operation SPLIT(T,v,L,R) with $L=\{x_3\}$ and $R=\{x_1,x_2\}$.

The following lemma proves that this operation on a tree results into another tree with same number of leaves.

Lemma 3.2.1. *Let T be a tree, v be an internal vertex of T and $N(v)$ is partitioned into two non-empty sets L and R. Let T' is the graph obtained from T after applying* SPLIT(T,v,L,R). *If T has at most ℓ leaves then T' is a tree with at most ℓ leaves.*

Proof. First, we prove that T' is a tree. Suppose not, then there exists a cycle in T'. Let C' be an induced cycle in T'. If C' contains at most one of v_1,v_2, then we can obtain a cycle C in T by replacing v_1 or v_2 by v. Otherwise, C contain both v_1 and v_2. Since, C' is an induced cycle and $v_1v_2 \in E(T')$, vertices v_1,v_2 appear consecutively in C'. Again, by replacing v_1v_2 by vertex v, we obtain a cycle in T which is a contradiction. Hence, T' is acyclic. Note that v_1v_2 is an edge in T' with $N_{T'}(v_1)\setminus\{v_2\}=L\neq\emptyset$ and $N_{T'}(v_2)\setminus\{v_1\}=R\neq\emptyset$, therefore v_1,v_2 are not leaves in T'. All leaves in T' remains as leaf vertices in T'. This implies that number of leaves in T' is no more than the number of leaves in T. □

We mention following simplifying assumption which is used in designing a lossy kernel for TREE CONTRACTION. It helps us to concentrate on 2-connected components of input graph.

Lemma 3.2.2 ([55])**.** *A connected graph is k-contractible to a tree if and only if each of its 2-connected components is contractible to a tree using at most k edge contractions in total.*

We make an observation on a tree witness structure of a 2-connected graph.

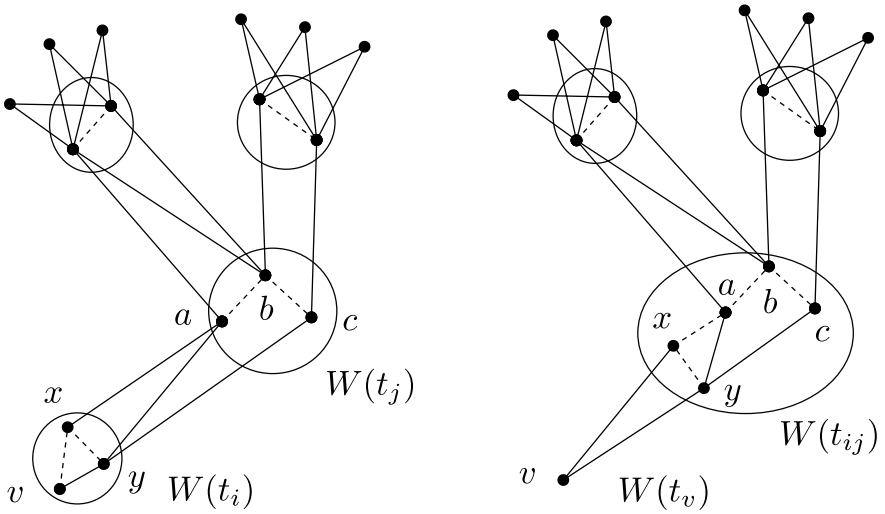


Figure 3.2: Modifying big witness sets which are leafs. All but one vertex in $W(t_i)$ has been moved to $W(t_j)$. See Observation 3.2.3.

Observation 3.2.3. *Consider a 2-connected graph G and let F be a set of edges in G such that G/F is a tree with at least three vertices. Then there exists a set F' of at most $|F|$ many edges such that G/F' is a tree and corresponding G/F'-witness structure $\mathscr{W}'$ satisfies following property: Witness set $W'(t')$ in $\mathscr{W}'$ is singleton if and only if t' is a leaf in G/F'.*

Proof. Let $\mathscr{W}$ be a T-witness structure of G where $T = G/F$. We first show that every vertex t in $V(T)$ for which $W(t)$ is singleton, is a leaf in T. Assume there exists a non-leaf t in T such that $W(t) = \{u\}$ for some vertex u in $V(G)$. Since t is not a leaf, $T - \{t\}$ has at least two non-empty subtrees, say T_1 and T_2. Define two sets $U_1 := \bigcup_{t \in V(T_1)} W(t)$ and $U_2 := \bigcup_{t \in V(T_2)} W(t)$. As $\mathscr{W}$ is a T-witness structure of G, there is no edge between a vertex in U_1 and a vertex in U_2 in $G - \{u\}$. This contradicts the fact that G is 2-connected. Hence our assumption is wrong and every singleton witness set corresponds to a leaf.

Consider a leaf t_i in T such that $W(t_i)$ is not a singleton set. Let t_j be the unique neighbor of t_i in T. Since T has at least three nodes, t_j is not a leaf. As $t_i t_j \in E(T)$, there exists an edge in G with one end-point in $W(t_i)$ and another in $W(t_j)$. Hence, $G[W(t_i) \cup W(t_j)]$ is connected (See Figure 3.2). We argue that $G[W(t_i) \cup W(t_j)]$ has a spanning tree which has

a leaf in $W(t_i)$. Observe that as $|W(t_i)| > 1$, any spanning tree of $G[W(t_i)]$ has at least two leaves. If there is a spanning tree of $G[W(t_i)]$ that has a leaf u which is not adjacent to any vertex in $W(t_j)$, then $G[(W(t_i) \cup W(t_j)) \setminus \{u\}]$ is connected and u is the required vertex. Otherwise every leaf in every spanning tree of $G[W(t_i)]$ is adjacent to some vertex in $W(t_j)$ and hence $G[(W(t_i) \cup W(t_j)) \setminus \{u\}]$ is connected for each vertex $u \in W(t_i)$. Therefore, $G[W(t_i) \cup W(t_j)]$ has a spanning tree which has a leaf u from $W(t_i)$.

Define sets $W_u := \{u\}$ and $W_{ij} := (W(t_j) \cup W(t_i)) \setminus \{u\}$. Let $\mathscr{W}'$ be a witness structure obtained from $\mathscr{W}$ by removing $W(t_i), W(t_j)$ and adding W_u, W_{ij}. Formally, $\mathscr{W}' = (\mathscr{W} \cup \{W_u, W_{ij}\}) \setminus \{W(t_i), W(t_j)\}$. Note that $\mathscr{W}'$ partitions $V(G)$ and for every witness sets W' in $\mathscr{W}'$, $G[W']$ is connected. Let T' is the graph obtained from G by contracting all witness sets in $\mathscr{W}'$. In other words, $\mathscr{W}'$ is T'-witness structure of G. We argue that T' is a tree. By Observation 3.2.1, for every vertex in v in $W(t_i)$, neighborhood of v is contained in $W(t_i) \cup W(t_j)$. Moreover, $W(t_j)$ is a subset of W_{ij}. Hence witness set W in $\mathscr{W}' \setminus \{W_u, W_{ij}\} = \mathscr{W} \setminus \{W(t_i), W(t_j)\}$ is adjancent with W_{ij} if and only if W is adjancent with W_j. By construction, W_u is adjancent with only W_{ij}. Hence T' can be obtained from T by renaming t_j to t_{ij} and t_i to t_u. This implies T' is a tree and it has same number of vertices as that of T. Note that the number of edges needed to contract all witness sets in $\mathscr{W}$ is same as the number of edges needed to contract all witness sets in $\mathscr{W}'$.

Recall that T has at least three vertices and hence every leaf vertex is adjacent to some non-leaf vertex. We repeat the above process for every non-singleton leaf until every witness set corresponding to a leaf is a singlton witness set. If F' is union of spanning trees of this modified witness structure then the number of edges in F' and F are same. This concludes proof of the lemma. □

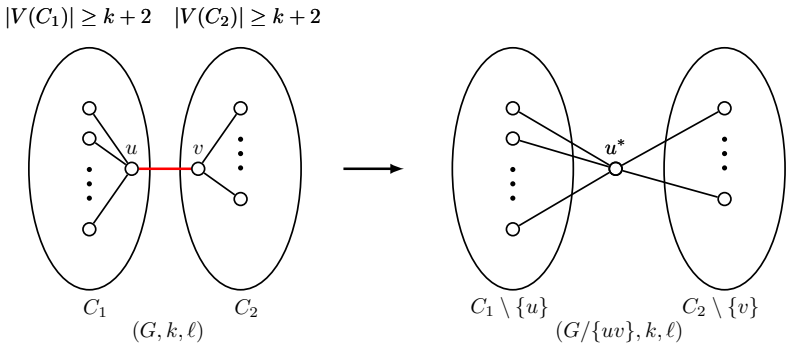


Figure 3.3: An illustration of Reduction Rule 3.3.1.

3.3 Kernel for BOUNDED TREE CONTRACTION

In this section we design a kernelization algorithm for BOUNDED TREE CONTRACTION (BOUNDED TC). Our algorithm is inspired by kernelization algorithm for PATH CONTRACTION presented in [55]. Let (G,k,ℓ) be an instance of BOUNDED TC. It is safe to assume that the input graph G is connected otherwise it is a trivial NO instance.

Kernelization algorithm has only one reduction rule which finds an contracts an *irrelevant* edge. We argue that a cut edge whose removal results in two *large* connected components is an irrelevant edge.

Reduction Rule 3.3.1. *Let uv be a cut-edge in G and C_1, C_2 be the connected components in $G - \{uv\}$. If $|V(C_1)|, |V(C_2)| \geq k+2$ then contract uv. The resulting instance is (G', k, ℓ), where $G' = G/\{uv\}$.*

Informally speaking, since edge uv is a cut-edge, it is not a part of any cycle. We do not need to contract it to destroy any cycle. The only reason we might include it in a solution is to reduce the number of leaves in resultant tree. As the sizes of both connected components of $G - \{uv\}$ is at least $k+2$, contracting at most k edges can not destroy either of connected component. Hence no end points of uv can be part of leaf in resulting graph. In other words, uv is irrelevant with respect to any solution of size at most k and can safely be contracted.

Lemma 3.3.1. *Reduction rule 3.3.1 is safe.*

Proof. We argue that (G,k,ℓ) is a YES instance of BOUNDED TC if and only if (G',k,ℓ) is a YES instance of BOUNDED TC.

To prove forward direction, let (G,k,ℓ) be a YES instance of BOUNDED TC. Let F be a set of at most k edges such that G/F be a tree with at most ℓ leaves. By Observation 3.2.2, graph $G/(F \cup \{uv\})$ is also a tree with at most ℓ leaves. Note that $G/(F \cup \{uv\}) = (G/\{uv\})/(F \setminus \{uv\}) = G'/(F \setminus \{uv\})$. Hence $G'/(F \setminus \{uv\})$ is a tree with at most ℓ leaves. Since $|F \setminus \{uv\}| \leq |F| \leq k$, we can conclude that (G',k,ℓ) is a YES instance of BOUNDED TC.

To prove reverse direction, let (G',k,ℓ) be a YES instance of BOUNDED TC. Let F' be a set of at most k edges such that $G'/F' = T'$ is a tree with at most ℓ leaves. We first argue that G is $(|F'|+1)$-contractible to a tree, say T_1, which has at most ℓ leaves. Using SPLIT operation on T_1 we argue that G is actually $|F'|$-contractible to a tree with at most ℓ leaves.

Let $\mathscr{W}'$ be a T'-witness structure of G'. Let u^* be the vertex resulting while contracting edge uv in G to get G'. Consider vertex t^* in $V(T')$ such that u^* is in $W(t^*)$. Define set $W(t_1) := (W(t^*) \setminus \{u^*\}) \cup \{u,v\}$. Let $\mathscr{W}_1$ be the witness structure obtained from $\mathscr{W}'$ by removing $W(t^*)$ and adding $W(t_1)$. Note that $\mathscr{W}_1$ partitions $V(G)$ and for each W in $\mathscr{W}_1$, $G[W]$ is connected. Let T_1 be a graph obtained from G by contracting witness sets in $\mathscr{W}_1$. In other words, $\mathscr{W}$ is a T_1-witness structure of G. Note that T_1 can be obtained from G by contracting all edges in $F' \cup \{uv\}$. This implies T_1 can be obtained from G' by contracting all edges in F' and hence it is a tree with at most ℓ leaves. We conclude that G is $(|F'|+1)$-contractible to a tree with at most ℓ leaves.

Since uv is a cut-edge in G, it is also a cut-edge in $G[W(t_1)]$. Let C_u and C_v be the connected components of $G[W(t_1)] - \{uv\}$ containing u and v, respectively. Further, let $W_u = V(C_u)$, $W_v = V(C_v)$. Consider a witness structure $\mathscr{W}$ of G obtained from $\mathscr{W}_1$ by removing $W(t_1)$ and adding W_u and W_v. Notice that $\mathscr{W}$ partitions $V(G)$ and for each W in $\mathscr{W}$, $G[W]$ is

connected. Moreover, we need $|F'|$ many edges to contract all witness sets in $\mathscr{W}$. Let T be a graph obtained by contracting all witness sets in $\mathscr{W}$. In other words, $\mathscr{W}$ is a T-witness structure of G. Note that G is $|F'|$-contractible to T. The only thing which remains to prove is that T is a tree with at most ℓ leaves. We prove this by showing that T can be obtained from T_1 by SPLIT operation at vertex t_1.

We start with proving that t_1 is an internal vertex in T_1 by showing that it has at least two neighbors.

Claim. Vertex t_1 in T_1 has at least two neighbors.

Proof. Each witness set in $\mathscr{W}_1$ is of size at most $k+2$ and hence $|W(t_1)| \leq k+2$. If t_1 is the only vertex in T_1, then all the vertices in $(V(C_1) \cup V(C_2)) \setminus \{u,v\}$ are in $W(t_1)$. This implies that $|W(t_1)| \geq 2k+3$ which is a contradiction. If t_1 has unique neighbor, say $\hat{t}$, in $V(T_1)$, then $V(C_1) \cap W(\hat{t})$ and $V(C_2) \cap W(\hat{t})$ are both non empty as $|V(C_1)|, |V(C_2)| \geq k+2$ and $|W(t_1) \setminus \{u,v\}| \leq k$. Since uv is a cut-edge, any path connecting vertices in $V(C_1)$ and $V(C_2)$ must contain an edge uv. Both sets $V(C_1) \cap W(\hat{t})$ and $V(C_2) \cap W(\hat{t})$ are not empty but $W(\hat{t})$ does not contain u,v. This implies that $G'[W(\hat{t})]$ is not connected contradicting the fact that it is a witness set. Hence t_1 has at least two neighbors in T_1. $\diamond$

Consider a vertex t in T_1 which is adjacent with t_1. From above arguments, we know that exactly one of $V(C_1) \cap W(t)$ and $V(C_2) \cap W(t)$ is an empty set. Partition vertices in $N_{T'}(t_1)$ into two sets L and R depending on whether corresponding witness sets intersect C_1 or C_2. Formally, $L := \{t \mid t \in N_{T'}(t) \text{ and } W(t) \cap V(C_1) \neq \emptyset\}$ and $R := \{t \mid t \in N_{T'}(t) \text{ and } W(t) \cap V(C_2) \neq \emptyset\}$. Note that (L,R) is a partition of $N_{T_1}(t)$ and none of this set is empty. Let T be the graph obtained after operation $\text{SPLIT}(T_1, t_1, L, R)$. By Lemma 3.2.1, T is a a tree with at most ℓ many leaves.

Hence, if there exist a set of edges F' in G' such that G/F' is tree with at most ℓ leaves then G is $|F'|$-contractible to a tree with at most ℓ leaves. This concludes the proof of reverse direction. $\square$

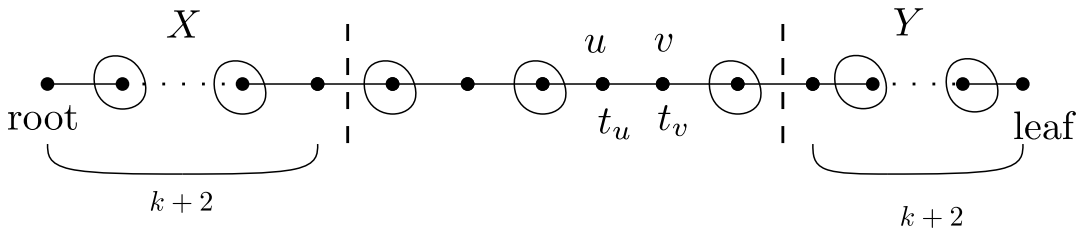


Figure 3.4: Parts of a longest path from root to a leaf. See Lemma 3.3.2.

We now argue that exhaustive application of Reduction Rule 3.3.1 either returns a reduced instance of bounded size or we can conclude that original instance is a NO instance.

Lemma 3.3.2. *Let (G,k,ℓ) be an instance of* BOUNDED TC *on which Reduction Rule 3.3.1 is not applicable. If (G,k,ℓ) is a* YES *instance of* BOUNDED TC*, then G has at most $\mathcal{O}(k\ell)$ vertices and $\mathcal{O}(k^2+k\ell)$ edges.*

Proof. Let (G,k,ℓ) be a YES instance of BOUNDED TC and $F \subseteq E(G)$ be a solution such that $T = G/F$ is a tree with at most ℓ leaves. Fix an arbitrary vertex of the tree T as its root. Let $\mathscr{W}$ be a T-witness structure of G. As T is obtained using at most k edge contractions from G, $|V(G)| \leq |V(T)|+k$. Note that $|V(T)|$ is upper bounded by the number of different paths from the root to leaves times the maximum length of a path. Since the number of leaves in T is bounded by ℓ, the number of paths from the root to leaves is also bounded by ℓ.

Let $P = \{t_1,t_2,\ldots,t_q\}$ be a longest path from the root to a leaf in T. If $q \leq 2k+5$ then $|V(T)| \leq \mathcal{O}(k\ell)$. Consider a case when $q > 2k+5$. We argue that there does not exist i in $\{k+2,\ldots,q-k-2\}$ such that both $W(t_i)$ and $W(t_{i+1})$ are of cardinality one. Define two sets $X := \cup_{j\in\{1,2,\ldots,k+2\}} W(t_j)$ and $Y := \cup_{j\in\{q-(k+2),\ldots,q\}} W(t_j)$ of $V(G)$. See Figure 3.4. Notice that $|X|,|Y| \geq k+2$. If there exists i in $\{k+3,\ldots,q-k-1\}$ such that $W(t_i) = \{u\}$ and $W(t_{i+1}) = \{v\}$ then uv is a cut-edge in G. Moreover, X,Y are in two different connected components of $G-\{uv\}$. Hence both the connected components of $G-\{uv\}$ are of size at least $k+2$. In this case, Reduction rule 3.3.1 is applicable. This contradicts the fact that (G,k,ℓ) is a reduced instance. Hence for i in $\{k+2,\ldots,q-k-2\}$, if $W(t_i)$ is a small

witness set then $W(t_{i+1})$ is a big witness set. Since there are at most k big witness sets, the number of vertices in path P is at most $2k+2(k+2) = 4k+4$. This implies $q \leq 4k+4$ and $|V(T)| \leq \ell(4k+4)$. Hence $|V(G)|$ is at most $\mathcal{O}(k\ell)$.

We now bound the number of edges in the graph G. Notice that the maximum degree of a vertex t in the tree T is bounded by ℓ. Since, every edge contraction reduces the number of vertices by 1, the maximum degree of a vertex in G is at most $\ell+k$. If G/F is a tree then $G-V(F)$ is a forest. Since the size of the solution F is at most k, $|V(F)| \leq 2k$. As G is a simple graph, the number of edges of G with both of its end-points contained in $V(F)$ is at most $\mathcal{O}(k^2)$. Since $G-V(F)$ is a forest on at most $\mathcal{O}(k\ell)$ many vertices, the number of edges of G whose both end points are in $V(G) \setminus V(F)$ is bounded by $\mathcal{O}(k\ell)$. The number of edges which has exactly one end point in $V(F)$ is upper bounded by the maximum degree of G multiplied by the cardinality of $V(F)$ which is at most $\mathcal{O}(k^2+k\ell)$. Hence the bound on number of edges in G follows. □

We are now ready to prove the main theorem of this section.

Theorem 3.3.1. BOUNDED TREE CONTRACTION *has a kernel with $\mathcal{O}(k\ell)$ vertices and $\mathcal{O}(k^2+k\ell)$ edges.*

Proof. Given an instance (G,k,ℓ), the algorithm applies Reduction Rule 3.3.1 as long as it is applicable. If the number of vertices and number of edges in reduced instance are upper bounded by $\mathcal{O}(k\ell)$ and $\mathcal{O}(k^2+k\ell)$, then algorithm returns reduced instance. If either of this upper bounds fails then the algorithm returns a trivial NO instance.

We now argue running time and correctness of this algorithm. To apply Reduction Rule 3.3.1, algorithm needs to find a cut edge and check the number of vertices in connected components after removing that edge. This step can be performed in polynomial time. Each application of the reduction rule decreases the number of edges and thus it can be applied at most $|E(G)|$ many times. This implies that kernelization algorithm terminates in polynomial time. Lemma 3.3.1 implies that Reduction Rule 3.3.1 is safe. Let (G',k,ℓ)

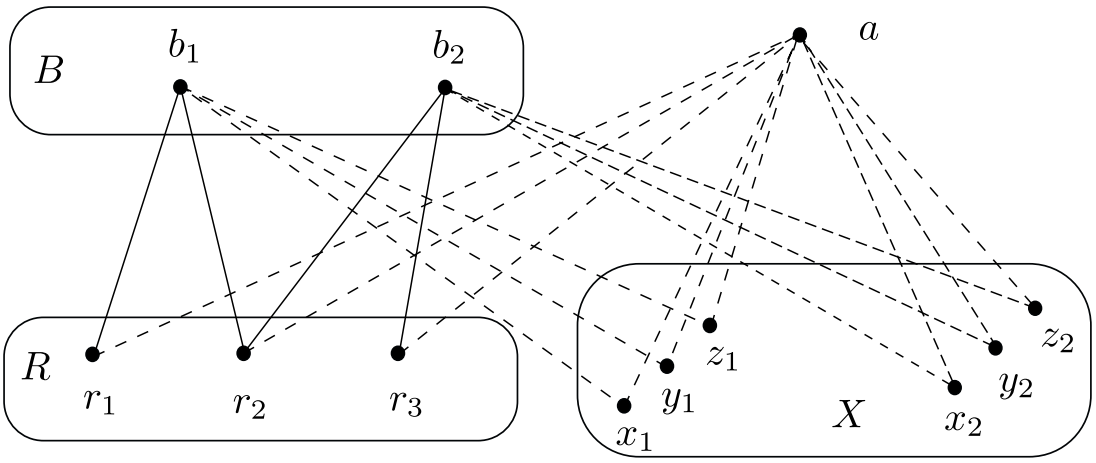


Figure 3.5: Kernel lower bound for BOUNDED TC.

be a reduced instance on which Reduction Rule 3.3.1 is not applicable. If G' does not have at most $\mathcal{O}(k\ell)$ vertices and $\mathcal{O}(k^2+k\ell)$ edges edges, algorithm correctly concludes that it is a NO instance. The correctness of this step follows from Lemma 3.3.2. Otherwise the algorithm returns a reduced instance as kernel. □

3.4 Kernel Lower Bound for BOUNDED TREE CONTRACTION

In this section we show that the kernelization algorithm presented in Section 3.3 for BOUNDED TC is optimal assuming NP $\not\subseteq$ coNP/poly. To prove this, we present a parameter preserving reduction which given an instance (G,R,B,k) of RED BLUE DOMINATING SET (RBDS), creates an instance (G',k',ℓ') of BOUNDED TC. Recall that in RBDS, given a bipartite graph $G(R,B)$ and an integer k, the task it to determine whether there exists a set of at most k vertices in R which dominates B.

Reduction. Let (G,R,B,k) be an instance of RBDS. We construct graph G' in the following way. See Figure 4.4. Initialize $V(G') = V(G)$ and $E(G') = \{br \mid b \in B, r \in$

R and $br \in E(G)\}$. Add a vertex a in $V(G')$ and for every vertex r in R, add an edge ar to $E(G')$. For every vertex b_i in B, add three new vertices x_i, y_i, z_i to $V(G')$ and edges b_ix_i, b_iy_i, b_iz_i to $E(G')$ *. Define set $X := \{x_i, y_i, z_i \mid b_i \in B\}$. For every vertex x in X, add an edge ax to $E(G')$. Set $k' = |B| + k$ and $\ell' = |R| + 3|B| - k$.

In the following lemma, we prove some structural properties of a solution for (G', k', ℓ').

Lemma 3.4.1. *Let (G', k', ℓ') be a* YES *instance of* BOUNDED TC. *There exists a solution $F^* \subseteq E(G')$ of size at most k' such that for each b_i in B one of the following holds.*

1. *b_i is in $W(t_a)$ or*
2. *x_i, y_i, z_i are in $W(t_a)$.*

Here, $W(t_a)$ is the witness set containing a in (G'/F^*)-witness structure of G'.

Proof. Let F be a set of edges of size at most k in G' such that G'/F is a tree with at most ℓ leaves. Let $\mathscr{W}$ be a T-witness structure of G' where $T = G'/F$. Let t_a be the vertex in $V(T)$ such that $W(t_a)$ contains vertex a. For a vertex b_i in B, if b_i is in $W(t_a)$ then the lemma holds. Consider a case when b_i is not in $W(t_a)$. There exists a vertex t_b, different from t_a, such that b_i is contained in $W(t_b)$. Similarly, consider vertices t_x, t_y and t_z such that x_i, y_i and z_i are contained in $W(t_x), W(t_y)$ and $W(t_z)$, respectively.

If neither of t_a or t_b is contained in set $\{t_x, t_y, t_z\}$, then no two vertices in $\{t_x, t_y, t_z\}$ can be same as only neighbors of x_i, y_i, z_i are a and b_i, and a witness set needs to be connected. But then, by construction, $T[\{t_a, t_x, t_y, t_z, t_b\}]$ is a cycle, contradicting the fact that T is a tree. Therefore, at least one of $\{t_x, t_y, t_z\}$ is same as t_a or t_b. Without loss of generality, let $t_x \in \{t_a, t_b\}$. This implies there is an edge t_at_b is in T. If t_y or t_z is not equal to t_a or t_b then there exist a cycle contradicting that T is a tree. Suppose, all t_x, t_y, t_z are same as t_a, then the second condition of the lemma is satisfied. Consider a case when at least one of t_x, t_y, t_z, say t_x, is not same as t_a, that is $t_x = t_b$. The only edges incident to x_i in G' are ax_i and bx_i.

*It is sufficient to add two vertices for each b_i in B. We add three vertices so that this proof can be re-used to prove similar results in case of CACTUS CONTRACTION problem in Chapter 4

This implies that $bx_i \in F$ and $W(t'_b) = W(t_b) \setminus \{x_i\}$ is connected. Since $ax_i \in E(G')$, set $W(t'_a) = W(t_a) \cup \{x_i\}$ is connected. Thus, replacing $W(t_b)$ by $W(t'_b)$ and $W(t_a)$ by $W(t'_a)$ in $\mathscr{W}$ yields another T-witness structure of G'. Furthermore, the spanning forest of the new witness structure, $F' = (F \setminus \{bx_i\}) \cup \{ax_i\}$ which has same cardinality as that of F. A similar swap can be carried out if $t_y = t_b$ or $t_z = t_b$. This concludes the proof. □

In the following lemma, we argue that the reduction is safe.

Lemma 3.4.2. (G,R,B,k) *is a* YES *instance of* RBDS *if and only if* (G',k',ℓ') *is a* YES *instance of* BOUNDED TC.

Proof. Let (G,R,B,k) be a YES instance of RBDS and S be a subset of R of size k such that S dominates every vertex in B. If S contains less than k vertices, then we take any of its superset of size exactly k. For each vertex b in B, we fix a vertex r_b in S such that b is neighbor of r_b in G. If there are multiple options for selecting r_b then we arbitrarily choose one of them. Let $F = \{br_b \mid b \in B\} \cup \{ar \mid r \in S\}$. Note that $|F| = |B| + k = k'$ and $G'[V(F)]$ is connected. Let T be the graph obtained from G' by contracting F. Let $\mathscr{W}$ be a T-witness structure of G'. Consider a vertex t_a such that a is in $W(t_a)$. Since all the edges in F are contracted to one vertex, set $S \cup B$ is also contained in $W(t_a)$. By construction, $R \cup X$ is an independent set in G'. No vertex in $(R \cup X) \setminus S$ is incident on edge which has been contracted. In other words, these vertices form singleton witness sets in $\mathscr{W}$. Since $R \cup X$ is an independent set in G', it follows that set $T_{RX} = \{t_v \mid v \in (R \cup X) \setminus S\}$ is an independent set in T of size $|R| + 3|B| - k = \ell'$. Moreover, for all v in X', $av \in E(T)$. Therefore, T is a star (which is a tree) with ℓ' leaves. This implies that F is a solution to (G',k',ℓ').

In the reverse direction, let (G',k',ℓ') be a YES instance of BOUNDED TC. By Lemma 3.4.1, there exists a solution F^* of size at most k' such that for every b_i in B, either b_i is in $W(t_a)$ or all of x_i, y_i, z_i are in $W(t_a)$. Here, $\mathscr{W}$ is the G'/F^*-witness structure of G' and t_a in $V(G'/F^*)$ such that vertex a is contained in witness set $W(t_a)$ in $\mathscr{W}$.

We partition vertices of B into two parts depending on whether they belong to $W(t_a)$ or not. Define $B_g = \{b_i \in B \mid b_i \in W(t_a)\}$. Let $R_a = R \cap W(t_a)$. Partition B_g into B_1 and B_2, depending on whether or not they have a neighbor in R_a. Formally, $B_1 = \{b_i \in B_g \mid N(b_i) \cap R_a \neq \emptyset\}$ and $B_2 = B_g \setminus B_1$. For a vertex b_i in B_2 at least one of x_i, y_i, z_i is present in $W(t_a)$ as there is no edge between b_i and a. Note that, by construction, x_i, y_i, z_i is not adjacent with b_j for $i \neq j$. This implies there exists a separate vertex for each b_i in B_2 which provides connectivity between a and b_i. Let XB_2 be set of vertices in $X \cap W(t_a)$ which provides adjacency between a and b_i for some b_i in B_2. For every b_i which is in $B \setminus B_g$, by Lemma 3.4.1, x_i, y_i, z_i are present in $W(t_a)$.

We can partition $W(t_a) \setminus \{a\}$ into following four parts: vertices in B (captured by B_g); vertices in R (captured by R_a); vertices in X which are present because corresponding b_i is not present (captured by $B \setminus B_g$); and vertices in X which are present because they are needed to provide connectivity between b_i and a (captured by XB_2). This implies $|B_g| + 3|B \setminus B_g| + |R_a| + |XB_2| + |\{a\}| \leq |W(t_a)|$.

We construct a solution S for RBDS by taking vertices in R_a and two more sets S_g and S_w. Informally, S_g dominates vertices in B_2 and S_w dominates vertices in $B \setminus B_g$. We construct S_g in following way. For every vertex b_i in B_2, arbitrary pick one of its neighbor in R and add it to S_g. Note that $|S_g| \leq |XB_2|$. We create another set S_w in the following way. Initialize S_w to an empty set. For each b in $B \setminus B_g$, we add an arbitrary neighbor of b in R to S_w. This implies $|S_w| \leq |B \setminus B_g|$. As cardinality of F^* is at most $k + |B|$, size of $W(t_a)$ is at most $|W(t_a)| \leq k + |B| + 1$.

Putting all inequalities together, we get $|R_a| + |S_g| + |S_w| \leq k$ and every vertex in B is dominated some vertex in $R_a \cup S_g \cup S_w$. This concludes the proof. □

We are now in position to present main result of this section.

Theorem 3.4.1. BOUNDED TREE CONTRACTION *does not admit a compression of size* $\mathcal{O}((k^2 + k\ell)^{1-\varepsilon})$, *for any* $\varepsilon > 0$ *unless* NP $\subseteq$ *coNP/poly.*

Proof. Assuming a contradiction, suppose BOUNDED TC admits a compression into $\Pi \subseteq \Sigma^*$ with bitsize in $\mathcal{O}((k^2+k\ell)^{1-\varepsilon})$, for some $\varepsilon > 0$. This implies that there exists an algorithm $\mathscr{A}$ which takes an instance $I = (G,k,\ell)$ of BOUNDED TC and in polynomial time returns an equivalent instance I' of Π with $|I'| \in \mathcal{O}((k^2+k\ell)^{1-\varepsilon})$.

Let (G,R,B,k) be an instance of RBDS, where G is a graph on n vertices. Using the reduction described, we create an instance (G,k',ℓ') of BOUNDED TC with $|V(G'_D)| \in \mathcal{O}(n)$, $|E(G'_D)| \in \mathcal{O}(n^2)$, $k' = k \leq |R| \in \mathcal{O}(n)$ and $\ell' = |B| + k \in \mathcal{O}(n)$. On the instance (G,k',ℓ') we run the algorithm $\mathscr{A}$ to obtain an instance I of Π such that $|I| \in \mathcal{O}((k'^2+k'\ell')^{1-\varepsilon})$. But then we have obtained a compression of size $\mathcal{O}(n^{2-\varepsilon})$ for RBDS, contradicting Proposition 2.3.2. □

Corollary 3.4.1. BOUNDED TREE CONTRACTION *does not admit a kernel of size* $\mathcal{O}((k^2+k\ell)^{1-\varepsilon})$*, for any* $\varepsilon > 0$ *unless* NP $\subseteq$ coNP/poly.

3.5 Lossy Kernel for TREE CONTRACTION

In this section, we present an α-lossy kernel of polynomial size for TREE CONTRACTION. We define parameterized minimization version of TREE CONTRACTION problem in the following way.

$$\mathrm{TC}(G,k,F) = \begin{cases} \infty & \text{if G/F is not a tree} \\ \min\{|F|, k+1\} & \text{otherwise} \end{cases}$$

We assume that input graph is connected as otherwise one can not obtain a tree only by contractions. If G has at most $k+3$ vertices then we already have a kernel of desired size. We assume that input graph has at least $k+3$ vertices. By definition of optimization problem, for a set of edges F, if G/F is a tree then maximum value $\mathrm{TC}(G,k,F)$ is $k+1$. Hence any spanning tree of G is a solution of cost $k+1$. We call it a *trivial solution* for

given instance. We denote a complete graph on four vertices by K_4. One need to contract at least two edges to obtain a tree from K_4. We call $(K_4, 1)$ as *trivial instance* of TREE CONTRACTION. If $\mathrm{OPT}(G,k) = k+1$ then we can return $(K_4, 1)$ as its α-lossy kernel. Note that for any c-factor solution of $(K_4, 1)$, solution lifting algorithm can return a trivial solution for original problem which is of cost $k+1$.

Lemma 3.2.2 states that a connected graph G is k-contractible to a tree if and only if each of its 2-connected components is contractible to a tree using at most k edge contractions in total. Cycles in different 2-connected components are edge-disjoints and hence contracting an edge in one component does not eliminate cycles in another component. If the number of 2-connected components in the input graph which are not a tree (more specifically, an edge) is more than $k+1$ then we can safely conclude that optimum solution for given instance is at least $k+1$. In this case we can return a trivial instance. Note that we do not guess the number of edges that needs to be contracted in each 2-connected component. We compute kernel for each 2-connected component using the budget of k. The output of our kernelization algorithm is a union of the kernels for each 2-connected component. We present first reduction rule which eliminate long path connecting two different 2-connected components.

Reduction Rule 3.5.1. *If uv is a cut-edge in G then contract the edge uv. The resulting instance is (G', k), where $G' = G/\{uv\}$.*

Informally speaking, since edge uv is a cut-edge, it is not a part of any cycle. We do not need to contract uv to destroy any cycle. This implies that edge uv is irrelevant with respect to any solution and can safely be contracted.

Lemma 3.5.1. *Reduction rule 3.5.1 is* 1*-safe.*

Proof. Consider a solution F' for (G', k). If $|F'| \geq k+1$, solution lifting algorithm returns a spanning tree F of G. If $|F'| \leq k$ then solution lifting algorithm returns $F = F'$. If $|F'| \geq k+1$ then for a spanning tree F of G, $\mathrm{TC}(G,k,F) = k+1$. Hence in this case,

$\mathrm{TC}(G,k,F) = k+1 = \mathrm{TC}(G',k,F')$. Consider a case when $|F'| \leq k$. Let $\mathscr{W}'$ be a T'-witness structure of G' where $T' = G'/F'$. Let u^* be the vertex new vertex added while contracting edge uv. Consider vertex t^* in $V(T')$ such that u^* in $W(t^*)$. Define set $W(t) := (W(t^*) \setminus \{u^*\}) \cup \{u,v\}$. Let $\mathscr{W}_1$ be a witness strucure obtained from $\mathscr{W}'$ by removing $W(t^*)$ and adding $W(t)$. Notice that $\mathscr{W}_1$ partitions $V(G)$ and for each W in $\mathscr{W}_1$, $G[W]$ is connected. Let T_1 be a graph such that $\mathscr{W}$ is a T_1-witness structure of G. Note that T_1 can be obtained from G by contracting all edges in $F' \cup \{uv\}$. This implies T_1 can be obtained from G' by contracting all edges in F'. Hence T_1 is a tree. This implies that G is $(|F'|+1)$-contractible to a tree. We argue that G is in fact $|F'|$-contractible to a tree.

Note that uv is a cut-edge in $G[W(t)]$. Let C_u and C_v be the connected components in $G[W(t)] - \{uv\}$ containing u and v, respectively. Further, let $W_u = V(C_u)$, $W_v = V(C_v)$. Consider a witness structure $\mathscr{W}$ of G obtained from $\mathscr{W}_1$ by removing $W(t)$ and adding W_u and W_v. Notice that $\mathscr{W}$ partitions $V(G)$ and for each W in $\mathscr{W}$, $G[W]$ is connected. Let T be a graph obtained by contracting all witness sets in $\mathscr{W}$. In other words, $\mathscr{W}$ is a T-witness structure of G. Note that G is $|F'|$-contractible to T. Let t_u, t_v in $V(T)$ to be the vertices such that $W(t_u) = W_u$ and $W(t_u) = W_u$. Note that T is obtained from T_1 by removing t and adding two vertices t_u, t_v. Edges incident on t_u, t_v are determined by witness structure $\mathscr{W}$ of G. We now argue that T is a tree. We claim that for any t' in $N_{T_1}(t)$, at most one of $t_u t'$ and $t_v t'$ is present in $E(T)$. Assume that both these edges exists for some t'. Let w be a vertex in $W(t')$. There exist a path between w and u which is entirely contained in $W(t') \cup W(t_u)$ and hence does not contain v. There also exists a path between w and v which is contained in $W(t') \cup W(t_u)$ and hence does not contain u. This contradicts the fact that uv is a cut-edge in G. Hence, T is a tree as no cycle has been introduced while removing t and adding t_u, t_v. This implies that G can be contracted to a tree by contracting all edges in F'. Hence, $\mathrm{TC}(G,k,F) = \mathrm{TC}(G',k,F')$.

We now argue that $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$. Let F be an optimum solution for (G,k). By Observation 3.2.2, $G/(F \cup \{uv\})$ is also a tree. Note that $G/(F \cup \{uv\}) = (G/\{uv\})/(F \setminus$

$\{uv\}) = G'/(F \setminus \{uv\})$. Hence G'/F is a tree. Since $|F \setminus \{uv\}| \leq |F|$, we can conclude that $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$.

Combining these two inequalities, we get $\frac{\mathrm{TC}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathrm{TC}(G',k,F')}{\mathrm{OPT}(G',k)}$ which concludes the proof. □

Exhaustive application of above reduction rule eliminates all cut edges in G. In rest of the section, we focus on 2-connected component of G. We assume that the input graph is 2-connected.

We now present a relationship between TREE CONTRACTION and CONNECTED VERTEX COVER. Consider a 2-connected graph G is contracted to a tree T. If every leaf corresponds to singleton witness set and vice-versa, the vertices in witness sets of non-singleton witness sets forms a connected vertex cover of graph.

Lemma 3.5.2. *If a 2-connected graph G is k-contractible to a tree, then G has a connected vertex cover of size at most $2k$.*

Proof. As G is k-contractible to a tree, there exists a (minimal) set of edges F such that $|F| \leq k$ and $T = G/F$ is a tree. Let $\mathscr{W}$ be a T-witness structure of G and $\mathscr{W}'$ denote a set of non-singleton sets in $\mathscr{W}$. Let X denote the set of vertices of G which are contained in $W(t)$ for some t in $\mathscr{W}'$. By Observation 3.2.3, we can assume that every leaf of T corresponds to a singleton witness set, and vice versa. Let L be the set of leaves of T. Then, $I = \{v \in V(G) \mid v \in W(t), t \in L\}$ is an independent set in G. Thus, X is a vertex cover of G. We have $|X| \leq 2k$ as every vertex in X has an edge incident on it which is in F and $|F| \leq k$. Finally, since the set of non-leaves of a tree induces a subtree, it follows that $G[X]$ is connected. □

We present following reduction rule which quickly returns a lossy kernels for graph which has large connected vertex cover.

Reduction Rule 3.5.2. *Given an instance (G,k), apply 2-factor approximation algorithm to compute a connected vertex cover X of G. If size of X is greater than $4k$ then return $(K_4,1)$.*

Lemma 3.5.3. *Reduction Rule 3.5.2 is 1-safe.*

Proof. Let (G,k) be an instance such that Reduction Rule 3.5.2 returns $(K_4,1)$ when applied on it. Solution lifting algorithm returns a spanning tree F of G.

Note that for a set of edges F', if K_4/F' is a tree then F' contains at least two edges. This implies $\mathrm{TC}(K_4,1,F') = 2$ and $\mathrm{OPT}(K_4,1) = 2$.

Since a 2-factor approximation algorithm returns a set of size strictly more than $4k$, size of minimum connected vertex cover of G is at least $2k$. But by Lemma 3.5.2, if G is k-contractible to a tree than it has a connected vertex cover of size at most $2k$. Hence for any set of edges F^* if G/F^* is a tree than size of F^* is at least $k+1$. This implies $\mathrm{OPT}(G,k) = k+1$. For a spanning tree F of G, $\mathrm{TC}(G,k,F) = k+1$.

Combining these values, we get $\frac{\mathrm{TC}(G,k,F)}{\mathrm{OPT}(G,k)} = \frac{k+1}{k+1} = \frac{2}{2} = \frac{\mathrm{TC}(K_4,1,F')}{\mathrm{OPT}(K_4,1)}$. This implies if F' is c-factor approximate solution for $(K_4,1)$ then F is 1-factor approximate solution for (G,k). This concludes the proof. □

We partition vertices of G into the following three parts: high degree vertices (H), independent set (I) and rest of the graph (R) (See Figure 3.6). These sets are defined as follows.

$$H = \{u \in V(G) \mid d(u) \geq 2k+1\}$$

$$I = \{v \in V(G) \setminus H \mid N(v) \subseteq H\}$$

$$R = V(G) \setminus (H \cup I)$$

By constructing H, we identify vertices which are in any connected vertex of size at most $2k$. Upper bound on connected vertex cover also provides upper bound on size of H.

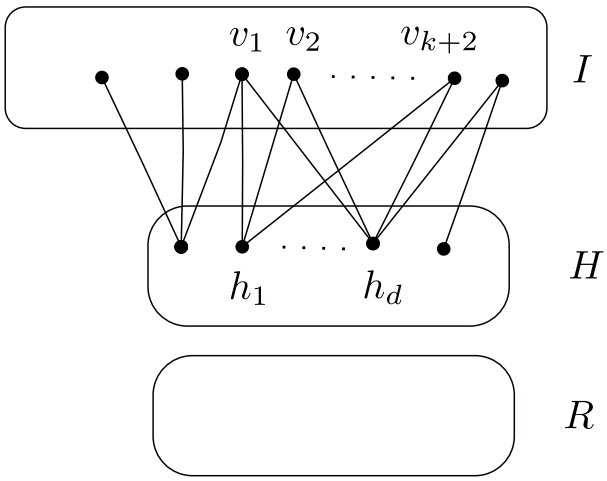


Figure 3.6: Partition of input graph. Please see Reduction 3.5.3

Set I contains vertices whose open neighborhood is contained in H and hence $G[I]$ is an independent set. Size of rest of the graph, $G[R]$, can be bounded by polynomial function of k. Inability to bound the number of vertices in I can be seen as a reason that TREE CONTRACTION does not have a polynomial kernel. We use *lossy reduction rules* to bound the cardinality of I.

Two vertices are said to be false twins if their open neighbourhood is same.

Reduction Rule 3.5.3. *If there is a vertex v in I that has at least $k+1$ false twins, then delete v. The resultant instance is $(G-\{v\},k)$.*

In the following lemma, we argue that v and its $k+1$ false twins *forces* some vertices to be in one witness set. By applying Reduction Rule 3.5.3, we ensure that we store only enough number of vertices in I which enforces such condition and delete rest of the vertices.

Lemma 3.5.4. *Reduction Rule 3.5.3 is* 1*-safe.*

Proof. Consider a solution F' of reduced instance (G',k) i.e. F' is a set of edges such that $G'/F' = T'$ is a tree. If $|F'| \geq k+1$, then the solution lifting algorithm returns a spanning tree F of G, otherwise it returns $F = F'$. We show that this solution lifting algorithm with the reduction rule constitutes a strict 1-approximate polynomial time preprocessing

algorithm. If $|F'| \geq k+1$ then $\mathrm{TC}(G',k,F') = k+1$ and $\mathrm{TC}(G,k,F) = k+1$ for a spanning tree F of G. In this case, $\mathrm{TC}(G,k,F) = \mathrm{TC}(G',k,F') = k+1$. Consider a case when $|F'| \leq k$. Let $\mathscr{W}'$ be a T'-witness structure of G'. Without loss of generality, we assume that F' satisfies the property mentioned in Observation 3.2.3. Notice that each edge in F' can be incident on at most one false twins of v. As v has at least $k+1$ false twins, one of these twins, say u, is not in $V(F')$. In other words, there is a vertex t in T' such that $W'(t) = \{u\}$. By Observation 3.2.3, t is a leaf in T'. Let t' denote the unique neighbor of t in T'. By Observation 3.2.1, $N_{G'}(u) \subseteq W'(t')$. Since $N_{G'}(u) = N_G(u) = N_G(v)$, all the vertices in $N_G(v)$ are in $W'(t')$. Define the partition $\mathscr{W}$ of $V(G)$ obtained from $\mathscr{W}'$ by adding a new set $\{v\}$. Let T be a graph obtained from G by contracting all edges in F. In other words, $\mathscr{W}$ is a T-witness structure of G. Note that T can be obtained from T' by adding a new vertex t_v as a leaf adjacent to t'. This implies that G/F is a tree. Hence, $\mathrm{TC}(G,k,F) \leq \mathrm{TC}(G',k',F')$.

Consider an optimum solution F^* for (G,k). If $|F^*| \geq k+1$ then $\mathrm{OPT}(G,k) = k+1 \geq \mathrm{OPT}(G',k')$. Otherwise, $|F^*| \leq k$ and let $T = G/F^*$. Let $\mathscr{W}^*$ be a T-witness structure of G. If there is a leaf t in T such that $W^*(t) = \{v\}$, then F^* is also a solution for (G',k') and $\mathrm{OPT}(G',k') \leq \mathrm{OPT}(G,k)$. Otherwise, as v has at least $k+1$ false twins, one of these twins, say u, is not in $V(F^*)$. That is, there is a leaf t in T such that $W^*(t) = \{u\}$. Define the partition $\mathscr{W}'$ of $V(G)$ obtained from $\mathscr{W}^*$ by replacing u by v and v by u. Then, the set F' of edges of G, obtained from F by replacing the edge xv with the edge xu for each x, is also an optimum solution for (G,k). Further, it is a solution for (G',k') and therefore, $\mathrm{OPT}(G',k') \leq \mathrm{OPT}(G,k)$. Hence, $\frac{\mathrm{TC}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathrm{TC}(G',k',F')}{\mathrm{OPT}(G',k')}$. □

We now present the reduction rule which introduces *lossy-ness*. Given $\alpha > 1$, let d be the minimum integer such that $\frac{d}{d-1} \leq \alpha$. In other words, $d = \lceil \frac{\alpha}{\alpha-1} \rceil$. If we add an extra edge for every $d-1$ edges in a solution, we obtain another solution which has cardinality at most $\frac{d}{d-1} \leq \alpha$ time the cardinality of original solution. We now state our next reduction rule.

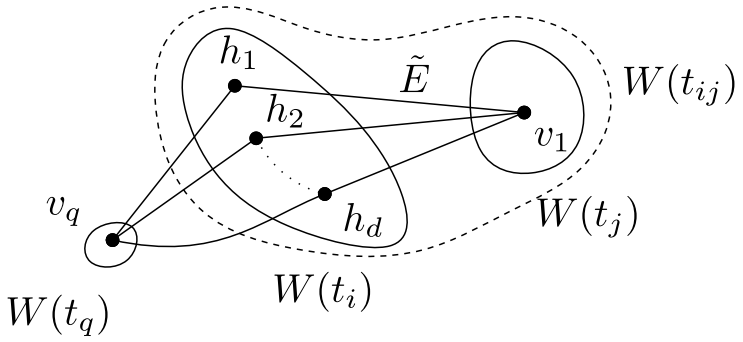


Figure 3.7: Please refer to Lemma 3.5.5

Reduction Rule 3.5.4. *If there are vertices $v_1, v_2, \ldots, v_{k+2} \in I$ and $h_1, h_2, \ldots, h_d \in H$ such that $\{h_1, \ldots, h_d\} \subseteq N(v_i)$ for each $i \in [k+2]$ then contract all edges in $\tilde{E} = \{v_1 h_i \mid i \in [d]\}$ and reduce the parameter by $d-1$. The resulting instance is $(G/\tilde{E}, k-d+1)$.*

The above rule can be applied in $\mathcal{O}(|H|^d \cdot n^{\mathcal{O}(1)})$ time, by considering each subset of H of cardinality at most d. As discussed in the paragraph following Reduction Rule 3.5.4, set of vertices $\{v_1, v_2, \ldots, v_{k+2}\}$ *forces* the vertices $\{h_1, h_2, \ldots, h_d\}$ to be in one witness set. But subgraph of G which induced on set $H' = \{h_1, h_2, \ldots, h_d\}$ may not be connected. We provide this connectivity by adding the vertex v_1 to set H'. To simplify G, we contract $H' \cup \{v_1\}$ into a single vertex and reduce the budget by $d-1$. Notice that we contract d many edges and reduce the budget by $d-1$. In other words, for every $d-1$ edges in optimum solution, we are using d edges. For every solution S' of modified graph, we can obtain a solution S of original graph which has cardinality at most $\frac{d}{d-1}$ times that of S'. We prove these things formally in the following lemma.

Lemma 3.5.5. *Reduction Rule 3.5.4 is α-safe.*

Proof. Consider a solution F' of the reduced instance (G', k'). If $|F'| \geq k'+1$, then the solution lifting algorithm returns a spanning tree F of G, otherwise it returns $F = F' \cup \tilde{E}$. We show that this solution lifting algorithm with the reduction rule constitutes a strict α-approximate polynomial time preprocessing algorithm. First, we prove that $\mathrm{TC}(G,k,F) \leq \mathrm{TC}(G',k',F') + d$. If $|F'| \geq k'+1$ then $\mathrm{TC}(G',k',F') = k'+1$. In this case, F is a spanning

tree of G and $\mathrm{TC}(G,k,F) = k+1 = k'+d = \mathrm{TC}(G',k',F')+d-1$. Consider the case when $|F'| \leq k'$. Let $\mathscr{W}'$ be a G'/F'-witness structure of G'. Let w denote the vertex in $V(G') \setminus V(G)$ obtained by contracting $\tilde{E}$. Let $W'(t_1)$ be a witness set in $\mathscr{W}'$ which contains w. Define $W_1 = (W'(t_1) \cup \{v_1, h_1, h_2, \ldots, h_d\}) \setminus \{w\}$. Let $\mathscr{W}$ be a witness structure obtained from $\mathscr{W}'$ by removing $W'(t_1)$ and adding W_1. Formally, $\mathscr{W} = (\mathscr{W}' \cup \{W_1\}) \setminus \{W'(t_1)\}$. Note that $V(G) \setminus \{v_1, h_1, h_2, \ldots, h_d\} = V(G') \setminus \{w\}$ and hence $\mathscr{W}$ is partition of $V(G)$. Further, $G[W_1]$ is connected as $G'[W'(t_1)]$ is connected as a spanning tree of $G'[W'(t_1)]$ along with $\tilde{E}$ is a spanning tree of $G[W_1]$. Also, $|W_1| = |W'(t_1)| + d$ and any vertex which is adjacent to w in G' is adjacent to at least one vertex in $\{v_1, h_1, h_2, \ldots, h_d\}$ in G. Thus, $G/F = G'/F'$. Note that the size of F is at most $|F'| + d \leq k' + d = k - d + 1 + d = k + 1$. Hence $\mathrm{TC}(G,k,F) = |F|$. This implies, $\mathrm{TC}(G,k,F) = |F| = k' + d \leq \mathrm{TC}(G',k',F') + d$.

We now show that $\mathrm{OPT}(G',k') \leq \mathrm{OPT}(G,k) - (d-1)$. Let F^* be an optimum solution for (G,k) and $\mathscr{W}$ be a T-witness structure of G, where $T = G/F^*$. If $|F^*| \geq k+1$, then $\mathrm{OPT}(G,k) = k+1 = k'+d \geq \mathrm{OPT}(G',k') + d - 1$. Now consider the case when $|F^*| \leq k$. Notice that every edge in F^* can be incident on at most one vertex in $\{v_1, v_2, \ldots, v_{k+2}\}$. There is at least one vertex, say v_q, in $\{v_1, v_2, \ldots, v_{k+2}\}$ which is not in $V(F^*)$. In other words, there exists a vertex in T such that $W(t_q) = \{v_q\}$. By Observation 3.2.3, t_q is a leaf in T. Let t_i be the unique neighbhor of t_q in T. By Observation 3.2.1, $N(v_q)$ is in a witness set $W(t_i)$. This implies that $\{h_1, h_2, \ldots, h_d\}$ are in $W(t_i)$. See Figure 3.7. If $v_1 \in W(t_i)$ then $F' = F^* \setminus \tilde{E}$ is a solution to (G',k') and so $\mathrm{OPT}(G',k') \leq |F'| = |F^*| - d = \mathrm{OPT}(G,k) - d$. Consider the case when v_1 is not $W(t_i)$ and let $t_j \in V(T)$ be the vertex such that $v_1 \in W(t_j)$. By definition of witness sets, t_i and t_j are adjacent in T. Define another partition $\mathscr{W}' = \mathscr{W} \cup \{W(t_{ij})\} \setminus \{W(t_i), W(t_j)\}$ of $V(G)$ where $W(t_{ij}) = W(t_i) \cup W(t_j)$. Let F be a solution associated with this witness structure. Graph $G[W(t_{ij})]$ is connected as t_i, t_j are adjacent in T. As $|W(t_i)| - 1 + |W(t_j)| - 1 = (|W(t_{ij})| - 1) - 1$, $\mathscr{W}'$ is a G/F-witness structure of G and $|F| = |F^*| + 1$. In particular, G/F is the tree obtained from G/F^* by contracting the edge $t_i t_j$. Graph induced on vertices which are end points of edges in $\tilde{E}$ is a star. Hence we can find spanning tree of $G[W(t_i)]$ which contains $\tilde{E}$. Hence,

without loss of generality $\tilde{E} \subseteq F$ and thus $F' = F \setminus \tilde{E}$ is a solution to (G', k'). Therefore, $\mathrm{OPT}(G', k') \leq |F'| = |F^*| + 1 - d = \mathrm{OPT}(G, k) - d + 1$. Combining these bounds, we have $\frac{\mathrm{TC}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathrm{TC}(G',k',F')+d}{\mathrm{OPT}(G',k')+(d-1)} \leq \max\left\{\frac{\mathrm{TC}(G',k',F')}{\mathrm{OPT}(G',k')}, \alpha\right\}$. □

We now prove that if G is k-contractible to a tree and none of the Reduction Rules mentioned above are applicable on instance (G,k), then the number of vertices in G is bounded by a function of k.

Lemma 3.5.6. *Let (G,k) be an instance of* TREE CONTRACTION *on which none of Reduction Rules 3.5.2; 3.5.3 and 3.5.4 are applicable. If G is 2-connected and k-contractible to a tree then the number of vertices in G is at most $\mathcal{O}((2k)^{d+1} + k^2)$.*

Proof. We bound cardinalities of sets H, I and R separately in order to bound $V(G)$. By Lemma 3.5.2, G has a connected vertex cover S of size at most $2k$. As H is the set of vertices of degree at least $2k+1$, $H \subseteq S$ and so $|H| \leq 2k$. Every vertex in R has degree at most $2k$. As $S \cap R$ is a vertex cover of $G[R]$, number of edges in $G[R]$ is $\mathcal{O}(k^2)$. Also, by the definition of I, every vertex in R has a neighbor in R and hence there are no isolated vertices in $G[R]$. Thus, size of R is $\mathcal{O}(k^2)$. Finally, we bound the size of I. For every set $H' \subseteq H$ of cardinality less than d, there are at most $k+1$ vertices in I which have H' as their neighborhood. Otherwise, Reduction Rule 3.5.3 would have been applied. Hence, there are at most $(k+1) \cdot \binom{2k}{d-1}$ vertices in I which have degree less than d. Further, for a d-size subset H' of H, there are at most $k+1$ vertices in I which contain H' in their neighborhood. Otherwise, Reduction Rule 3.5.4 would have been applied. As a vertex in I of degree at least d is adjacent to all vertices in at least one such subset of H, there are at most $(k+1)\binom{2k}{d}$ vertices of I of degree at least d. Therefore, $|I|$ is $\mathcal{O}((2k)^{d+1})$. □

We now present main result of this section.

Theorem 3.5.1. TREE CONTRACTION *admits a strict PSAKS with $\mathcal{O}((2k)^{\lceil \frac{\alpha}{\alpha-1} \rceil + 2} + k^3)$ vertices.*

Proof. For a given instance (G,k), kernelization algorithm exhaustively apply Reduction Rule 3.5.1. If number of 2-connected components which contains a cycle is more than $k+1$ then the algorithm returns a trivial instance as a lossy kernel. Otherwise, algorithm computes α-lossy kernel for each of 2-connected components separately. If algorithm finds trivial instance as lossy kernel for any of 2-connected component then it returns a trivial instance as a lossy kernel for entire graph.

For a 2-connected component, say C, the algorithm creates an instance $(G[C],k)$. Let I,H,R be partition of $V(C)$ as defined before Reduction Rule 3.5.3. It is possible that cut vertices in C are part of I and may get deleted while computing a lossy kernel. We avoid this by marking these vertices. Since there are at most k many 2-connected components in G, C has at most $k-1$ many cut vertices. Marking these vertices increase the size of reduced instance by $\mathcal{O}(k)$.

Given $\alpha > 1$, the algorithm fixes $d = \lceil \frac{\alpha}{\alpha-1} \rceil$. It applies Reduction Rule 3.5.2; 3.5.3; and 3.5.4 exhaustively on instance $(G[C],k)$. If reduced graph G' has more than $\mathcal{O}((2k)^{d+1} + k^2)$ vertices, then by Lemma 3.5.6, graph G' is not k-contractible to a tree. This implies that $\mathrm{OPT}(G[C],k)$ is $k+1$. In this case the algorithm returns trivial instance as a lossy kernel. Otherwise the reduced graph has $\mathcal{O}((2k)^{d+1} + k^2)$ vertices. There are at most k many 2-connected components, and summing over each component, the reduced graph has at most $\mathcal{O}((2k)^{d+2} + k^3)$ vertices. The correctness of the algorithm follows from Lemma 3.5.1; 3.5.3; 3.5.4; and 3.5.5 □

3.6 Conclusion

In this chapter, we analyse the structure of the family of paths that allows PATH CONTRACTION to admit a polynomial kernel but forbids TREE CONTRACTION. Apart from solution size k, we make number of leaves, ℓ, as additional parameter to bridge the gap between kernels of these two problem. We call this problem as BOUNDED TREE CONTRACTION.

We present a polynomial kernel for this problem. We also prove that this kernel is optimal under certain complexity assumption. In this chapter, we also present a lossy kernel of polynomial size for TREE CONTRACTION problem.

Chapter 4

Cactus Contraction

4.1 Introduction

In this chapter, we study a problem of contracting given graph into a graph class which is superset of trees. A *cactus* is a connected graph in which every edge is a part of at most one cycle. We generalize techniques used in Chapter 3 to present a lossy kernel and an FPT algorithm for following problem.

CACTUS CONTRACTION	**Parameter:** k
Input: A graph G and an integer k	
Question: Is it possible to obtain a cactus from G with at most k edge contractions?	

We prove that this problem is NP-Complete by presenting a reduction from RED BLUE DOMINATING SET. This reduction also implies that CACTUS CONTRACTION does not have a polynomial kernel when parameterized by solution size. This raises two questions similar to the ones which we addressed in Chapter 3 regarding TREE CONTRACTION.

−*What are the additional parameter we need to add to obtain a polynomial kernel for* CACTUS CONTRACTION*?*

−*If we allow small loss in accuracy, can we get a polynomial kernel for* CACTUS CON-

TRACTION*?*

In case of TREE CONTRACTION, we answered first question by describing polynomial kernel parameterized by solution size and the number of leaves in resulting tree. We deploy similar approach for CACTUS CONTRACTION. In case of cactus, we define *number of leaves* using *block decomposition* of given cactus. (Formal definitions are in Section 4.2.) A block is a maximal 2-connected subgraph. A block in a cactus can be either a cycle or an edge or an isolated vertex. Block decomposition is an auxiliary graph which encodes how blocks and cut vertices of graph intersects. This auxiliary graph of any connected graph is a tree. The *number of leaves* in a cactus is defined as the number of leaves in its block decomposition. We formally define BOUNDED CACTUS CONTRACTION problem in the following way.

BOUNDED CACTUS CONTRACTION(BOUNDED CC) **Parameter:** $k+\ell$
Input: A graph G and integers k, ℓ
Question: Is it possible to obtain a cactus with at most ℓ leaves from graph G with at most k edge contractions?

In Section 4.3, we present a kernel of size $\mathcal{O}(k^2+k\ell)$ for this problem. Although the approach involved in designing this kernel is similar to that of BOUNDED TREE CONTRACTION, technical analysis is considerably more in this case. In Section 4.4, we prove that this kernel is optimal under certain complexity assumption.

We answer the second question by designing lossy kernel of polynomial size for CACTUS CONTRACTION. We prove that given a graph G on n vertices, an integer k and an approximation parameter $\alpha > 1$, there is an algorithm that runs in $n^{\mathcal{O}(1)}$ time and outputs a graph G' on $\mathcal{O}((2k)^{2\lceil \frac{\alpha}{\alpha-1} \rceil+1}+k^5)$ vertices and an integer k' such that for every $c > 1$, a c-approximate solution for (G',k') can be turned into a $(c\alpha)$-approximate solution for (G,k) in $n^{\mathcal{O}(1)}$.

Heggerners et al. presented FPT algorithms for TREE-CONTRACTION and PATH-CONTRACTION.

We present an FPT algorithm for CACTUS-CONTRACTION additing it to the small list of graph classes $\mathscr{F}$ for which $\mathscr{F}$-CONTRACTION is known to be FPT. We present an algorithm running in time $c^k n^{\mathcal{O}(1)}$, where c is a fixed constant. Our algorithm builds upon ideas presented in [55], but requires a more involved structural analysis of the graph.

Results presented in Section 4.3 and 4.4 are from [1]. Lossy kernel presented in Section 4.5 for CACTUS CONTRACTION is based on [69]. The FPT algorithm described in Section 4.6 can be found in [70].

4.2 Preliminaries

In this section, we mention some known properties of cactus. As in case of tree contraction, we define graph class called bounded cactus contraction. We mention some property of these graph classes. We later mention simplifying assumption that helps us to concentrate on 2-connected components of while finding a lossy kernel and designing an FPT algorithm.

A *block* is a connected maximal connected subgraph which is 2-connected. A block in a graph is either an induced maximal 2-connected subgraph or an edge or an isolated vertex. Two distinct blocks in graph can intersect in at most one vertex. A vertex contained in at least two block must be a cut-vertex in graph. Let K be the set of cut-vertices and $\mathscr{B}$ be the set of blocks in G. A *block-decomposition* of G is a bipartite graph $\mathscr{D}$ with the vertex set $K \uplus \mathscr{B}$. Furthermore, $aB \in E(\mathscr{D})$ for $a \in K$ and $B \in \mathscr{B}$ if and only if $a \in V(B)$. Here, we slightly abuse the notation, and use $\mathscr{B}$ to denote the set of blocks in G as well as vertices corresponding to the blocks of G in $\mathscr{D}$. It is known that a block decomposition of a connected graph is unique and is a tree [29, Proposition 3.1.2]. For the sake of clarity, we call vertices in $\mathscr{D}$ as nodes. See Figure 4.1. The number of leaves of cactus is defined as the number of leaves in its block decomposition.

Since every edge in cactus is part of at most one cycle, if G is a cactus then a block of G is

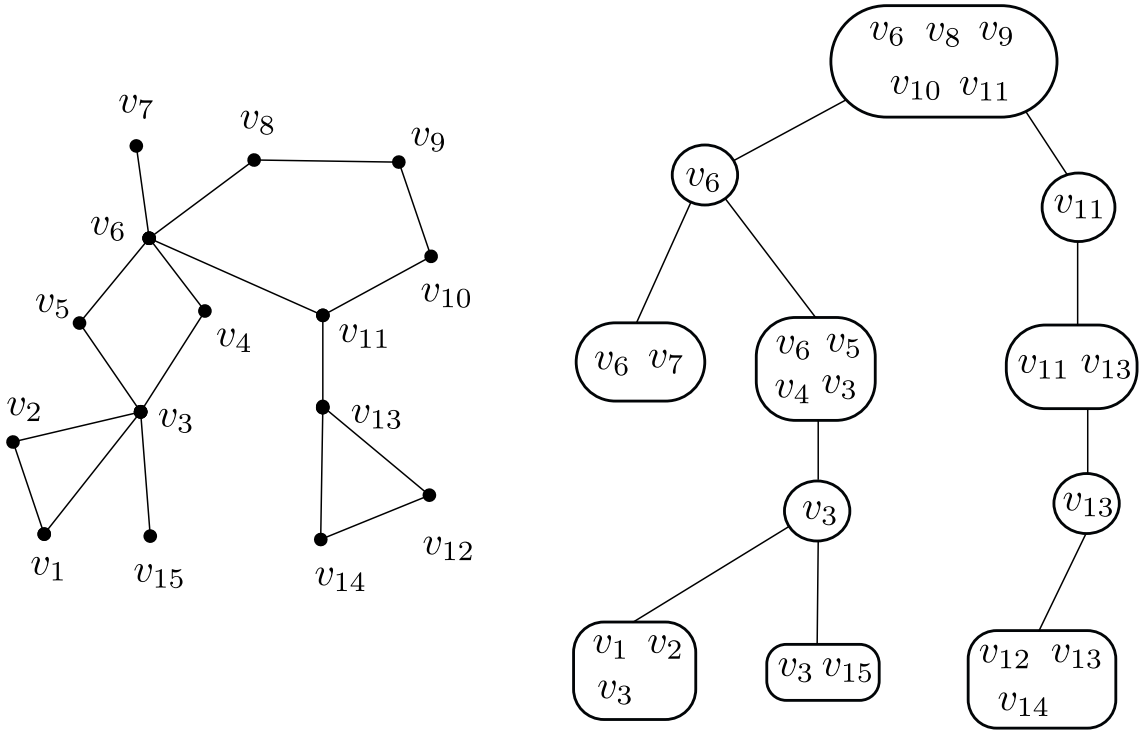


Figure 4.1: Cactus graph and its block decomposition.

either a cycle or an edge. We define pendant cycle in cactus.

Definition 4.2.1 (Pendant Cycle)**.** *For a cactus T, a cycle C is called a pendant cycle if it contains exactly one cut vertex.*

For example, in Figure 4.1, cycles $\{v_1, v_2, v_3\}$ and $\{v_{12}, v_{13}, v_{14}\}$ are pendent cycles.

Observation 4.2.1. *The following statements hold for a cactus T.*

1. $|E(T)| \leq 2|V(T)|$
2. *The vertices of T can be properly colored using 3 colors.*
3. *Every vertex of degree at least 3 is a cut-vertex.*

Proof. For a given cactus T, let $\mathscr{D}$ be its block decomposition.

(1) We prove this using the induction on number of blocks in cactus graph. Our induction hypothesis is: if number of blocks in T is strictly less than q then $|E(T)| \leq 2|V(T)|$. For

base case, consider when T has exactly one block. In this case, T is either an edge or a cycle. In either case, $|E(T)| \leq 2|V(T)|$.

Consider T which has q blocks. Let a block B corresponds to a leaf in $\mathscr{D}$. For this block, $|E(B)| \leq 2|V(B)| - 2$ as B is either an edge or a cycle. Let u be the unique cut vertex in B. Consider a cactus $T_1 = T - (V(B) \setminus \{u\})$. Since T_1 has $q-1$ blocks, by induction hypothesis, $|E(T_1)| \leq 2|V(T_1)|$.

Any edge in T is present in exactly one block. Hence $|E(T)| = |E(T_1)| + |E(B)|$. By construction, $|V(T)| = |V(T_1)| + |V(B)| - 1$ as u is counted in $V(T_1)$ and also in $V(B)$. Substituting upper bounds for $|E(T_1)|$ and $|E(B)|$, we get $|E(T)| \leq 2|V(T)|$.

(2) We again use induction on number of blocks in cactus to prove the statement. Our induction hypothesis is: if number of blocks in T is strictly less than q then T can be properly colored using at most 3 colors. For base case, consider a case when T has exactly one block. In this case, T is either an edge or a cycle. In either case, T can be properly colored using at most 3 colors.

Consider a cactus T which has q blocks. Let a block B corresponds to a leaf in $\mathscr{D}$. Let u be the unique cut vertex in B. Consider a cactus $T_1 = T - (V(B) \setminus \{u\})$. Since T_1 has $q-1$ blocks, by induction hypothesis, we can properly color T_1 using at most 3 colors. Since B is a block in cactus, it is either a cycle or an edge. One can properly color vertices in B using at most 3 colors when color of u is fixed. Hence T can be colored with at most 3 colors.

(3) Consider a vertex u which has degree at least three. Since any block B is cycle or an edge, any vertex u has at most 2 neighbors in B. Since u has degree at least 3, u is present in at least two block. This implies that u is a cut vertex. □

The operation of *subdividing* an edge uv results in the graph obtained by deleting uv and adding a new vertex w adjacent to both u and v.

Observation 4.2.2. *Consider a cactus T with at most ℓ leaves. Let T' be the graph*

obtained from T by one of the following operations.

1. *subdividing an edge;*
2. *contracting an edge;*
3. *deleting a cut-edge uv and add two vertex disjoint path between u, v.*

Then, T' is a cactus with at most ℓ leaves.

Proof. Let $\mathscr{D}$ being the block decomposition of T with $\mathscr{B}$ being the set of block and K being the set of cut-vertices in T.

(1) Let T' be the graph obtained by subdividing an edge uv in T and w be the resulting vertex after subdivision. Since degree of w is 2 in T', any cycle which contains w must contain its neighbors u and v. Assume that T' is not a cactus then, there exists two distinct cycles C'_1, C'_2 in T' such that $E(C'_1) \cap E(C'_2) \neq \emptyset$. But then, by replacing w with the edge uv in C'_1, C'_2 (if present), we obtain cycles $\hat{C}_1$ and $\hat{C}_2$ in T with at least one common edge, contradicting that T is cactus.

Consider the case when the edge uv is a block, say B in T. In $\mathscr{D}$, by replacing B by B_1, B_2, each containing the edges uw, wv respectively, and adding w to K, we obtain a block decomposition of $\mathscr{D}'$ of T'. Since block decomposition of a connected graph is a tree, notice that $\mathscr{D}'$ can be obtained from $\mathscr{D}$ by sub-dividing an edge twice. In a tree, a sub-division of an edge does not increase the number of leaves. Hence follows that T' is a cactus with at most ℓ leaves. The remaining case is when the edge uv not a block. Let B be a block containing the edge uv in $\mathscr{D}$. Then, by replacing B by $B \cup \{w\}$, we obtain a block decomposition of T' with exactly the same number of leaves. This concludes the proof.

(2) Let T' be the graph obtained by contracting an edge uv in T and u^* be the resulting vertex. Suppose T' is not a cactus then there exists two distinct cycles C'_1, C'_2 in T' such that $E(C'_1) \cap E(C'_2) \neq \emptyset$. But then, by replacing w with the edge uv in C'_1, C'_2 (if present), we obtain cycles $\hat{C}_1$ and $\hat{C}_2$ in T with at least one common edge, contradicting that T is cactus.

Let B be the block containing the edge uv. Consider the case when B is just the edge uv. In this case u,v must be in K. But then, by contracting the edges $uB, Bv \in E(\mathscr{D})$ we can obtain a block decomposition of T'. Notice that contracting an edge in a tree (block decomposition) cannot increase the number of leaves. Hence, it follows that T' is a cactus with at most ℓ leaves. The remaining case is when B contains some other vertex. Notice that if $u,v \notin K$, then by replacing B by $B' = (B \setminus \{u,v\}) \cup \{u^*\}$ in $\mathscr{D}$ we obtain a block decomposition of T', with exactly same number of leaves. If $u \in K$ and $v \notin K$, then by contracting the edge $uB \in E(\mathscr{D})$ we obtain a block decomposition of T' with exactly the same number of leaves.

(3) Let T' be the graph obtained from T by deleting a cut-edge uv and replacing it by two vertex disjoint paths. Let C be the cycle obtained by adding these two vertex disjoint paths between u,v. Assume that T' is not a cactus then there exists two distinct cycles C'_1, C'_2 in T' such that $E(C'_1) \cap E(C'_2) \neq \emptyset$. Since u,v are cut-vertices in graph T', any cycle which is different from C, intersect with C in at most one vertex. Hence both C'_1, C'_2 are distinct from C which implies C'_1 and C'_2 are two distinct cycles with at least one edge common in T which contradicts that it is cactus. Since uv is a cut-edge it is a block with u,v as cut-vertices. Let B be the block containing the edge uv, then we have $uB, Bv \in E(\mathscr{D})$. By replacing B with $V(C)$ we can obtain a block decomposition $\mathscr{D}'$ of T' with same number of leaves. This concludes the proof. □

We define operation SPLIT on cactus in similar way as we defined for trees with one additional condition. Consider a cactus T and one of its cut vertex, say v. Let L,R be a partition of $N(v)$ such that none of them is an empty set and there is no path between vertices of L and R in $G - \{v\}$.

SPLIT(T,v,L,R)**:** Remove vertex v and add two vertices v_1 and v_2. Make v_1 adjacent with every vertex in L and v_2 adjacent with every vertex in R. Add edge v_1v_2. If T' is the graph obtained from T by this operation then $V(T') = (V(T) \setminus \{v\}) \cup \{v_1, v_2\}$ and $E(T') = (E(T) \setminus (\{vu \mid u \in N(v)\})) \cup \{v_1u \mid u \in L\} \cup \{v_2u \mid u \in R\} \cup \{v_1v_2\}$.

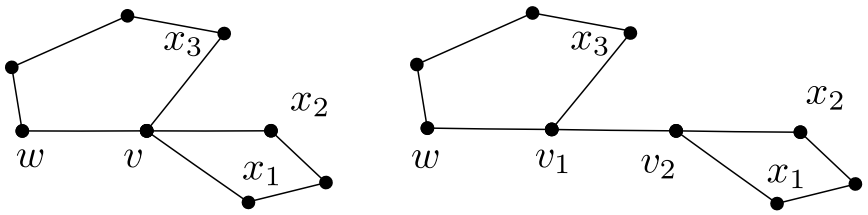


Figure 4.2: Operation SPLIT(T,v,L,R) with $L=\{w,x_3\}$ and $R=\{x_1,x_2\}$.

See Figure 4.2 for illustration. Second condition on (L,R) ensures that v_1v_2 is not a part of any cycle in new graph. The following observation, we prove that this operation on a cactus results into another cactus with same number of leaves.

Observation 4.2.3. *Let T be a cactus, v be a cut vertex of T and $N(v)$ is partitioned into two non-empty sets L and R such that there is no path between L and R in $T-v$. Let T' is the graph obtained from T after applying* SPLIT(T,v,L,R). *If T has at most ℓ leaves then T' is a cactus with at most ℓ leaves.*

Proof. For a cactus T and cut-vertex v, let $\mathscr{B}_v$ be set of blocks in T which contains a vertex v. Since v is a cut-vertex, there are at least two blocks in $\mathscr{B}_v$. Let L' and R' be the partition of $\mathscr{B}_v$ which vertices vertices from L and R respectively. Formally, $L'=\{B|\ x\in N_T(v)\cap B \text{ for some } x\in L\}$ and $R'=\{B|\ y\in N_T(v)\cap B \text{ for some } y\in R\}$. As there is no path between vertices of L,R in $T-\{v\}$, if block B is in L' then it can not be in R'.

Let T' be the graph obtained from T by deleting a cut-vertex v and adding an edge v_1v_2 such that $N_{T'}(v_1)=L\cup\{v_2\}$ and $N_{T'}(v_2)=R\cup\{v_1\}$. Notice that v_1v_2 is an cut edge in T'. We can get a block decomposition $\mathscr{D}'$ of T' from the block decomposition $\mathscr{D}$ of T by following operations : (a) Delete v from K and adding v_1,v_2 to K. (b) Replace every B in L' by $(B\cup v_1)\setminus\{v\}$ and add edge v_1B in $E(\mathscr{D}')$. (c) Replace every B in R' by $(B\cup v_2)\setminus\{v\}$ and add edge v_2B in $E(\mathscr{D}')$ (d) Add new block $B=\{v_1,v_2\}$ and add edges v_1B and v_2B in $E(\mathscr{D}')$. It is easy to see that $\mathscr{D}'$ is a block decomposition of T'. Since every block is either an edge or a cycle, T' is a cactus. Moreover, the number of leaves in $\mathscr{D}'$ is equal to the

number of leaves in $\mathscr{D}$ as newly added block is adjacent to two vertices in K. □

We make few observations regarding a cactus witness structure of a graph. Let T be a cactus obtained by contracting a set of edges in the graph G and $\mathscr{W}$ be a T-witness structure of G. Consider a pendant cycle (uPu) in cactus T where u is the unique cut vertex, and for every other vertex t in $V(P)$, witness set $W(t)$ is singleton. Then P corresponds to a simple path in G. With a slight abuse of notation let us use P to denote the path in G as well. We observe the following.

Observation 4.2.4. *If $(u_1P_1u_1)$ and $(u_1P_2u_1)$ are two pendant cycles in T, corresponding to simple paths P_1 and P_2 in G, then $V(P_1) \cap V(P_2) = \emptyset$.*

We say that a simple path P in G forms a pendant cycle in T if there is a cut vertex u in T such that uPu is a pendant cycle in T.

Following lemma says that if input graph contains a long induced path then we can find an edge which can be safely contracted.

Lemma 4.2.1. *Suppose graph G has a path $P = (u_0, u_1, \ldots, u_q)$ with $q \geq k+1$ such that all its internal vertices are of degree two. If $F \subseteq E(G)$ is a minimal set of edges of size at most k such that G/F is a cactus then F does not contain an edge in $E(P)$.*

Proof. Assume on the contrary that F contains an edge in $E(P)$. As there are at least $k+1$ edges in $E(P)$ and $|F| \leq k$, therefore there exists a vertex u_i in $V(P) \setminus \{u_0, u_q\}$ such that exactly one out of the two edges incident on it is contained in solution. Without loss of generality assume that $u_{i-1}u_i \in F$ and $u_iu_{i+1} \notin F$. Let $T = G/F$ and $\mathscr{W}$ be a T-witness structure of G. Let $t, t' \in V(T)$ such that $u_{i-1}, u_i \in W(t)$ and $u_{i+1} \in W(t')$. Consider the case when $t = t'$. F must contain all the edges in some spanning tree of $G[W(t)]$. Since $u_iu_{i+1} \notin F$, any spanning tree of $G[W(t)]$ not containing u_iu_{i+1} must contains all the edges in $E(P) \setminus \{u_iu_{i+1}\}$. But this implies $|W(t)| \geq k+2$ which is a contradiction to fact that each witness set is of size at most $k+1$. Therefore, we have that $t \neq t'$ which implies

that $tt' \in E(T)$. Recall that u_i is a degree two vertex in G. This implies that u_i is not a cut-vertex in $G[W(t)]$ as there is exactly one edge incident to it in $G[W(t)]$. Therefore, $G[W(t) \setminus \{u_i\}]$ is connected. Let $\mathscr{W}' = (\mathscr{W} \setminus \{W(t)\}) \cup \{u_i\} \cup \{W(t) \setminus \{u_i\}\}$. Observe that $\mathscr{W}'$ is a partition of $V(G)$ which is a G/F'-witness structure of G where $F' = F \setminus \{u_{i-1}u_i\}$. Notice that G/F' is the graph obtained by subdividing the edge tt' in the cactus T and by Observation 4.2.2(1) it follows that G/F' is a cactus. This contradicts the minimality of F. □

We need to work with witness structures which has certain properties.

Observation 4.2.5. *Consider a graph G and let F be a set of edges in G such that G/F is a cactus with at least three vertices. Then there exists a set F' of at most $|F|$ many edges such that G/F' is a cactus and G/F'-witness structure satisfies following property: if t' is a leaf in G/F' then $W'(t')$ is a singleton witness set.*

Proof. Let $\mathscr{W}$ be a T-witness structure of G, where $T = G/F$. Consider a leaf t_i in T such that $|W(t_i)| > 1$. Let t_j be the unique neighbor of t_i and note that t_j is not a leaf in T. As $t_it_j \in E(T)$, there exists an edge in G between a vertex in $W(t_i)$ and a vertex in $W(t_j)$. Therefore, $G[W(t_i) \cup W(t_j)]$ is connected. We claim that $G[W(t_i) \cup W(t_j)]$ has a spanning tree which has a leaf from $W(t_i)$. Observe that as $|W(t_i)| > 1$, any spanning tree of $G[W(t_i)]$ has at least two leaves. If there is a spanning tree of $G[W(t_i)]$ that has a leaf u which is not adjacent to any vertex in $W(t_j)$, then $G[(W(t_i) \cup W(t_j)) \setminus \{u\}]$ is connected too and u is the required vertex. Otherwise, every leaf in every spanning tree of $G[W(t_i)]$ is adjacent to some vertex in $W(t_j)$ and hence $G[(W(t_i) \cup W(t_j)) \setminus \{u\}]$ is connected for any vertex $u \in W(t_i)$. Therefore, as claimed, $G[W(t_i) \cup W(t_j)]$ has a spanning tree which has a leaf v from $W(t_i)$. Now consider the partition $\mathscr{W}' = (\mathscr{W} \cup \{W_v, W_{ij}\}) \setminus \{W(t_i), W(t_j)\}$ of G where $W_v = \{v\}$ and $W_{ij} = (W(t_j) \cup W(t_i)) \setminus \{v\}$. Since t_j is the only vertex adjacent to t_i, $N(u) \subseteq W(t_i) \cup W(t_j)$ for every vertex in u in $W(t_i)$. Hence $\mathscr{W}'$ is another T-witness structure of G. This leads to a set F' of at most $|F|$ edges of G such that $T = G/F'$

is a cactus. We repeat this process to ensure that the each leaf of the resulting cactus corresponds to singleton witness sets. □

Since $K_{3,2}$ can not be an induced subgraph of cactus, for any three vertices which are singleton witness sets, intersection of their neighborhood should be in one witness set.

Observation 4.2.6. *Consider a graph G and let F be a set of edges in G such that $G/F = T$ is a cactus with at least three vertices. For any three vertices u_1, u_2 and u_3 such that, $W(t_1) = \{u_1\}, W(t_2) = \{u_2\}, W(t_3) = \{u_3\}$, there is a vertex $t \in V(T)$ such that $(N(u_1) \cap N(u_2) \cap N(u_3)) \subseteq W(t)$. Similarly, for any three simple paths P_1, P_2 and P_3 in G, such that each of them form a pendant cycle in T, there is a vertex $t \in V(T)$ such that $N_G(P_1) \cap N_G(P_2) \cap N_G(P_3) \subseteq W(t)$.*

Proof. Let $X = N(u_1) \cap N(u_2) \cap N(u_3)$. If there exists $t \neq t'$ such that $X \cap W(t)$ and $X \cap W(t')$ are non-empty, T contains two cycles (t_1, t, t_2, t', t_1) and (t_1, t, t_3, t', t_1) that share more than one vertex leading to a contradiction. We argue the case of simple paths that form pendant cycles in T in a similar way. Let $X = N_G(P_1) \cap N_G(P_2) \cap N_G(P_3)$. We claim that there is a vertex t_i in T' such that $X \subseteq W'(t_i)$. If not, then there are two vertices, t_i and t_j in T' such that $X \cap W'(t_i) \neq \emptyset$ and $X \cap W'(t_j) \neq \emptyset$. Therefore, both t_i and t_j are adjacent to P_1, P_2 and P_3 in T'. This implies that there are two cycles in T' with more than one common vertex, which contradicts the fact that T' is a cactus. □

We list the following simplifying assumption analogous to Lemma 3.2.2 in case of TREE CONTRACTION.

Lemma 4.2.2. *A connected graph is k-contractible to a cactus if and only if each of its 2-connected components is contractible to a cactus using at most k edge contractions in total.*

Proof. We prove the claim by induction on the number of vertices in the graph. The claim holds for a graph on a single vertex and assume that it holds for graphs with less than n

vertices. Consider a connected graph G on n vertices. Suppose G is k-contractible to a cactus. Then, there is a set $F \subseteq E(G)$ of size at most k such that $T = G/F$ is a cactus. Let $\mathscr{W}$ be the corresponding T-witness structure of G. Let v be a cut vertex in G and let C be a connected component of $G - \{v\}$. Let G_1 denote the subgraph of G induced on $V(C) \cup \{v\}$ and G_2 denote the subgraph of G induced on $V(G) \setminus V(C)$. Then, G_1 and G_2 are connected graphs satisfying $V(G_1) \cap V(G_2) = \{v\}$. Further, the sets $E(G_1)$ and $E(G_2)$ partition $E(G)$. We claim that $G_1/(F \cap E(G_1))$ and $G_2/(F \cap E(G_2))$ are both cactus graphs. Consider the vertex $t_0 \in V(T)$ such that $v \in W(t_0)$. As the deletion of a vertex in $G_2 - \{v\}$ cannot disconnect G_1, every set in $\mathscr{W}_1 = \{W(t) \setminus V(G_2) \mid t \neq t_0, W(t) \in \mathscr{W}\} \cup \{W(t_0) \setminus (V(G_2) \setminus \{v\})\}$ induces a connected subgraph of G. Then, $F \cap E(G_1)$ is the associated set of solution edges and $G_1/(F \cap E(G_1))$ is the subgraph of G/F induced on $\{t \in V(T) \mid W(t) \cap V(G_1) \neq \emptyset\}$. Since an induced subgraph of a cactus is also a cactus, it follows that $G_1/(F \cap E(G_1))$ is a cactus. A similar argument holds for $G_2/(F \cap E(G_2))$. As $E(G_1)$ and $E(G_2)$ form a partition of $E(G)$, $|F \cap E(G_1)| + |F \cap E(G_2)| \leq k$. By induction hypothesis, the required claim holds for G_1 and G_2 and the result follows.

Conversely, let $G_1, G_2, \ldots G_l$ be the 2-connected components of G and let $F_i \subseteq E(G_i)$ be a set of edges such that G_i/F_i is a cactus and $\sum_{i \in [l]} |F_i| \leq k$. Let $\mathscr{W}_i$ be the G_i/F_i-witness structure of G_i. Define $\mathscr{W} = \bigcup_{i \in [l]} \mathscr{W}_i$. Now, $\mathscr{W}$ is made into a partition of $V(G)$ as follows: if a vertex v is contained in $W(t_1)$ and in $W(t_2)$ then add $W(t_{12}) = W_1 \cup W_2$ to $\mathscr{W}$ and delete both $W(t_1)$ and $W(t_2)$. Then, $F = \bigcup_{i \in [l]} F_i$ contains the edges of a spanning tree of every witness set in $\mathscr{W}$ and $|F| \leq k$. It remains to argue that G/F is a cactus. If G/F is not a cactus, then there exists two cycles C_1, C_2 which share at least two vertices. As any cycle can have vertices from only a single 2-connected component of a graph, C_1, C_2 are both in some 2-connected component of G leading to a contradiction. □

Following observation, which is analogous to Observation 3.2.3, helps us utilise the fact that input graph is 2-connected while designing a lossy kernel and an FPT algorithm.

Observation 4.2.7. *Consider a 2-connected graph G and let F be a set of edges in G such*

that G/F is a cactus. If t is a cut vertex in G/F then witness set $W(t)$, in G/F-witness structure, contains at least two vertices.

Proof. For sake of contradiction, assume that t is a cut-vertex in $T = G/F$ and $W(t)$ is a singleton set in T-witness structure. Let $W(t) = \{u\}$. We argue that u is a cut-vertex in G. Let T_1 and T_2 be any two connected components obtained by removing t from cactus T. Consider a set V_1 which is collection of vertices present in witness sets corresponding to vertices in T_1. Formally, $V_1 = \{u |\ u \in W(t_1)$ for some $t_1 \in V(T_1)\}$. We define set V_2 in similar way. Since T_1, T_2 are non-empty, so are V_1, V_2. There is no edge between T_1, T_2 in T and since T is obtained from graph G by contracting edges, there is no edge between V_1, V_2 in G. This implies that $G - v$ has at least two connected component viz V_1, V_2. This contradicts the fact that G is a 2-connected graph. Hence for every cut vertex t in T, associated witness sets $W(t)$ contains at least two vertices. □

If v is high degree vertex in G and v is contained in $W(t)$ then t is a cut vertex in T.

Observation 4.2.8. *Let F be a minimal set of edges of a 2-connected graph G such that $G/F = T$ is a cactus and $\mathscr{W}$ be a T-witness structure of G. Consider a vertex v in G and let t be a vertex in T such that v is in witness set $W(t)$. If $|F| \leq k$ and $d(v) \geq k+3$, then $W(t)$ is not a singleton witness set.*

Proof. For the sake of contradiction, assume that $W(t)$ is a singleton witness set. Note that every edge contraction reduces the number of vertices by exact one. Since, $|F| \leq k$, degree of t is at least three in T. By Observation 4.2.1(3), t is a cut vertex. But this is contradiction to Observation 4.2.7 which says that witness set corresponding to a cut vertex in T is a big witness set. Hence our assumption is wrong and $W(t)$ can not be a singleton witness set. □

4.3 Kernel for BOUNDED CACTUS CONTRACTION

In this section, we design a kernelization algorithm for BOUNDED CACTUS CONTRACTION. We assume that input graph is a connected otherwise we can return a trivial NO instance. Exhaustive application of first reduction rule contracts an induced path of arbitrarily large length to a path of length $\mathcal{O}(k)$.

Reduction Rule 4.3.1. *If G has a path $P = (u_0, u_1, \ldots, u_{k+1}, u_{k+2})$ such that all of its internal vertex are of degree two, then contract $u_{k+1}u_{k+2}$. The resulting instance is (G', k, ℓ) where $G' = G/\{u_{k+1}u_{k+2}\}$.*

We prove that this reduction rule is safe using Lemma 4.2.1.

Lemma 4.3.1. *Reduction Rule 4.3.1 is safe.*

Proof. Let u^*_{k+1} be the resulting vertex after contraction of the edge $u_{k+1}u_{k+2}$. Given an instance (G, k, ℓ), one can find a path P which satisfies required property, in one exists, and apply reduction rule in polynomial time. We need to prove that (G, k, ℓ) is a YES instance of BOUNDED CC if and only if (G', k, ℓ) is a YES instance of BOUNDED CC.

Let (G, k, ℓ) be a YES instance of BOUNDED CC and $F \subseteq E(G)$ such that $|F| \leq k$ and G/F is a cactus with at most ℓ leaves. From Observation 4.2.2 (2), we know that $G/(F \cup \{u_{k+1}u_{k+2}\})$ is also a cactus with at most ℓ leaves. This implies, $G/(F \cup \{u_{k+1}u_{k+2}\}) = (G/\{u_{k+1}u_{k+2}\})/(F \setminus \{u_{k+1}u_{k+2}\}) = G'/(F \setminus \{u_{k+1}u_{k+2}\})$ is a cactus with at most ℓ leaves. Also, $|F \setminus \{u_{k+1}u_{k+2}\}| \leq |F| \leq k$. Hence, it follows that (G', k, ℓ) is a YES instance of BOUNDED CC.

Let (G', k, ℓ) be a YES instance of BOUNDED CC and $F' \subseteq E(G')$ of size at most k be a minimal set such that $T' = G'/F'$ is a cactus with at most ℓ leaves. Let $\mathscr{W}'$ be a T'-witness structure of G'. Notice that in path $(u_0, u_1, \ldots, u_k, u^*_{k+1})$ every internal vertex is of degree exactly two. From Lemma 4.2.1, F' does not contain any edge incident to a vertex in $\{u_1, u_2, \ldots, u_k\}$, in particular to u_k. There exists $t'_k, t'_{k+1} \in T'$ such that $t'_k t'_{k+1} \in E(T)$ and

$W(t'_k) = \{u_k\}$ and $u^*_{k+1} \in W(t'_{k+1})$. Let $\mathscr{W} = (\mathscr{W}' \setminus W(t'_{k+1})) \cup \{W(t_{k+1}), W(t_{k+2})\}$, where $W(t_{k+1}) = \{u_{k+1}\}$ and $W(t_{k+2}) = (W(t'_{k+1}) \cup \{u_{k+2}\}) \setminus \{u^*_{k+1}\}$. Since $N_{G'}(u^*_{k+1}) \setminus \{u_k\} = N_G(u_{k+2}) \setminus \{u_{k+1}\}$, $G[W(t_{k+2})]$ is connected. Let T be the graph obtained from G by contracting each witness set to a vertex. In other words, $\mathscr{W}$ is T-witness structure of graph G. Note that T can be obtained from T' by subdividing an edge $t'_k t'_{k+1}$. From Observation 4.2.2 (1) it follows that T is also a cactus with at most ℓ leaves. Since $F' \subseteq E(G)$ and it is also a spanning forest for $\mathscr{W}$, we can conclude that (G, k, ℓ) is also a YES instance of BOUNDED CC. □

Reduction Rule 4.3.1 can be applied in polynomial time. After exhaustive application of Reduction Rule 4.3.1 in the resulting graph G any induced path with internal vertices of degree 2 is of length at most $k+2$.

Suppose input graph G has a cut-edge uv. An optimal solution may contract one of the connected components of $G - \{uv\}$, along with edge uv, to reduce the number of leaves in the resulting cactus. Consider the case when both connected components of $G - \{uv\}$ are *large* enough that neither of them is contained entirely in one witness set. In this case, no minimal solution contains the edge uv. Following reduction rule is based on this observation.

Reduction Rule 4.3.2. *If G has a cut-edge uv with C_1, C_2 being two connected components in $G - \{uv\}$ and $|V(C_1)|, |V(C_2)| \geq k+2$, then contract uv. The resulting instance is (G', k, ℓ) where $G' = G/\{uv\}$.*

Lemma 4.3.2. *Reduction Rule 4.3.2 is safe.*

Proof. Let u^* be the vertex obtained by contracting the edge uv. Given an instance (G, k, ℓ), one can find a cut-edge uv which satisfies required property, if one exists, and apply reduction rule in polynomial time. We need to prove that (G, k, ℓ) is a YES instance of BOUNDED CC if and only if (G', k, ℓ) is a YES instance of BOUNDED CC.

Let (G,k,ℓ) be a YES instance of BOUNDED CC and $F \subseteq E(G)$ of size at most k such that G/F is a cactus T with at most ℓ leaves. As a consequence of Observation 4.2.2 (2), $G/(F \cup \{uv\})$ is also a cactus. Hence, $G/(F \cup \{uv\}) = (G/\{uv\})/(F \setminus \{uv\}) = G'/(F \setminus \{uv\})$ is a cactus with at most ℓ leaves. Also $|(F \setminus \{uv\}| \leq |F| \leq k$. This concludes that (G',k,ℓ) is a YES instance of BOUNDED CC.

To prove reverse direction, let (G',k,ℓ) be a YES instance of BOUNDED CC. Let F' be a set of at most k edges such that $G'/F' = T'$ is a cactus with at most ℓ leaves. We first argue that G is $(|F'|+1)$-contractible to a cactus, say T_1, which has at most ℓ leaves. Using SPLIT operation on T_1 we argue that G is actually $|F'|$-contractible to a cactus with at most ℓ leaves.

Let $\mathscr{W}'$ be a T'-witness structure of G'. Let u^* be the vertex resulting while contracting edge uv in G to get G'. Consider vertex t^* in $V(T')$ such that u^* is in $W(t^*)$. Define set $W(t_1) := (W(t^*) \setminus \{u^*\}) \cup \{u,v\}$. Let $\mathscr{W}_1$ be the witness structure obtained from $\mathscr{W}'$ by removing $W(t^*)$ and adding $W(t_1)$. Note that $\mathscr{W}_1$ partitions $V(G)$ and for each W in $\mathscr{W}_1$, $G[W]$ is connected. Let T_1 be a graph obtained from G by contracting witness sets in $\mathscr{W}_1$. In other words, $\mathscr{W}$ is a T_1-witness structure of G. Note that T_1 can be obtained from G by contracting all edges in $F' \cup \{uv\}$. This implies T_1 can be obtained from G' by contracting all edges in F' and hence it is a cactus with at most ℓ leaves. We conclude that G is $(|F'|+1)$-contractible to a cactus with at most ℓ leaves.

Since uv is a cut-edge in G, it is also a cut-edge in $G[W(t_1)]$. Let C_u and C_v be the connected components of $G[W(t_1)] - \{uv\}$ containing u and v, respectively. Further, let $W_u = V(C_u)$, $W_v = V(C_v)$. Consider a witness structure $\mathscr{W}$ of G obtained from $\mathscr{W}_1$ by removing $W(t_1)$ and adding W_u and W_v. Notice that $\mathscr{W}$ partitions $V(G)$ and for each W in $\mathscr{W}$, $G[W]$ is connected. Moreover, we need $|F'|$ many edges to contract all witness sets in $\mathscr{W}$. Let T be a graph obtained by contracting all witness sets in $\mathscr{W}$. In other words, $\mathscr{W}$ is a T-witness structure of G. Note that G is $|F'|$-contractible to T. The only thing which remains to prove is that T is a cactus with at most ℓ leaves. We prove this by showing that T can be

obtained from T_1 by SPLIT operation at vertex t_1. We start with following claim.

Claim. Vertex t_1 is a cut vertex in T_1.

Proof. Each witness set in $\mathscr{W}_1$ is of size at most $k+2$ and hence $|W(t_1)| \leq k+2$. If t_1 is the only vertex in T_1, then all the vertices in $(V(C_1) \cup V(C_2)) \setminus \{u,v\}$ are in $W(t_1)$. This implies that $|W(t_1)| \geq 2k+3$ which is a contradiction. If t_1 has unique neighbor, say $\hat{t}$, in $V(T_1)$, then $V(C_1) \cap W(\hat{t})$ and $V(C_2) \cap W(\hat{t})$ are both non empty as $|V(C_1)|, |V(C_2)| \geq k+2$ and $|W(t_1) \setminus \{u,v\}| \leq k$. Since uv is a cut-edge in G, any path connecting vertices in $V(C_1)$ and $V(C_2)$ must contain an edge uv. Both sets $V(C_1) \cap W(\hat{t})$ and $V(C_2) \cap W(\hat{t})$ are not empty but $W(\hat{t})$ does not contain u,v. This implies that $G'[W(\hat{t})]$ is not connected contradicting the fact that it is a witness set. Hence, t_1 has at least two neighbors, say $\hat{t}_1, \hat{t_2}$ in T' such that $V(C_1) \cap W(\hat{t}_1) \neq \emptyset$ and $V(C_2) \cap W(\hat{t}_2) \neq \emptyset$. Assume that t_1 is not a cut vertex in T_1. There exist a path between $\hat{t}_1$ and $\hat{t}_2$ in $T_1 - \{t_1\}$. This implies there exists a path between $V(C_1)$ and $V(C_2)$ which does not contains an edge uv. This contradicts the fact that uv is an cut edge in G. Hence our assumption is wrong and t_1 is a cut vertex in T_1. $\diamond$

Consider a vertex t in T_1 which is adjacent with t_1. From above arguments, we know that exactly one of $V(C_1) \cap W(t)$ and $V(C_2) \cap W(t)$ is an empty set. Partition vertices in $N_{T'}(t_1)$ into two sets L and R depending on whether corresponding witness sets intersect C_1 or C_2. Formally, $L := \{t \mid t \in N_{T'}(t) \text{ and } W(t) \cap V(C_1) \neq \emptyset\}$ and $R := \{t \mid t \in N_{T'}(t) \text{ and } W(t) \cap V(C_2) \neq \emptyset\}$. Note that (L,R) is a partition of $N_{T_1}(t)$ and none of this set is empty. Moreover, there is no path between vertices in L and R. Let T be the graph obtained after operation SPLIT(T_1, t_1, L, R). By Observation 4.2.3, T is a cactus with at most ℓ many leaves.

Hence, if there exist a set of edges F' in G' such that G/F' is tree with at most ℓ leaves then G is $|F'|$-contractible to a tree with at most ℓ leaves. This concludes the proof of reverse direction. $\square$

We generalize notion of cut-edge to cycle whose removal disconnects the graph.

Definition 4.3.1 (Cut-Cycle)**.** *For a cycle C in graph G, C is a cut-cycle if in the block*

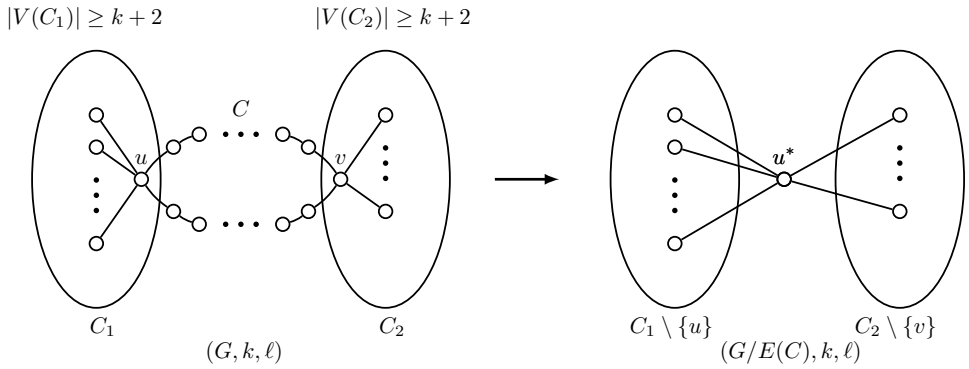


Figure 4.3: An illustration of Reduction Rule 4.3.3.

decomposition of G, there exists a block B such that $B = V(C)$ that contains exactly two cut-vertices.

For example, in Figure 4.1, $\{v_3, v_4, v_5, v_6\}$ is a cut-cycle. Let C be a cut-cycle in G and u, v be the cut-vertices that it contains. Observe that $G - E(C)$ has exactly two non-trivial connected components (components with at least two vertices), one containing u and another containing v. Following reduction rule states that it is safe to contract certain cut-cycles.

Reduction Rule 4.3.3. *Let C be a cut-cycle in G containing cut-vertices u, v and C_1, C_2 be the non-trivial components of $G - E(C)$ such that $|V(C_1)|, |V(C_2)| \geq k+2$, then contract edges in $E(C)$. The resulting instance is (G', k, ℓ), where $G' = G/E(C)$.*

Lemma 4.3.3. *Reduction Rule 4.3.3 is safe.*

Proof. We prove the safeness of this reduction rule using an intermediate instance. Reduction Rule 4.3.3 can be applied in two steps. In first step, we delete all edges in $E(C)$ and add edge uv. In second step, we apply Reduction Rule 4.3.3 on cut edge uv. Let E_1 be set of edges in $E(C)$ which are not incident on u. Then, first step is equivalent to contracting all edges E_1 in G and renaming new vertex to v. Let $\tilde{G}$ be the graph obtained from G by contracting edges in E_1. To prove the lemma, we only need to argue that (G, k, ℓ) is an

YES instance if and only if $(\tilde{G}, k, \ell)$ is an YES instance. The correctness of second step is implied by Lemma 4.3.2.

In the forward direction, let (G, k, ℓ) be a YES instance of BOUNDED CC and $F \subseteq E(G)$ of size at most k such that G/F is a cactus T, with at most ℓ leaves. As a consequence of Observation 4.2.2(2) it follows that $G/(F \cup E_1)$ is also a cactus with at most ℓ leaves. Hence $G/(F \cup E_1) = (G/E_1)/(F \setminus E_1) = \tilde{G}/(F \setminus E_1)$ is a cactus with at most ℓ leaves. Also, $|(F \setminus E_1)| \leq |F| \leq k$. This implies that $(\tilde{G}, k, \ell)$ is a YES instance of BOUNDED CC.

Let $(\tilde{G}, k, \ell)$ is a YES instance of BOUNDED CC. There exists $\tilde{F} \subseteq E(\tilde{G})$ such that $\tilde{G}/\tilde{F}$ is a cactus $\tilde{T}$ with at most ℓ leaves. Let $\tilde{\mathscr{W}}$ be $\tilde{T}$-witness structure of $\tilde{G}$ such that $u \in W(\tilde{t}_u)$ and $v \in W(\tilde{t}_v)$. Consider a witness structure $\mathscr{W}$ obtained from $\tilde{\mathscr{W}}$ by adding a singleton witness set for every vertex in $V(C) \setminus \{u, v\}$. Formally, $\mathscr{W} = \tilde{\mathscr{W}} \cup \{\{x\} \mid x \in V(C) \setminus \{u, v\}\}$. Notice that $\mathscr{W}$ partitions $V(G)$ and for each $W \in \mathscr{W}$, $G[W]$ is connected. Let T be the graph obtained from G by contracting witness sets in $\mathscr{W}$. In other words, $\mathscr{W}$ is T-witness structure of G. Notice that T is a graph obtained by replacing a cut-edge $\tilde{t}_u\tilde{t}_v$ in cactus $\tilde{T}$ by pair of vertex disjoint paths between vertices $\tilde{t}_u, \tilde{t}_v$. Hence, from Observation 4.2.2(3), T is a cactus with at most ℓ leaves. This concludes the proof of reverse direction.

Hence, if there exist a set of edges F' in G' such that G/F' is tree with at most ℓ leaves then G is $|F'|$-contractible to a tree with at most ℓ leaves. □

We say (G, k, ℓ) is a reduced instance of BOUNDED CC if none of the Reduction Rules 4.3.1, 4.3.2 and 4.3.3 are applicable.

Lemma 4.3.4. *Let (G, k, ℓ) be a reduced instance of* BOUNDED CC. *If (G, k, ℓ) is a* YES *instance of* BOUNDED CC*, then the number of vertices and edges in G is bounded by $\mathcal{O}(k^2 + k\ell)$.*

Proof. Suppose G is k-contractible to a cactus T with at most ℓ leaves. Let $\mathscr{W}$ be the T-witness structure of G and $\mathscr{D}$ be the block decomposition of T. By definition of cactus,

every block of T is either an edge or a cycle. We use the bound on the number of nodes in $\mathscr{D}$ and upper bound on size of a block to bound the number of vertices in T. Let B be a block in T. If B is an edge in T, then it contains exactly two vertices. Otherwise, B contains at least 2 vertices. Let B_C, B_W are two subsets of B, defined as follows: B_C be the set of cut-vertices in T that belongs to B and B_W be the set of vertices $t \in B$ such that $|W(t)| > 1$. We bound the size of a block using following claim.

Claim 1: $|B| \leq (k+3)|B_C \cup B_W|$.

Proof. Since the number of vertices in block B is more than 2, B induces a cycle in T. By Observation 4.2.1 and construction, for every vertex t in $B \setminus (B_C \cup B_W)$, $\deg_T(t) = 2$ and $|W(t)| = 1$. Consider a path $P = (t_x, t_1, t_2, \ldots, t_q, t_y)$ in T between two vertices $t_x, t_y \in B_C \cup B_W$ such that $\{t_1, t_2 \ldots, t_q\} \cap (B_C \cup B_W) = \emptyset$. Let $u_i \in W(t_i)$ for $i \in \{1, 2, \ldots, q\}$. Note that $|W(t_i)| = 1$, for all $i \in \{1, 2, \ldots, q\}$. Then, there exists a path $P' = (x, u_1, u_2, \ldots, u_q, y)$ in G such that $x \in W(t_x)$, $y \in W(t_y)$ and $\deg_G(u_i) = 2$ for all $i \in [q]$. Since Reduction Rule 4.3.1 is not applicable, therefore, $q \leq k$. Since B induces a cycle in T, there are at most $|B_C \cup B_W|$ such path and each path contains at most $k+3$ many vertices. Hence $|B| \leq (k+3)|B_C \cup B_W|$. ◇

By the property of block decomposition of a graph, a node t_B corresponding to block B in $\mathscr{D}$ has degree equal to $|B_C|$. Let V_1, V_2, V_3 be the set of nodes of $\mathscr{D}$ which corresponds to a block in T and are of degree at most 1, degree 2 and degree at least 3 respectively. Since $\mathscr{D}$ has at most ℓ leaves, $|V_1| \leq \ell$ which in turn implies that $|V_3| \leq \ell$. From Proposition 2.1.1, it follows that the number of cut-vertices present in blocks with at least 3 cut-vertices is bounded by the following.

$$\sum_{t_B \in V_3} |B_C| \leq 3\ell \tag{4.1}$$

Note that the number of vertices in T corresponds to big witness set is at most k therefore we have the following inequality.

$$\sum_{t_B \in V_1 \cup V_2 \cup V_3} |B_W| \leq k \tag{4.2}$$

We fix an arbitrary vertex as the root of $\mathscr{D}$ (preferable vertex of degree at least 2). For counting purpose, we apply the following marking scheme to the nodes in $\mathscr{D}$. We start by marking all the leaves in $\mathscr{D}$. For a leaf t_B, keep marking the nodes on path from the leaf to the root of that tree until the total number of vertices in T from the marked blocks is at least $k+2$. We say these marked vertices are *close* to the leaf t_B. Also mark all the nodes t_B in $\mathscr{D}$ for which B_W is not empty. This completes the marking procedure. For leaf node t_B, let t_{B^*} be the last node marked by above marking scheme to ensure that we have covered at least $k+2$ many vertices of T. Hence there are at most $k+1+|B^*|$ many vertices marked for the leaf t_B. Let $L' = \{t_{B^*} \mid t_B \in V_1\}$, *i.e.* the set of all the nodes which were the last marked node corresponding to some leaf. Notice that $|L'| \leq |V_1|$. Consider the subgraph $\mathscr{D}'$ of $\mathscr{D}$ induced on the vertices in $V_1 \cup L'$ and the cut-vertices their corresponding block contains. Note that in a block decomposition no two cut-vertices or two vertices corresponding to blocks are adjacent. This implies that the number number of vertices in $\mathscr{D}'$ is bounded by $\mathscr{O}(\ell)$. This helps us in establishing the following.

$$\sum_{t_{B^*} \in L'} |B^*_C| = \sum_{t_{B^*} \in L'} \mathsf{deg}_{\mathscr{T}}(t_{B^*}) \in \mathscr{O}(\ell)$$

Using the above relation, Claim 1 and Equation 4.2, we have the following.

$$\sum_{t_{B^*} \in L'} |B^*| \leq \sum_{t_B \in L'} (|B^*_C| + |B^*_W|)(k+2) \in \mathscr{O}(k^2 + k\ell)$$

Hence the total number of marked vertices which are close to leaf nodes are,

$$\sum_{t_B \in V_1} ((k+1) + |B^*|) \leq \sum_{t_B \in V_1} (k+1) + \sum_{t_{B^*} \in L'} |B^*| \in \mathscr{O}(k^2 + k\ell)$$

Let V_M be set of nodes t_B which are marked because B_W is not empty. By Equation 4.2, $|V_M| \leq k$. For $t_B \in V_M \cap (V_1 \cup V_2)$, $|B_C| \leq 2$ which implies $\sum_{t_B \in V_M \cap (V_1 \cup V_2)} |B_C| \leq 2k$.

$$\sum_{t_B \in V_M \cap (V_1 \cup V_2)} |B| \leq \sum_{t_B \in V_M \cap (V_1 \cup V_2)} (|B_C| + |B_W|)(k+2) \in \mathscr{O}(k^2)$$

For $t_B \in V_3 \cap V_M$, we use Equation 4.1 to obtain following bound.

$$\sum_{t_B \in V_3 \cap V_M} |B| \leq \sum_{t_B \in V_3 \cap V_M} (|B_C| + |B_W|)(k+2) \in \mathcal{O}(k^2 + k\ell)$$

We now count the number of vertices in blocks corresponding to unmarked nodes. We first argue that every unmarked node, associated block contains at least three cut-vertices. In other words, all the nodes in V_1, V_2 have been marked.

Claim 2: If t_B is not marked by above marking scheme, then $t_B \in V_3$.

Proof. We prove this by contradiction. Since all the nodes in V_1 are marked, assume that there exists unmarked node t_B in V_2 such that $|B_W| = 0$. Since B contains exactly two cut-vertices, $T - E(B)$ has exactly two non-trivial connected components, say T_1, T_2. Notice that each T_1, T_2 contains marked vertices corresponding to at least one leaf node and hence $|V(T_1)|, |V(T_2)| \geq k+2$. Since B does not contain any vertex t such that $|W(t)| > 1$, vertex set $X = \bigcup_{t \in B} W(t)$ is either a cut-edge or a cut-cycle in graph G. Moreover, $G - E(X)$ has two non-trivial connected components C_1, C_2 such that $V(C_1) = \bigcup_{t \in V(T_1)} W(t)$ and $V(C_2) = \bigcup_{t \in V(T_2)} W(t)$ which implies $|V(C_1)|, |V(C_2)| \geq k+2$. But in this case, Reduction Rule 4.3.2 or 4.3.3 is applicable on the instance. This contradicts that (G, k, ℓ) is a reduced instance. ◇

Let U be the set of nodes which are unmarked. By Claim 2, $U \subseteq V_3$. By Equation 4.1 and using the fact that $|B_W| = 0$ for $t_B \in U$,

$$\sum_{t_B \in U} |B| = \sum_{t_B \in U} (k+3)|B_C| = (k+3) \cdot \sum_{t_B \in U} |B_C| \in \mathcal{O}(k\ell)$$

Combining all these upper bounds, we get $|V(T)| \leq \mathcal{O}(k^2 + k\ell)$. Since T is obtained from G with at most k edge contractions, it follows that $|V(G)| \leq |V(T)| + k$. This implies the desired bound on the vertices of input graph. We now bound the number of edges in G. Notice that maximum degree of a node in $\mathscr{D}$ is at most ℓ as the number of leaves in $\mathscr{D}$ is at most ℓ. This implies that any cut-vertex in T can be part of at most ℓ blocks. Since,

every vertex can be adjacent to at most 2 vertices in a block, maximum degree of a vertex t in cactus T is at most 2ℓ. Every edge contraction can reduce the number of vertices by 1 hence the maximum degree of a vertex in G is at most $2\ell + k$. If G/F is a cactus then each component in $G - V(F)$ is also a cactus. Since the size of solution F is at most k, $|V(F)| \leq 2k$. As G is a simple graph, the number of edges of G with both of its end points in $V(F)$ is at most $\mathcal{O}(k^2)$. $G - V(F)$ is cactus on at most $\mathcal{O}(k^2 + k\ell)$ many vertices and hence by Observation 4.2.1, the number of edges of G whose both end points are in $V(G) \setminus V(F)$ is at most $\mathcal{O}(k^2 + k\ell)$. The number of edges which has exactly one end point in $V(F)$ is upper bounded by maximum degree of G multiplied by cardinality of F which is at most $\mathcal{O}(k^2 + k\ell)$. Hence the bound on number of edges in G follows. □

We are now ready to prove the main theorem of this section.

Theorem 4.3.1. BOUNDED CACTUS CONTRACTION *admits a kernel of size* $\mathcal{O}(k^2 + k\ell)$.

Proof. Given an instance (G, k, ℓ) of BOUNDED CC the kernelization algorithm exhaustively applies Reduction Rules 4.3.1, 4.3.2 or 4.3.2. If the number of vertices and edges in reduced graph is not upper bounded by $\mathcal{O}(k^2 + k\ell)$ then it returns a trivial no instance.

By Lemma 4.3.1; 4.3.2; and 4.3.3, these reduction rules are safe and can be applied in polynomial time. Each application of reduction rule decreases the number of edges thus it can be applied only $|E(G)|$ times. If none of the reduction rules are applicable then then either the size of the instance is bounded by $\mathcal{O}(k^2 + k\ell)$, in which case we return a kernel of desired size. Otherwise, the algorithm correctly concludes that the instance is a NO instance of BOUNDED CC. Lemma 4.3.4 proves the correctness of this step of the algorithm. □

4.4 Kernel Lower Bound for BOUNDED CACTUS CONTRACTION

In this section, we present a parameter preserving reduction from a given instance (G,R,B,k) of RBDS to an instance (G',k',ℓ') of BOUNDED CACTUS CONTRACTION. This reduction is same as the one presented in Section 3.4. We use this reduction to prove three things. First, we show that CACTUS CONTRACTION is NP-Hard. Second, CACTUS CONTRACTION parameterized by solution size k does not admit a polynomial kernel assuming NP $\not\subseteq$ coNP/poly. Third, the kernel presented for BOUNDED CACTUS CONTRACTION in Section 4.3 is optimal under the same assumption. Recall that in RBDS, given a bipartite graph $G(R,B)$ and an integer k, the task it to determine whether there exists a set of at most k vertices in R which dominates B.

Reduction. Let (G,R,B,k) be an instance of RBDS. We construct graph G' in the following way. See Figure 4.4. Initialize $V(G') = V(G)$ and $E(G') = \{br \mid b \in B, r \in R \text{ and } br \in E(G)\}$. Add a vertex a in $V(G')$ and for every vertex r in R, add an edge ar to $E(G')$. For every vertex b_i in B, add three new vertices x_i, y_i, z_i to $V(G')$ and edges b_ix_i, b_iy_i, b_iz_i to $E(G')$. Define set $X := \{x_i, y_i, z_i \mid b_i \in B\}$. For every vertex x in X, add an edge ax to $E(G')$. Set $k' = |B| + k$ and $\ell' = |R| + 3|B| - k$.

Following the same spirit of proof as described in Section 3.3, we prove following lemmas. Note that lemma implies if b_i is not present in $W(t_a)$ then at least two vertices in $\{x_i, y_i, z_i\}$ are present in $W(t_a)$ unlike in case of TREE CONTRACTION where all three were present.

Lemma 4.4.1. *Let (G',k',ℓ') be a* YES *instance of* BOUNDED CC. *There exists a solution $F^* \subseteq E(G')$ of size at most k' such that for each $b_i \in B$ one of the following holds.*

- *b_i is in $W(t_a)$ or*
- *at least two of $\{x_i, y_i, z_i\}$ are in $W(t_a)$.*

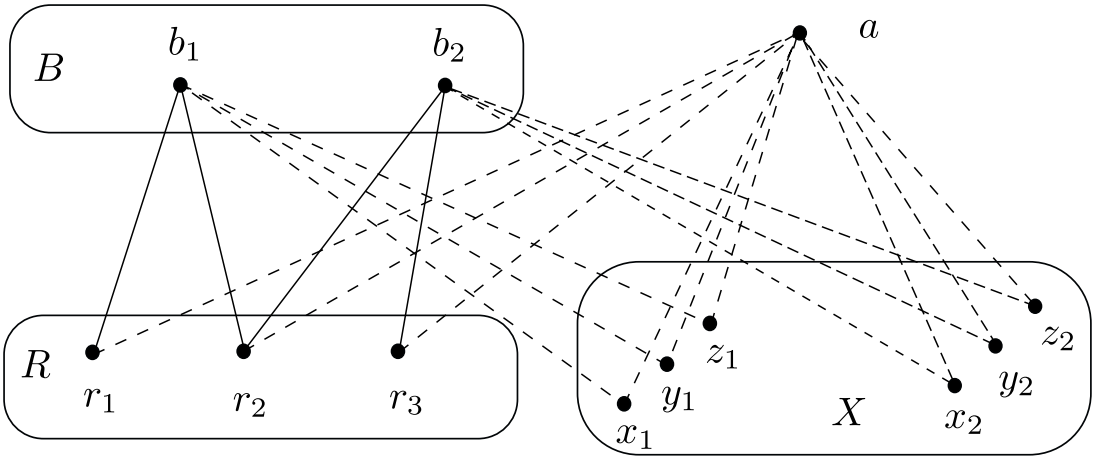


Figure 4.4: Kernel lower bound for BOUNDED CC.

Here, $W(t_a)$ is the witness set containing a in (G'/F^*)-witness structure of G'.

Proof. Let F be a set of edges of size at most k in G' such that G'/F is a tree with at most ℓ leaves. Let $\mathscr{W}$ be a T-witness structure of G' where $T = G'/F$. Let t_a be the vertex in $V(T)$ such that $W(t_a)$ contains a. For a vertex b_i in B, if b_i is in $W(t_a)$ then the lemma holds. Consider a case when b_i is not in $W(t_a)$. There exists a vertex t_b, different from t_a, such that b_i is in $W(t_b)$. Similarly, consider vertices t_x, t_y and t_z such that x_i, y_i and z_i are contained in $W(t_x), W(t_y)$ and $W(t_z)$, respectively.

If neither of t_a or t_b is contained in set $\{t_x, t_y, t_z\}$, then no two vertices in $\{t_x, t_y, t_z\}$ can be same as only neighbors of x_i, y_i, z_i are a and b_i, and a witness set needs to be connected. But then, by construction, $T[\{t_a, t_x, t_y, t_z, t_b\}]$ has at least two cycles which share an edge, contradicting that F is a solution. Without loss of generality, let $t_x \in \{t_a, t_b\}$. This implies there is an edge $t_a t_b$ is in T. If t_a and t_b not equal to t_y or t_z then, $T[\{t_a, t_y, t_z, t_b\}]$ has at least two cycles which share $t_a t_b$, contradicting that F is a solution. Therefore, at most one of t_x, t_y, t_z can be different from t_a or t_b. Without loss of generality, assume that $\{t_x, t_y\}$ is a subset of $\{t_a, t_b\}$. If both t_x, t_y are same as t_a, then the second condition of the lemma is satisfied. Therefore, we assume that at least one of t_x, t_y, say t_x, is not same as t_a which implies $t_x = t_b$. By construction, the only edges incident to x_i in G are ax_i and

bx_i. This implies that $bx_i \in F$ and $W(t'_b) = W(t_b) \setminus \{x_i\}$ is connected. Since $ax_i \in E(G)$, $W(t'_a) = W(t_a) \cup \{x_i\}$ is connected. Thus, replacing $W(t_b)$ by $W(t'_b)$ and $W(t_a)$ by $W(t'_a)$ in $\mathscr{W}$ yields another T-witness structure of G. Furthermore, the spanning forest of the new witness structure, $F' = (F \setminus \{bx_i\}) \cup \{ax_i\}$ which has same cardinality as that of F. A similar swap can be carried out if $t_y = t_b$. Hence there a witness structure such that for each $b_i \in B$ if b_i is not in $W(t_a)$ then at least two of $\{x_i, y_i, z_i\}$ are in $W(t_a)$. □

In the following lemma, we argue that the reduction is safe.

Lemma 4.4.2. (G, R, B, k) *is a* YES *instance of* RBDS *if and only if* (G, k', ℓ') *is a* YES *instance of* BOUNDED CC.

Proof. Let (G, R, B, k) be a YES instance of RBDS and S be a subset of R of size k such that S dominates every vertex in B. If S contains less than k vertices, then we take any of its superset of size exactly k. For each vertex b in B, we fix a vertex r_b in S such that b is neighbor of r_b in G. If there are multiple options for selecting r_b then we arbitrarily choose one of them. Let $F = \{br_b \mid b \in B\} \cup \{ar \mid r \in S\}$. Note that $|F| = |B| + k = k'$ and $G'[V(F)]$ is connected. Let T be the graph obtained from G' by contracting F. Let $\mathscr{W}$ be a T-witness structure of G'. Consider a vertex t_a such that a is in $W(t_a)$. Since all the edges in F are contracted to one vertex, set $S \cup B$ is also contained in $W(t_a)$. By construction, $R \cup X$ is an independent set in G'. No vertex in $(R \cup X) \setminus S$ is incident on edge which has been contracted. In other words, these vertices form singleton witness sets in $\mathscr{W}$. Since $R \cup X$ is an independent set in G', it follows that set $T_{RX} = \{t_v \mid v \in (R \cup X) \setminus S\}$ is an independent set in T of size $|R| + 3|B| - k = \ell'$. Moreover, for all v in X', $av \in E(T)$. Therefore, T is a star (which is a cactus) with ℓ' leaves. This implies that F is a solution to (G', k', ℓ').

In the reverse direction, let (G, k', ℓ') be a YES instance of BOUNDED CC and $F \subseteq E(G)$ be one of its solution. Then by Lemma 4.4.1, there exists a solution F^* of size at most k' such that for all $b_i \in B$, either $b_i \in W(t_a)$ or at least two of x_i, y_i, z_i are in $W(t_a)$. Here, $\mathscr{W}$

is a G/F^*-witness structure of G and $t_a \in V(G/F^*)$ such that $a \in W(t_a)$.

We partition vertices of B into two parts depending on whether they belong to $W(t_a)$ or not. Define $B_g = \{b_i \in B \mid b_i \in W(t_a)\}$. Let $R_a = R \cap W(t_a)$. Partition B_g into B_1 and B_2, depending on whether or not they have a neighbor in R_a. Formally, $B_1 = \{b_i \in B_g \mid N(b_i) \cap R_a \neq \emptyset\}$ and $B_2 = B_g \setminus B_1$. For a vertex b_i in B_2 at least one of x_i, y_i, z_i is present in $W(t_a)$ as there is no edge between b_i and a. Note that, by construction, x_i, y_i, z_i are not adjacent with b_j for $i \neq j$. This implies there exists a separate vertex for each b_i in B_2 which provides connectivity between a and b_i. Let XB_2 be set of vertices in $X \cap W(t_a)$ which provides adjacency between a and b_i for some b_i in B_2. For every b_i which is in $B \setminus B_g$, by Lemma 4.4.1, at least two of vertices in $\{x_i, y_i, z_i\}$ are present in $W(t_a)$.

We can partition $W(t_a) \setminus \{a\}$ into following four parts: vertices in B (captured by B_g); vertices in R (captured by R_a); vertices in X which are present because corresponding b_i is not present (captured by $B \setminus B_g$); and vertices in X which are present because they are needed to provide connectivity between b_i and a (captured by XB_2). This implies $|B_g| + 2|B \setminus B_g| + |R_a| + |XB_2| + |\{a\}| \leq |W(t_a)|$.

We construct a solution S for RBDS by taking vertices in R_a and two more sets S_g and S_w. Informally, S_g dominates vertices in B_2 and S_w dominates vertices in $B \setminus B_g$. We construct S_g in following way. For every vertex b_i in B_2, arbitrary pick one of its neighbor in R and add it to S_g. Note that $|S_g| \leq |XB_2|$. We create another set S_w in the following way. Initialize S_w to an empty set. For each b in $B \setminus B_g$, we add an arbitrary neighbor of b in R to S_w. This implies $|S_w| \leq |B \setminus B_g|$.

As cardinality of F^* is at most $k + |B|$, size of $W(t_a)$ is at most $|W(t_a)| \leq k + |B| + 1$. Putting all inequalities together, we get $|R_a| + |S_g| + |S_w| \leq k$ and every vertex in B is dominated some vertex in $R_a \cup S_g \cup S_w$. This concludes the proof. □

RED BLUE DOMINATING SET is NP-complete [44] and it does not have a polynomial kernel when parameterized by $(|B|, k)$ [32]. The existence of the polynomial parameter

transformation described above and Proposition 2.3.1 implies that CACTUS CONTRACTION does not have a kernel with size polynomial in k, unless $\mathsf{NP} \subseteq \mathsf{coNP}/poly$.

Theorem 4.4.1. CACTUS CONTRACTION *does not have a polynomial kernel unless* $\mathsf{NP} \subseteq \mathsf{coNP}/poly$.

We are now in position to present main result of this section.

Theorem 4.4.2. BOUNDED CACTUS CONTRACTION *does not admit a compression of size* $\mathcal{O}((k^2+k\ell)^{1-\varepsilon})$, *for any* $\varepsilon > 0$ *unless* $\mathsf{NP} \subseteq \mathsf{coNP}/poly$.

Proof. Assuming a contradiction, suppose BOUNDED CC admits a compression into $\Pi \subseteq \Sigma^*$ with bitsize in $\mathcal{O}((k^2+k\ell)^{1-\varepsilon})$, for some $\varepsilon > 0$. This implies that there exists an algorithm $\mathscr{A}$ which takes an instance $I = (G,k,\ell)$ of BOUNDED CC and in polynomial time returns an equivalent instance I' of Π with $|I'| \in \mathcal{O}((k^2+k\ell)^{1-\varepsilon})$.

Let (G,R,B,k) be an instance of RBDS, where G is a graph on n vertices. Using the reduction described, we create an instance (G,k',ℓ') of BOUNDED CC with $|V(G'_D)| \in \mathcal{O}(n)$, $|E(G'_D)| \in \mathcal{O}(n^2)$, $k' = k \leq |R| \in \mathcal{O}(n)$ and $\ell' = |B| + k \in \mathcal{O}(n)$. On the instance (G,k',ℓ') we run the algorithm $\mathscr{A}$ to obtain an instance I of Π such that $|I| \in \mathcal{O}((k'^2 + k'\ell')^{1-\varepsilon})$. But then we have obtained a compression of size $\mathcal{O}(n^{2-\varepsilon})$ for RBDS, contradicting Proposition 2.3.2. □

Corollary 4.4.1. BOUNDED CACTUS CONTRACTION *does not admit a kernel of size* $\mathcal{O}((k^2+k\ell)^{1-\varepsilon})$, *for any* $\varepsilon > 0$ *unless* $\mathsf{NP} \subseteq \mathsf{coNP}/poly$.

4.5 Lossy Kernel for CACTUS CONTRACTION

In previous section, we established that CACTUS CONTRACTION does not admit polynomial kernel under standard complexity assumption (Theorem 4.4.1). In this section, we compliment that result by providing a lossy kernel of polynomial size for the problem. We

define parameterized minimization version of CACTUS CONTRACTION in the following way.

$$\mathrm{CC}(G,k,F) = \begin{cases} \infty & \text{if G/F is not a cactus} \\ \min\{|F|, k+1\} & \text{otherwise} \end{cases}$$

If G has at most $k+3$ vertices then we already have a kernel of desired size. We assume that input graph has at least $k+3$ vertices. By definition of optimization problem, for a set of edges F, if G/F is a cactus then maximum value of $\mathrm{CC}(G,k,F)$ is $k+1$. Hence any spanning tree of G is a solution of cost $k+1$. We call it a *trivial solution* for given instance. We denote a complete graph on five vertices by K_5. One need to contract at least two edges to obtain a cactus from K_5. We call $(K_5,1)$ as *trivial instance* of CACTUS CONTRACTION. If $\mathrm{OPT}(G,k) = k+1$ then we can return trivial instance as its α-lossy kernel. Note that for any c-factor solution for trivial instance, solution lifting algorithm can return a trivial solution for original instance which is of cost $k+1$. Since $\mathrm{OPT}(G,k)$ is equal to $k+1$, it is 1-factor solution. We assume that input graph is connected as otherwise one can not obtain a cactus only by edge contractions.

Lemma 4.2.2 implies that a connected graph G is k-contractible to a cactus if and only if each of its 2-connected components is contractible to a cactus using at most k edge contractions in total. If a 2-connected component of graph is not a cactus then there exists an edge which is part of at least two cycles. Cycles in each of 2-connected component are edge-disjoints and hence contracting an edge in one component does not eliminate cycles in another component. If the number of 2-connected components in the input graph which are not cactus are more than $k+1$ then we can safely conclude that optimum solution for given instance is at least $k+1$. In this case we can return trivial instance otherwise we consider each 2-connected component separately. Note that we do not guess the number of edges needs to be contracted in each 2-connected component. We compute a kernel for each 2-connected component using the budget of k. The output of our kernelization

algorithm is disjoint union of kernels for each 2-connected component. We present first reduction rule which eliminate long chain of paths and/or cycles which connects two different 2-connected components. Let K be the set of cut-vertices and $\mathscr{B}$ be the set of blocks in G.

Reduction Rule 4.5.1. *If B is a block in G which is an edge or a cycle then contract all edges in $E(B)$. The resulting instance is (G',k), where $G' = G/E(B)$.*

Informally speaking, since no edges in $E(B)$ is part of more than one cycle, we do not need to contract any edge in it to construct a cactus. This implies that edges in $E(B)$ are irrelevant with respect to any solution and can safely be contracted.

Lemma 4.5.1. *Reduction rule 4.5.1 is* 1*-safe.*

Proof. Consider a solution F' for (G',k). If $|F'| \geq k+1$, solution lifting algorithm returns a spanning tree F of G. If $|F'| \leq k$ then solution lifting algorithm returns $F = F'$. If $|F'| \geq k+1$ then for a spanning tree F of G, $\mathrm{CC}(G,k,F) = k+1$. Hence in this case, $\mathrm{CC}(G,k,F) = k+1 = \mathrm{CC}(G',k,F')$. Consider a case when $|F'| \leq k$. Let $\mathscr{W}'$ be a T'-witness structure of G' where $T' = G'/F'$. Since B is a block, when all edges in $E(B)$ are contracted, there is a unique new vertex. Let u^* be the new vertex added after contracting all edges in $E(B)$. Consider vertex t^* in $V(T')$ such that u^* in $W(t^*)$. Define set $W(t) := (W(t^*) \setminus \{u^*\}) \cup V(B)$. Let $\mathscr{W}_1$ be a witness structure of G obtained from $\mathscr{W}'$ by removing $W(t^*)$ and adding $W(t)$. Notice that $\mathscr{W}_1$ partitions $V(G)$ and for each W in $\mathscr{W}_1$, $G[W]$ is connected. Let T_1 be a graph such that $\mathscr{W}$ is a T_1-witness structure of G. Note that T_1 can be obtained from G by contracting all edges in $F' \cup E(B)$. This implies T_1 can be obtained from G' by contracting all edges in F'. Hence T_1 is a cactus. This implies that G is $(|F'| + |E(B)|)$-contractible to a cactus. We argue that G is in fact $|F'|$-contractible to a cactus.

Consider a witness structure $\mathscr{W}$ obtained from $\mathscr{W}_1$ by removing $W(t)$ and adding each connected component in $G[W(t)] - E(B)$. All vertices in B which are not cut vertices are

now singleton witness sets. Cut vertices in B are either singleton witness sets or present in witness set which has vertices from other blocks containing that cut vertex. Notice that $\mathscr{W}$ partitions $V(G)$ and for each W in $\mathscr{W}$, $G[W]$ is connected. Let T be a graph obtained by contracting all witness sets in $\mathscr{W}$. In other words, $\mathscr{W}$ is a T-witness structure of G.

For a vertex u in $V(B)$, let t_u be the $V(T)$ such that $W(t_u) = \{u\}$. Define set B_T as set of vertices in t_u whose corresponding witness set is singleton and it contains vertices in $V(B)$. Edges incident on vertices in B_T are determined by witness structure $\mathscr{W}$ of G. We now argue that T is a cactus.

Assume that T is not a cactus for the sake of contradiction. By construction, $T/E(B_T) = T_1$ and T_1 is a cactus. This implies that if T is not a cactus then there exists an edge $t_u t_v$ in $E(B_T)$ which is contained in two cycles. Since no edge in $E(B)$ is contracted while constructing T, $T[B_T]$ is a cycle. Let C_T be the another cycle which contains $t_u t_v$. Let X be a union of witness sets corresponding to vertices in $B_T \cup C_T$. Formally, $X = \bigcup_{t \in B_T \cup C_T} W(t)$. Note that B is a proper subset of X. For any two vertices in X, there exists at least two paths connecting these two vertices. Hence X is a 2-connected set. This contradicts the fact that B is a block which is maximal 2-connected set in G. Hence, our assumption is wrong and T is a cactus. This implies that G can be contracted to a cactus by contracting all edges in F'. Hence, $\mathrm{CC}(G,k,F) = \mathrm{cCC}(G',k,F')$.

We now argue that $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$. Let F be an optimum solution for (G,k). By Observation 4.2.2(2), $G/(F \cup E(B))$ is also a cactus. Note that $G/(F \cup E(B)) = (G/E(B))/(F \setminus E(B)) = G'/(F \setminus E(B))$. Hence G'/F is a cactus. Since $|F \setminus E(B)| \leq |F|$, we can conclude that $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$.

Combining these two inequalities, we get $\frac{\mathrm{CC}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathrm{CC}(G',k,F')}{\mathrm{OPT}(G',k)}$ which concludes the proof. □

Exhaustive application of above reduction rule eliminates all blocks in G which are already a cactus. In rest of the section, we focus on 2-connected component of G. We assume that

the input graph is 2-connected.

Following reduction rules states that we can replace *long* path in input graph by *shorter* paths.

Reduction Rule 4.5.2. *If G has a path $P = (u_0, u_1, \ldots, u_{k+1}, u_{k+2})$ such that all of its internal vertex are of degree 2, then contract $u_{k+1}u_{k+2}$. The resulting instance is (G', k, ℓ) where $G' = G/\{u_{k+1}u_{k+2}\}$.*

We observe that this rule can be applied in polynomial time by considering each simple path in the graph of length more than $k+1$.

Lemma 4.5.2. *Reduction Rule 4.5.2 is 1-safe.*

Proof. Consider a minimal set $F' \subseteq E(G)$ such that $T' = G'/F'$ is a cactus. If $|F'| \geq k'+1$, then the solution lifting algorithm a spanning tree F of G. In this case, $\mathrm{CC}(G,k,F) = k+1 = \mathrm{CC}(G',k',F')$. In case $|F'| \leq k'$, the solution lifting algorithm returns $F = F'$. Let $\mathscr{W}'$ denote a T'-witness structure of G' where $T' = G'/F'$. Let u'_{k+1} be the new vertex added while contracting $u_{k+1}u_{k+2}$. Let P' be the path obtained from P by contracting $u_{k+1}u_{k+2}$. By Lemma 4.2.1, F' has no edge incident on $V(P') \setminus \{u_0, u_{k+2}\}$. Hence, every vertex in $V(P') \setminus \{u_0, u_{k+2}\}$ is in a singleton set of $\mathscr{W}'$. Let $\mathscr{W}$ to a witness structure obtained from $\mathscr{W}'$ by removing $\{u'_{k+1}\}$ and adding two sets $\{u_{k+1}\}, \{u_{k+2}\}$. Note that $\mathscr{W}$ is a partition of G and for every W in $\mathscr{W}$, $G[W]$ is connected. Let T be the graph obtained from G by contracting witness sets in $\mathscr{W}$. In other words, $\mathscr{W}$ is a T-witness structure of G. Note that T can be obtained from T' by subdividing edge $u_k u'_{k+1}$. By Observation 4.2.1(1), T is a cactus as T' is a cactus. Hence, $\mathrm{CC}(G,k,F) \leq \mathrm{CC}(G',k',F')$.

We now argue that $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$. Let F be an optimum solution for (G,k). By Observation 4.2.1(2), $G/(F \cup \{u_{k+1}u_{k+2}\})$ is also a cactus. Note that $G/(F \cup \{u_{k+1}u_{k+2}\}) = (G/\{u_{k+1}u_{k+2}\})/(F \setminus \{u_{k+1}u_{k+2}\}) = G'/(F \setminus \{u_{k+1}u_{k+2}\})$. Hence G'/F is a cactus. Since $|F \setminus \{uv\}| \leq |F|$, we can conclude that $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$.

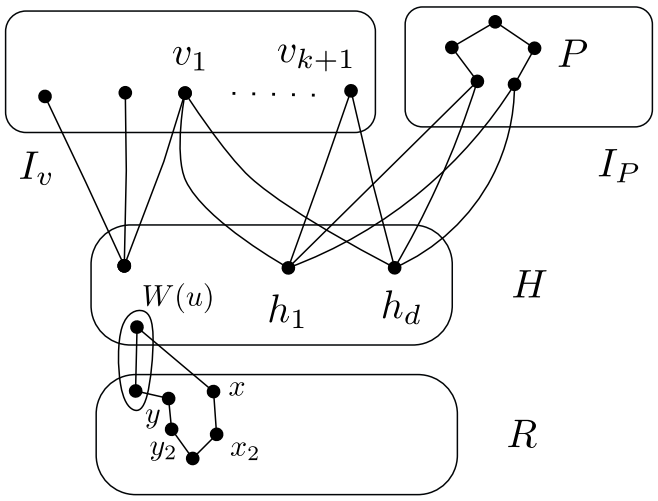


Figure 4.5: Partition of input graph.

Combining two inequalities, we get $\frac{\text{CC}(G,k,F)}{\text{OPT}(G,k)} \leq \frac{\text{CC}(G',k',F')}{\text{OPT}(G',k')}$. This concludes the proof. □

We apply Reduction Rule 4.5.2 exhaustively to the input graph. Any simple path in resulting graph contains at most $k+4$ vertices. We partition vertices of G into the following four parts: high degree vertices (H), independent set (I), collections of simple paths (I_p) and rest of the graph (R). See Figure 4.5. These sets are defined as follows.

$$H = \{u \in V(G) \mid d(u) \geq k+3\}$$

$$I_v = \{v \in V(G) \setminus H \mid N_G(v) \subseteq H\}$$

$$I_p = \{V(P) \mid P \text{ is a simple path in } G \setminus I_v \text{ and } N_G(P) \subseteq H\}$$

$$R = V(G) \setminus (H \cup I_v \cup I_p)$$

With a slight abuse of notation, we say that a path P is contained in I_p (i.e. $P \in I_p$) if $V(P) \subseteq I_P$. Let us make the following observation.

Observation 4.5.1. *For any two paths $P_1, P_2 \in I_p$, we have $V(P_1) \cap V(P_2) = \emptyset$.*

We construct graph G' from G by contracting each path P in I_p to a single vertex. All the

vertices present in $H \cup R$ are contained in $V(G')$. We use this graph to bound the cardinality of set $H \cup R$. By construction, if G is 2-connected than G' is also a 2-connected graph. We mention few simple observations which directly follows from construction of G'.

Observation 4.5.2. *There is no simple path in G' which is an isolated path in $G' \setminus H$.*

Observation 4.5.3. *If G is k-contractible to cactus then G' is also k-contractible to a cactus.*

Our main aim of constructing G' is to bound the number of pendent cycles in $G - H$.

Lemma 4.5.3. *If G' is k-contractible to a cactus T', then the number of pendant cycles in T' is bounded by $2k(k+2)$.*

Proof. Let the graph G' be k-contractible to the cactus T' via a solution set F, and let $R_F = V(F) \cap R$. Since the number of edges in F is at most k, $|R_F| \leq |V(F)| \leq 2k$. Consider a pendant cycle (uPu) in T'. It follows that $V(P) \subseteq R$ and let x, y be the endpoints P. See Figure 4.5. Observe that $N(x) \subseteq V(F') \cup \{x_2\}$ and $N(y) \subseteq V(F') \cup \{y_2\}$, where x_2 and y_2 are the respective neighbors of x and y on the path P in G'. We show that at least one of x and y is neighbor to a vertex in R_F. If this is not the case, then the neighborhoods of both x and y, except for x_2 and y_2 respectively, are contained in $V(F) \setminus R_F = H$. Hence, P is a simple path in G' which is an isolated path in $G' \setminus H$, which is a contradiction.

We conclude that each pendant cycle in T' corresponds to a simple path in G', which has at least one is incident on at least one vertex in R_F. Furthermore, the simple paths in G' corresponding to any two pendant cycles in T' are vertex disjoint (Observation 4.5.1). Hence, the number of simple paths is upper bounded by total number of neighbors of the vertices in R_F. Since $|R_F| \leq 2k$ and each vertex in R has degree at most $k+2$ in G', the lemma follows. □

We now argue that if G' is k-contractible to a cactus then its connected vertex cover is bounded. Define $\boxed{f(k) = (2k^2 + 9k + 2)(k+4)}$ for all integers k.

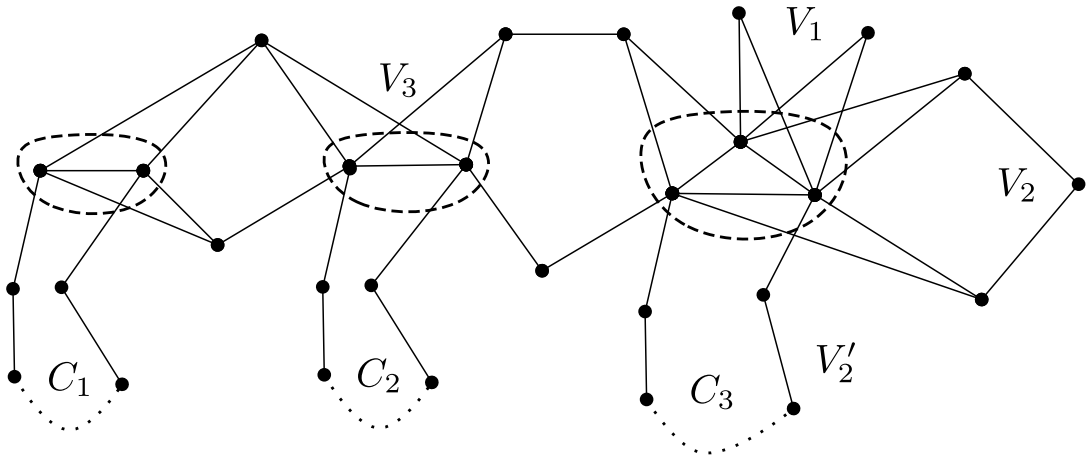


Figure 4.6: Construction of G'' from G' by adding cycles C_1, C_2 and C_3. Dotted boundary denotes big witness set in T'-witness structure of G'.

Lemma 4.5.4. *If G' is k-contractible to a cactus and Reduction Rule 4.5.2 is not applicable on (G', k), then G' has a connected vertex cover of size $f(k)$.*

Proof. Suppose G' is k-contractible to the cactus T' via a solution set F where $\mathscr{W}'$ is the corresponding T'-witness structure of G'. Consider the graph G'' obtained from graph G', by adding new vertices and edges as follows. For each $t \in T'$ such that $|W'(t)| > 1$, we arbitrarily choose an edge $u_t v_t \in G[W'(t)]$ and add a path of length $k+2$ between these two vertices. Observe that, G'' is 2-connected, and furthermore, G''/F is a cactus T'' that is obtained from T' by adding a pendant cycle of length $k+2$ to each $t \in T'$ such that $|W'(t)| > 1$. See Figure 4.6. Furthermore, T'' has at most k additional pendant cycles, as compared to T'. Let $\mathscr{W}''$ be a T''-witness structure of G''. We construct G'' to ensure that this graph has following two properties. (1) If $|W''(t)| > 1$, then t is a cut-vertex in T''. (2) In any pendant cycle (tPt) of T'' where t is the cut-vertex, $|W(t')| = 1$ for every $t' \in P$, and $|W''(t)| > 1$. We show that G'' has a connected vertex cover S'' of size $f(k)$ such that $S = V(G') \cap S''$ is a connected vertex cover of G'.

Let V_1, V_3 be the set of vertices of T'' of degree 1 and at least 3 respectively. Let $V_2' = \{t \in V(T'') \mid d(t) = 2 \text{ and } t \text{ is part of pendant cycle in } T''\}$. Now, since any simple path corresponding to a pendant cycle in T'' has at most $k+4$ vertices in G'', and by Lemma 4.5.3

in G' and the construction of G'', there are at most $2k^2+5k$ pendant cycles in T'', we conclude that $|V_2'| \leq (2k^2+5k)(k+4)$. Since every vertex of V_2' corresponds to a singleton witness set in T'', we abuse notation slightly to denote the set of corresponding vertices in G'' by V_2' as well. Next, let V_2 denote the set of degree 2 vertices in T which are not part of a pendant cycle, and note that $V_2 \cup V_2'$ cover the set of all degree 2 vertices in T''. We claim that $S'' = \bigcup_{t \in V_2 \cup V_2' \cup V_3} W(t)$ is a connected vertex cover of G''. As $T''[V_2 \cup V_2' \cup V_3]$ is connected, S'' is a connected set in G''. Without loss of generality, we assume that F follows the property mentioned in Observation 4.2.5. Hence, if $t_i \in V_1$ then $|W''(t_i)| = 1$. Consider two vertices t_i and t_j in V_1. Let $W''(t_i) = \{u\}$ and $W''(t_j) = \{v\}$. Then, as $t_i t_j \notin E(T'')$, we have that $uv \notin E(G'')$. Hence S'' is a vertex cover of G''.

We now argue that $|S''|$ is at most $f(k)$. For every vertex $t \in V_3$, by Observation 4.2.1 and 4.2.7, we have $|W''(t)| > 1$. Then, there are at most $2k$ vertices in V_3. i.e. $\bigcup_{t \in V_3} W''(t)$ is upper bounded by $2k$. Further note that, by construction of G'' and T'', for any vertex $t \in V(T'') \setminus V_3$, $|W''(t)| = 1$. We have a bound of $(2k^2+5k)(k+4)$ on the number of vertices in V_2'. It remains to bound the number of vertices in G'' corresponding to V_2. Again, since V_2 corresponds to singleton witness sets in $\mathscr{W}''$, we slightly abuse notation, and denote the set of these vertices in G'' by V_2 as well. Now, let T_s be the graph obtained from T'' by short-circuiting all vertices in V_2. By Observation 4.2.1, T_s is a cactus with $|V_3| \leq k$ vertices. Since no vertex in V_2 is contained in a pendant cycle in T'', short-circuiting a maximal path in with all internal vertices in V_2 results in an edge with two distinct endpoints in the cactus T_s. Furthermore, there can be at most two paths in T'' such that contracting them gives the same edge of T_s. By Observation 4.2.1(1), the number of edges in T_s is bounded by $2|V(T_s)| \leq 2k$. Hence, V_b can be partitioned into a collection of $4k$ simple paths in G', and recall that each one contains at most $k+2$ vertices. Therefore, $|V_2| \leq 4k(k+4)$. Putting together all these bounds we have $|S''| \leq f(k)$.

Finally, observe that $S = S'' \cap V(G')$ is a connected set in G', and S is a vertex cover of G'. This completes the proof of this lemma. □

We present following reduction rule which returns a lossy kernels for graph which has large connected vertex cover.

Reduction Rule 4.5.3. *Given an instance (G,k), let G' be the graph obtained from G by contracting each path P in I_p to a single vertex. Apply 2-factor approximation algorithm to compute a connected vertex cover X of G'. If size of X is greater than $2 \cdot f(k)$ then return* $(K_5, 1)$.

Lemma 4.5.5. *Reduction Rule 4.5.3 is 1-safe.*

Proof. Let (G,k) be an instance such that Reduction Rule 4.5.3 returns $(K_5,1)$ when applied on it. Solution lifting algorithm returns a spanning tree F of G.

Note that for a set of edges F', if K_5/F' is a tree then F' contains at least two edges. This implies $\mathrm{CC}(K_5,1,F') = 2$ and $\mathrm{OPT}(K_5,1) = 2$.

Since a 2-factor approximation algorithm returns a set of size strictly more than $2 \cdot f(k)$, size of minimum connected vertex cover of G' is strictly more than $f(k)$. By Lemma 4.5.4, if G' is k-contractible to a cactus than it has a connected vertex cover of size at most $f(k)$. By Observation 4.5.3, if G is k-contractible to a cactus then G' is also k-contractible to a catus. Hence for any set of edges F^* if G/F^* is a cactus than size of F^* is at least $k+1$. This implies $\mathrm{OPT}(G,k) = k+1$. For a spanning tree F of G, $\mathrm{CC}(G,k,F) = k+1$.

Combining these values, we get $\frac{\mathrm{CC}(G,k,F)}{\mathrm{OPT}(G,k)} = \frac{k+1}{k+1} = \frac{2}{2} = \frac{\mathrm{CC}(K_5,1,F')}{\mathrm{OPT}(K_5,1)}$. This implies if F' is c-factor approximate solution for $(K_5,1)$ then F is 1-factor approximate solution for (G,k). This concludes the proof. □

For our next reduction rule, we extend the notion of false twins to simple paths. We call two paths, P_1 and P_2 in I_p, *false twins* if $N(P_1) = N(P_2)$. The reduction rule states that we can delete all but $2k+3$ vertices (respectively paths) in I_v (respectively in I_p) which has identical neighborhood.

Reduction Rule 4.5.4. *If there is a vertex $v \in I_v$ that has at least $2k+3$ false twins, then delete v. That is, the resultant instance is $(G-\{v\},k)$. Similarly, if there is a path P in I_p that has at least $2k+3$ false twins, then delete P.*

This reduction rule can be applied in polynomial time. The rationale behind this reduction rule is same as we have mentioned in case of TREE-CONTRACTION.

Lemma 4.5.6. *Reduction Rule 4.5.4 is 1-safe.*

Proof. Let us consider the case of a path $P \in I_p$ that has at least $2k+3$ false twins. The case of a vertex $v \in I_v$ can be argued similarly. Consider a solution F' of the reduced instance (G',k'). If $|F'| \geq k'+1$, then the solution lifting algorithm returns a spanning tree F of G, and $\mathrm{CC}(G,k,F) = k+1 = \mathrm{CC}(G',k,F')$. In other case $|F'| \leq k$, and the solution lifting algorithm returns $F = F'$ as a solution for the instance (G,k). Let T' denote the cactus G'/F' and $\mathscr{W}'$ denote the corresponding T'-witness structure of G'. Then, as P has at least $2k+3$ false twins, at least three of these twins, say P_1,P_2,P_3, are disjoint from $V(F')$. Let $X = N_{G'}(P_1) = N_{G'}(P_2) = N_{G'}(P_3)$, and note that X is also the neighborhood the paths P,P_1,P_2 and P_3 in G. By Observation 4.2.6, there exists $t_i \in V(T')$ such that $X \subseteq W'(t_i)$. Now, let T be the cactus obtained from T' by adding P as a pendant cycle adjacent to t_i. Define the partition $\mathscr{W}$ of $V(G)$ obtained from $\mathscr{W}'$ by adding the new witness set $\{v\}$ for every vertex $v \in V(P)$. Then T is G/F and $\mathscr{W}$ is a T-witness structure of G. Hence, $\mathrm{CC}(G,k,F) \leq \mathrm{CC}(G',k',F')$.

We now show that $\mathrm{OPT}(G',k') \leq \mathrm{OPT}(G,k)$. Consider an optimum solution F^* for (G,k). If $|F^*| \geq k+1$ then by definition, $\mathrm{OPT}(G',k') \leq k'+1 = k+1 = \mathrm{OPT}(G,k)$. In case $|F^*| \leq k$, let T be the cactus G/F^*. Let $\mathscr{W}^*$ denote the corresponding T-witness structure of G. By a similar argument as above, we know that there exists $t_j \in V(T)$ such that $N_G(P) \subseteq W(t_j)$. It follows that P is disjoint from $V(F^*)$, and it forms a pendant cycles in T attached at t_j. Hence F^* is also a solution to the instance (G',k). Therefore, $\mathrm{OPT}(G',k') \leq \mathrm{OPT}(G,k)$. Finally, by combining the above, we conclude that, $\frac{\mathrm{CC}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathrm{CC}(G',k',F')}{\mathrm{OPT}(G',k')}$.

This concludes the proof of this lemma. □

Given $\alpha > 1$, let d be the minimum integer such that $\frac{d}{d-1} \leq \alpha$. In other words, $d = \lceil \frac{\alpha}{\alpha-1} \rceil$. For every simple path $P \in I_p$, such that $N(P)$ contains at least $2d$ vertices of H, pick one of its endpoints, that is adjacent to at least d vertices of H, into the set $\tilde{I}_p$. We apply the following reduction rule to the set $I = I_v \cup \tilde{I}_p$.

Reduction Rule 4.5.5. *If there are vertices $v_1, v_2, \ldots, v_{2k+3} \in I$ and $h_1, h_2, \ldots, h_d \in H$ such that $\{h_1, \ldots, h_d\} \subseteq N(v_i)$ for all $i \in [2k+3]$ then contract all edges in $\tilde{E} = \{v_1 h_i \mid i \in [d]\}$ and reduce the parameter by $d-1$. The resulting instance is $(G/\tilde{E}, k-d+1)$.*

The above rule can be applied in $\mathcal{O}((2k)^d \cdot n^3)$ time, by considering each subset of H of cardinality at most d.

Lemma 4.5.7. *Reduction Rule 4.5.5 is α-safe.*

Proof. Consider a solution F' of the reduced instance (G', k'). If $|F'| \geq k'+1$, then the solution lifting algorithm returns a spanning tree F of G, otherwise it returns $F = F' \cup \tilde{E}$. If $|F'| \geq k'+1$ then $\mathrm{CC}(G', k', F') = k'+1 = k-d$. In this case, F is a spanning tree of G and $\mathrm{CC}(G, k, F) \leq k+1 = k'+d = \mathrm{CC}(G', k', F') + d - 1$. Now, consider the case when $|F'| \leq k'$ and let $\mathscr{W}'$ be a G'/F'-witness structure of G. Let w denote the vertex in $V(G') \setminus V(G)$ obtained by contracting $\tilde{E}$. Let $W'(t_1)$ be the witness set in $\mathscr{W}'$ which contains w. Define $W_1 = (W'(t_1) \cup \{v_1, h_1, h_2, \ldots, h_d\}) \setminus \{w\}$. Let $\mathscr{W}$ be a witness structure obtained from $\mathscr{W}'$ by removing $W'(t_1)$ and adding W_1. Formally, $\mathscr{W} = (\mathscr{W}' \cup \{W_1\}) \setminus \{W'(t_1)\}$. Note that $V(G) \setminus \{v_1, h_1, h_2, \ldots, h_d\} = V(G') \setminus \{w\}$ and hence $\mathscr{W}$ is a partition of $V(G)$. Further, $G[W_1]$ is connected as $G'[W'(t_1)]$ is connected and a spanning tree of $G'[W'(t_1)]$ along with $\tilde{E}$ is a spanning tree of $G[W_1]$. Also, $|W_1| = |W'(t_1)| + d$ and any vertex which is adjacent to w in G' is adjacent to at least one vertex in $\{v_1, h_1, h_2, \ldots, h_d\}$ in G. Thus, $G/F = G'/F'$. Size of F is at most $|F'| + d \leq k' + d = k - d + 1 + d = k + 1$. Hence $\mathrm{CC}(G, k, F) = |F|$. This implies, $\mathrm{CC}(G, k, F) = |F| = k' + d \leq \mathrm{CC}(G', k', F') + d$.

We now show that $\mathrm{OPT}(G',k') \leq \mathrm{OPT}(G,k)-(d-1)$. Let F^* be an optimum solution for (G,k) and $\mathscr{W}$ be the T-witness structure of G where $T = G/F^*$. If $|F^*| \geq k+1$, then $\mathrm{OPT}(G,k) = k+1 = k'+d = \mathrm{OPT}(G',k')+d-1$. In case $|F^*| \leq k$, there are at least 3 vertices, say v_p, v_q, v_r in $\{v_1, v_2, \ldots, v_{2k+3}\}$ which are not in $V(F^*)$. That is, they are in singleton witness sets of $\mathscr{W}$. Then, by Observation 4.2.6, $\{h_1, h_2, \ldots, h_d\}$, which is a subset of the common neighborhood of these three vertices, is a subset of the same witness set, say $W(t_i)$ where $t_i \in V(T)$. Suppose that $v_1 \in W(t_i)$, and hence we can assume that $\tilde{E} = \{v_1 h_i \mid i \in [d]\} \subseteq F^*$. Then, $F' = F^* \setminus \tilde{E}$ is solution to (G',k') and so $\mathrm{OPT}(G',k') \leq |F'| \leq |F^*| - d = \mathrm{OPT}(G,k) - d$. Otherwise $v_1 \notin W(t_i)$, and let $t_j \in V(T)$ be the vertex such that $v_1 \in W(t_j)$. Then observe that t_i and t_j are adjacent in T. Let T' denote the cactus obtained from T by contracting the edge (t_i, t_j) and let t_{ij} denote the vertex so formed. Define another partition $\mathscr{W}' = \mathscr{W} \cup \{W(t_{ij})\} \setminus \{W(t_i), W(t_j)\}$ of $V(G)$ where $W(t_{ij}) = W(t_i) \cup W(t_j)$. Clearly, $G[W(t_{ij})]$ is connected, and hence $\mathscr{W}'$ is a T'-witness structure of G. From $\mathscr{W}'$ we can obtain a solution F that contains $\tilde{E}$, and note that $|F| = |F^*| + 1$. Now observe that, $F' = F \setminus \tilde{E}$ is solution to (G',k') leading to $\mathrm{OPT}(G',k') \leq |F'| = |F^*| + 1 - d = \mathrm{OPT}(G,k) - d + 1$.

Combining these bounds, we have, $\frac{\mathrm{CC}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathrm{CC}(G',k',F')+d}{\mathrm{OPT}(G',k')+(d-1)} \leq \max\left\{\frac{\mathrm{CC}(G',k',F')}{\mathrm{OPT}(G',k')}, \alpha\right\}$. This concludes the proof. □

We now prove that if G is k-contractible to a cactus and none of the reduction rules mentioned above are applicable, then the number of vertices in G is bounded by a function of k.

Lemma 4.5.8. *Let (G,k) be an instance of* CACTUS CONTRACTION *on which none of the Reduction Rules 4.5.2; 4.5.3; 4.5.4 and 4.5.5 are applicable. If G is 2-connected and k-contractible to a cactus then the number of vertices in G is at most $\mathcal{O}((2k)^{2d} + k^4)$.*

Proof. We first bound the size of $H \cup R$. The set H consists of only vertices of degree at least $k+3$ and by Observation 4.2.1(3) and 4.2.7, every vertex in H is incident on some

solution edge hence $|H| \leq 2k$. Since Reduction Rule 4.5.2 is not applicable on graph G, it is also not applicable on graph G'. By Lemma 4.5.4, G' has connected vertex cover S of size $\mathcal{O}(k^3)$. Notice that $V(G) \setminus (I_v \cup I_p) = V(G') \setminus (I_v \cup I'_p)$ and hence it is suffices to bound the size of R in graph G'. By construction, every vertex in R has degree at most $k+2$. Therefore, $S \cap R$ is a vertex cover of $G[R]$. The number of edges with both end points in R is at most $\mathcal{O}(k^4)$. Also every vertex in R has a neighbor in R and hence there are no isolated vertices in $G[R]$. Thus, number of vertices in R is $\mathcal{O}(k^4)$.

We now bound the size of $I_v \cup I_p$. For every set $H' \subseteq H$ of cardinality less than d, there are at most $2k+3$ vertices in I_v which have H' as their neighborhood. Otherwise, Reduction Rule 4.5.4 would have been applicable. Hence, there are at most $(2k+3) \cdot \binom{2k}{d-1}$ vertices in I_v which have degree less than d. Similarly, there are at most $2k+3$ paths in I_p which have a subset H' of H as their neighborhood, where H' has cardinality at most $2d-1$ Hence it follows that the number of such paths is at most $(2k+3) \cdot \binom{2k}{2d-1}$.

Now, any path in P, such that $|N(P)| \geq 2d$, has an endpoint in the set $I = I_v \cup \tilde{I}_p$, and this endpoint has at least d neighbors in H. Observe that, any vertex in I, with at least d neighbors in H, is adjacent to all vertices in of a subset H' of H, of cardinality d. For such a subset H', there are at most $2k+3$ vertices in I which have H' in their neighborhood. Otherwise, Reduction Rule 4.5.5 would have been applied. Thus, there are at most $(2k+3)\binom{2k}{d}$ vertices of I with d or more neighbors in H. Hence, $|I|$ is $\mathcal{O}((2k)^{d+1})$, and this also bounds the number of paths in I_p which has an endpoint in I. Combining the above, we obtain that the number of paths contained in I_p is $\mathcal{O}((2k)^{2d-1})$. By Reduction Rule 4.5.2, the length of any path in I_p is $k+4$ and hence, the total number of vertices in I_p is $\mathcal{O}((2k)^{2d})$. Similarly it follows that the number of vertices in I_v is bounded by $\mathcal{O}((2k)^d)$. Since (I_v, I_P, H, R) is a partition of G, this concludes proof of the lemma. □

Now, we put things together to present a PSAKS for CACTUS CONTRACTION.

Theorem 4.5.1. CACTUS CONTRACTION *admits a strict PSAKS with* $\mathcal{O}((2k)^{2\lceil \frac{\alpha}{\alpha-1} \rceil + 1} + k^5)$ *vertices.*

Proof. For a given instance (G,k), kernelization algorithm exhaustively apply Reduction Rule 4.5.1. If number of 2-connected components which are not cactus is more than $k+1$ then the algorithm returns a trivial instance as a lossy kernel. Otherwise, algorithm computes α-lossy kernel for each of 2-connected components separately. If algorithm finds trivial instance as lossy kernel for any of 2-connected component then it returns a trivial instance as a lossy kernel for entire graph.

For a 2-connected component, say C, the algorithm creates an instance $(G[C],k)$. Let I_v, I_P, H, R be partition of $V(C)$ as defined after 4.5.2. It is possible that cut vertices in C are part of $I_v \cup I_P$ and may get deleted while computing a lossy kernel. We avoid this by marking these vertices. Since there are at most k many 2-connected components in G, C has at most $k-1$ many cut vertices. Since each path in I_P is of length at most $k+4$, marking these vertices increase the size of reduced instance by at most $\mathcal{O}(k^2)$.

Given $\alpha > 1$, the algorithm fixes $d = \lceil \frac{\alpha}{\alpha-1} \rceil$. It applies Reduction Rule 4.5.2; 4.5.3; 4.5.4; and 4.5.5 exhaustively on instance $(G[C],k)$. If reduced graph G^* has more than $\mathcal{O}((2k)^{2d}+k^4)$ vertices then by Lemma 4.5.8, graph G^* is not k-contractible to a cactus. This implies $\mathrm{OPT}(G[C],k)$ is $k+1$. In this case, the algorithm returns a trivial instance as a lossy kernel. Otherwise reduced graph has at most $\mathcal{O}((2k)^{2d}+k^4)$ vertices. There are at most k many 2-connected components, and summing over each component, the reduced graph has at most $\mathcal{O}((2k)^{2d+1}+k^5)$. The correctness of algorithm follows from Lemma 4.5.1; 4.5.2; 4.5.5; 4.5.6; and 4.5.7. □

4.6 An FPT Algorithm for Cactus Contraction

We start with outline of the algorithm. We can think of graph contraction problem as partition problem. The task is to find a partition where each part, also called *witness set*, is connected and contracting each part to a vertex leads to a graph with desired property. Towards this, we color the input graph such that every colored component contains at most

one big witness set. A witness set, or a set which is equally good, is then extracted from color class via structural properties of the graph. See Figure 4.7. In first phase, we color $V(G)$ using three colors $\{1,2,3\}$ with hope that all vertices of a big witness set receive the same color and that two big witness sets are separated. We then identify some vertices that are not part of any big witness set and recolor them using new colors 4 and 5. For instance, we identify certain induced paths that do not intersect with any minimal solution and are *adjacent* to only one big witness set (Lemma 4.6.3). The vertices of such paths are colored 4 (Ex. v_1, v_2 in Figure 4.7). After this we identify vertices that are not part of any big witness set and lie on a path between two big witness sets (Lemma 4.6.4) and re-color them to 5 (Ex. v_4, v_5, v_6, v_{10} in Figure 4.7). This completes the first phase. In the second phase, we extract the big witness sets from the components highlighted in the first phase (Ex. in a color component with color 3, identifying v_8, v_9 as vertices not included in the witness set). For this purpose, we define the notion of a *connected core* (Definition 4.6.2) which can be thought of as generalization of connected vertex cover. For every monochromatic component colored with $\{1,2,3\}$ by the first phase, we find connected core containing certain boundary vertices. The desired solution is the set of edges of spanning forests of connected cores.

Rest of the chapter is organized as follows. Following the approach of [55], we first give a randomized algorithm for the problem on 2-connected graphs, which is then used to give an algorithm in general graphs. Algorithm can be divided into two phases viz coloring phase (Subsection 4.6.1) and extracting a solution from colored graph (Subsection 4.6.2). Finally, in Subsection 4.6.3, we present overall algorithm and illustrate how this algorithm can be derandomized via (n,k)-universal sets. We remark that the main goal of this work is to provide a $c^k n^{\mathcal{O}(1)}$ algorithm for CACTUS CONTRACTION, where c is a fixed constant. For the sake of simplicity, we do not attempt to optimize the running time.

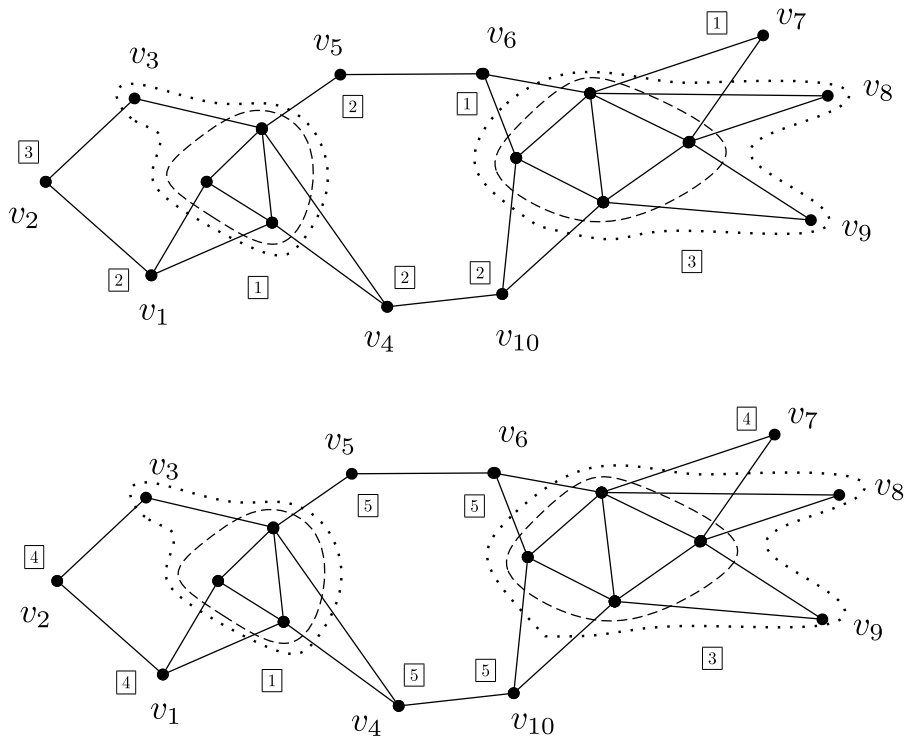


Figure 4.7: Coloring and Re-coloring of input graph. Dashed boundaries denote big witness sets while dotted boundaries corresponds to color classes.

4.6.1 Phase 1: The Coloring Phase

In this phase, we assign one of colors $\{1,2,3\}$ to vertices of input graph uniformly at random. Once we obtain a coloring, we identify certain vertices of the graph which are contained in small witness sets. We re-color them using new colors $\{4,5\}$ and move on to Phase 2 of algorithm to extract a solution from components of G which are colored $1,2$ or 3.

We need notion of *compatible coloring* to argue the correctness of coloring step. Consider a 2-connected graph G and a *minimal* set of edges F in $E(G)$ such that $G/F = T$ is a cactus. Fix a T-witness structure $\mathscr{W}$ of G. We define a compatible coloring of G with respect to $\mathscr{W}$. Informally speaking, for each big witness set, a compatible coloring assign same color to every vertex in it. It separates two big witness sets which are adjacent with each other. If two big witness sets are connected by a path then the color of an end point is

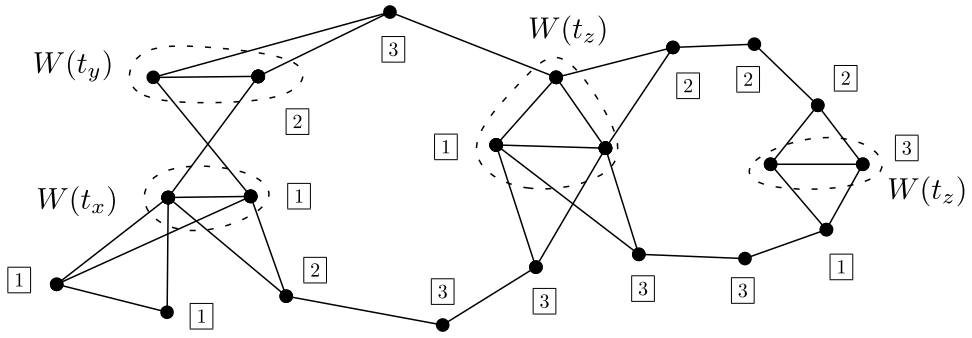


Figure 4.8: A compatible coloring of input graph. Dotted boundaries denote big witness sets. Please refer to Definition 4.6.1

different then the color of big witness set it is adjacent with. See Figure 4.8.

Definition 4.6.1 (Compatible Coloring)**.** *A coloring ϕ of G is* compatible *with a fixed T-witness structure $\mathscr{W}$ of G if following three conditions are satisfied.*

1. *For all $W(t)$ in $\mathscr{W}$, $W(t)$ is monochromatic. Hence $\phi(W(t))$ is a well defined.*
2. *For all t_x, t_y in $V(T)$, $t_x \neq t_y$, such that $W(t_x), W(t_y)$ are big witness sets and $t_x t_y$ is an edge in T, we have $\phi(W(t_x)) \neq \phi(W(t_y))$.*
3. *For all t_x, t_y in $V(T)$, such that $W(t_x), W(t_y)$ are big witness sets and there exists a simple path $P = (t_x, t_1, t_2, \ldots, t_q, t_y)$ in T such that $W(t_i)$ is small witness set for all $1 \leq i \leq q$, we have $\phi(W(t_x)) \neq \phi(W(t_1))$ and $\phi(W(t_y)) \neq \phi(W(t_q))$.*

In Figure 4.8, all three big witness sets, $W(t_x), W(t_y), W(t_z)$ are monochromatic. Since t_x and t_y are adjacent and $W(t_x), W(t_y)$ are big witness sets, these two have different colors. Note the path between $W(t_x)$ and $W(t_z)$ whose internal vertices corresponds to singleton witness sets. Coloring of this path satisfy third property of the definition. Notice the path which starts and ends in $W(t_x)$ and all internal vertices corresponds to singleton witness sets. Definition of compatible coloring allows all vertices in this path to have same color as that of $W(t_x)$.

We say that ϕ is compatible with set of edges F if G/F is a cactus and ϕ is compatible with a G/F-witness structure of G. We later argue that if (G,k) is a YES instance of CACTUS

CONTRACTION than any random 3-coloring is compatible coloring with respect to an optimum solution with high probability (Observation 4.6.4). For this section, we assume that we are given a 3-coloring ϕ of G which is compatible with an optimum solution.

A subset X of $V(G)$ is called a *colored component* of ϕ, if X is a maximal connected set of vertices that have the same color in ϕ. Let $\mathscr{X}$ be the set of all colored components of ϕ. Given a coloring ϕ, we are only interested in finding an optimum solution which is compatible with this coloring. Hence, for any two components X, Y in $\mathscr{X}$, no edge with one end point in X and another in Y is in an optimum solution. We prune coloring components and re-color them in order to move closer to an optimum solution. We note few properties of colored components in $\mathscr{X}$.

Observation 4.6.1. *For every color component X in $\mathscr{X}$, either all vertices of X are in small bags or X contains exactly one big witness set.*

Lemma 4.6.1. *If a colored component X in $\mathscr{X}$ is a simple path in G then either all vertices of X are in small bags or X is a big witness set in $\mathscr{W}$.*

Proof. Let X be a simple path $P = (v_1, v_2 \ldots, v_\ell)$. If X does not contain any big witness set then the lemma is true. By Observation 4.6.1, there exists at most one big witness set in X. We consider a case when X contains a big witness set $W(t)$. We argue that, in this case, $W(t) = X$. For the sake of contradiction assume that there exists a vertex in $X \setminus W(t)$. Since $W(t)$ is a connected subgraph and it is entirely contained in X, either v_1 or v_ℓ are not contained in $W(t)$. Without loss of generality, let v_1 be a vertex not contained in $W(t)$. Let v_{i+1} be the smallest indexed vertex which is in $W(t)$. Since a color class contains at most one big witness set, which in this case $W(t)$, each vertex in $\{v_1, v_2, \ldots, v_i\}$ are part of singleton witness sets.

Since $W(t)$ is a big witness set, v_{i+1}, v_{i+2} are in $W(t)$. Notice that v_{i+1} is a vertex in $W(t)$ such that $d_G(v_{i+1}) = 2$ and it has exactly one neighbor, v_{i+2}, in $W(t)$. The other neighbor of v_{i+1}, v_i, is not in $W(t)$. There is no neighbor of v_i in $W(t)$ apart from v_{i+1}, as X is a

simple path.

We now argue that such situation is not possible in a witness structure associated with a *minimal* solution. Let F be a minimal solution associated with witness structure $\mathscr{W}$. Since F contains a spanning tree for each big witness set, there is unique edge $v_{i+1}v_{i+2}$ in $G[W(t)]$ which is incident on v_{i+1}. The edge $v_{i+1}v_{i+2}$ is present in F. Consider a witness structure $\mathscr{W}'$ obtained from $\mathscr{W}$ by removing $W(t)$ and adding two new sets $\{v_{i+1}\}$ and $W(t) \setminus \{v_{i+1}\}$. Let T' be the graph from G by contracting all witness sets in $\mathscr{W}'$. In other words, $\mathscr{W}'$ is a T'-witness structure of G. Moreover, $F \setminus \{v_{i+1}v_{i+2}\}$ contains spanning trees of witness sets in $\mathscr{W}'$.

We now argue that T' is a cactus. For a vertex u in $V(G)$, let t_u denotes the vertex of T such that $u \in W(t_u)$. In graph T, consider edge $t_{v_i}t_{v_{i+1}}$. Graph T' can be obtained from cactus T by subdividing the edge $t_{v_i}t_{v_{i+1}}$. By Observation 4.2.2(1), T' is also a cactus. This contradicts to the fact that F is a minimal solution. Hence our assumption was wrong and $X = W(t)$. This concludes the proof of lemma. □

Identifying *Few* Vertices in Pendant Cycles and Leaves

We specify the criteria to identify *few* vertices in G that are contained singleton witness sets which corresponds to vertices in pendant cycles or are leaves in T. Note that we can not identify all such vertices in G.

Consider a pendent cycle C_T in T such that t is a unique cut vertex in C_T and all vertices $C_T \setminus \{t\}$ corresponds to singleton witness sets. Let X be the colored component which contains $W(t)$. Let V_1 be the vertices in $G - X$ which are contained in singleton witness sets corresponding to vertices in C_T. It is easy to see that V_1 induces a simple path in G. Following re-coloring is based on this observation. Later, we argue that all vertices in V_1 are re-colored in this step (Lemma 4.6.5).

Re-coloring I: For any colored component X in $\mathscr{X}$, if $G - X$ contains a vertex or a simple

path as its connected component then recolor vertices in that connected component with color 4.

For example, in Figure 4.7, path v_1v_2 is a connected component of $G-X$ where X is a colored component with color 1. Similarly, v_7 is re-colored in this step. Note that we can not identify v_3 or v_8 in this step.

For a colored component X, let a simple path P be a connected component of $G-X$. In Lemma 4.6.3, we argue that all vertices in $V(P)$ are singleton witness sets in $\mathscr{W}$ or we are dealing with simple instance mentioned in Lemma 4.6.2. See Figure 4.9.

Lemma 4.6.2. *If G is a 2-connected graph such that $V(G)$ can be partitioned into two simple paths P and Q in G, then we can solve the instance (G,k) of* CACTUS CONTRACTION *in polynomial time.*

Proof. Let p_1,p_2 and q_1,q_2 be the endpoints of the simple paths P and Q, respectively. Observe that G has a hamiltonian cycle, as G is 2 connected and p_1,p_2,q_1,q_2 are the only vertices that can have degree greater than two. If G is an induced cycle, then the optimal solution is the empty set. Otherwise, G is a cycle with either one or two additional edges between p_1,p_2 and q_1,q_2. It follows that any optimal solution requires at most 3 edge contractions. □

We assume that instance we are working with does not satisfy the premise of Lemma 4.6.2.

Lemma 4.6.3. *For a colored component X in $\mathscr{X}$, let P be a connected component of $G-X$. If P is a simple path in G whose neighborhood is contained in X then the all the vertices of P lie in small witness sets.*

Above lemma holds when P contains only one vertex. For a colored component X in $\mathscr{X}$, suppose there is an isolated vertex v which is connected component of $G-X$. Since ϕ is compatible with optimum solution, all big witness sets are monochromatic. This implies v can not be part of any big witness set and remains as singleton witness set.

Proof. (of Lemma 4.6.3) For the sake of contradiction assume the lemma is false. Therefore there is some big witness set in $\mathscr{W}$ that contains a vertex of P. Let $Y \in \mathscr{X}$ be a colored component that contains this witness set. As ϕ is a compatible coloring, and $N_G(P) \subseteq X$, we have $Y \subseteq V(P)$. Hence Y is a simple path in G, and by Lemma 4.6.1, color component Y is a big witness set.

We argue that $Y = V(P)$. Let $P = (v_1, v_2, \ldots, v_\ell)$. Suppose that Y is proper subset of $V(P)$ then at least one of v_1 or v_ℓ is not present in Y. Without loss of generality, let $v_1 \notin Y$. Let v_{i+1} be the smallest indexed vertex in Y. Let t_i be the vertex in T such that $v_i \in W(t_i)$. Observe that $W(t_i) \subseteq P$. There is no edge between Y and $W(t_i)$ except for $v_i v_{i+1}$. Consider a witness structure $\mathscr{W}'$ obtained from $\mathscr{W}$ by replacing Y with $Y \setminus \{v_{i+1}\}$ and $\{v_{i+1}\}$. Let T' is the graph obtained from G by contracting all witness sets in $\mathscr{W}'$. In other words, $\mathscr{W}'$ is a T'-witness structure of G. Note that T' can be obtained from T by sub-dividing $t_i t_Y$, where $W(t_Y) = Y$. Hence by Observation 4.2.2(1), T' is a cactus. By similar arguments to that of proof of Lemma 4.6.1, this contradicts the minimality of solution associated with $\mathscr{W}$

Hence no proper subset of edges in P is contained in the minimal solution, say F, associated with witness structure $\mathscr{W}$. Since all the edges of P are in F, this implies that $P \in \mathscr{X}$. Let t_P be the vertex corresponding to P in T. It is adjacent to $t \in T$ if and only if $W(t)$ contains a vertex from $N_G(P)$ which is a subset of X. We consider two cases depending on the number of edges across P and X. If $|E_G(P,X)| \leq 2$, then either t_i is adjacent with one vertex, say t_i or two vertices t_i, t_j in T. By subdividing edge $t_i t_P$, we get another cactus (Observation 4.2.2(1)). This contradicts the minimality of F

For rest of the proof, we assume that $|E_G(P,X)| \geq 3$. By Observation 4.6.1, X contains at most one big witness set. We consider two cases depending on whether X contains a big witness set or not.

Case 1. X does not contain a big witness set

Let T_X denote vertices in T which corresponds to singleton witness set containing vertices in X. If $N_G(P)$ corresponds to at least 3 vertices in X, then $T[T_X \cup t_P]$ contains two cycles with a common edge, i.e. T is not a cactus, which is a contradiction. Hence, $N_G(P)$ contains exactly two vertices, x_1 and x_2, of X and $E_G(P,X)$ contains either 3 or 4 edges. See Figure 4.9. By Observation 4.2.7, no vertex of T_X is a cut-vertex in T. As X is a connected set, there exists a path, say Q, between x_1 and x_2 which is contained in X. Let T_Q denote vertices in T which corresponds to singleton witness set containing vertices in Q. Observe that $C = T[T_Q \cup t_P]$ is a cycle in T with t_P being the only vertex corresponding to a big witness set in C. We claim that there is no other vertex in T apart from vertices C i.e. $T = C$. If this is the case then $G = P \uplus Q$ and both P and Q are simple paths in G. This contradicts our assumption that instance under consideration does not satisfy premise of Lemma 4.6.2.

We now argue that $T = C$. Assume this is not the case, then $V(T) \setminus (t_P \cup V(Q))$ is non-empty. There is a vertex $t_y \in V(T) \setminus (V(P) \cup V(Q))$, such that there are two internally vertex disjoint paths between t_y and t_P in T. Indeed, we can start with a arbitrarily chosen t_y, and consider a minimum separator between t_y and t_P in T. If the minimum separator is a single vertex t'_y, then observe that $y' \notin V(Q)$, as vertices of Q are not cut-vertices in T. We substitute t_y with t'_y and start over. Since the shortest path between t'_y and t_P in T is strictly shorter than the shortest path between t_y and t_P, we obtain the vertex t_y in finitely many iterations. See Figure 4.9. Let T_{R_1} and T_{R_2} be two internally vertex disjoint paths in T between t_y and t_P. Paths T_{R_1} and T_{R_2} contains vertices t_{x_1}, t_{x_2}. Without loss of generality, let $t_{x_1} \in T_{R_1}$ and $t_{x_2} \in T_{R_2}$, and hence $T_{R_1} \cup T_{R_2}$ contains a path between t_{x_1} and t_{x_2}, say T_R in T. The path is distinct from the path T_Q, as $Q_1 = V(T_Q) \cap V(T_{R_1})$ and $Q_2 = V(T_Q) \cap V(T_{R_2})$ are disjoint and therefore at least one edge of T_Q is absent from T_R. This implies that T contains three distinct paths between t_{x_1} and t_{x_2}, namely $P_T = (x_1, t_P, x_2), T_Q$ and $T_{R_1} \cup T_{R_2}$. This contradicts the fact that T is a cactus. Hence our assumption is wrong and $T = C$.

Case 2. X contains a big witness set

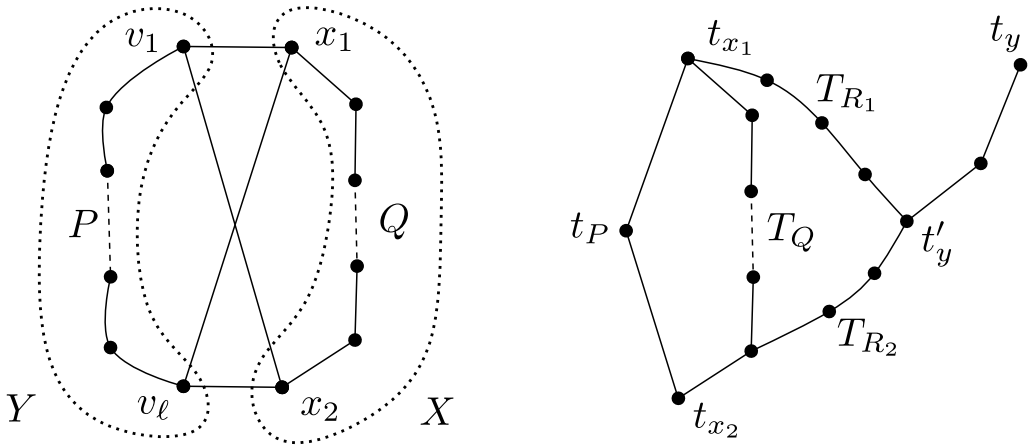


Figure 4.9: Please refer to Lemma 4.6.3

Let Z be the big witness set contained in X and let t_Z be the vertex in T obtained by contracting Z. We claim that $N_G(P)$ is a subset of Z. If this is not the case, consider a vertex v in $N_G(P) \setminus Z$. Note that v is contained in singleton witness set, say $W(t_v)$. In T, vertex t_v lies on a path between t_Z and t_P. As t_Z, t_P are big witness sets, $\phi(t_v) = \phi(t_Z)$ contradicts the fact that ϕ is a compatible coloring (Definition 4.6.1 (3)). Hence $N_G(P) \subseteq Z$ which implies $N_T(t_P) = t_Z$ in T. Hence $G/(F \setminus E(P))$ is also a cactus contradicting the minimality of F.

In either case, we derive a contradiction. Hence our assumption is wrong and lemma is true. □

Identifying *All* Vertices in Simple Paths Between Two Big Witness Sets

Recall that in simple path no internal vertex is adjacent to any vertex outside this path. A simple path is *maximal* if it is not contained in any other simple path. In other words, in *maximal simple path* every internal vertex has degree exactly two and end points have degree strictly greater than two. Since we are working with a 2-connected graph, we do not have to consider the case when end points of maximal simple path have degree one.

Consider a simple path $P = (t_x, t_1, t_2, \ldots, t_q, t_y)$ in T such that $W(t_i)$ is singleton witness set

for all $1 \leq i \leq q$, and $W(t_x), W(t_y)$ are big witness sets. Let X, Y are the colored components containing $W(t_X), W(t_Y)$, respectively. Since coloring ϕ is compatible with $\mathscr{W}$, we know that t_1, t_q are not contained in X and Y, respectively. Let V_1 be the vertices in G which are contained in $W(t_i)$ for $1 \leq i \leq q$. It is easy to see that V_1 induces a maximal simple path in G. Moreover, V_1 is a connected component of $G - (X \cup Y)$. Following re-coloring is based on this observation. Later, we argue that all vertices in V_1 are re-colored in this step (Lemma 4.6.5).

We mention that we exhaustively apply Re-coloring I before starting Re-coloring II. Also, once a vertex is re-colored to 4, we do not re-color it to 5. This ensures that for two colored components Y and Z, a vertex or simple path which is a connected component of $G - (Y \cup Z)$ is not a vertex or simple path in $G - Y$ or $G - Z$.

Re-coloring II: For any two colored component Y, Z in $\mathscr{X}$, if $G - (Y \cup Z)$ contains a vertex or a maximal simple path as its connected component then recolor vertices in that connected component with color 5.

For example, in Figure 4.7, path v_5v_6 is a maximal path when two colored components are deleted from graph. These two vertices are recolored to 5 in this step.

We state following lemma when P is maximal simple path but it also holds for a vertex.

Lemma 4.6.4. *For two colored components Y, Z in $\mathscr{X}$, let P be a connected component of $G - (Y \cup Z)$. Suppose that P is a maximal simple path in G such that $P = (v_1, v_2, \ldots, v_\ell)$; $N_G(v_1) \subseteq Y \cup \{v_2\}$ and $N_G(v_\ell) \subseteq Z \cup \{v_{\ell-1}\}$. Then all vertices of P lie in small witness sets. Furthermore, both Y and Z contain big witness sets.*

We mention that maximality of P is used to prove second part of the lemma.

Proof. (*of Lemma 4.6.4*) For the sake of contradiction, assume first part of lemma is false, i.e there is a big witness set that contains a vertex of P. Let $A \in \mathscr{X}$ be a colored component that contains this witness set. As ϕ is a compatible coloring, and $N_G(P) \subseteq Y \cup Z$, we have

$A \subseteq P$. Therefore, for any $A \in \mathscr{X}$ which intersects P, we have $A \subseteq P$. This implies A is a simple path in G. Since A contains a big witness set, by Lemma 4.6.1, A itself is a big-witness set.

We now argue that $A = V(P)$. Suppose that A is proper subset of $V(P)$ then at least one of v_1, v_ℓ is not present in Y. Without loss of generality, let $v_1 \notin A$. Let v_{i+1} be the smallest indexed vertex in A, and let t_i be the vertex in T such that $v_i \in W(T_i)$. Observe that $W(t_i) \subseteq P$. There is only one edge, $v_i v_{i+1}$, between A and $W(t_i)$. Consider a witness structure $\mathscr{W}'$ obtained from $\mathscr{W}$ by replacing A with $A \setminus \{v_{i+1}\}$ and $\{v_{i+1}\}$. Let T' is the graph obtained from G by contracting all witness sets in $\mathscr{W}'$. In other words, $\mathscr{W}'$ is a T'-witness structure of G. Note that T' can be obtained from T by sub-dividing $t_i t_Y$, where $W(t_Y) = Y$. Hence by Observation 4.2.2(1). This contradicts the minimality of solution associated with $\mathscr{W}$ by similar arguments to that of proof of Lemma 4.6.1.

Hence no proper subset of edges in P is contained in the minimal solution, say F, associated with witness structure $\mathscr{W}$. Since all the edges of P are in F, this implies that $P \in \mathscr{X}$. Let t_P be the vertex corresponding to P in T. Let $YP = Y \cap N(P) = Y \cap N(v_1)$ and $ZP = Z \cap N(P) = Z \cap N(v_\ell)$. First, we claim that YP is in one witness set of $\mathscr{W}$. Suppose that $W(t_1), W(t_2)$ are in two different witness sets in $\mathscr{W}$ which contains vertices from YP. See Figure 4.10. As G is a 2-connected graph and P is a simple path in G, the graph $G - V(P)$ is connected. As G is a 2-connected graph, there exists a path, say P_1, between YP and ZP which does not contains v_1. Since v_1 is part of simple path P, path P_1 does not contain any vertex in $V(P)$. This implies that there exists path between any two vertices among $\{t_1, t_2, t_{ZP}\}$ in $T - \{t_P\}$, where t_{ZP} is a vertex in T such that $W(t_{ZP}) \cap ZP \neq \emptyset$. Since t_P is adjacent to t_1, t_2 and t_{ZP}, there exists two cycles that have a common edge in T. This is a contradiction to the fact that T is a cactus. Hence, all of YP lies in one witness set in $\mathscr{W}$. By similar arguments, we can show that ZP is contained in one witness set in $\mathscr{W}$.

Consider vertices t_P, t_{YP}, t_{ZP} in T where and $YP \subseteq W(t_{YP})$ and $ZP \subseteq W(t_{ZP})$. Clearly $(t_{YP} t_P)$ and $(t_P t_{ZP})$ are edges in T, and t_P is a vertex of degree 2 in T. Consider a witness

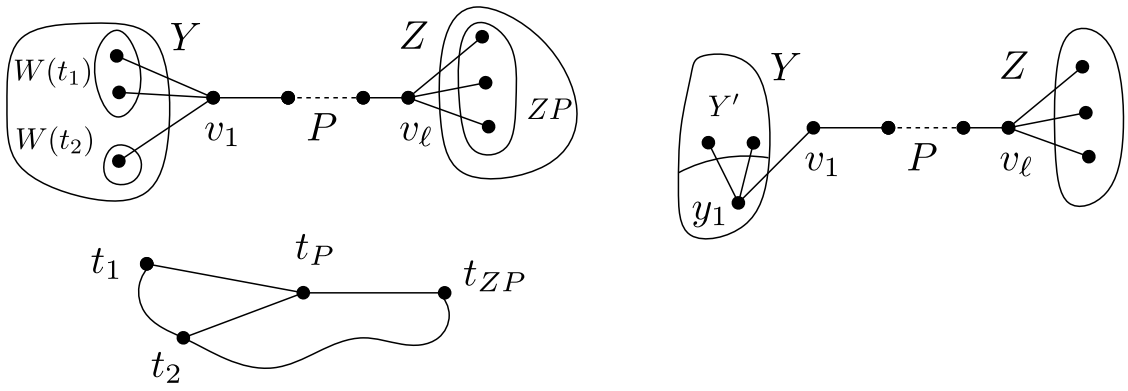


Figure 4.10: Refer to Lemma 4.6.4.

structure $\mathscr{W}'$ obtained from $\mathscr{W}$ by removing $W(t_P)$ and adding two new sets $\{v_1\}$ and $W(t_P) \setminus \{v_1\}$. Let T' be the graph from G by contracting all witness sets in $\mathscr{W}'$. In other words, $\mathscr{W}'$ is a T'-witness structure of G. Moreover, $F \setminus \{v_1v_2\}$ contains spanning trees of witness sets in $\mathscr{W}'$. We now argue that T' is a cactus. Graph T' can be obtained from cactus T by subdividing the edge $t_{YP}t_P$. By Observation 4.2.2 (1), T' is also a cactus. This contradicts to the fact that F is a minimal solution. Hence our assumption was wrong and first part of lemma is true.

Next, we argue that Y contain a big witness set in $\mathscr{W}$ as argument for Z are symmetric. If v_1 has at least two neighbors in Y then the above arguments imply that all these vertices are in a single witness set of Y and therefore Y contain a big witness set. Otherwise, v_1 only has only one neighbor, say y_1 in Y. This contradicts the maximality of P in G. Hence v_1 is adjancent with at least two vertices in Y which are contained in one big witness set. By similar arguments, we conclude that Z contains a big witness set. This concludes the proof of the lemma. □

Properties of Recoloring

By definition of compatible coloring, every colored component contains at most one big witness set. In Lemma 4.6.5, we argue that after re-coloring, all colored components

which contains at least two vertices and are colored with $\{1,2,3\}$ contain a big witness set. We can think of Lemma 4.6.5 as *completeness* part for Lemma 4.6.3 and 4.6.4. In Lemma 4.6.5, we claim that *all* vertices in colored component which do not contain a big witness set, satisfies the premise of Lemma 4.6.3 or 4.6.4.

Lemma 4.6.5. *If a colored component X in $\mathscr{X}$ which contains at least two vertices and is monochromatic with color from $\{1,2,3\}$ after exhaustive application of two re-coloring rules then X contains a big witness set.*

Proof. Let T_B and T_S are set of vertices in T which corresponds to big witness sets and singleton witness set respectively. By Observation 4.2.1 (3) and 4.2.7, any vertex in T_S has degree at most two in T. Hence $T - T_B$ is a collection of isolated vertices and simple paths.

Let X be a colored component in $\mathscr{X}$ which has not been re-colored. If for some t in T_B, $W(t)$ is contained in X then the lemma is true. Assume that there exists X which is not been re-colored and it does not contain any vertex from T_B. Let T_X be the set of vertices in T such that corresponding witness set contains vertices in X.

Any vertex in T_S, and hence in T_X, is either a leaf or part of path starting and ending at same vertex in T_B (in other words, part of pendent cycle) or part of path connecting two different vertices in T_B. Consider a vertex t' in T_X and t_1, t_2 in T_B. Let $x' = W(t')$ and X', X_1, X_2 be the color components containing $W(t'), W(t_1), W(t_2)$, respectively. If t' is a leaf adjacent to t_1 then X' is a connected component of $G - X_1$ and hence it was re-colored to 4. If t' is a part of path starting and ending at t_1 then x' is part of simple path in $G - X_1$ and hence it was re-colored to 4. Similarly, if t' is a part of path connected t_1, t_2 then x' is part of simple path in $G - (X_1 \cup X_2)$ and re-colored to 5. □

This also implies exhaustive application of re-coloring identify almost all the vertices in G that form small bags in T. The only exceptions being those vertices that are contained in some colored component X in $\mathscr{X}$ which also contains a big witness set. In the next section, we see how to identify those singleton witness sets.

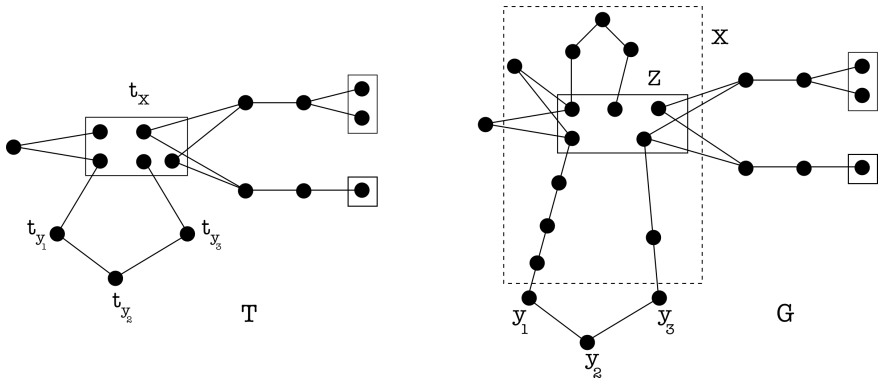


Figure 4.11: Square represents a connected component in graph. Consider a colored component X in graph G on right hand side. Instead of contracting all of X to a vertex t_X (left side graph), we contract connected core Z of $G[\hat{H}]$ to a single vertex which require smaller edges to be contracted. We replace X by Z and singleton set for every vertex in $\hat{X} \setminus Z$ in $\mathscr{X}$.

4.6.2 Phase 2: Identifying Big Witness Sets

By Lemma 4.6.5, any colored component in $\mathscr{X}$ which is not recolored and is of size at least two, contains a big witness set. For a colored component X in $\mathscr{X}$, let $W(t)$ be the big witness set contained in X. Our objective in this section is to find subset X' of X which is *at least as good as* $W(t)$ (See Figure 4.13). Informally speaking, this means we can replace edges in spanning tree of $G[W(t)]$ by edges in spanning tree of $G[X']$ in an optimum solution F, compatible with ϕ, and get another optimum solution F'.

We examine the properties of $W(t)$ in graph $G[X]$. In fact, we consider a superset $\hat{X}$ of X and examine the properties of $W(t)$ with respect to graph $G[\hat{X}]$. Let $\hat{X}$ be the superset of X which contains vertices in the connected components of $G-X$ that are either isolated vertices or a simple path in G and whose neighborhood is contained in X. We now define the notion of connected core. See Figure 4.11.

Definition 4.6.2 (Core)**.** *A* core *of a graph G is a subset Z of $V(G)$ such that every connected component of $G-Z$ is either an isolated vertex or a simple path whose neighborhood is contained in Z. If a core Z is a connected set in G, then we call it a* connected core *of G.*

Following observation is a direct consequence of the definition.

Observation 4.6.2. *For a given graph G and its connected core Z, let S be a spanning tree of $G[Z]$. Then, G/S is a cactus.*

Notice that any superset of a connected-core which induces a connected subgraph is also a connected core. In the following lemma, we claim that $W(t)$ is a connected core of $G[\hat{X}]$.

Lemma 4.6.6. *For a colored component X in $\mathscr{X}$, if $W(t)$ is the big witness set contained in X then $W(t)$ is a connected core of $G[\hat{X}]$.*

Proof. Since $W(t)$ is a witness set, by definition $G[W(t)]$ is connected. For the sake of contradiction assume that $W(t)$ is not a core of $G[\hat{X}]$. This implies that at least one connected component C of $G[\hat{X}] \setminus W(t)$ is neither a simple path nor a isolated vertex. Hence, C contains at least 3 vertices and there exists a vertex x in C such that $d_{G[\hat{X}]}(x)$ is at least 3 and it is adjacent to at least two vertices in C. If x is in $\hat{X} \setminus X$, then by Lemma 4.6.3, it is contained in a small bag. Otherwise, x is in $X \setminus W(t)$ and it is again contained in a small bag. This implies that there exits a vertex t_x in T such that $W(t_x) = \{x\}$ and $d_T(t_x)$ is at least 3. By Observation 4.2.1 (3), x is a cut-vertex in T. However, this contradicts Observation 4.2.7 which states that that every cut-vertex in T corresponds to big witness set. Hence our assumption is wrong and $W(t)$ is a connected core of $G[\hat{X}]$. □

We point out that there might exists a proper subset of $W(t)$ which is a connected core of $G[\hat{X}]$. In other words, every vertex in $W(t)$ is either part of a connected core of $G[X]$ and/or it is in $W(t)$ because of external constraints. Lemma 4.6.7 and 4.6.8 states that if vertices in X specify certain conditions then they are part of $W(t)$ because of external constraints.

Lemma 4.6.7. *If there exists v in $N_G(X)$ such that v is colored 5 then $N_G(v) \cap X$ is contained in a big witness set of X.*

Proof. If v is colored 5 then by Lemma 4.6.4, v is contained in a simple path P in G between two components X, X' in $\mathscr{X}$, such that all the vertices of P are in small witness

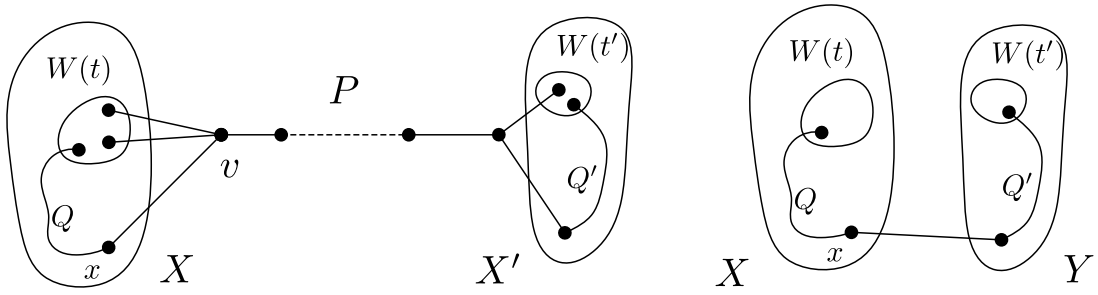


Figure 4.12: Please refer to Lemma 4.6.7 and 4.6.8.

sets in $\mathscr{W}$. Furthermore, both X and X' contain big witness sets in $\mathscr{W}$. Let $W(t')$ be the big witness set contained in X'. Assume that there exists x in $W(t) \setminus N(v)$. Consider a path Q from $W(t)$ to x which is contained entirely in $G[X]$. Let Q' be a path from $W(t')$ to an endpoint of P, whose internal vertices are in X'. See Figure 4.12. Since xv is an edge in G, we know that Q along with edge xv and paths P, P' form a path from $W(t)$ to $W(t')$ in G. This path in G gives a path between t and t' in T, such that all the internal vertices of this path correspond to small witness sets in $\mathscr{W}$. Notice that every vertex on the path from $W(t)$ to x, has same color as that of $W(t)$. All these vertices are in small bags. This is a contradiction to the fact that ϕ is compatible coloring with $\mathscr{W}$. Hence our assumption is wrong and all vertices in $N(v) \cap X$ are contained in $W(t)$. This concludes the proof of the lemma. □

Lemma 4.6.8. *Let X, Y be two colored component in $\mathscr{X}$ which contain big witness sets, say W_X and W_Y, respectively. Then, $N(X) \cap Y$ and $N(Y) \cap X$ are contained in W_Y and W_X respectively.*

Proof. If $E(X,Y)$ is empty then the statement is vacuously true. Assume that there is a vertex x in $(N(Y) \setminus W_X) \cap X$, and let t, t' be the vertices of T corresponding to the big witness sets W_X, W_Y respectively. See Figure 4.12. Since X is connected, there exists a path between W_X and x which is entirely contained in X. As X may contain only one big witness set, x lies in a small bag in $\mathscr{W}$. This implies that there is path between t and t'

in T (via x) such that the neighbor of t has the same color as vertices in W_X. This is a contradiction to the fact that ϕ is compatible coloring with $\mathscr{W}$. □

We point out that these conditions for including vertices in X in big witness set depend on other color components and not on big witness set contained in them. Hence, given $\mathscr{X}$ we can mark all vertices which are part of big witness set in each colored component. We introduce following marking scheme to mark vertices which are in big witness set because of external constraints.

Marking Scheme 4.6.1. *For a colored component X in $\mathscr{X}$,*

1. *If there exists y in $N(X)$ such that $\phi(y) = 5$ then mark all the vertices in $N(y) \cap X$.*
2. *For a colored component Y in $\mathscr{X}$, if $\phi(Y) \in \{1,2,3\}$ and it contains at least two vertices then mark all vertices in $N(Y) \cap X$*

We note that Re-coloring-I indirectly contributes to second point in marking scheme. Because of Re-coloring-I and Lemma 4.6.5, we can be sure that any colored component colored with $1,2$ or 3 contains a big witness sets.

Once we mark vertices which are present in big witness set because of external constraints, we find *any* connected core of minimum cardinality which contains these vertices. We argue that this connected core is *as good as* the big witness set for our purposes. For example, consider Figure 4.13 and let $R = \{x_1, x_2, x_5, x_6, x_7\}$. Set $W(t) = R \cup \{x_3\}$ is the unique big witness set in X. Our objective is to find a set which is as good was $W(t)$. In this case, apart from $W(t)$ itself, set $Z = R \cup \{x_4\}$ is as good as $W(t)$. Note that vertices x_5, x_6, x_7 need not be included in any minimal connected core of $G[\hat{X}]$. Vertices x_6, x_7 are marked because of first point in Marking Scheme while x_5 is marked because of second point. We formally prove these things in rest of the section. We postpone discussion on how to find a connected core of minimum size for a given graph to last part of this section.

Recall that $\hat{X}$ is the superset of X which contains vertices in the connected components of $G - X$ that are either isolated vertices or a simple path in G and whose neighborhood is

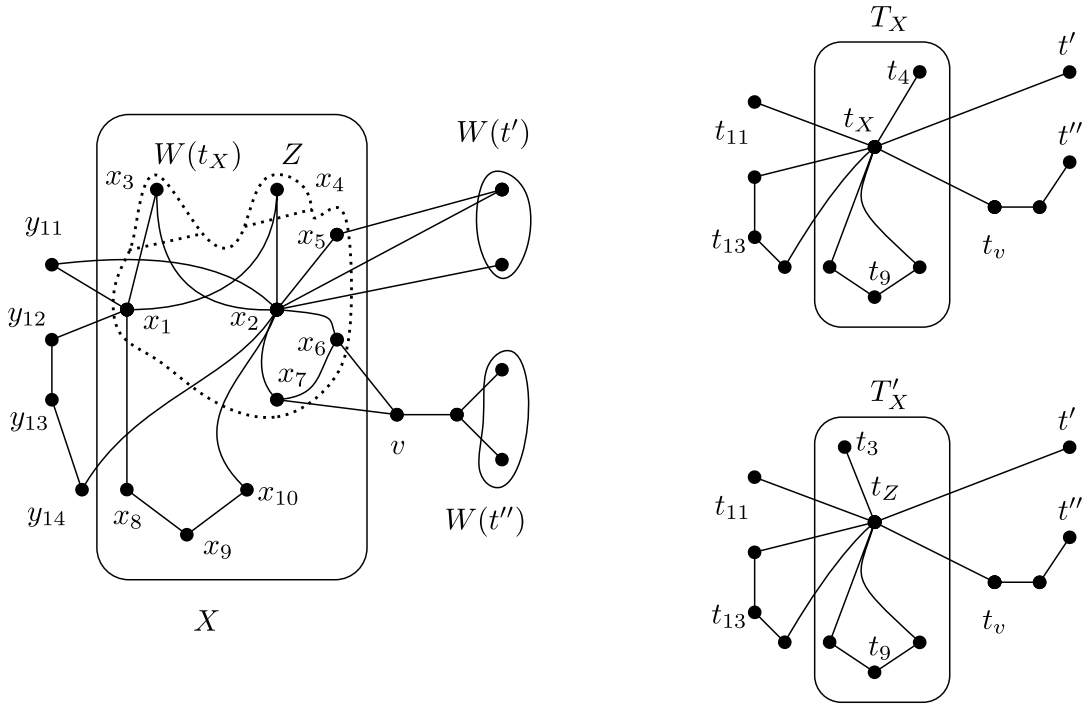


Figure 4.13: Replacing $W(t_x)$ by Z in $\mathscr{X}$. Please refer to Lemma 4.6.9.

contained in X.

Pruning Operation: For a colored component X in $\mathscr{X}$ which contains a big witness set, let M_X be set of marked vertices in X by Marking Scheme 4.6.1. Let Z_X be a connected core of $G[\hat{X}]$ of minimum cardinality which contains set M_X. In $\mathscr{X}$, remove X and add Z_X along with $\{v\}$ for every vertex v in $\hat{X} \setminus Z_X$.

We stop the pruning operation when no colored component is replaced in $\mathscr{X}$. Since pruning operation consider a colored component at most once, it stops in at most $|V(G)|$ many steps. As final lemma in this section, we argue that if we start applying pruning operation on set of colored classes obtained from compatible coloring ϕ, we end up with a witness structure corresponding with an optimum solution. Recall that F is a minimum set of edges such that G/F is a cactus and $\mathscr{W}$ is a G/F-witness structure of G. Also, ϕ is coloring of $V(G)$ which is compatible with $\mathscr{W}$. Set $\mathscr{X}$ is collection of colored components of ϕ.

Lemma 4.6.9. *Let set $\mathscr{X}^*$ be obtained from $\mathscr{X}$ by exhaustive application of Pruning Operations. If F^* is a union of spanning trees of graph induced on colored component in $\mathscr{X}^*$ then G/F^* is a cactus and $|F^*| \leq |F|$.*

Proof. For the sake of simplicity, we consider a case when pruning operation replaces exactly one colored component in $\mathscr{X}$. Let X be a colored component in $\mathscr{X}$; M_X are marked vertices in X and Z is a connected core of $G[\hat{X}]$ which contains all marked vertices. Let $W(t_x)$ be the unique big witness set contained in X. This implies that a spanning tree, say S_t, of $G[W(t_x)]$ is contained in F. Let S_Z be a spanning tree of $G[Z]$. Consider a set of edges F' obtained from F by removing S_t and adding S_Z. By Lemma 4.6.6; 4.6.7; and 4.6.8, $W(t_x)$ is a connected core of $G[\hat{X}]$ which contains M_X. Since Z is a minimum sized connected core which contains M_X, $|Z| \leq |W(t_x)|$ which implies $|F'| \leq |F|$. In the remaining proof, we argue that G/F' is a cactus.

Let $\mathscr{W}'$ be a T'-witness structure of G where $T' = G/F'$. We argue that $\mathscr{W}'$ can be obtained from a T-witness structure $\mathscr{W}$ of G. We first claim that if a witness set in $\mathscr{W}$ intersects $\hat{X}$ then it is a subset of $\hat{X}$. Assume that there exists a witness set $W(t)$ which intersects $\hat{X}$ and contains a vertex y in $W \setminus \hat{X}$. As $W(t)$ is a connected set, there is a path in $G[W(t)]$ between y and x. Since, X is a separator between $\hat{X} \setminus X$ and $V(G) \setminus \hat{X}$, this path intersects X. This implies that there exists a witness set with one vertex in X and another outside X. Since every witness set is monochromatic, no such witness set exists. Hence, there is no witness set which contains $\hat{X}$ and vertex outside $\hat{X}$. This implies $\mathscr{W}'$ can be obtained from $\mathscr{W}$ by removing all witness sets which are contained in $\hat{X}$ and adding Z and singleton witness set for every vertex in $\hat{X} \setminus Z$. For a witness structure $\mathscr{W}$, let $\mathscr{W}_{\hat{X}}$ be set of all witness set contained in $\hat{X}$. Formally, $\mathscr{W}' = (\mathscr{W} \setminus \mathscr{W}_{\hat{X}}) \cup \mathscr{W}'_1$ where $\mathscr{W}'_1 = \{Z\} \cup \{\{v\} | \ v \in \hat{X} \setminus Z\}$.

We consider two resulting graph T and T'. See Figure 4.13. Let T_X be the induced subgraph on vertices in T whose corresponding witness sets are contained in $\hat{X}$. Formally, $V_1 = \{t | \ t \in V(T) \text{ and } W(t) \subseteq \hat{X}\}$ and $T_X = T[V_1]$. We similarly define T'_X. Since $\mathscr{W}$ and $\mathscr{W}'$ are T-witness structure and T'-witness structure of G, graphs $T - V(T_X)$ and

$T' - V(T'_X)$ are isomorphic to each other.

Recall that $W(t_x)$ is the big witness set in X. Let t_z be the vertex in T' such that $W'(t_z) = Z$. We now argue that $T - (V(T_X) \setminus \{t_x\})$ and $T' - (V(T'_X) \setminus \{t_z\})$ are isomorphic as well. It is sufficient to prove that neighbors of t_x in $T - V(T_X)$ are identical to that of t'_x in $T' - V(T'_X)$, or formally, $N_{T'}(t_z) \setminus V(T'_X) = N_T(t_x) \setminus V(T_X)$. Consider a vertex x in $\hat{X}$ which has a neighbor y in $V(G) \setminus \hat{X}$. Let t_y be a vertex in T such that y is in $W(t_y)$. There are three possibilities for t_y in T: (1) t_y is part of a path between t_x and some other vertex in T which corresponds to a big witness set; (2) t_y is part pendent cycle in which t_x is the unique (cut) vertex which corresponds to a big witness set; or (3) t_y is a leaf adjacent to t_x. In first case, y is colored to 5 and hence x is in M_X. In second and third case, y is a isolated vertex or part of simple path in $G - W(t)$ and hence in $G - X$. This would imply y is part of $\hat{X}$. Since we started with assumption that y is in $V(G) \setminus \hat{X}$, these cases do not occur. Hence M_X contains every vertex in X that has a neighbor in $V(G) \setminus \hat{X}$. This implies both $W(t_x)$ and Z contains every vertex in X that has a neighbor in $V(G) \setminus \hat{X}$, and therefore $N_{T'}(t_z) \setminus V(T'_X) = N_T(t) \setminus V(T_X)$. Hence $T - (V(T_X) \setminus \{t_x\})$ and $T' - (V(T'_X) \setminus \{t_z\})$ are isomorphic with each other.

By Observation 4.6.2, both $G[\hat{X}]/S_t$ and $G[\hat{X}]/S_Z$ are cactus. Once again, since M_x (hence $W(t_x)$ and Z) contains all vertices in $\hat{X}$ which has neighbors outside, t_x and t_z are the only vertices in T and T' which has neighbors outside T_X and T'_X. Graph $T' - (V(T'_X) \setminus \{t_z\})$ is cactus as it is isomorphic to $T - (V(T_X) \setminus \{t_x\})$ which is a cactus. Since T'_X is also a cactus and t_z is the only vertex which has neighbors outside T'_X, T' is a cactus. This concludes the proof that G/F' is a cactus.

Next, we consider all the sets $X \in \mathscr{X}$ and fix an arbitrary order among them. Now, starting with a given solution F, we apply the above arguments for each X in $\mathscr{X}$ one by one. Here, we update the set F to F' each time, before proceeding to the next X. Observe that F' obtained at the end of the process, say F^*, is a solution, i.e. G/F^* is a cactus, and $|F^*| \leq |F|$. Since F was optimum solution it follows that $|F| \leq |F^*|$ which concludes the

proof. □

Finding Connected Cores

Recall that a connected-core of a graph G is a subset Z of vertices such that, $G[Z]$ is connected and each connected component of $G-Z$ is either an isolated vertex or a simple path whose both end points have neighbors in Z. Here we present a simple branching algorithm that determines if G has a connected core of size at most k or not. We use algorithm for STEINER TREE problem as subroutine. In STEINER TREE problem, we are given a graph G and set of vertices, called *terminals*, and a positive integer ℓ. The goal is to determine whether there is a tree with at most ℓ edges that connects all the terminals. We present the following lemma in the form which we use it later on.

Lemma 4.6.10. *There is an algorithm that given a connected graph G on n vertices, a subset X of its vertices and an integer k, either computes a minimum connected core of G which contains X and is of size at most k or correctly concludes that no such connected core exists in* $6^k \cdot n^{\mathcal{O}(1)}$ *time.*

Proof. We first construct a core of G via a branching algorithm. At each leaf of the branching tree, we extend the core constructed by the branching algorithm to a connected set by applying an algorithm for the STEINER TREE problem.

Let Z denote a partial solution to the instance. Initialize Z to X and decrease k by $|X|$. The following branching rule is derived from the observation that if (u,v,w) is a path outside connected core of G then degree of v is two in G.

Branching Rule 1. *If there is a path* (u,v,w) *in* $G-Z$ *such that* $|N_G(v)| \geq 3$*, then branch into three cases where each of u or v or w is added to Z. Decrease k by one in each of the branches.*

Observe that when this rule is no longer applicable, all vertices of $G-Z$ have degree at

most two. Hence the components of $G - Z$ are simple paths in G, or isolated vertices. Next, we have the following reduction rule that follows from observation that if (x, y) is an isolated edge obtained by removing minimum connected core of G, then x and y have degree two or more in G.

Reduction Rule 4.6.1. *If there is an edge uv in $G - Z$ such that u is an unique neighbor of v then add u into Z and reduce k by one.*

Since the only neighbor of v is u, the edge uv cannot be part of a simple path in G whose both endpoints have neighbours in Z. Now, if there exists an optimal solution Z^* that does not contain u, then $v \in Z^*$ and $Z' = (Z^* \setminus \{v\}) \cup \{u\}$ is also a connected core of G. This justifies the correctness of the rule.

We apply the above rules exhaustively, and consider the search tree constructed. Note that each node of the search tree is labeled with either a triple (u, v, w) indicating that the Branching rule 1 was applied, or an edge (x, y) indicating that Reduction rule 4.6.1 was applied at this node. If at any node in the search tree, k is 0 and the set Z is not a connected core of G, we abort that node. If all the leaves of the current search tree are aborted, then we output NO as a solution to this instance.

Next, we claim that if none of the rules are applicable at a leaf of the search tree, then the corresponding Z is a core of G. Assume to the contrary that Z is not a core of G. Then there is a component C of $G - Z$ that is neither an isolated vertex, nor it is a simple path in G whose both endpoints have neighbours in Z. Hence such a C has at least two vertices. Furthermore recall that the branching-rule is not applicable at this node of the search tree, and therefore all vertices in $G - Z$ have maximum degree 2. Consider the case when C is a cycle in $G - Z$. As G is connected, C has a vertex v that has a neighbour in Z. Let u and w be the neighbours of v in C. Then, it follows that (u, v, w) is a path in $G - Z$ with $|N_G(v)| \geq 3$. However, this leads to a contradiction as Branching rule 1 is not applicable. Now, consider the case when C is a path in $G - Z$ with end-points u and v. If there is an internal vertex on this path that has a neighbor in Z, then as before, we obtain a

contradiction. Hence, C is a simple path in G, with end-points u and v. As Z is not a core of the connected graph G, one of u or v has no neighbour in Z, i.e. it is a vertex of degree 1 in G. But then, Reduction rule 4.6.1 is applicable, which is a contradiction. Hence Z is a core of the graph G.

However, as Z may not be connected in G, we may have to add additional vertices to ensure connectivity. Observe that this can be achieved by computing a minimum STEINER TREE for Z in G. Given a graph G and a set S of vertices of G, the STEINER TREE problem is the task of computing a minimum cardinality connected subgraph that contains S. This problem is known to admit an algorithm with $\mathcal{O}^*(2^{|S|})$ running time [82]. The above algorithm computes a minimum cardinality connected set of vertices, $Z' \supseteq Z$, in time $\mathcal{O}(2^k)$. Observe that Z' is a connected-core of G, as $G-Z$ is a collection of isolated vertices and simple paths in G. Let $\bar{Z}$ be the minimum cardinality connected-core over all the leaves of the search-tree. If $|\bar{Z}| \leq k$, we output $\bar{Z}$ and otherwise we output NO as the solution to the instance.

Let us now argue the correctness of this algorithm. Assume Z^* is an optimal solution of size at most k. We claim that above algorithm finds a connected core $\bar{Z}$ such that $|\bar{Z}| \leq |Z^*|$. To argue this, we associate a path on the search tree of branching algorithm to the set Z^*.

Now consider an internal node in search tree that is labeled with (a,b,c). Since Branching rule 1 is applied at this node, we have that (a,b,c) is a path in $G-Z$ and $|N_G(b)| \geq 3$. As Z^* is a core of G, at least one of a,b,c must be present in it. Similarly, for any node labeled with an edge (x,y), one of these vertices, say y, is of degree 1 in G, and hence Z^* must contain one of them. Recall that, by previous arguments, we may assume $x \in Z^*$. Hence, we start from the root of the search tree and navigate to a leaf along the choices consistent with Z^*. If more than one choices are consistent with Z^*, we arbitrarily pick one of the them and proceed. Consider the set $\tilde{Z}$ obtained at the leaf via this navigation consistent with Z^* from the root-node of the search tree. Clearly $\tilde{Z} \subseteq Z^*$ and $\tilde{Z}$ is a core (not necessarily connected) of G. Let T be an optimal solution for an instance of $(H,\tilde{Z})$ of

STEINER TREE as defined above. Since Z^* is a connected core of G and $\tilde{Z} \subseteq Z^*$ we know that $Z^* \setminus \tilde{Z}$ is a solution to this Steiner Tree instance. By the optimality of T, $|T| \leq |Z^* \setminus \tilde{Z}|$ and hence $\bar{Z} = \tilde{Z} \cup T$ is a desired solution.

Let us now consider the running time of this algorithm. At each application of the Branching rule 1, we have a three-way branch and the measure drops by 1 branching vector is $(1,1,1)$. This leads to the recurrence $T(k) \leq 3T(k-1)$ which solution is $3^k \cdot n^{\mathcal{O}(1)}$. Next, at each leaf of the search tree, we run the algorithm for finding a minimum Steiner tree, which runs in time $2^k \cdot n^{\mathcal{O}(1)}$. If Steiner tree obtained is of size strictly more than k then we discard this node. Therefore, the overall running time is $6^k \cdot n^{\mathcal{O}(1)}$. □

4.6.3 Putting it all Together: The Overall Algorithm

Recall that a connected graph is k-contractible to a cactus if and only if each of its 2-connected components is contractible to a cactus using at most k edge contractions in total (Lemma 4.2.2). In this section, we first present a randomized algorithm for CACTUS CONTRACTION when input graph is 2-connected (Theorem 4.6.1). Using the arguments presented in [55], we present a randomized algorithm to solve CACTUS CONTRACTION on connected graphs (Theorem 4.6.2). Finally, we describe how to derandomize this algorithm using (n,k)-universal sets.

Consider a 2-connected graph G which is contractible to a cactus T. Let $\mathscr{W}$ be a T-witness structure of G. By Observation 4.2.7, if t is a cut-vertex in T then $W(t)$ is a big witness set. Without loss of generality, we assume $\mathscr{W}$ satisfy the property mentioned in Observation 4.2.5, i.e. if t is a leaf in T then $W(t)$ is a singleton witness set. We use following observation to bound the vertices in G which are adjacent to big witness sets; contained in singleton witness sets and are part of a path between two big witness sets.

Observation 4.6.3. *For a given cactus T, let T_B be the set of vertices in T that correspond to big-witness sets. Then, there are at most $4|T_B|$ vertices which lie on a path between two*

different vertices in T_B and are adjacent to vertices in T_B.

Proof. Formally, $T_B = \{t|\ t \in V(T)$ such that $W(t)$ is a big witness set.$\}$. Let V_1 be the set of vertices t in $V(T)$ such that there exists $t_1 \neq t_2 \in T_B$; $t \in N(t_1)$; and there exists a path between between t_1 and t_2 which contains t. Delete all vertices in T which are not contained in path between two different vertices in T_B to get a graph T_1. Since all cut vertices in T are in T_B, resultant graph is still connected and hence a cactus. Moreover, no vertex in V_1 is deleted.

If $|T_B| = 1$ then the statement is vacuously true. We consider a case when $|T_B| \geq 2$. Let $\mathscr{D}$ be the block decomposition of T_1. We prove the bound using the induction on number of blocks in cactus graph. Our induction hypothesis is: if number of blocks in T_1 is strictly less than q then $|V_1| \leq 4|T_B|$. For base case, consider a case when T_1 has exactly one block. In this case, T_1 is either an edge or a cycle. In either case, $|V_1| \leq 4|T_B|$. In fact, in this case, $|V_1| \leq 4(|T_B| - 1)$ as $|T_B| \geq 2$.

Consider cactus T_1 which has q blocks. Let D be a block corresponding to a leaf in $\mathscr{D}$. Let t be the unique cut vertex in this block. There exists at least one vertex in D, apart from t, which corresponds to a big witness set. If this is not the case then all vertices in $D \setminus \{t\}$ would have been deleted while obtaining T_1 from T. Consider cactus T_1' induced on $V(T) \setminus (D \setminus \{t\})$. Since T_1' has $q-1$ blocks in its block decomposition, by induction hypothesis, $|V_1'| \leq 4|T_B'|$ where $V_1' = V_1 \cap V(T_1')$ and $T_B' = T_B \cap D$. Now, consider the cactus T'' induced on D. Since it is either an edge or cycle, $|V_1''| \leq 4(|T_B''| - 1)$ where $V_1'' = V_1 \cap V(T_1'')$ and $T_B'' = T_B \cap D$.

As t is not in V_1 and it is the only vertex in both $V(T_1')$ and $V(T_1'')$. Hence, we have $|V_1| = |V_1'| + |V_1''|$ and $|T_B| = |T_B'| + |T_B''| - 1$. Substitute the values, we get desired bound for T_1. □

Since every vertex in $N(T_B)$ is correspond to singleton witness set, this bound also applies to the number of desired vertices in G. We present following observation which states that

if (G,k) is an YES instance of CACTUS CONTRACTION then any random 3-coloring of is *good* with certain probability.

Observation 4.6.4. *Consider a 2-connected graph G which is k-contractible to a cactus T. Fix a T-witness structure $\mathscr{W}$ of G. If $\phi : V(G) \to \{1,2,3\}$ is a coloring where colors are chosen uniformly at random for each vertex then ϕ is compatible with $\mathscr{W}$ with probability at least $1/3^{6k}$.*

Proof. Let S be the set of all vertices in $V(G)$ which are either a part of big witness sets in $\mathscr{W}$ or are adjacent to a big witness set and are part of paths between two big witness sets. Since G is k-contractible to T, there are at most k big witness sets. This implies the total number of vertices in S is at most $2k+4k = 6k$ (by Observation 4.6.3). We can ensure $|S| = 6k$ by arbitrarily adding some extra vertices to it. By the definition of compatible coloring (Definition 4.6.1), to determine whether a random coloring ϕ is a compatible with $\mathscr{W}$ or not, we only need to check color of vertices in S.

Let ψ be a 3-coloring of G which is compatible with $\mathscr{W}$. For a random coloring ϕ and a vertex v in S, probability that $\phi(v) = \psi(v)$ is $1/3$. Since colors are chosen uniformly at random for each vertex while constructing ϕ, the probability that ψ and ϕ color S identically is at least $1/3^{6k}$. Hence ϕ is compatible with $\mathscr{W}$ with probability at least $1/3^{6k}$. □

We are now in a position to present first algorithm in this section.

Theorem 4.6.1. *Let (G,k) be an instance of* CACTUS CONTRACTION *where G is a 2-connected graph on n vertices. There is an one-sided error Monte Carlo algorithm with false negatives which determines whether (G,k) is a* YES *instance or not in time $c^k n^{\mathcal{O}(1)}$. It returns correct answer with constant probability.*

Proof. Consider an algorithm which uses Algorithm 4.6.1 as subroutine and runs it 3^{4k} many times. If any of these runs return a solution F, then the algorithm returns F otherwise

Algorithm 4.6.1: Randomized Algorithm for Cactus Contraction

Input: A 2-connected graph G and an integer k

Output: Return a set of edges F such that G/F is a cactus and $|F| \leq k$ if such set exists otherwise return NO.

1 Generate a random coloring $\phi : V(G) \to \{1,2,3\}$ and let $\mathscr{X}$ be the set of colored components.

2 **for** *each* $X \in \mathscr{X}$ **do**

3 | **if** *P is a simple path or a isolated vertex in* $G - X$ **then**

4 | | for all $u \in P$: set color of u to 4

5 **for** *each pair* $X_1, X_2 \in \mathscr{X}$ **do**

6 | **if** *P is a simple maximal path or an isolated vertex in* $G - (X_1 \cup X_2)$ **then**

7 | | for all $u \in P$: set color of u to 5

8 **for** *each* $X \in \mathscr{X}$ **do**

9 | Apply Pruning Operation to obtain marked vertices M_X in X

10 | $Z_X \leftarrow$ minimum connected core of $G[\hat{X}]$ containing M_X

11 | Replace X by Z_X & singleton set for every vertex in $X \setminus Z_X$ in $\mathscr{X}$.

12 **if** *a spanning forest* F *of* $\mathscr{X}$ *has at most* k *edges* **then**

13 | return F

14 **else**

15 | return NO

after all iterations are over, it returns NO. This finishes the description of the algorithm.

We first argue the correctness of this algorithm. Since Algorithm 4.6.1 returns a solution only if it has found a witness structure with desired properties, it never returns false positives. We argue that if these is a solution then the algorithm returns it with constant probability. Consider a graph G which is k-contractible to a cactus T. Fix a T-witness structure $\mathscr{W}$ of G.

To argue the correctness, we first claim that given graph G and a compatible coloring ϕ compatible with $\mathscr{W}$, Algorithm 4.6.1 returns a correct answer. By Lemma 4.6.3, every vertex which is re-colored to 4 in Step 3, is a singleton witness set in $\mathscr{W}$. By Lemma 4.6.4, every vertex which is re-colored to 5 in Step 6, is a singleton witness set in $\mathscr{W}$. At Step 8, each colored component which contains at least two vertices also contains a big witness set (Lemma 4.6.5). This fact allows algorithm to perform Pruning Operation. The correctness of replacement step follows from Lemma 4.6.9. Hence given a coloring ϕ

which is compatible with $\mathscr{W}$, Algorithm 4.6.1 returns a correct answer.

By Observation 4.6.4, any random 3-coloring of G is compatible with $\mathscr{W}$ with probability at least $1/3^{6k}$. Since the algorithm runs 3^{6k} many iterations of Algorithm 4.6.1, probability that none of these colorings (which are generated uniformly at random) is compatible with $\mathscr{W}$ is at most $(1-\frac{1}{3^{6k}})^{3^{6k}} < 1/e$. Hence Algorithm 4.6.1 returns a solution on positive instances with probability at least $1-1/e$.

By Lemma 4.6.10, each iteration of Algorithm 4.6.1 takes $6^k \cdot n^{\mathcal{O}(1)}$ time and hence the total running time of the algorithm is $c^k \cdot n^{\mathcal{O}(1)}$ for a fixed constant c. This concludes the proof of the theorem. □

We apply the arguments presented in [55] to extend above theorem to solve CACTUS CONTRACTION on general graphs.

Theorem 4.6.2. *Let (G,k) be an instance of* CACTUS CONTRACTION *where G is a connected graph on n vertices. There is an one-sided error Monte Carlo algorithm with false negatives which determines whether (G,k) is a* YES *instance or not in time $c^k n^{\mathcal{O}(1)}$. It returns correct answer with constant probability.*

Proof. Let $G_1, G_2, \ldots, G_q$ be 2-connected components of G such that G_i is not a cactus for all i in $[q]$. If $q \leq 1$ then we use algorithm presented in Theorem 4.6.1. If $q \geq k+1$ then we return NO as at least one edge needs to be contracted in each of these 2-connected components. We now consider the case when $2 \leq q \leq k$.

For each G_i and each possible values k_j between 1 and k, we run algorithm in Theorem 4.6.1 on instance (G_i, k_j). We repeat each run $3\log k$ times on each instance. Since there are at most k^2 such pairs, algorithm in Theorem 4.6.1 has been ran at most $3k^2 \log k$ time. If algorithm returns NO for all the values of k_j for some G_i then we return NO. Otherwise let k_i' be the smallest value for which algorithm returns a solution for G_i. Since algorithm in Theorem 4.6.1 returns no false positive, G_i is k_i'-contractible to a cactus. On the other

hand if (G_i, k_i) is a YES instance of CACTUS CONTRACTION then probability that no run will output right answer is at most $(\frac{1}{e})^{3\log k} = \frac{1}{k^3}$. Since there are at most k^2 pairs (G_i, k_j), and by the union bound on probabilities, the probability that there is a pair (G_i, k_j) for which the algorithm returns false negative is upper bounded by $k^2 \cdot \frac{1}{k^3} \geq \frac{1}{k}$. If such a failure does not occur, then for every i we have that k'_i is the smallest value of k_j such that G_i is k_i-contractible to a cactus. Finally, the algorithm answers YES only if $\sum_{i=1}^{q} k'_i \leq k$, and answers NO otherwise. The correctness of this algorithm follows from Lemma 4.2.2. Consequently, the algorithm cannot give false positives, and it may give false negatives with probability at most $1/k \leq 1/q \leq 1/2$, where the two inequalities follows from the assumption that $2 \leq q \leq k$. □

Derandomization

We can derandomize our algorithms by constructing a family of coloring functions that is derived from a universal set.

Definition 4.6.3 (Universal Set)**.** *A (n,k)-universal set is a family $\mathscr{H}$ of subsets of $[n]$ such that for any $S \subseteq [n]$ of size at most k, $\{S \cap H \mid H \in \mathscr{H}\}$ contains all subsets of S.*

Given integers n, k, one can construct a (n,k)-universal set using following result.

Proposition 4.6.1 ([81])**.** *For any $n, k \geq 1$, we can construct a (n,k)-universal set of size $2^k k^{\mathscr{O}(\log k)} \log n$ in time $2^k k^{\mathscr{O}(\log k)} n \log n$.*

We use this (n,k)-universal set to construct a 3-coloring family of $V(G)$.

Lemma 4.6.11. *Consider a graph G and a subset S of $V(G)$ of size 6k. There is a family of 3-coloring functions, $\mathscr{F}$, such that for a given 3-coloring ϕ of $V(G)$, there exists coloring ψ in $\mathscr{F}$ that agrees with ϕ on S. This family has size $4^{6k} k^{\mathscr{O}(\log k)} \log^2 n$ and it can be constructed in time $4^{6k} k^{\mathscr{O}(\log k)} n \log^2 n$.*

Proof. Let coloring ϕ partitions S into 3 parts, say S_1, S_2, S_3. Let $\mathscr{H}$ be a $(n, 6k)$-universal set, that is constructed by Proposition 4.6.1. We define a family of partitions of $V(G)$ as follows.

$$\mathscr{F}' = \{(A,B,C) \mid A \in \mathscr{H}, B = Y \setminus A \text{ where } Y \in \mathscr{H}, C = V(G) \setminus Y\}$$

Observe that $\mathscr{F}'$ can be constructed by considering each pair of sets in $\mathscr{H}$. We claim that there is a triple $(A,B,C) \in \mathscr{F}'$ such that $S \cap A = S_1$, $S \cap B = S_2$ and $S \cap C = S_3$. Since $\mathscr{H}$ is a $(n, 6k)$-universal-set, there is some set $Y \in \mathscr{H}$ such that $S \cap Y = S_1 \cup S_2$, and there is some $A \in \mathscr{H}$ such that $A \cap S = S_1$. Hence, $S \cap (Y - A) = (S \cap Y) \setminus (S \cap A) = S_2$. We can easily convert the family $\mathscr{F}'$ into a family of coloring functions, where for each $(A,B,C) \in \mathscr{F}'$ maps all vertices in A, B, C to $1, 2, 3$ respectively. Hence if ϕ partitions S into S_1, S_2, S_3, then there is a function $\psi \in \mathscr{F}$, which also partitions S into S_1, S_2, S_3. Since the family $\mathscr{H}$ has size $2^{6k}k^{\mathscr{O}(\log k)} \log n$, size of $\mathscr{F}$ is at most $4^{6k}k^{\mathscr{O}(\log k)} \log^2 n$ and it can be constructed in time $4^{6k}k^{\mathscr{O}(\log k)} n \log^2 n$ □

In the algorithm mentioned in Theorem 4.6.2, instead of repeatedly generating random a random coloring, we use coloring family mentioned in Lemma 4.6.11, to get following result.

Theorem 4.6.3. *Let (G,k) be an instance of* CACTUS CONTRACTION *where G is a connected graph on n vertices. There is a deterministic algorithm which determines whether (G,k) is a* YES *instance or not in time $c^k n^{\mathscr{O}(1)}$.*

4.7 Conclusion

In this chapter, we take a closer look at methods developed for TREE CONTRACTION problems and generalize it for CACTUS CONTRACTION. We prove that CACTUS CONTRACTION does not have a polynomial kernel when parameterized by solution size. We

compliment this result in two ways. We present a polynomial kernel for CACTUS CONTRACTION using solution size and number of leaves in resulting cactus as combined parameter. We argue that this kernel is optimal under certain polynomial complexity assumption. We also present a lossy kernel of polynomial size for this problem. We end this chapter with an FPT algorithm running in time $c^k n^{\mathcal{O}(1)}$ for this problem.

Chapter 5

Contraction to Generalization of Trees

5.1 Introduction

As in Chapter 4, we study a problem of contracting an input graph to a graph class which is superset of trees. We define this graph class in a "parameterized way". Let $\mathbb{T}_\ell$ be a collection of graph which can be made into a tree by deleting at most ℓ edges. In other words, $\mathbb{T}_\ell$ is a set of connected graph whose feedback edge set is of size at most ℓ. We study $\mathbb{T}_\ell$-CONTRACTION problem which is formally defined as follows.

$\mathbb{T}_\ell$-CONTRACTION	**Parameter:** k
Input: A graph G and an integer k.	
Question: Is it possible to obtain a graph in $\mathbb{T}_\ell$ from G with at most k edge contractions?	

Note that for $\ell = 0$, problem $\mathbb{T}_\ell$-CONTRACTION is same as TREE CONTRACTION. In Section 5.3, we show that the problem does not admit a polynomial kernel when parameterized by k (alone). In fact, this reduction proves that the problem does not admit a polynomial kernel when parameterized by k for any (fixed) integer ℓ. This implies that, unlike in case of BOUNDED TREE CONTRACTION, we can not get a polynomial kernel when parameterized by k and additional parameter ℓ. Inspired by this negative result for kernelization, we

design an α-lossy kernel for $\mathbb{T}_\ell$-CONTRACTION of size $\mathcal{O}([k(k+2\ell)]^{(d+1)})$ in Section 5.4, where $d = \lceil \frac{\alpha}{\alpha-1} \rceil$. Note that this lossy kernel has polynomial dependency on both k and ℓ.

In Section 5.5, we design an FPT algorithm for $\mathbb{T}_\ell$-CONTRACTION running in time $(2\sqrt{\ell}+2)^{\mathcal{O}(k+\ell)} \cdot n^{\mathcal{O}(1)}$. Our algorithm follows the general approach of designing the algorithm for TREE CONTRACTION by Heggernes et al. [55]. They presented a randomized algorithm for the problem and then derandomize it using (n,k)-*universal sets*. A (n,k)-universal sets is a collection $\mathscr{F}$ of functions from $[n]$ to $[2]$ such that for each subset S of $[n]$ of size k and a function $\phi : S \to [q]$, there exists function f in $\mathscr{F}$ such that f restricted on set S is identical with ϕ. We rename (n,k)-universal sets to $(n,k,2)$-*universal family*. In Section 5.6, we present an algorithm to compute (n,k,q)-universal family for any integer $q \geq 2$. We use these families to de-randomize our algorithm for $\mathbb{T}_\ell$-CONTRACTION.

The work presented in this chapter is based on [3].

5.2 Preliminaries

For a given graph, *feedback edge set* is defined as set of edges whose removal deletes all cycles in the graph. For an integer ℓ, by $\mathbb{T}_\ell$ we denote collection of all connected graphs which has a feedback edge set of size at most ℓ. It can also be defined as collection of graphs which can be obtain from a tree by adding at most ℓ edges. For any graph G in $\mathbb{T}_\ell$, we have $|E(G)| \leq |V(G)| - 1 + \ell$. For a connected graph H, if $|E(H)| \leq |V(H)| - 1 + \ell$ then H is in $\mathbb{T}_\ell$.

A k-coloring of a graph G is a function $\phi : V(G) \to [k]$. A k-coloring ϕ of G is a *proper coloring* if for all uv in $E(G)$ we have $\phi(u) \neq \phi(v)$. The *chromatic number* of a graph is the minimum number of colors needed for its proper coloring. For a k-coloring ϕ of G, a subset S of $V(G)$ is said to be *monochromatic* with respect to ϕ if for all s, s' in S, $\phi(s) = \phi(s')$. Observe that ϕ partitions $V(G)$ into (at most) k pairwise disjoint sets. A subset S of $V(G)$ is said to be *monochromatic component* with respect to ϕ if S is monochromatic and $G[S]$

is connected.

We start with few observation regarding the graph class $\mathbb{T}_\ell$.

Observation 5.2.1. *For each $T \in \mathbb{T}_\ell$ the following statements hold.*

1. *The chromatic number of T is at most $2\sqrt{\ell}+2$.*
2. *If T' is a graph obtained by subdividing an edge in T then $T' \in \mathbb{T}_\ell$.*
3. *If T' is a graph obtained by contracting an edge in T then $T' \in \mathbb{T}_\ell$.*

Proof. (Proof of Part 1.) We first prove that for any graph G with at least one edge, its chromatic number is upper bounded by $2\sqrt{|E(G)|}$. Let $C_1, C_2, \ldots, C_q$ be the color classes in a proper coloring of G which uses the minimum number of colors. Observe that there is at least one edge between C_i, C_j, where $i, j \in [q]$, $i \neq j$. This implies that $\binom{q}{2} \leq |E(G)|$, which proves the claim. Next, consider $T_\ell \in \mathbb{T}_\ell$, and fix a spanning tree T of T_ℓ. Let $T' = E(T_\ell) \setminus E(T)$. If $\ell > 0$ then from the claim above, we can properly color graph $T_\ell[V(T')]$ using at most $2\sqrt{\ell}$ many colors. Since $T_\ell - T'$ is a tree, we can properly color T_ℓ by coloring the vertices in $T_\ell - V(T')$ using two new colors.

(Proof of Part 2.) For any connected graph T if $|E(T)| \leq |V(T)| - 1 + \ell$ then T is contained in $\mathbb{T}_\ell$. Subdividing an edge adds a new vertex and an edge and hence this inequality is satisfied while maintaining the connectivity of graph. This implies $T' \in \mathbb{T}_\ell$, where T' is obtained from T by sub-dividing an edge in T.

(Proof of Part 3.) Similar to the proof of part 2, contracting an edge decreases the number of vertices by one and number of edges by at least one. This implies $|E(T')| \leq |V(T')| - 1 + \ell$. Contracting an edge maintains the connectivity of the graph and hence $T' \in \mathbb{T}_\ell$. □

Observation 5.2.2. *For a graph $T \in \mathbb{T}_\ell$, the graph $T' \in \mathbb{T}_\ell$ whenever T' is obtained from T as follows. Consider a vertex $v \in V(T)$, and a partition N_1, N_2 of $N_T(v)$. Let $V(T') = (V(T) \setminus \{v\}) \cup \{v_1, v_2\}$ and $E(T') = E(T - \{v\}) \cup \{(v_1, u) \mid u \in N_1\} \cup \{(v_2, u) \mid u \in N_2\} \cup \{(v_1, v_2)\}$.*

Proof. Consider a vertex $v \in V(T)$, and a partition N_1, N_2 of $N_T(v)$. Let $V(T') = (V(T) \setminus \{v\}) \cup \{v_1, v_2\}$ and $E(T') = E(T - \{v\}) \cup \{(v_1, u) \mid u \in N_1\} \cup \{(v_2, u) \mid u \in N_2\} \cup \{(v_1, v_2)\}$. Notice that T' is a connected graph. We have $|V(T')| = |V(T)| + 1$ and $|E(T')| = |E(T)| + 1 \leq |V(T)| - 1 + \ell + 1 = |V(T')| - 1 + \ell$. This concludes the proof. □

Following lemma helps us find an edge which can safely be contracted.

Lemma 5.2.1. *Let (G,k) be an instance of $\mathbb{T}_\ell$-*CONTRACTION *and $P = (u_0, u_1, \ldots, u_q, u_{q+1})$ be a path in G, where $q \geq k+2$, and for each $i \in [q]$ we have $deg(u_i) = 2$. Then no minimal solution F to $\mathbb{T}_\ell$-*CONTRACTION *in (G,k) with $|F| \leq k$ contains an edge incident to $V(P) \setminus \{u_0, u_{q+1}\}$.*

Proof. Assume the contrary that F contains at least one such edge. Observe that there are at least $k+1$ edges with endpoints in $V(P) \setminus \{u_0, u_{q+1}\}$. Therefore, there exists $i \in [q]$ such that $u_{i-1}u_i \in F$ and $u_iu_{i+1} \notin F$, or $u_{i-1}u_i \notin F$ and $u_iu_{i+1} \in F$. Let us assume that there exists $i \in [q]$ such that $u_{i-1}u_i \in F$ and $u_iu_{i+1} \notin F$ (other case is symmetric). Let $T = G/F$ with $V(T) = \{t_1, \cdots, t_p\}$, and $\mathscr{W}$ be the T-witness structure of G. Furthermore, let t and t' be the vertices in T such that $u_{i-1}, u_i \in W(t)$ and $u_{i+1} \in W(t')$. If $t = t'$ then consider the following. Notice that $G[W(t)]$ is connected, $u_{i-1}, u_i, u_{i+1} \in W(t)$, and $u_iu_{i+1} \notin F$. Therefore, $W(t)$ must contain the vertices of the sub-path $(u_{i+1}, \ldots, u_q, u_{q+1})$ and the vertices of the subpath $(u_0, u_1, \ldots, u_{i-1}, u_i)$. But then, we have $|W(t)| > k+1$, a contradiction. Therefore, we have $t \neq t'$. Notice that u_i is not a cut vertex in $G[W(t)]$, as there is exactly one edge incident on it. Therefore, $G[W(t) \setminus \{u_i\}]$ is connected. Let $\mathscr{W}' = (\mathscr{W} \setminus \{W(t)\}) \cup \{u_i\} \cup \{W(t) \setminus \{u_i\}\}$. Observe that $\mathscr{W}'$ is a partition of $V(G)$ which is a G/F'-witness structure of G, where $F' = F \setminus \{u_{i-1}u_i\}$. Here, G/F' is the graph obtained by subdividing the edge tt' in T, and by Observation 5.2.1, G/F' is also a graph in $\mathbb{T}_\ell$, which contradicts the minimality of F. □

We end this section with two lemmas regarding witness structure of input graph.

Lemma 5.2.2. *Let F be a set of edges in a graph G such that $T = G/F$ is in $\mathbb{T}_\ell$ and $|V(G/F)| \geq 3$, and $\mathscr{W}$ be a T-witness structure of G. Then, there exists a set F' of at most $|F|$ edges in G such that G/F' is in $\mathbb{T}_\ell$ and the G/F'-witness structure $\mathscr{W}'$ of G satisfies the property that for every leaf t in G/F', witness set $W'(t)$ in $\mathscr{W}'$ is a singleton set.*

Proof. If for each leaf $t \in V(T)$ we have $|W(t)| = 1$ then $F' = F$ is a desired solution. Otherwise, consider a leaf t in T such that $|W(t)| > 1$. Let t' be the unique neighbour of t in T. Notice that $W(t)$ and $W(t')$ are adjacent in G, and $G[W(t) \cup W(t')]$ is connected. Fix a spanning tree Q of $G[W(t)]$, and (arbitrarily) choose a vertex $u^* \in V(Q)$ that is adjacent to a vertex in $W(t')$, which exists since $tt' \in E(T)$. Furthermore, choose a leaf $v^* \in V(Q) \setminus \{u^*\}$, which exists as $|V(Q)| > 1$. Let $W'(t') = (W(t') \cup W(t)) \setminus \{v^*\}$, $W'(t) = \{v^*\}$, and $\mathscr{W}' = (\mathscr{W} \setminus \{W(t), W(t')\}) \cup \{W'(t), W'(t')\}$. Notice that $\mathscr{W}'$ is a T-witness structure of G, and the number of leaves corresponding to singleton witness sets is strictly more than that of $\mathscr{W}$. Hence, by repeating this argument for each (non-adjacent) leaves in T and their corresponding witness set in $\mathscr{W}$, we can obtain the desired result. □

Lemma 5.2.3. *Let F be a set of edges in a graph G such that $T = G/F$ is in $\mathbb{T}_\ell$, and $\mathscr{W}$ be a T-witness structure of G. If G is 2-connected and t is cut vertex in T then witness set $W(t)$ in $\mathscr{W}$ is not a singleton set.*

Proof. Let t be a cut vertex in T such that $W(t) = \{u\}$, where $u \in V(G)$. Notice that $T - \{t\}$ has at least two components, say T_1 and T_2. Consider $U_1 = \bigcup_{t \in V(T_1)} W(t)$ and $U_2 = \bigcup_{t \in V(T_2)} W(t)$. As $\mathscr{W}$ is a T-witness structure of G, it follows that there is no edge between a vertex in U_1 and a vertex in U_2 in G. This contradicts the fact that G is 2-connected. □

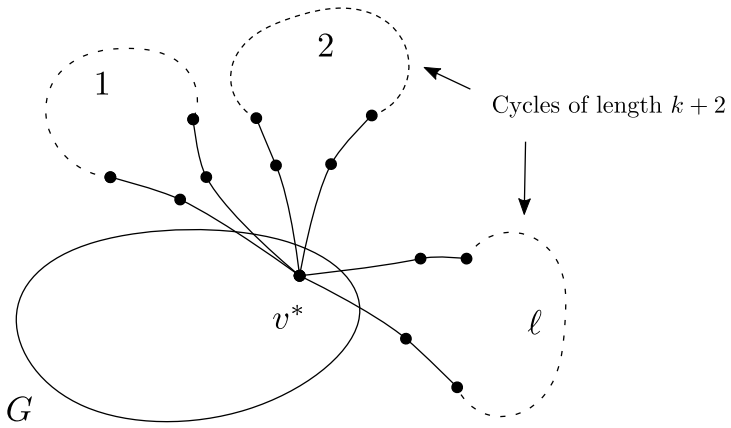


Figure 5.1: Reduction from TREE CONTRACTION to $\mathbb{T}_\ell$-CONTRACTION.

5.3 Hardness results for $\mathbb{T}_\ell$-CONTRACTION

Following reduction shows that $\mathbb{T}_\ell$-CONTRACTION is NP-Hard. Moreover, it implies that the problem does not admit a polynomial kernel when parameterized by k unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. We present a parameter preserving reduction from TREE CONTRACTION which as we have mentioned does not admit a polynomial kernel under same complexity assumption [55].

Reduction. Let (G,k) be an instance of TREE CONTRACTION. We create an instance (G',k') of $\mathbb{T}_\ell$-CONTRACTION as follows. Initially, we have $G = G'$. Let v^* be an arbitrarily chosen vertex in $V(G)$. For each $i \in [\ell]$, we add a cycle $(v^*, w_1^i, w_2^i, \ldots, w_{k+1}^i)$ on $k+2$ vertices to G', which pairwise intersect at v^*, and we set $k' = k$. This completes the description of the reduction. See Figure 5.1.

In the following lemma we establish equivalence between the two instances.

Lemma 5.3.1. *(G,k) is a* YES *instance of* TREE CONTRACTION *if and only if (G',k') is a* YES *instance of* $\mathbb{T}_\ell$-CONTRACTION.

Proof. In the forward direction, let (G,k) be a YES instance of TREE CONTRACTION, and S be one of its solution. Notice that $G'/S \in \mathbb{T}_\ell$, and $|S| \leq k' = k$. Therefore, (G',k')

is a YES instance of $\mathbb{T}_\ell$-CONTRACTION. In the reverse direction, let (G',k') be a YES instance of $\mathbb{T}_\ell$-CONTRACTION, and S be one of its (minimal) solution. Recall that for each $i \in [\ell]$ we have a cycle $C_i = (v^*, w_1^i, w_2^i, \ldots, w_{k+1}^i)$ on $k+2$ vertices in G', which pairwise intersect at v^*. This together with minimality of $|S|$ implies that $S \cap E(C_i) = \emptyset$. Furthermore, $G'[\{v^*\} \cup (\cup_{i\in[\ell]} V(C_i))]$ belongs to $\mathbb{T}_\ell \setminus \mathbb{T}_{\ell-1}$. Therefore, $G'[V(G)]/S$ must be a tree. □

Following theorem follows from construction of an instance (G',k') of $\mathbb{T}_\ell$-CONTRACTION for a given instance (G,k) of TREE CONTRACTION and Lemma 5.3.1.

Theorem 5.3.1. $\mathbb{T}_\ell$-CONTRACTION *does not admit a polynomial kernel parameterized by solution size unless* NP $\subseteq$ coNP/poly.

5.4 Lossy Kernel for $\mathbb{T}_\ell$-CONTRACTION

To complement the result that $\mathbb{T}_\ell$-CONTRACTION does not admit a polynomial kernel assuming NP $\not\subseteq$ coNP/poly (Section 5.3) we design a PSAKS for $\mathbb{T}_\ell$-CONTRACTIONin this section. For a given graph G, a set of edges F in G is said to a solution if G/F is in $\mathbb{T}_\ell$. We define parameterized minimization version of $\mathbb{T}_\ell$-CONTRACTION problem in the following way.

$$\mathbb{T}_\ell\mathrm{C}(G,k,F) = \begin{cases} \infty & \text{if F is not a solution} \\ \min\{|F|, k+1\} & \text{otherwise} \end{cases}$$

If G has at most $k+3$ vertices then we already have a kernel of desired size. We assume that input graph has at least $k+3$ vertices. By definition of optimization problem, for a set of edges F, if G/F is a graph in $\mathbb{T}_\ell$ then maximum value $\mathrm{TC}(G,k,F)$ is $k+1$. Hence any spanning tree of G is a solution of cost $k+1$. We call it a *trivial solution* for given instance. Consider complete graph $K_{\ell+4}$. One need to contract at least two edges from this

graph to obtain a graph in $\mathbb{T}_\ell$. Hence any solution for instance $(K_{\ell+4}, 1)$ is of cost two. We call this instances as *trivial instance* for $\mathbb{T}_\ell$C problem. If we are able to conclude that an optimum solution for instance (G,k) is of size at least $k+1$ then we can return this trivial instance as its lossy kernel. Note that for any c-factor solution of this trivial instance, we can return of a trivial solution for original problem which is of cost $k+1$. If input graph is not connected, we can not obtain a tree by edge contraction operations only. We assume that input graph is connected.

The algorithm starts by applying Reduction Rules 5.5.1 to 5.5.4 (if applicable, in that order). Next, we state the following lemma which prove that we can shorten long induced paths without changing the value of optimum solution.

In following reduction rule we find an edge, if exists, which can be safely contracted.

Reduction Rule 5.4.1. *If G has a path $P = (u_0, u_1, \ldots, u_q, u_{q+1})$ such that $q > k+2$ and for all $i \in [q]$, we have $deg(u_i) = 2$. Then contract the edge $u_{q-1}u_q$,* i.e. *the resulting instance is $(G/\{u_{q-1}u_q\}, k)$.*

Note that Reduction Rule 5.4.1 can be applied in polynomial time by searching for such a path (if it exists) in the subgraph induced on the vertices of degree 2 in G. In the following lemma, we show that Reduction Rule 5.4.1 is safe.

Lemma 5.4.1. *Reduction Rule 5.4.1 is* 1*-safe.*

Proof. Let $P = (u_0, u_1, \ldots, u_q, u_{q+1})$ be a path in G such that $q > k+2$ and for all $i \in [q]$, we have $deg(u_i) = 2$. Furthermore, let $G' = G/\{u_{q-1}u_q\}$, $P' = (u_0, u_1, \ldots, u_{q-2}, u^*, u_{q+1})$, where u^* is the vertex resulting after contracting the edge $u_{q-1}u_q$. We consider the instances (G,k) and (G',k) of $\mathbb{T}_\ell$-CONTRACTION, and show that $\frac{\mathbb{T}_\ell\text{C}(G,k,F)}{\text{OPT}(G,k)} \leq \frac{\mathbb{T}_\ell\text{C}(G',k',F')}{\text{OPT}(G',k')}$. Here, $\mathbb{T}_\ell$C is a shorthand notation for the parameterized minimization problem for $\mathbb{T}_\ell$-CONTRACTION.

Consider a minimal set $F' \subseteq E(G')$ such that $T' = G'/F'$ is in $\mathbb{T}_\ell$. If $|F'| \geq k+1$, then the solution lifting algorithm returns $E(G)$, otherwise it returns $F = F'$. If $|F'| \geq k+1$ then

$\mathbb{T}_\ell\mathrm{C}(G,k,F) \leq k+1 = \mathbb{T}_\ell\mathrm{C}(G',k,F')$. Otherwise, let $V(T') = \{t_1,\cdots,t_r\}$ and $\mathscr{W}'$ denote the T'-witness structure of G'. By Lemma 5.2.1, F' has no edge incident on vertices in $V(P)\setminus\{u_0,u_{q+1}\}$. Therefore, every vertex in $V(P')\setminus\{u_0,u_{q+1}\}$ is in a singleton set of $\mathscr{W}'$. Let $\mathscr{W} = (\mathscr{W}'\setminus\{u^*\})\cup\{\{u_{q-1}\},\{u_q\}\}$ to be a partition of $V(G)$. Then, $\mathscr{W}$ is a T-witness structure of G where T is G/F, which is obtained from T' by subdividing an edges. From Observation 5.2.1, T is in $\mathbb{T}_\ell$. Therefore, $\mathbb{T}_\ell\mathrm{C}(G,k,F) \leq \mathbb{T}_\ell\mathrm{C}(G',k,F')$.

Next, consider an optimum solution F^* to $\mathbb{T}_\ell$-CONTRACTION in (G,k). If $|F^*| \geq k+1$ then $\mathrm{OPT}(G,k) = k+1$ and by definition, $\mathrm{OPT}(G',k) \leq k+1 = \mathrm{OPT}(G,k)$. Otherwise, we have $|F^*| \leq k$. Let $T = G/F^*$, and $\mathscr{W}$ be the T-witness structure of G. By Lemma 5.2.1, F^* has no edge incident on $V(P)\setminus\{u_0,u_{q+1}\}$. Therefore, every vertex in $V(P)\setminus\{u_0,u_{q+1}\}$ is in a singleton set in $\mathscr{W}$. Let $\mathscr{W}' = (\mathscr{W}\setminus\{\{u_{q-1}\},\{u_q\}\})\cup\{\{u^*\}\}$ be a partition of $V(G')$. Then, $\mathscr{W}'$ is a T'-witness structure of G', where $T' = G'/F^*$. Finally, T' is the graph obtained from T by contracting an edge. Hence, $T' \in \mathbb{T}_\ell$, and $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$. Hence, we have $\frac{\mathbb{T}_\ell\mathrm{C}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathbb{T}_\ell\mathrm{C}(G',k',F')}{\mathrm{OPT}(G',k')}$. □

Let (G,k) be an instance obtained by exhaustively applying Reduction Rule 5.4.1 on input instance. At this stage, we derive a structural property of graph G which is k-contractible to a graph in $\mathbb{T}_\ell$. In the following lemma, we argue that such graphs should have a small connected vertex cover.

Lemma 5.4.2. *Consider an instance (G,k) of $\mathbb{T}_\ell$-CONTRACTION on which Reduction Rule 5.4.1 is not applicable. If G is k-contractible to a graph in $\mathbb{T}_\ell$ then G has a connected vertex cover of size at most $2(k+3)(k+2\ell)$.*

Proof. Let (G,k) be a instance of $\mathbb{T}_\ell$-CONTRACTION such that G is k-contractible to a graph in $\mathbb{T}_\ell$. Let F be one of its solution, $T = G/F$, where $T \in \mathbb{T}_\ell$, and $\mathscr{W}$ be the T-witness structure of G. Let L be the set of leaves in T, and $X = V(T)\setminus L$. If $|V(T)| \leq 2$ then the claim trivially holds since $|F| \leq k$. Otherwise, we have $|V(T)| \geq 3$. In this case, by Lemma 5.2.2 we can assume that each vertex in L belongs to a singleton witness set in

$\mathscr{W}$. Notice that for $t_i, t_j \in L$, where $t_i \neq t_j$, and $W(t_i) = \{u\}$ and $W(t_j) = \{v\}$ we have $t_i t_j \notin E(T)$ (since $|V(T)| \geq 3$), and therefore $uv \notin E(G)$. As $T[X]$ is connected, it follows that $S = \bigcup_{t \in X} W(t)$ is a connected vertex cover of G. We now argue that $|S|$ is at most $2(k+3)(k+2\ell)$.

Let $X_1 \subseteq X$ be the set comprising of vertices in T such that for each $t \in X_1$ we have $|W(t)| > 1$, and $X_2 = X \setminus X_1$. Since Reduction Rule 5.5.4 is not applicable on G, we can assume that every leaf in T is adjacent to a vertex in X_1. Notice that any connected induced subgraph of T is in $\mathbb{T}_\ell$. Fix a spanning tree of $T - L$, and let F be the set of edges which are not in this spanning tree. Since, $T - L \in \mathbb{T}_\ell$ therefore, we have $|F| \leq \ell$. Next, we create a set of marked vertices M. We add both the endpoints of edges in F to M, and add vertices in X_1 to M. Consider a graph T' obtained from $T - L$ by deleting edges in F and contracting all vertices with degree exactly two in the graph $T - L$. It is easy to see that T' is a tree with all its leaves marked and every internal vertex of degree at least 3. Hence the number of vertices in T' is at most twice the number of marked vertices. Since there are at most $k + 2\ell$ marked vertices, we get $|V(T')| \leq 2(k+2\ell)$. Every edge in $E(T')$ corresponds to a simple path (or an edge) in T. Recall that the number of internal vertices in each such path is bounded by $k+2$ as Reduction Rule 5.4.1 is not applicable. Hence, $|X_2|$ is at most $2(k+2)(k+2\ell)$. Since, there are at most k more vertices in $W(t)$ for $t \in X_1$, $|S|$ is at most $2(k+2)(k+2\ell)+k$. This concludes the proof of lemma. □

Using Lemma 5.4.2, we can identify graphs which are not k-contractible to a graph in $\mathbb{T}_\ell$. Note that we have 2-factor approximation algorithm to find a connected vertex cover of input graph. It is easy to see that following reduction rule is 1-safe.

Reduction Rule 5.4.2. *Given an instance (G,k), apply 2-factor approximation algorithm to compute a connected vertex cover X of G. If size of X is greater than $4(k+3)(k+2\ell)$ then return the trivial instance $(K_{\ell+4}, 1)$.*

Lemma 5.4.3. *Reduction Rule 5.4.2 is 1-safe.*

Proof. Let (G,k) be an instance such that Reduction Rule 5.4.2 returns $(K_{\ell+4},1)$ when applied on it. Solution lifting algorithm returns a spanning tree F of G.

Note that for a set of edges F', if $K_{\ell+4}/F'$ is a graph in $\mathbb{T}_\ell$ then F' contains at least two edges. This implies $\mathbb{T}_\ell\text{C}(K_{\ell+4},1,F') = 2$ and $\text{OPT}(K_{\ell+4},1) = 2$.

Since a 2-factor approximation algorithm returns a set of size strictly more than $4(k+3)(k+2\ell)$, size of minimum connected vertex cover of G is at least $2(k+3)(k+2\ell)$. But by Lemma 5.4.2, if G is k-contractible to a graph in $\mathbb{T}_\ell$ than it has a connected vertex cover of size at most $2(k+3)(k+2\ell)$. Hence for any set of edges F^* if G/F^* is in $\mathbb{T}_\ell$ than size of F^* is at least $k+1$. This implies $\text{OPT}(G,k) = k+1$. For a spanning tree F of G, $\text{TC}(G,k,F) = k+1$.

Combining these values, we get $\frac{\mathbb{T}_\ell\text{C}(G,k,F)}{\text{OPT}(G,k)} = \frac{k+1}{k+1} = \frac{2}{2} = \frac{\mathbb{T}_\ell\text{C}(K_{\ell+4},1,F')}{\text{OPT}(K_{\ell+4},1)}$. This implies if F' is c-factor approximate solution for $(K_{\ell+4},1)$ then F is 1-factor approximate solution for (G,k). This concludes the proof. □

For the remaining section, we assume that above reduction rule did not return the trivial instance. In other words, we assume that graph has connected vertex cover of size at most $4(k+3)(k+2\ell)$. Before describing the next reduction rule, we define a partition of $V(G)$ into the following sets.

$$H = \{u \in V(G) \mid deg(u) \geq 2(k+3)(k+2\ell)+1\}$$

$$I = \{v \in V(G) \setminus H \mid N(v) \subseteq H\}$$

$$R = V(G) \setminus (H \cup I)$$

Vertices v,u are said to be *false twins* if $N(v) = N(u)$. We use Lemma 5.4.4 to reduce the number of vertices in I which have many false twins. Let G be k-contractible to a graph T in $\mathbb{T}_\ell$ and $\mathscr{W}$ be the T-witness structure of G.

Lemma 5.4.4. *Consider sets $X, U \subseteq V(G)$ such that U is an independent set in G and for all $v \in U$ we have $X \subseteq N(v)$. If $|U| \geq k+\ell+2$ then there is a vertex $t \in V(T)$ such that $X \subseteq W(t)$.*

Proof. We prove this by contradiction. Assume there exists $t \neq t'$ such that $X \cap W(t)$ and $X \cap W(t')$ are non-empty. Since U is an independent set and $|U| \geq k+\ell+2$, there are at least $\ell+2$ vertices in U which are not contained in any big witness sets. Consider the subgraph of T (on at least $\ell+4$ vertices) induced on the vertices $\{t,t'\} \cup \{t_i \mid W(t_i)$ is a singleton witness set containing a vertex in $U\}$. After deleting any set of ℓ edges in T, there still exists a cycle in T. This is a contradiction the fact that $T \in \mathbb{T}_\ell$. □

Recall that two vertices are said to be *false twins* of each other if their open neighbourhoods are same. Following reduction rule removes a vertex in I which has too many false twins.

Reduction Rule 5.4.3. *If there is a vertex $v \in I$ that has at least $k+\ell+2$ false twins in I then delete v,* i.e. *the resulting instance is $(G-\{v\}, k)$.*

In the following lemma we prove that this reduction rule is 1-safe.

Lemma 5.4.5. *Reduction Rule 5.4.3 is 1-safe.*

Proof. Let $v \in I$ such that v has at least $k+\ell+2$ false twins in I, and let $G' = G-\{v\}$. We consider instances (G,k) and (G',k) of $\mathbb{T}_\ell$-CONTRACTION, and show that $\frac{\mathbb{T}_\ell\mathrm{C}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathbb{T}_\ell\mathrm{C}(G',k,F')}{\mathrm{OPT}(G',k)}$. Here, $\mathbb{T}_\ell$C is a shorthand notation for the parameterized minimization problem for $\mathbb{T}_\ell$-CONTRACTION.

Consider a solution F' to $\mathbb{T}_\ell$-CONTRACTION in (G',k). If $|F'| \geq k+1$ then the solution lifting algorithm returns $E(G)$, otherwise it returns $F = F'$. If $|F'| \geq k+1$ then $\mathbb{T}_\ell\mathrm{C}(G,k,F) \leq k+1 = \mathbb{T}_\ell\mathrm{C}(G',k,F')$. Otherwise, $|F'| \leq k$, and let $T' = G'/F$, where $T' \in \mathbb{T}_\ell$ with $\mathscr{W}'$ being the T'-witness structure of G'. Let U be set of false twins of v in I. Recall that $|U| \geq k+\ell+2$. From Lemma 5.4.4, there exists $t_i \in V(T')$ such

that $N_{G'}(u_1) \subseteq W'(t_i)$ for u_1 in U. Let T be the graph obtained from T' by adding a new vertex t_v as a leaf adjacent to t_i. Notice that $T \in \mathbb{T}_\ell$, which follows from the fact that $N_{G'}(u_1) = N_G(u_1) = N_G(v)$, and $N_G(u_1) \subseteq W'(t_i)$. Let $\mathscr{W} = \mathscr{W}' \cup \{\{v\}\}$ be a partition of $V(G)$. Then, T is G/F and $\mathscr{W}$ is the T-witness structure of G. Hence, $\mathbb{T}_\ell\mathrm{C}(G,k,F) \leq \mathbb{T}_\ell\mathrm{C}(G',k,F')$.

Next, consider an optimum solution F^* to $\mathbb{T}_\ell$-CONTRACTION in (G,k). If $|F^*| \geq k+1$ then by definition, $\mathrm{OPT}(G,k) \leq k+1 = \mathrm{OPT}(G,k)$. Otherwise, we have $|F^*| \leq k$. Let $T = G/F^*$, and $\mathscr{W}^*$ denote the T-witness structure of G. By an argument analogous to the proof of $\mathbb{T}_\ell\mathrm{C}(G,k,F) \leq \mathbb{T}_\ell\mathrm{C}(G',k',F')$, we know that there exists $t_j \in V(T)$ such that $N(v) \subseteq W(t_j)$. Let $t \in V(T)$ such that $v \in W(t)$. If $W(t) = \{v\}$ then t is a leaf in T, which implies that F^* is also a solution to $\mathbb{T}_\ell$-CONTRACTION in (G',k), thus giving the desired relation. Otherwise, consider the following. Recall that v has at least $k+\ell+2$ false twins, and at least one of them, say u, belongs to a singleton witness set. That is, there exists a vertex t' in T such that $W(t') = \{u\}$. Let $\mathscr{W}'$ be the partition of $V(G)$ obtained from $\mathscr{W}^*$ by swapping the appearances of u and v. Furthermore, let F' be the set of edges obtained from F by replacing each edge xv with the edge xu, where for each $xv \in F$. Notice that F' is also an optimal solution to $\mathbb{T}_\ell$-CONTRACTION in (G,k), and a solution to $\mathbb{T}_\ell$-CONTRACTION in (G',k). Therefore, $\mathrm{OPT}(G',k) \leq \mathrm{OPT}(G,k)$. Hence, $\frac{\mathbb{T}_\ell\mathrm{C}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathbb{T}_\ell\mathrm{C}(G',k,F')}{\mathrm{OPT}(G',k)}$. □

For $\alpha > 1$, let d be the smallest integer such that $\frac{d+1}{d} \leq \alpha$. This implies $d = \lceil \frac{\alpha}{\alpha-1} \rceil$.

Reduction Rule 5.4.4. *If there are vertices $v_1, v_2, \cdots, v_{k+\ell+2} \in I$ and $h_1, h_2, \cdots, h_d \in H$ such that for all $i \in [k+\ell+2]$, we have $\{h_1, \ldots, h_d\} \subseteq N(v_i)$ then contract all edges in $\tilde{E} = \{v_1 h_i \mid i \in [d]\}$, and decrease k by $d-1$. The resulting instance is $(G/\tilde{E}, k-d+1)$.*

We note that the *lossy-ness* is introduced only in the Reduction Rule 5.4.4. We have determined that $H' = \{h_1, h_2, \ldots, h_d\}$ need to be in one witness bag but $G[H']$ may not be connected. To simplify the graph, we introduce additional vertex v_1 to the bag which contains H'. By doing this we are able to contract $H' \cup \{v_1\}$ into a single vertex. In the

following lemma, we argue that the number of extra edge contracted in this process is α times that of the optimum solution.

Lemma 5.4.6. *Reduction Rule 5.4.4 is α-safe.*

Proof. Let $v_1, v_2, \cdots, v_{k+\ell+2} \in I$ and $h_1, h_2, \cdots, h_d \in H$ such that for all $i \in [k+\ell+2]$, we have $\{h_1, \ldots, h_d\} \subseteq N(v_i)$. Furthermore, let $\tilde{E} = \{v_1 h_i \mid i \in [d]\}$, $G' = G/\tilde{E}$, and $k' = k - d + 1$. We consider instances (G,k) and (G',k') of $\mathbb{T}_\ell$-CONTRACTION, and show that $\frac{\mathbb{T}_\ell \mathrm{C}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \max\left\{\frac{\mathbb{T}_\ell \mathrm{C}(G',k',F')}{\mathrm{OPT}(G',k')}, \alpha\right\}$.

Consider a solution F' of $\mathbb{T}_\ell$-CONTRACTION in (G',k'). If $|F'| \geq k'+1$, then the solution lifting algorithm returns $E(G)$, otherwise it returns $F = F' \cup \tilde{E}$. If $|F'| \geq k'+1$ then $\mathbb{T}_\ell \mathrm{C}(G',k',F') = k'+1 = k-d$. In this case, $F = E(G)$ and $\mathbb{T}_\ell \mathrm{C}(G,k,F) \leq k+1 = k'+d = \mathbb{T}_\ell \mathrm{C}(G',k',F') + d - 1$. Next, consider the case when $|F'| \leq k'$, and let $\mathscr{W}' = \{W'(t_1), W'(t_2), \ldots, W'(t_q)\}$ be the G'/F'-witness structure of G. Let w denote the vertex in $V(G') \setminus V(G)$ obtained by contracting the edges in $\tilde{E}$. Without loss of generality, assume that $w \in W'(t_1)$. Let $\mathscr{W} = (\mathscr{W}' \setminus \{W'(t_1)\}) \cup \{W_1\}$, where $W_1 = (W'(t_1) \setminus \{w\}) \cup \{v_1, h_1, h_2, \ldots, h_d\}$. Note that $V(G) \setminus \{v_1, h_1, h_2, \ldots, h_d\} = V(G') \setminus \{w\}$ and hence $\mathscr{W}$ is partition of $V(G)$. Furthermore, $G[W_1]$ is connected as $G'[W'(t_1)]$ is connected, and therefore, $E(G'[W_1 \setminus \{w\}]) \cup \tilde{E}$ contains a spanning tree of $G[W_1]$. Also, $|W_1| = |W'(t_1)| + d$, and any vertex which is adjacent to w in G' is adjacent to at least one vertex in $\{v_1, h_1, h_2, \ldots, h_d\}$ in G. Thus, $\mathscr{W}'$ is a G/F-witness structure of G, where $G/F \in \mathbb{T}_\ell$. Therefore, $\mathbb{T}_\ell \mathrm{C}(G,k,F) \leq \mathbb{T}_\ell \mathrm{C}(G',k',F') + d$.

Next, consider an optimum solution F^* to $\mathbb{T}_\ell$-CONTRACTION in (G,k), and let T be G/F^* with $\mathscr{W}$ being the T-witness structure of G. If $|F^*| \geq k+1$, then $\mathrm{OPT}(G,k) = k+1 = k'+d = \mathrm{OPT}(G',k') + d - 1$. Otherwise, we have $|F^*| \leq k$, and there are at least $\ell+3$ vertices, in $\{v_1, v_2, \ldots, v_{k+\ell+2}\}$ ($\subseteq I$) which are not in $V(F^*)$. That is, they are in singleton witness sets of $\mathscr{W}$. Then, by Lemma 5.4.4, $\{h_1, h_2, \ldots, h_d\}$ are in the same witness set, say $W(t_i)$ where $t_i \in V(T)$. Consider the case when $v_1 \in W(t_i)$. Let $\tilde{F}$ be the edge set obtained from F by replacing each edge uv by uw, where $v \in \{v_1, h_1, \cdots, v_d\}$ and $u \notin \{v_1, h_1, \cdots, v_d\}$.

Furthermore, let $F' = \tilde{F} \setminus \tilde{E}$. Notice that $|F'| \leq |F^*| - d$, and F' is solution to (G', k'). Therefore, $\mathrm{OPT}(G', k') \leq |F^*| - d = \mathrm{OPT}(G, k) - d$. Next, we consider the case when $v_1 \notin W(t_i)$, and let $t_j \in V(T)$ be the vertex such that $v_1 \in W(t_j)$. Then, t_i and t_j are adjacent in T. Let $\mathscr{W}' = \mathscr{W} \cup \{W(t_{ij})\} \setminus \{W(t_i), W(t_j)\}$ of $V(G)$, where $W(t_{ij}) = W(t_i) \cup W(t_j)$. Clearly, $G[W(t_{ij})]$ is connected. Thus, $\mathscr{W}'$ is a G/F-witness structure of G, where $|F| = |F^*| + 1$ as $|W(t_i)| - 1 + |W(t_j)| - 1 = (|W(t_{ij})| - 1) - 1$. Furthermore, F can be assumed to contain $\tilde{E}$, and therefore $F' = F \setminus \tilde{E}$ is solution to $\mathbb{T}_\ell$-CONTRACTION in (G', k'). This implies that $\mathrm{OPT}(G', k') \leq |F'| = |F^*| + 1 - d = \mathrm{OPT}(G, k) - d + 1$. Thus, we have * $\frac{\mathbb{T}_\ell \mathrm{C}(G,k,F)}{\mathrm{OPT}(G,k)} \leq \frac{\mathbb{T}_\ell \mathrm{C}(G',k',F')+d}{\mathrm{OPT}(G',k')+(d-1)} \leq \max\left\{\frac{\mathbb{T}_\ell \mathrm{C}(G',k',F')}{\mathrm{OPT}(G',k')}, \alpha\right\}$. □

In the following lemma we argue that if after applying all these reduction rules exhaustively, if the number of vertices in resulting graph is large, we can safely conclude that given instance can not be contracted to a graph in $\mathbb{T}_\ell$ with at most k edge contractions.

Lemma 5.4.7. *Let (G, k) be an instance of $\mathbb{T}_\ell$-CONTRACTION where none of the Reduction Rules 5.4.1 to 5.4.4 are applicable. If G is k-contractible to a graph in $\mathbb{T}_\ell$ then number of vertices in G is at most $\mathscr{O}(k(k+2\ell)]^{d+1})$.*

Proof. Since Reduction Rule 5.4.1 and 5.4.2 are not applicable, from Lemma 5.4.2 it follows that G has a connected vertex cover S of size at most $2(k+3)(k+2\ell)$. The set H consists of vertices of degree at least $2(k+3)(k+2\ell)+1$, and hence every vertex in H is included in any connected vertex cover of G, which is of size at most $2(k+3)(k+2\ell)$. This implies that $|H| \leq 2(k+3)(k+2\ell)$. Every vertex in R has degree at most $2(k+3)(k+2\ell)$. Therefore, if $S \cap R$ is a vertex cover of $G[R]$, then $|E(G[R])|$ is bounded by $4(k+3)^2(k+2\ell)^2$. Also, by the definitions of I and R, every vertex in R has a neighbour in R. Therefore, there are no isolated vertices in $G[R]$. Thus, $|R|$ is bounded by $8(k+3)^2(k+2\ell)^2$. Now, we bound the size of I. For every set $H' \subseteq H$ of size at most d, there are at most $k+\ell+2$ vertices in I which have H' as their neighbourhood. Otherwise, Reduction Rule 5.4.3 would have been applicable. Hence, there are at most $(k+\ell+2) \cdot \binom{2(k+3)(k+2\ell)}{d-1}$ vertices in

*We use the bound, $\frac{x+p}{y+q} \leq \max\{\frac{x}{y}, \frac{p}{q}\}$ for any positive real numbers x, y, p, q.

I which have degree at most d. A vertex in I which is of degree at least $d+1$, is adjacent to all vertices in at least one subset of size d of H. For a such a subset H' of H, there are at most $k+\ell+2$ vertices in I which have H' in their neighbourhood since Reduction Rule 5.4.4 is not applicable. Thus, there are at most $(k+\ell+2)\binom{2(k+3)(k+2\ell)}{d}$ vertices in I of degree at least d. Hence, $|I| \leq c'[k(k+2\ell)]^{(d+1)}$, for some fixed c'. Since $H \cup R$ is $\hat{c}k^2(k+2\ell)^2$ (where $\hat{c}$ is a constant) and $d > 1$, the claim follows. □

We are now in position to state main theorem of this section.

Theorem 5.4.1. $\mathbb{T}_\ell$-CONTRACTION *admits a strict PSAKS, where the number of vertices is bounded by* $\mathcal{O}([k(k+2\ell)]^{(\lceil \frac{\alpha}{\alpha-1} \rceil+1)})$.

Proof. For given $\alpha > 1$, kernelization algorithm fix $d = \lceil \frac{\alpha}{\alpha-1} \rceil$ and apply Reduction Rules 5.4.1; 5.4.2; 5.4.3; and 5.4.4 on the instance as long as they are applicable. If Reduction Rule 5.4.2 returns a trivial instance then statement is true. Otherwise, we know that there exists a connected vertex cover of size at most $2(k+3)(k+2\ell)$. This implies size of H is at most $2(k+3)(k+2\ell)$. If the number of vertices in graph is more than $\mathcal{O}([k(k+2\ell)]^{d+1})$ then the reduction rules can be applied in $\mathcal{O}([k(k+2\ell)]^{(d+1)}n^{\mathcal{O}(1)}) = n^{\mathcal{O}(1)}$ time, where n is the number of vertices in the input graph. If the number of vertices in resulting graph is more than $\mathcal{O}([k(k+2\ell)]^{d+1})$, then by Lemma 5.4.7 we have $\mathrm{OPT}(G,k) = k+1$ and the algorithm returns a spanning tree of $(K_{\ell+4}, 1)$ as a reduced instance. Otherwise, reduced instance has $\mathcal{O}([k(k+2\ell)]^{(\lceil \frac{\alpha}{\alpha-1} \rceil+1)})$ vertices. The correctness of algorithm follows from Lemma 5.4.1; 5.4.3; 5.4.5; and 5.4.6. □

5.5 Randomized FPT Algorithm for $\mathbb{T}_\ell$-CONTRACTION

In this section, we design an FPT algorithm for $\mathbb{T}_\ell$-CONTRACTION. Our algorithm proceeds as follows. We start by applying some simple reduction rules. Then by branching we ensure that the resulting graph is 2-connected. Finally, we give an FPT algorithm

running in time $\mathcal{O}((2\sqrt{\ell}+2)^{\mathcal{O}(k+\ell)} \cdot n^{\mathcal{O}(1)})$ on 2-connected graphs. The approach we use for designing the algorithm for the case when the input graph is 2-connected follows the approach of Heggernes et al. [55] for designing an FPT algorithm for contracting to trees. Also, whenever we are dealing with an instance of $\mathbb{T}_\ell$-CONTRACTION we assume that we have an algorithm running in time $\mathcal{O}((2\sqrt{\ell'}+2)^{\mathcal{O}(k+\ell')} \cdot n^{\mathcal{O}(1)})$ for $\mathbb{T}_{\ell'}$-CONTRACTION, for every $\ell' < \ell$. That is, we give family of algorithms inductively for each $\ell' \in \mathbb{N}$, where the algorithm for TREE CONTRACTION by Heggernes et al. forms the base case of our inductive hypothesis.

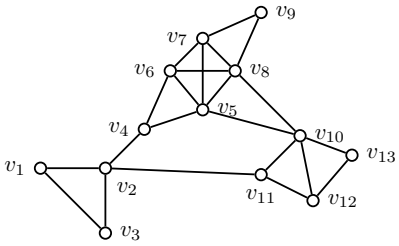
Let (G,k) be an instance of $\mathbb{T}_\ell$-CONTRACTION. The measure we use for analyzing the running time of our algorithm is $\mu = \mu(G,k) = k$. We start by applying some simple reduction rules.

Reduction Rule 5.5.1. *If $k < 0$ then return that (G,k) is a* NO *instance of* $\mathbb{T}_\ell$-CONTRACTION.

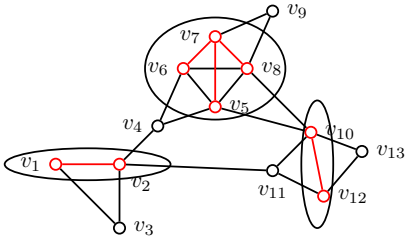
Reduction Rule 5.5.2. *If $k = 0$ and $G \in \mathbb{T}_\ell$ then return that (G,k) is a* YES *instance of* $\mathbb{T}_\ell$-CONTRACTION.

Reduction Rule 5.5.3. *If G is a disconnected, or $k = 0$ and $G \notin \mathbb{T}_\ell$ then return that (G,k) is a* NO *instance.*

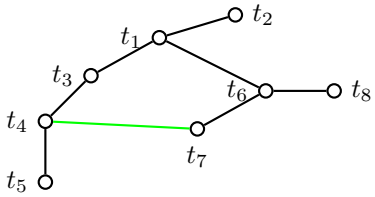
We assume that the input graph is 2-connected, and design an algorithm for input restricted to 2-connected graphs. Later, we will show how we can remove this constraint. The key idea behind the algorithm is to use a coloring of $V(G)$ with at most $2\sqrt{\ell}+2$ colors to find a T-witness structure (if it exists) of G, where G is contractible to $T \in \mathbb{T}_\ell$ using at most k edge contractions. Moreover, if such a T does not exist then we must correctly conclude that (G,k) is a NO instance of $\mathbb{T}_\ell$-CONTRACTION. If such T exists then T can be colored with $2\sqrt{\ell}+2$ (Observation 5.2.1.1). Fix one such coloring of T. When we *uncontract* a vertex, say t, in T, all the vertices in G which has been contracted to t gets same color. Proper coloring of T insures that vertices obtained by *uncontracting* adjacent vertices, say t_1, t_2 in T have different colors in G and are easy to differentiate. We will see that the



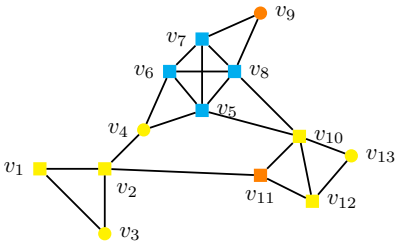
(a) Input Graph



(b) Witness Structure $\mathscr{W}$



(c) Reduced Graph T



(d) Compatible Coloring

Figure 5.2: Compatible coloring

before mentioned property holds only when each of t_1, t_2 are obtained by contracting at least two vertices in G. We make this notion more formal in definition.

Definition 5.5.1 (Compatible Coloring)**.** *Let G be a connected graph, T be a graph in $\mathbb{T}_\ell$, $\mathscr{W}$ be a T-witness structure of G, and $\phi : V(G) \to [2\sqrt{\ell}+2]$ be a coloring of $V(G)$. Furthermore, let T_S be a (fixed) spanning tree of T, $M = \{t,t' \mid tt' \in E(T) \setminus E(T_S)\} \cup \{t \in V(T) \mid d_T(t) \geq 3\}$, and $B = \{t \in V(T) \mid |W(t)| \geq 2\}$. We say that ϕ is $\mathscr{W}$-compatible if the following conditions are satisfied.*

1. *For all $W \in \mathscr{W}$, and $w, w' \in W$ we have $\phi(w) = \phi(w')$.*
2. *For all $t,t' \in M \cup B$ such that $tt' \in E(T)$ we have $\phi(W(t)) \neq \phi(W(t'))$.*
3. *For all $t,t' \in M \cup B$ (not necessarily distinct), and a path $P = (t,t_1,\ldots,t_z,t')$, where $z \in \mathbb{N}$ such that for all $i \in [z]$ we have $t_i \notin M \cup B$ then $\phi(W(t)) \neq \phi(W(t_1))$ and $\phi(W(t_z)) \neq \phi(W(t'))$.*

We refer to the set $M \cup B$ as the set of marked vertices.

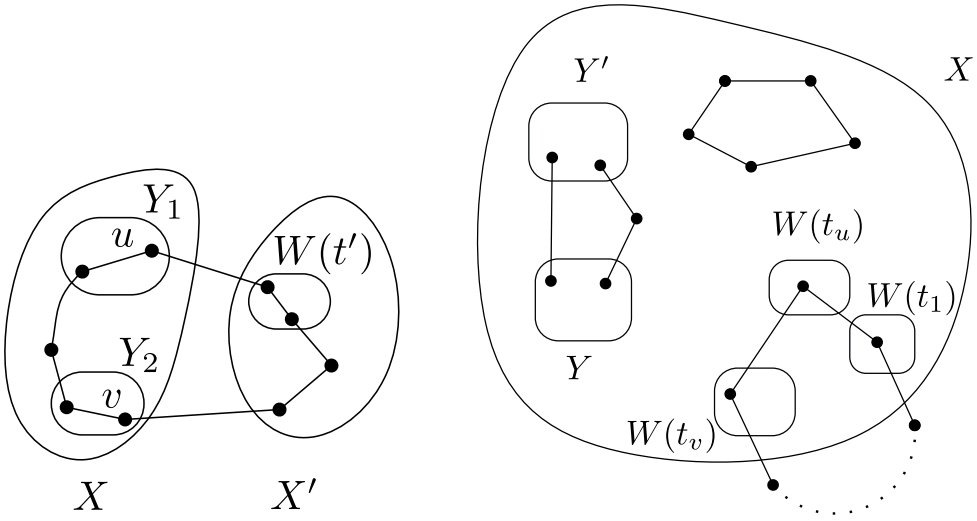
Consider an example in Figure 5.2. Input graph G is contractible to a tree T in $\mathbb{T}_1$. Let $\mathscr{W}$ be a T-witness structure of G. We fix a spanning tree T_s in T which excludes edge t_4t_7 in graph T. Coloring of vertices of $V(G)$ is $\mathscr{W}$-compatible for fixed spanning tree T_S.

Assume that (G,k) is a YES instance of $\mathbb{T}_\ell$-CONTRACTION, and F be one of its (inclusion-wise) minimal solution. Furthermore, let $T = G/F$, and $\mathscr{W}$ be the T-witness structure of G. Suppose we are given G and a $\mathscr{W}$-compatible coloring $\phi : V(G) \to [2\sqrt{\ell}+2]$ of G, but we are neither given $\mathscr{W}$ nor T. We will show how we can compute a T'-witness structure $\mathscr{W}'$ of G such that $|V(T')| \geq |V(T)|$, where $T' \in \mathbb{T}_\ell$. Informally, we will find such a witness structure by either concluding that none of the edges are part of the solution, some specific set of edges are part of the solution, or finding a star-like structure of the monochromatic components of size at least 2 in G, with respect to ϕ. Towards this, we will employ the algorithm for CONNECTED VERTEX COVER (CVC) by Cygan [24] which runs in time $2^k n^{\mathcal{O}(1)}$. Here, k is the size of a solution and n is the number of vertices in the input graph.

Consider the case when G is k-contractable to a graph, say $T \in \mathbb{T}_\ell$, and let $\mathscr{W}$ be a T-witness structure of G. Furthermore, let $\phi : V(G) \to [2\sqrt{\ell}+2]$ be a $\mathscr{W}$-compatible coloring of G, and $\mathscr{X}$ be the set of monochromatic components of ϕ. We prove some lemmata showing useful properties of $\mathscr{X}$.

Lemma 5.5.1. *Let T' be the graph with $\mathscr{X}$ as the T'-witness structure of G. Then $T' \in \mathbb{T}_\ell$ and $|V(T')| \leq |V(T)|$.*

Proof. Every witness set of $\mathscr{W}$ is monochromatic with respect to ϕ (see item 1 of Definition 5.5.1). Therefore, for every $W \in \mathscr{W}$ there exists $X \in \mathscr{X}$ such that $W \subseteq X$. Moreover, by the definition of $\mathscr{X}$, $G[X]$ is connected. There exists $Y \subseteq V(T)$ such that $T[Y]$ is a connected subgraph of T and $X = \cup_{y \in Y} W(y)$. Graph T' can beobtained from T by contracting spanning tree in X for every such X. Since $\mathbb{T}_\ell$ is closed under edge contraction (item 3 of Observation 5.2.1), T' is also in $\mathbb{T}_\ell$ with $|V(T')| \leq |V(T)|$. □



(a) Please refer to Lemma 5.5.2. (b) Please refer to Lemma 5.5.4.

Figure 5.3: Set X, X' are monochromatic components. Rectagular box represents big witness sets in $\mathscr{W}$.

Next, we proceed to show how we can partition each $X \in \mathscr{X}$ into many smaller witness sets such that either we obtain $\mathscr{W}$ or a T'-witness structure of G for some $T' \in \mathbb{T}_\ell$ which has at least as many vertices as T. Towards this, we introduce the following notions.

For $X \in \mathscr{X}$, by $\hat{X}$ we denote the set of vertices that have a neighbor outside of X, *i.e.* $\hat{X} = N(V(G) \setminus X)$. A *shatter* of X is a partition of X into sets such that one of them is a connected vertex cover C of $G[X]$ containing all the vertices in $\hat{X}$ and all other sets are of size 1. The size of a shatter of X is the size of C. Furthermore, a shatter of X is minimum if there is no other shatter with strictly smaller size.

From Definition 5.5.1, it follows that each $X \in \mathscr{X}$ is union of some witness sets in $\mathscr{W}$. Formally, for every $X \in \mathscr{X}$, there is $\mathscr{W}_X \subseteq \mathscr{W}$ such that $X = \cup_{Y \in \mathscr{W}_X} Y$. Following lemma says that if X has some particular structure then X is a big witness in itself. In other words, $\mathscr{W}_X$ contains only one witness set.

Lemma 5.5.2. *Consider $X \in \mathscr{X}$ with $|X| \geq 2$, $\mathscr{W}_X \subseteq \mathscr{W}$ such that $X = \cup_{Y \in \mathscr{W}_X} Y$, and all of the following conditions are satisfied.*

- *$G[X] = (u, v_1, \ldots, v_q, v)$ is an induced path, where $q \in \mathbb{N}$.*

- *For each $i \in [q]$ we have $deg(v_i) = 2$.*
- *There exists $X' \in \mathscr{X} \setminus \{X\}$ such that $N(u) \cap X' \neq \emptyset$ and $N(v) \cap X' \neq \emptyset$.*

Then $|\mathscr{W}_X| = 1$.

Proof. Let $X = (u, v_1, v_2, \ldots, v_q, v)$, where for each $i \in [q]$ we have $deg_G(v_i) = 2$. Also, let $X' \in \mathscr{X} \setminus \{X\}$ such that $N(u) \cap X' \neq \emptyset$ and $N(v) \cap X' \neq \emptyset$. Assume that $|\mathscr{W}_X| \geq 2$. Let Y_1 and Y_2 be the witness sets containing u and v, respectively. See Figure 5.3a. Since, $|\mathscr{W}_X| \geq 2$, and each of the witness sets are connected therefore, we have $Y_1 \neq Y_2$. Notice that in T, for which $\mathscr{W}$ is a T-witness structure of G there is a cycle C containing t_{Y_1}, t_{Y_2}, and vertices corresponding to some of the witness sets included in $X' \cup (X \setminus (Y_1 \cup Y_2))$. Here, t_{Y_1} and t_{Y_2} are vertices in T such that $W(t_{Y_1}) = Y_1$ and $W(t_{Y_2}) = Y_2$. Notice that C must contain at least two marked vertices which are adjacent (see Definition 5.5.1). By definition, X and X' are monochromatic, and therefore, these marked vertices can not belong to X or X'. Without loss of generality assume that t_{Y_1} is one of the marked vertex and one of its neighbor, say t', such that $W(t') \subseteq X'$ is another marked vertex on this cycle. This implies that t_{Y_2} is contained in a path between two marked vertices namely, t_{Y_1} and t'. But all the nodes on the path between t_{Y_1} and t_{Y_2} have the same color. This contradicts the fact that ϕ is a $\mathscr{W}$-compatible coloring of G (see item 3 of Definition 5.5.1). □

Following lemma says that if X has particular structure than all vertices in X are part of singleton witness sets in $\mathscr{W}$.

Lemma 5.5.3. *Consider $X \in \mathscr{X}$ with $|X| \geq 2$, $\mathscr{W}_X \subseteq \mathscr{W}$ such that $X = \cup_{Y \in \mathscr{W}_X} Y$, and all the following conditions are satisfied.*

- *$G[X] = (u, v_1, \ldots, v_q, v)$ is an induced path, where $q \in \mathbb{N}$.*
- *For each $i \in [q]$ we have $deg(v_i) = 2$.*
- *There exists no $X' \in \mathscr{X} \setminus X$ such that $N(u) \cap X' \neq \emptyset$ and $N(v) \cap X' \neq \emptyset$.*

Then $|\mathscr{W}_X| = |X|$.

Proof. Recall that F is a (inclusion wise) minimal solution corresponding to the witness structure $\mathscr{W}$. Assume that $|\mathscr{W}_X| < |X|$. This implies that there exists $Y \in \mathscr{W}_X$ such that $|Y| \geq 2$. Let t_Y be a vertex in T such that $Y = W(t_Y)$. Also, let v_i be the smallest i such that $v_i \in Y$. Since $|Y| \geq 2$, v_{i+1} is also present in Y. We can partition neighbors of t_Y into N_1 and N_2 such that N_1 is adjacent to v_i and N_2 is adjacent to $Y \setminus \{v_i\}$. Since there is no X' in $\mathscr{X} \setminus X$ such that $N(u) \cap X' \neq \emptyset$ and $N(v) \cap X' \neq \emptyset$, N_1 and N_2 are different. If graph G obtained by spanning tree of Y is in $\mathbb{T}_\ell$ then by Observation 5.2.2, $G/(F \setminus \{(v_i, v_{i+1})\})$ is also a graph in $\mathbb{T}_\ell$. This contradicts the minimality of F. Notice that if N_1, N_2 are not different then $G/(F \setminus \{(v_i, v_{i+1})\})$ may have one more cycle than the graph G/F. □

Next, we show that each $X \in \mathscr{X}$ for which Lemma 5.5.2 and 5.5.3 are not applicable must contain exactly one big witness set. Moreover, the unique big witness set (together with other vertices as singleton sets) forms one of its shatters.

Lemma 5.5.4. *For $X \in \mathscr{X}$ with $|X| \geq 2$, let $\mathscr{W}_X \subseteq \mathscr{W}$ such that $X = \cup_{Y \in \mathscr{W}_X} Y$. Furthermore, the set X does not satisfy the conditions of Lemma 5.5.2 or 5.5.3. Then there is exactly one big witness set in $\mathscr{W}_X$.*

Proof. Consider $X \in \mathscr{X}$ with $|X| \geq 2$. Assuming a contradiction, suppose $\mathscr{W}_X$ contains two big witness sets say Y and Y'. Notice that there cannot be an edge between a vertex in Y and a vertex in Y' (see item 2 of Definition 5.5.1). This together with the connectedness of X implies that there is a path from a vertex in $y \in Y$ and a vertex in $y' \in Y'$ which contains a neighbor of y in some $Z \in \mathscr{W}_X \setminus \{Y, Y'\}$. But then from item 2 and 3 of Definition 5.5.1 we have $\phi(Y) \neq \phi(Z)$, a contradiction. Therefore, X can contain at most one big witness set from $\mathscr{W}$.

Suppose X does not contain any big witness set. Consider the case when $deg_G(v) = 2$ for all $v \in X$ and $X = V(G)$. Since all the witness sets in X are singleton, there exists a cycle in T such that all the vertices on this cycle have same color. This contradicts the fact that ϕ is $\mathscr{W}$-compatible (see item 3 Definition 5.5.1). We now consider case when X

is a proper subset of $V(G)$ or it contains a vertex of degree 3. Since X does not satisfies conditions of Lemma 5.5.2 or 5.5.3, it is not an induced path or it is an induced path but one of its internal vertex has degree other than 2. Since X is connected, in either case there exists $v \in X$ such that $deg_G(v) \geq 3$. If X contains all singleton witness set then $deg_T(t_v) \geq 3$ where $W(t_v) = \{v\}$. Let $u \in X$ be a vertex adjacent to v and $W(t_u) = \{u\}$. Since $|W(t_v)| = |W(t_u)| = 1$, neither t_v nor t_u is a cut vertex in T (Lemma 5.2.3) which implies $deg_T(t_u) > 1$. Let t_v, t_1 are two neighbors of t_u. There exists a path between t_v, t_1 which does not contain vertex t_u. This implies there exists a cycle in T containing t_v, t_u, t_1. There are at least two vertices marked on this cycle. Hence, either t_u is marked or t_u is contained between two marked vertices. In either case, it contradicts the fact that X is color class of a coloring which is $\mathscr{W}$-compatible (see item 2, 3 Definition 5.5.1). □

Lemma 5.5.5. *Consider $X \in \mathscr{X}$ such that $|X| \geq 2$ and it contains a big witness set, and it does not satisfy conditions of Lemma 5.5.2 or 5.5.3. Let $\mathscr{W}_X \subseteq \mathscr{W}$ such that $X = \cup_{Y \in \mathscr{W}_X} Y$, and W^* be the (unique) big witness set in X. Then W^* is a connected vertex cover of $G[X]$ and it contains $\hat{X}$.*

Proof. Suppose X contains a big witness set, say W^*. From Lemma 5.5.4, for each $Y \in \mathscr{W}_X \setminus \{W^*\}$ we have $|Y| = 1$. We first prove that W^* is a vertex cover of $G[X]$. Assume that W^* is not a vertex cover of $G[X]$, then there is an edge y_1y_2 such that $y_1, y_2 \notin W^*$. For $i \in \{1,2\}$, let $\{y_i\} = Y_i = W(t_i)$. Since $G[X]$ is connected, there exists a path between y_1, y_2 and a vertex in W^*, which is contained in X. Without loss of generality, assume that there exist a path P_1 in X from y_1 to W^* which does not contain y_2. Since G is 2-connected, there exists a path, say P_2 from y_2 to a vertex in W^*, which does not contain y_1. Notice that there is a cycle in T containing nodes t^*, t_1, t_2, where $W^* = W(t^*)$. At least two nodes of the vertices from this cycle must be marked, and have different colors from each other. Hence, the path P_2 can not be contained in X. We know that t^* is contained in $M \cup B$. Let the other marked vertex in this cycle be t. The vertex t is obtained by contracting some vertices on the path P_2. Notice that t_1 is vertex contained in the path between two

vertices in $M \cup B$ and all the nodes in path from t^* to t_1 has the same color. This contradicts the fact that X is a color class in a coloring which is $\mathscr{W}$-compatible (see item 2, 3 of Definition 5.5.1). Hence our assumption was wrong and no such edge y_1y_2 exits. Since, W^* is a witness set, by definition, it is connected and therefore W^* is a connected vertex cover of $G[X]$. Notice that all the above argument still holds if the path P_1 is simply an edge and y_2 is outside X. In other words, if there exists y_1 in $X \setminus W^*$ there is no edge y_1y_2 such that y_2 is not contained in X. This implies that $\hat{X}$ is contained in W^*. □

Using Lemma 5.5.3 to Lemma 5.5.5 we show how we can replace each $X \in \mathscr{X}$ with the sets of its shatter. Recall that we are given only G and ϕ, and therefore we know $\mathscr{X}$, but we do not know $\mathscr{W}$. In the Lemma 5.5.6, we show how we can find a T'-witness structure of G for some $T' \in \mathbb{T}_\ell$, which has at least as many vertices as T (without knowing $\mathscr{W}$).

Lemma 5.5.6. *Given $\mathscr{X}$, we can obtain a T'-witness structure of G in time $2^k n^{\mathcal{O}(1)}$ time, where $T' \in \mathbb{T}_\ell$ and $|V(T')| \geq |V(T)|$.*

Proof. Consider $X \in \mathscr{X}$. If $|X| = 1$ then we let $\mathscr{W}_X = \{X\}$, which is the unique shatter of X. We now consider $X \in \mathscr{X}$ such that $|X| \geq 2$. If there is $X \in \mathscr{X}$ which satisfies the premise of Lemma 5.5.2 then contract all edges in X and reduce k by $|X| - 1$. If there exists $X \in \mathscr{X}$ which satisfies the premise of Lemma 5.5.3 then replace X in $\mathscr{X}$ with $|X|$ many singleton sets $\{v\}$ for each $v \in X$. If there exists $X \in \mathscr{X}$ which does not satisfy the conditions of Lemma 5.5.2 and 5.5.3 then from Lemma 5.5.4 we know that X contains exactly one big-witness set, say $\hat{W}$. Moreover, Lemma 5.5.5 implies that $\hat{W}$ is a connected vertex cover of $G[X]$ containing $\hat{X}$. In this case, we will find a shatter W^* of X, which has size at most $|\hat{W}|$ as follows. Let G' be the graph obtained from $G[X]$ by adding a (new) vertex v^* for each vertex $v \in \hat{X}$, and adding the edge (v, v^*). Then we find a minimum sized connected vertex cover of C of G' by using the algorithm given by Proposition 2.1.2. Notice that a minimum connected vertex cover of a graph does not contain any degree one vertex therefore, $\hat{X} \subseteq C$. From the definition of minimum shatter and the minimality of set C, it follows that $\mathscr{W}_X = \{C\} \cup \{\{x\} \mid x \in X \setminus C\}$ is a minimum shatter of X. Notice

that apart from computing connected vertex cover, all other steps can be performed in polynomial time. Since the size of each witness set in $\mathscr{W}$ is bounded by $k+1$, therefore there exists a connected vertex cover of size at most $k+1$. Moreover, we can compute connected vertex cover in time $2^{k+1}n^{\mathcal{O}(1)}$ (Proposition 2.1.2), and there are at most n sets in $\mathscr{X}$. Therefore, the overall running time is bounded by $2^k n^{\mathcal{O}(1)}$. □

Now we are ready to present our randomized algorithm for $\mathbb{T}_\ell$-CONTRACTION when input graph is 2-connected.

Theorem 5.5.1. *There is a Monte Carlo algorithm for solving* $\mathbb{T}_\ell$-CONTRACTION *on* 2*-connected graphs running in time* $\mathcal{O}((2\sqrt{\ell}+2)^{\mathcal{O}(k+\ell)} \cdot n^{\mathcal{O}(1)})$*, where n is the number of vertices in the input graph. It does not return false positive and returns correct answer with probability at least* $1-1/e$.

Proof. Let (G,k) be an instance of $\mathbb{T}_\ell$-CONTRACTION, where G is a 2-connected graph. Furthermore, the Reduction Rules 5.5.1 and 5.5.3 are not applicable, otherwise we can correctly decide whether or not (G,k) is a YES instance. The algorithm starts by computing a random coloring $\phi : V(G) \to [2\sqrt{\ell}+2]$, by choosing a color for each vertex uniformly and independently at random. Let $\mathscr{X}$ be the set of monochromatic connected components with respect to ϕ in G. The algorithm applies Lemma 5.5.6 in time $2^k n^{\mathcal{O}(1)}$ and tries to compute T' such that $T' \in \mathbb{T}_\ell$ and G is k-contractible to T'. It runs $(2\sqrt{\ell}+2)^{6k+8\ell}$ many iterations of two steps mentioned above. If for any such iteration it obtains a desired T'-witness structure of G then it returns YES. If none of the iterations yield YES then the algorithm returns NO. This completes the description of the algorithm.

Observe that the algorithm returns YES only if it has found a $T' \in \mathbb{T}_\ell$ such that G is contractible to T' using at most k edge contractions. Therefore, when it outputs YES, then indeed (G,k) is a YES instance of $\mathbb{T}_\ell$-CONTRACTION. We now argue that if (G,k) is a YES instance then using a random coloring the algorithm (correctly) returns the answer with sufficiently high probability. Let T be a graph in $\mathbb{T}_\ell$, such that G is k-contractible to T, and

$\mathscr{W}$ be a T-witness structure of G. Furthermore, let T_S be a (fixed) spanning tree of T, and vertex set M, B are set of vertices defined in Definition 5.5.1. Let $\psi : V(G) \to [2\sqrt{\ell}+2]$ be a coloring where colors are chosen uniformly at random for each vertex. The total number of vertices contained in big witness sets of $\mathscr{W}$ is at most $2k$. By our assumption, every leaf is a singleton witness set and it is adjacent to a big witness set. Here, we assume that the number of vertices in T is at least 3, otherwise we can solve the problem in polynomial time. This implies that no leaf is in $M \cup B$. Consider graph T' obtained from T by deleting all the leaves and deleting edges in $E(T_\ell) \setminus E(T_S)$. All the marked vertices of T_ℓ and all the paths connecting two marked vertices are also present in T'. Notice that T' is tree with at most $k+2\ell$ leaves. Since the number of vertices of degree three is at most the number of leaves in any tree, there are at most $k+2\ell$ vertices of degree at least 3. There are at most k vertices in T which are big witness sets and at most 2ℓ vertices incident to edges in $E(T_\ell) \setminus E(T_S)$. Hence the total number of marked vertices is at most $2k+4\ell$. Since T' is a tree, there are at most $2k+4\ell$ vertices which lie on a path between two vertices in $M \cup B$ and are adjacent to one of these. The number of vertices of G which are marked vertices or vertices which are adjacent to it in T' is at most $2(2k+4\ell)+2k$. Therefore, the probability that ψ is compatible with $\mathscr{W}$ is at least $1/(2\sqrt{\ell}+2)^{6k+8\ell}$. Since the algorithm runs $(2\sqrt{\ell}+2)^{6k+8\ell}$ many iterations, probability that none of these colorings which is generated uniformly at random is compatible with $\mathscr{W}$ is at most $(1-1/(2\sqrt{\ell}+2)^{6k+8\ell})^{(2\sqrt{\ell}+2)^{6k+8\ell}} < 1/e$. Hence, algorithm returns a solution on positive instances with probability at least $1-1/e$. Each iteration takes $2^k \cdot n^{\mathcal{O}(1)}$ time and hence the total running time of the algorithm is $\mathcal{O}((2\sqrt{\ell}+2)^{\mathcal{O}(k+\ell)} \cdot n^{\mathcal{O}(1)})$. □

Next, we design reduction rules and a branching rule whose (exhaustive) application will ensure that the instance of $\mathbb{T}_\ell$-CONTRACTION we are dealing with is 2-connected. Either we apply one of these reduction rules or branching rule, or we resolve the instance using the algorithm for $\mathbb{T}_{\ell'}$-CONTRACTION, where $\ell' < \ell$. This together with Theorem 5.5.1 gives us an algorithm for $\mathbb{T}_\ell$-CONTRACTION on general graphs.

Lemma 5.5.7. *If for some $0 \leq \ell' < \ell$, (G,k) is a* YES *instance of* $\mathbb{T}_{\ell'}$-CONTRACTION *then return that (G,k) is a* YES *instance of* $\mathbb{T}_{\ell}$-CONTRACTION.

Our next reduction rule deals with vertices of degree of 1.

Reduction Rule 5.5.4. *If there is $v \in V(G)$ such that $d(v) = 1$ then delete v from G. The resulting instance is $(G - \{v\}, k)$.*

If a connected graph G is not 2-connected graph then there is a cut vertex say, v in G. Let $C_1, C_2, \ldots, C_t$ be the components of $G - \{v\}$. Furthermore, let $G_1 = G[V(C_1) \cup \{v\}]$ and $G_2 = G - V(C_1)$. Next, we try to resolve the instance (if possible) using the following lemma.

Lemma 5.5.8. *If there exists ℓ_1 and ℓ_2 with $\ell_1 + \ell_2 = \ell$, where $\ell_1, \ell_2 > 0$, and k_1 and k_2 with $k_1 + k_2 = k$ such that (G_1, k_1) is a* YES *instance of* $\mathbb{T}_{\ell_1}$-CONTRACTION *and (G_2, k_2) is a* YES *instance of* $\mathbb{T}_{\ell_2}$-CONTRACTION *then return that (G,k) is a* YES *instance of* $\mathbb{T}_{\ell}$-CONTRACTION.

Notice that if Lemma 5.5.8 is not applicable then one of G_1 or G_2 must be contracted to a tree. Let k_1 be the smallest integer such that (G_1, k_1) is a YES instance of $\mathbb{T}$-CONTRACTION, and k_2 be the smallest integer such that (G_2, k_2) is a YES instance of $\mathbb{T}$-CONTRACTION. Notice that k_1 and k_2 can be computed in (deterministic) time $4^k n^{\mathcal{O}(1)}$ using the algorithm for $\mathbb{T}$-CONTRACTION [55]. We next proceed with the following branching rule.

Branching Rule 2. *We branch depending on which of the graphs among G_1 and G_2 are contracted to a tree. Therefore, we branch as follows.*

- *Contract G_1 to a tree, and the resulting instance is $(G_2, k - k_1)$.*
- *Contract G_2 to a tree, and the resulting instance is $(G_1, k - k_2)$.*

Note that the measure strictly decreases in each of the branches of the Branching Rule 2 since Reduction Rule 5.5.4 is not applicable. If we are unable to resolve the instance using

Lemma 5.5.7 and 5.5.8, and Reduction Rules 5.5.3 and 5.5.4 and Branching Rule 2 are not applicable then the input graph is 2-connected. And, then we resolve the instance using Theorem 5.5.1.

Theorem 5.5.2. *For each $\ell \in \mathbb{N}$, there is a Monte Carlo algorithm for solving $\mathbb{T}_\ell$-*CONTRACTION *with running in time $\mathcal{O}((2\sqrt{\ell}+2)^{\mathcal{O}(k+\ell)} \cdot n^{\mathcal{O}(1)})$. It does not return false positive and returns correct answer with probability at least $1-1/e$.*

Proof. Let (G,k) be an instance of $\mathbb{T}_\ell$-CONTRACTION. If G is 2-connected then we resolve the instance using Theorem 5.5.1 with the desired probability bound. If G is not connected then we correctly resolve the instance using Reduction Rule 5.5.3. Moreover, the Reduction Rule 5.5.3 can be applied in polynomial time. Hereafter, we assume that G is connected, but not 2-connected.

In this case, we proceed by either resolving the instance using Lemma 5.5.7 or Lemma 5.5.8, or applying the Reduction Rule 5.5.4, or applying the Branching Rule 2. We prove the claim by induction on the measure $\mu = \mu(G,k) = k$.

If $\ell = 0$ then we can resolve the instance using the (deterministic) algorithm for $\mathbb{T}_\ell$-CONTRACTION in [55] in time $4^k n^{\mathcal{O}(1)}$. We note here that though the deterministic algorithm presented in [55] has been mentioned to run in time $4.98^k n^{\mathcal{O}(1)}$ but, it uses the algorithm for CONNECTED VERTEX COVER as a black-box, which has been improved in [24]. This also improves the running time of the deterministic algorithm in [55].

Hereafter, we inductively assume that whenever we are dealing with an instance of $\mathbb{T}_\ell$-CONTRACTION, we have an algorithm for $\mathbb{T}_{\ell'}$-CONTRACTION with the desired runtime and success probability bound, where $0 \leq \ell' < \ell$. We note that this does not interfere with the probability computation since the only randomized step (recursively) in our algorithm is when we employ Theorem 5.5.1, in which case we directly resolve the instance.

If $k \leq 0$ then we correctly resolve the instance using Reduction Rules 5.5.1 and 5.5.2. If (G,k) is a YES instance of $\mathbb{T}_{\ell'}$-CONTRACTION, for some $0 \leq \ell' < \ell$ then we correctly

conclude that (G,k) is a YES instance of $\mathbb{T}_\ell$-CONTRACTION. Moreover, we obtain the desired probability and runtime bound using the assumption of existence of an algorithm with desired properties for every $0 \leq \ell' < \ell$. If $k > 0$, and there is a vertex of degree 1 then we remove this vertex (in polynomial time) to obtain an equivalent instance using Reduction Rule 5.5.4. If none of the above are applicable the G has a cut vertex say.

We consider the following case. If Lemma 5.5.8 is applicable then we correctly resolve the instance in allowed running time with the desired success probability. This again relies on the existence of an algorithm for $\mathbb{T}_{\ell'}$-CONTRACTION with desired properties, for every $0 \leq \ell' < \ell$. Otherwise, we know that Branching Rule 2 must be applicable, where the measure drops at least by 1 in each of the branches since Reduction Rule 5.5.4 is not applicable. Moreover, when none of the Reduction Rules 5.5.1 to 5.5.4 are applicable, we cannot resolve the instance using one of Lemma 5.5.7 and Lemma 5.5.8, and Branching Rule 2 is not applicable then the graph is 2-connected, and we resolve the instance using Theorem 5.5.1. Notice the number of nodes in the search tree is bounded by $2^{\mathcal{O}(k)}$, all the reduction rules can be applied in polynomial time, and at the leaves of the search tree and at the internal nodes we require time which is bounded by $\mathcal{O}((2\sqrt{\ell}+2)^{\mathcal{O}(k+\ell)} \cdot n^{\mathcal{O}(1)})$. Thus, we obtain the desired running time and probability bound. □

5.6 Derandomization of the FPT Algorithm

In this section, we derandomize the algorithm presented in Section 5.5. Before proceeding forward we define the following important object of this section.

Definition 5.6.1 (Universal Family)**.** *A (n,k,q)-universal family is a collection $\mathscr{F}$, of functions from $[n]$ to $[q]$ such that for each $S \subseteq [n]$ of size k and a function $\phi : S \to [q]$, there exists function $f \in \mathscr{F}$ such that $f|_S \equiv \phi$.*

Here, $f|_S$ denotes the function f when restricted to the elements of S. For $q = 2$, the universal family defined above is called an (n,k)-universal set [81]. Hence, (n,k,q)-

universal family is a generalization of (n,k)-universal set. The main result of this section is the following theorem (Theorem 5.6.1), which we use to derandomize the algorithm presented in Section 5.5.

Theorem 5.6.1. *For any $n,k,q \geq 1$, one can construct an (n,k,q)-universal family of size $\mathcal{O}(q^k \cdot k^{\mathcal{O}(k)} \cdot \log n)$ in time $\mathcal{O}(q^k \cdot k^{\mathcal{O}(k)} \cdot n \log n)$.*

Before proceeding to the proof of Theorem 5.6.1, we state how we use it to derandomize the algorithm presented in Section 5.5. Let (G,k) be an instance of $\mathbb{T}_\ell$-CONTRACTION. Assume that (G,k) is a YES instance of $\mathbb{T}_\ell$-CONTRACTION, and let F be one of its solution. Furthermore, let $T = G/F$, where $T \in \mathbb{T}_\ell$ and $\mathscr{W}$ be the T-witness structure of G, and $\phi : V(G) \to [2\sqrt{\ell}+2]$ be a $\mathscr{W}$-compatible coloring of G. Recall that our randomized algorithm starts by coloring vertices in G uniformly and independently at random, and then uses this coloring to extract a witness structure out of each color classes. We then argued that any random coloring is "equally good" as that of ϕ with sufficiently high probability, which is given by a function of k (and ℓ). To derandomize this algorithm, we construct a family $\mathscr{F}$ of (coloring) functions from $[n]$ to $[2\sqrt{\ell}+2]$. We argue that one of the colorings in the family that we compute is "equally good" as that of ϕ. Recall that the number of vertices which we need to be colored in a specific way for a coloring to be $\mathscr{W}$-compatible is bounded by $6k+8\ell$ (see Definition 5.5.1 and Theorem 5.5.1). Let S be the set of vertices in G which needs to be colored in a specific way as per the requirements of Definition 5.5.1. We can safely assume that $|S| = 6k+8\ell$. If this is not the case we can add arbitrary vertices in S to ensure this. Notice that any coloring f of G such that $f|_S = \phi|_S$ also satisfies the requirements of Definition 5.5.1. Let $\mathscr{F}$ be an $(n, 6k+8\ell, 2\sqrt{\ell}+2)$-universal family constructed using Theorem 5.6.1. Instead of using random coloring in the algorithm presented in Section 5.5, we can iterate over functions in $\mathscr{F}$. Notice that we do not know S but for any such S, we are guaranteed to find an appropriate coloring in one of the functions in $\mathscr{F}$, which gives us the desired derandomization of the algorithm.

In rest of the section, we focus on the prove of Theorem 5.6.1. Overview of the proof is

as follows: Let S be a set of size k in an n-sized universe U. We first *reduce* this universe U to another universe U' whose size is bounded by k^2. We ensure that all elements of S are mapped to different elements of U' during this reduction. Let Y be the range of S in U'. We further partition U' into $\log k$ parts such that Y is almost equally divided among these partition. In other words, each partition contains (roughly) $k/\log k$ many elements of Y. For each of these parts, we explicitly store functions which represents all possible q-coloring of elements of Y in this partition. Finally, we "pull back" these functions to obtain a coloring of S.

Definition 5.6.2 (Splitter [81])**.** *An (n,k,q)-splitter $\mathscr{F}$ is a family of functions from $[n]$ to $[q]$ such that for every set $S \subseteq [n]$ of size k there exists a function $f \in \mathscr{F}$ that splits S evenly. That is, for every $1 \leq z,z' \leq q$, $|f^{-1}(z) \cap S|$ and $|f^{-1}(z') \cap S|$ differ by at most 1.*

Lemma 5.6.1. *For every $1 \leq k,q \leq n$ there is a family of (n,k,q)-splitter of size $\mathscr{O}(n^{\mathscr{O}(q)})$ which can be constructed in the same time.*

Proof. Let $x_0 = 0$ and $x_q = n$. For every choice of $q-1$ elements in $[n]$ such that $1 \leq x_1 < x_2 < \cdots < x_{q-1} \leq n$ define a function $f : [n] \to [q]$ as follows. For $x \in [n]$ we set $f(x) = j$ if $x_{j-1} < x \leq x_j$ where $j \in [q]$. This family has size $\binom{n}{q-1}$, and can be constructed in time $\mathscr{O}(n^{\mathscr{O}(q)})$. □

Following is another well known result for construction of splitter when $q = k^2$. We use this result to *reduce* the size of the universe.

Proposition 5.6.1 ([81])**.** *For any $n,k \geq 1$ one can construct an (n,k,k^2)-splitter of size $\mathscr{O}(k^{\mathscr{O}(1)} \log n)$ in time $\mathscr{O}(k^{\mathscr{O}(1)} n \log n)$.*

Next, we look at the k-RESTRICTION problem defined by Naor et al. [81]. Before defining the problem, we define some terminologies that will be useful. For a fixed set of alphabets, say $\{1,2,\ldots,b\}$ and a vector *vector* V, which is an ordered collection of alphabets, the length of V is the size of the collection. We represent n length vector V as $(v_1,v_2,\ldots,v_n)$.

For a positive integer $i \in [n]$, $V[i]$ denotes the alphabet at the i^{th} position of V. Similarly, for an (index) set $S \subseteq [n]$, $V[S]$ denotes the $|S|$ sized vector obtained by taking alphabet at i^{th} position in V, for each $i \in S$. In other words, if $S = \{i_1, i_2, \ldots, i_k\}$ for $i_1 < i_2 < \cdots < i_k$, then $V[S] = (V[i_1], V[i_2], \ldots, V[i_k])$. An input to the k-RESTRICTION problem is a set $\mathscr{C} = \{C_1, C_2, \ldots, C_m\}$ called as a k-restrictions, where $C_j \subseteq [b]^k$ for $j \in [m]$ and an integer n. Here, $[b]^k$ denotes the set of all possible vectors of length k over $[b]$, and m denotes the size of the k-restrictions. We say that a collection $\mathscr{V}$ of vectors *obeys* $\mathscr{C}$ if for all $S \subseteq [n]$ which is of size k and for all $C_j \in \mathscr{C}$, there exists $V \in \mathscr{V}$ such that $V[S] \in C_j$. The goal of k-RESTRICTION problem is to find a collection $\mathscr{V}$ of as small cardinality as possible, which obeys $\mathscr{C}$. Let $c = \min_{j \in [m]} |C_i|$, and let T be the time needed to check whether or not the vector V is in C_j. We next state the result of Naor et al. [81], which will be useful for proving Theorem 5.6.1.

Proposition 5.6.2 (Theorem 1 [81])**.** *For any k-RESTRICTION problem with $b \leq n$, there is a deterministic algorithm that outputs a collection obeying k-restrictions, which has size at most $(k \log n + \log m)/\log(b^k/(b^k - c))$. Moreover, the algorithm runs in time $\mathscr{O}\left(\frac{b^k}{c}\binom{n}{k} \cdot m \cdot T \cdot n^k\right)$. Here, b is the size of the alphabet set, m is the size of the k-restrictions, n is the size of the vectors in the output set, and c is the size of the smallest collection in the k-restrictions.*

Notice that a function from $[n]$ to $[q]$ can be seen as an n-length vector over the alphabet set $[q]$. Consider the case when each C_j contains exactly one vector of length k over $[q]$, *i.e.* $\mathscr{C} = \{\{C\} \mid C \in [q]^k\}$, $m = q^k$, $c = 1$, and $T = \mathscr{O}(n)$. The output of k-RESTRICTION on this input is exactly an (n,k,q)-universal family. Therefore, we obtain the following corollary.

Corollary 5.6.1. *For any $n, k, q \geq 1$, one can construct an (n,k,q)-universal family of size $\mathscr{O}(q^k \cdot k \cdot (\log n + \log q))$ in time $\mathscr{O}(q^k \cdot n^{\mathscr{O}(k)})$.*

Notice that we can not directly employ Corollary 5.6.1 to construct the desired family,

since its running time is $\mathcal{O}(q^k \cdot n^{\mathcal{O}(k)})$. Therefore, we carefully use splitter to construct an (n,k,q)-universal family to obtain the desired running time.

Proof of Theorem 5.6.1. For the sake of clarity in the notations, we assume that $\log k$ and $k/\log k$ are integers. Let $\mathscr{A}$ be a (n,k,k^2)-splitter obtained by Proposition 5.6.1. Let $\mathscr{B}$ be a $(k^2,k,\log k)$-splitter obtained by Lemma 5.6.1. Let $\mathscr{D}$ be a $(k^2,k/\log k,q)$-universal family obtained by Corollary 5.6.1. We construct $\mathscr{F}$ as follows. For every function f_a in $\mathscr{A}$, f_b in $\mathscr{B}$, and $\log k$ functions $g_1,g_2,\ldots,g_{\log k}$ in $\mathscr{D}$, we construct a tuple $f=(f_a,f_b,g_1,g_2,\ldots,g_{\log k})$, and add it to $\mathscr{F}$. We note here that $g_1,g_2,\ldots,g_{\log k}$ need not be different functions. For $f\in\mathscr{F}$, we define $f:[n]\to[q]$ as follows. For $x\in[n]$, we have $f(x)=g_r(f_b(f_a(x)))$, where $r=f_b(f_a(x))$.

We first argue about the size of $\mathscr{F}$ and the time needed to construct it. Notice that $|\mathscr{F}|\leq|\mathscr{A}||\mathscr{B}||\mathscr{D}|^{\log k}$. We know $|\mathscr{A}|\leq k^{\mathcal{O}(1)}\log n$, $|\mathscr{B}|\leq\mathcal{O}(k^{\mathcal{O}(\log k)})$ and $|\mathscr{D}|\leq q^{k/\log k}k^{\mathcal{O}(k/\log k)}$ by Proposition 5.6.1, Lemma 5.6.1, and Corollary 5.6.1, respectively. This implies that $|\mathscr{F}|\in\mathcal{O}(q^k\cdot k^{\mathcal{O}(\log k)}\cdot\log n)$. Note that $\mathscr{A},\mathscr{B},\mathscr{D}$ can be constructed in time $\mathcal{O}(k^{\mathcal{O}(1)}n\log n)$, $\mathcal{O}(k^{\mathcal{O}(\log k)})$, and $\mathcal{O}(q^k\cdot k^{\mathcal{O}(k/\log k)})$, respectively. This implies that time required to construct $\mathscr{F}$ is bounded by $\mathcal{O}(q^k\cdot k^{\mathcal{O}(k)}\cdot n\log n)$.

It remains to argue that $\mathscr{F}$ has the desired properties. Consider $S\subseteq[n]$ of size k and $\phi:S\to[q]$. We prove that there exists a function $f\in\mathscr{F}$ such that $f|_S\equiv\phi$. By the definition of splitter, there exists $f_a\in\mathscr{A}$ such that f_a evenly splits S (see Definition 5.6.2). Since $|S|<k^2$, for every $y\in[k^2]$, $|f_a^{-1}(y)\cap S|$ is either 0 or 1. Let $Y=\{y_1,y_2,\ldots,y_k\}$ be a subset of $[k^2]$ such that $y_1<y_2<\cdots<y_k$ and $|f_a^{-1}(y_i)\cap S|=1$, for all $i\in[k]$. For $j=k/\log k$, we mark every j^{th} element in set Y marking $\log k-1$ indices altogether. In other words, construct a subset Y' of Y of cardinality $\log k-1$ such that $Y'=\{y_{1j},y_{2j},y_{3j}\ldots,y_{(\log k-1)j}\}$. We use the set Y' to partition $[k^2]$ in a way that every partition contains almost $k/\log k$ many elements of Y. Let $y_0=0$ and $y_{(\log k)j}=k^2$ and define set $Y_r=\{y\in Y\mid y_{r-1}<y\leq y_r\}$ for $r\in[\log k]$. Recall that a $\mathscr{B}$ is $(k^2,k,\log k)$-splitter family obtained by Lemma 5.6.1. By construction, there exists a function f_b which corresponds to subset Y' of $\log k-1$ many

indices. In other words, there is a function f_b such that $f_b^{-1}(r)$ contains all the elements in Y_r, for each r in $[\log k]$. We note that size of $f_b^{-1}(r)$ could be as large as k^2. Recall that $\mathscr{D}$ is a $(k^2, k/\log k, q)$-universal family. Therefore, for every $r \in [\log k]$ there exists $g_r \in \mathscr{D}$ such that $g_r|_{Y_r} \equiv \phi|_{Y_r}$. Consider a function $f = (f_a, f_b, g_1, g_2, \ldots, g_{\log k})$ in $\mathscr{F}$ where f_a, f_b and g_r satisfies the property mentioned above. The function f_a is bijective on S and $f(S) = Y$. The function f_b partitions Y into $\log k$ many parts by mapping Y into $Y_1, Y_2, \ldots, Y_{\log k}$. For each Y_r there exists a function g_r which gives the desired coloring of elements in Y_r and hence for the elements in S. Since we considering all possible combinations of f_a, f_b and $\log k$ functions in $\mathscr{D}$, there exists a function f such that $f|_S \equiv \phi$, which proves the theorem.

5.7 Conclusion

We continue the study of a problem of contracting given graph into a graph class which is *generalization of trees*. For an integer ℓ, we define superclass of trees, denoted by $\mathbb{T}_\ell$, as collection of graphs which can be obtained from a tree by adding at most ℓ edges. We prove that $\mathbb{T}_\ell$-CONTRACTION does not admit a polynomial kernel when parameterized by solution size. We also proved that the additional parameter ℓ is not useful to get polynomial kernel for this problem. But this parameter is useful to get an α-lossy kernel of polynomial size. We presented an FPT algorithm to solve this problem.

Chapter 6

Out-Tree Contraction

6.1 Introduction

In this chapter, we study contraction problem on directed graphs. An *out-tree*, informally speaking, is a rooted tree in which each edge is directed away from root. We study the problem of contracting a given directed graphs into an out-tree. Formally, the problem is defined as follows.

OUT-TREE CONTRACTION	**Parameter:** k
Input: A digraph D and an integer k	
Question: Is it possible to obtain an out-tree from G with at most k edge contractions?	

In Section 6.4, we show that this problem is NP-Hard. This reduction also implies that it does not admit a polynomial kernel when parameterized by k. As in case of TREE CONTRACTION and CACTUS CONTRACTION, we study this problem with number of leaves in resultant out-tree as additional paramter. For the following problem, we present a kernel with $\mathcal{O}(k^2 + k\ell)$ vertices and arcs and prove that this kernel is optimal.

BOUNDED OUT-TREE CONTRACTION (BOUNDED OTC) **Parameter:** $k+\ell$
Input: A digraph D and integers k, ℓ
Question: Is it possible to obtain an out-tree which has at most ℓ leaves, from G with at most k edge contractions?

We address OUT-TREE CONTRACTION as a candidate problem to show an example of *lossy kernelization* for contractions in directed graphs contractions. Given a digraph D on n vertices, an integer k and an approximation parameter $\alpha > 1$, we present an α-lossy kernel which runs in time $n^{\mathcal{O}(1)}$ time and outputs a digraph D' on $\mathcal{O}(k^{2d+1})$ vertices and an integer k' such that for every $c > 1$, a c-approximate out-tree contraction solution for (D', k') can be turned into a $(c\alpha)$-approximate out-tree contraction solution for (G, k) in $n^{\mathcal{O}(1)}$. Here $d = \lceil \frac{\alpha}{\alpha - 1} \rceil$.

Kernel presented in this chapter for BOUNDED OUT-TREE CONTRACTION is based work in [1] whereas lossy kernel for OUT-TREE CONTRACTION is presented in [69].

6.2 Preliminaries

For a *directed graph (or digraph) D*, by $V(D)$ and $A(D)$ we denote the sets of vertices and directed edges (arcs) in D, respectively. Vertex u is said to be adjacent with vertex v in D if there is an arc $uv \in A(D)$ and u, v are said to be endpoints of the arc uv. For $v \in V(D)$, $N_D^-(v)$ denotes the set $\{u \in V(D) \mid uv \in A(D)\}$ of its in-neighbors and $N_D^+(v)$ denotes the set $\{u \in V(D) \mid vu \in A(D)\}$ of its out-neighbors. The neighbourhood of a vertex $v \in V(D)$ is the set $N_D(v) = N_D^+(v) \cup N_D^-(v)$. The closed neighbourhood of a vertex is $N_G[v] = N_G(v) \cup \{v\}$. The in-degree and out-degree of a vertex v, denoted by $d_D^-(v)$, $d_D^+(v)$, is $|N_D^-(v)|$ and $|N_D^+(v)|$ respectively. The (total) degree of v, denoted by $d_G(v)$, is the sum of its in-degree and out-degree. The subscripts in the notation for neighbourhood and degree is omitted if the context is clear. For $F \subseteq A(D)$, $V(F)$ denotes the set of endpoints of arcs in F.

For every digraph we associate an underdirected graph, called *underlying graph* obtained by forgetting the direction of arcs. Formally, for a digraph D, underlying graph G_D is defined as $V(G_D) = V(D)$ and $E(G_D) = \{uv|\ uv \in A(D) \text{ or } vu \in A(D)\}$. A *digraph* is connected (disconnected, 2-connected) if its underlying undirected graph is connected (disconnected, 2-connected). A *spanning tree* of a digraph is defined as set of arcs F such that $V(F) = V(G)$ and unerlying graph induced on $V(F)$ is a tree. A sequence $P = (v_1, \cdots, v_q)$ of distinct vertices of D is called a directed path in D if $v_1v_2, \cdots, v_{q-1}v_q \in A(D)$. Path P is called induced path if $d^-(v_i) = 1$ for all $i \in \{2,3,\ldots,q\}$ and $d^+(v_i) = 1$ for all $i \in \{1,2,\ldots,q-1\}$.

An *out-tree* T is a digraph where each vertex has in-degree at most one and underlying undirected graph is a tree. A vertex v of an out-tree is called a *leaf* if $d^-(v) = 1$ and $d^+(v) = 0$. The *root* of an out-tree is the unique vertex that has no in-neighbour. The number of leaves in an out-tree is the number of vertices whose out-degree is zero.

For a digraph D, contracting an arc $e = uv$ in D is deletion of vertices u, v in D and the addition of a new vertex w and adding arcs to w from in-neighbors of u and v apart from u, v, and from w to out-neighbors of u and v apart from u, v. The resulting graph is denoted by D/e. Formally $V' = (V(D) \setminus \{u,v\}) \cup \{w\}$ with $A(D/e) = \{xy \mid x,y \in V', xy \in A(D)\} \cup \{wx|\ x \in (N_D^+(u) \cup N_D^+(v)) \setminus \{u,v\}\} \cup \{xw|\ x \in (N_D^-(u) \cup N_D^-(v)) \setminus \{u,v\}\}$.

The notion of witness structures and witness sets are extended to digraphs as follows. We say digraph D is *contractible* to digraph H if there exists an onto function $\psi : V(D) \to V(H)$ such that following properties hold.

- For any vertex h in $V(H)$, underlying graph $G_D[W(h)]$ is connected, where set $W(h) := \{v \in V(G) \mid \psi(v) = h\}$.
- For any two vertices h, h' in $V(H)$, arc hh' is present in H if and only if there exists an arc in D with one end point in $W(h)$ and another in $W(h')$.

For a vertex h in H, set $W(h)$ is called *witness set* associated with vertex h. We define

H-witness structure of D, denoted by $\mathscr{W}$, as collection of all witness set. Formally, $\mathscr{W} = \{W(h) \mid h \in V(H)\}$. Witness structure $\mathscr{W}$ is a partition of vertices in D. If a *witness set* contains more than one vertex then we call it *big* witness-set, otherwise it is *small/singleton* witness set. For a fixed H-witness structure, let F be union of spanning trees of all witness sets. We say digraph D is *k-contractible* to H if cardinality of F is at most k. Following observation is direction consequence of definitions.

Observation 6.2.1. *If digraph D is k-contractible to digraph H then following statements are true.*

- $|V(D)| \leq |V(H)| + k$.
- *For any witness set W in a H-witness structure of G, cardinality of W is at most $k+1$.*
- *Any H-witness structure of D has at most k big witness sets.*
- *For a fixed H-witness structure, the number of vertices in D which are contained in big witness sets is at most $2k$.*

Digraph obtained by *subdividing* an arc of an out-tree results in another out-tree. The operation of subdividing an arc uv in D is consists of deletion of the arc uv and addition of a new vertex w as an out-neighbor of u and an in-neighbor of v.

Observation 6.2.2. *Consider an out-tree T with at most ℓ leaves. Let T' be the out-tree obtained from T by one of the following operations.*

1. *subdividing an arc;*
2. *contracting an arc;*

Then, T' is an out-tree with at most ℓ leaves.

Proof. Proof of Part (1) Let t_1t_2 be an arc in T which is subdivided to obtain graph T'. Let t^* be newly added vertex while subdividing arc t_1t_2. Note that $d^-_T(t) = d^-_{T'}(t)$ and $d^+_T(t) = d^+_{T'}(t)$ for any vertex in t in $V(T') \setminus \{t^*\} = V(T)$. Also, $d^-_{T'}(t^*) = d^+_{T'}(t^*)$. This

also implies that t^* is not a leaf in T'. Hence the number of leaves in T and T' is same. Every vertex in T' has in-degre at most one. If there exists a cycle in $G_{T'}$ which passes through t^* then the same cycle passes through t_1, t_2. This implies there exists a cycle in G_T which passes through t_1, t_2. This contradicts the fact that G_T is an underlying graph of an out-tree. Hence T' is an out-tree with at most ℓ leaves.

Proof of Part (2) Let t_1t_2 be an arc in T which is contracted to obtain graph T'. Let t^* be newly added vertex while contracting arc t_1t_2. Note that no vertex in T is has an arc to or from both t_1 and t_2. This implies $d^-_T(t) = d^-_{T'}(t)$ and $d^+_T(t) = d^+_{T'}(t)$ for any vertex in t in $V(T') \setminus \{t^*\} = V(T) \setminus \{t_1, t_2\}$. Moreover, by contruction, $d^-_T(t_1) = d^-_{T'}(t^*)$ and $d^+_T(t_2) = d^+_{T'}(t^*)$. Hence T' is an out-tree. Also, t^* is a leaf in T' if and only if t_2 is a leaf in T. This implies T' is an out-tree with at most ℓ leaves. □

In the following lemma, we argue that if D is k-contractible to an out-tree and there exists a long induced path then no minimal solution is incident on any vertex of this path.

Lemma 6.2.1. *Suppose D has a directed path $P = (v_0, v_1, \ldots, v_q, v_{q+1})$ with $q > k+1$ and $d^-(v) = d^+(v) = 1$ for each $i \in [q]$. Let F be a set of arcs of D such that $|F| \leq k$ and D/F is an out-tree with at most ℓ vertices. If F is minimal then it does not contain an edge incident on $V(P) \setminus \{v_0, v_{q+1}\}$.*

Proof. Assume that F contains at least one such arc. There are at least $k+1$ arcs with endpoints in $V(P) \setminus \{v_0, v_{q+1}\}$. Since $|F| \leq k$, there exists v_i in $\{v_0, v_1, \ldots, v_q, v_{q+1}\}$ such that $v_{i-1}v_i \in F$ and $v_iv_{i+1} \notin F$. Let $\mathscr{W}$ denote the corresponding T-witness structure of D where $T = D/F$. Now, let t and t' denote the vertices of T such that $\{v_{i-1}, v_i\} \subseteq W(t)$ and $v_{i+1} \in W(t')$. If $t = t'$ then $v_{i-1}, v_i, v_{i+1} \in W(t)$ and $v_iv_{i+1} \notin F$. As $G_D[W(t)]$ is connected, there must be a path connecting v_i, v_{i+1} in G_D which is entirely contained in $W(t)$. Any path between v_i, v_{i+1} which does not contain edges v_iv_{i+1} must contain a path from v_i to v_0 and the path from v_{q+1} to v_{i+1}. It implies that $W(t)$ contains the vertices of the subpath $(v_{i+1}, \ldots, v_q, v_{q+1})$ and the vertices of the subpath $(v_0, v_1, \ldots, v_{i-1}, v_i)$. This

implies $|W(t)| > k+1$ which is a contradiction to the fact that T is obtained from D by contracting at most k edges. Hence $t \neq t'$. We now focus on $W(t)$ which, as argued above, does not contain v_{i+1}. Vertex v_i is not a cut vertex in $G_D[W(t)]$ as there is exactly one edge incident on it. This implies $G_D[W(t) \setminus v_i]$ is a connected graph. Define $\mathscr{W}' = (\mathscr{W} \setminus \{W(t)\}) \cup \{\{v_i\}\} \cup \{W(t) \setminus \{v_i\}\}$. Graph $D/(F \setminus \{v_{i-1}v_i\})$ is isomorphic to graph obtained by subdividing the arc tt' in the out-tree T. Thus, $\mathscr{W}'$ is an out-tree witness structure of D leading to the solution $F \setminus \{v_{i-1}v_i\}$ which contradicts the minimality of F. □

Note that in the above proof, we did not use the fact that T has at most ℓ leaves. Hence this claim is true for any out-tree. We mention the result explicitely in the following lemma.

Lemma 6.2.2. *Suppose D has a directed path $P = (v_0, v_1, \ldots, v_q, v_{q+1})$ with $q > k+1$ and $d^-(v) = d^+(v) = 1$ for each $i \in [q]$. Let F be a set of arcs of D such that $|F| \leq k$ and D/F is an out-tree. If F is minimal then it does not contain an edge incident on $V(P) \setminus \{v_0, v_{q+1}\}$.*

6.3 Kernel for BOUNDED OUT-TREE CONTRACTION

In this section we design a polynomial kernel for BOUNDED OUT-TREE CONTRACTION. Our algorithm is inspired by kernelization algorithm for PATH CONTRACTION presented in [55].

Let (D, k, ℓ) be an instance of BOUNDED OTC. Without loss of generality we assume that D is connected, else (D, k, ℓ) is a NO instance. Recall that D is connected if its underlying undirected graph G_D is connected.

The algorithm has only one reduction rule.

Reduction Rule 6.3.1. *Let $P = (v_0, v_1, \ldots, v_q, v_{q+1})$ be an indueced path in D with $q > k+3$ and $d^-(v) = d^+(v) = 1$ for each $i \in [q]$. Then contract the arc $v_{q-1}v_q$ and let the*

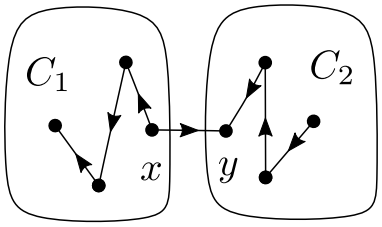


Figure 6.1: Different between reduction rules in case of directed and un-directed graphs.

resulting instance be (D', k, ℓ)*, where* $D' = D/\{v_{q-1}v_q\}$.

We note that unlike in case of undirected graph (Reduction Rule 3.3.1), it is not enough to find an cut arc whose remove results into two connected components of size at least $k+1$. We might still have to contract this edge because of direction constraints. See Figure 6.1.

Lemma 6.3.1. *Reduction rule 6.3.1 is safe and can be applied in polynomial time.*

Proof. We need to show that (D, k, ℓ) is a YES instance of BOUNDED OTC if and only if (D', k, ℓ) is a YES instance of BOUNDED OTC. Clearly, given D and P one can apply Reduction Rule 6.3.1 in polynomial time.

In the forward direction, let (D, k, ℓ) be a YES instance of BOUNDED OTC and let $F \subseteq A(D)$ such that $|F| \leq k$ and $T = D/F$ is an out-tree with at most ℓ leaves. By Observation 6.2.2, we know that $D/(F \cup \{v_{q-1}v_q\})$ is also an out-tree with at most ℓ leaves. However, $D/(F \cup \{v_{q-1}v_q\}) = (D/\{v_{q-1}v_q\})/(F \setminus \{v_{q-1}v_q\}) = D'/(F \setminus \{v_{q-1}v_q\})$. This implies that $D'/(F \setminus \{v_{q-1}v_q\})$ is an out-tree with at most ℓ leaves and $|F \setminus \{v_{q-1}v_q\}| \leq |F| \leq k$. Hence, it follows that (D', k, ℓ) is a YES instance of BOUNDED OTC.

In the reverse direction, let (D', k, ℓ) be a YES instance of BOUNDED OTC and let $F' \subseteq A(D')$ of size at most k such that $T' = D'/F'$ is an out-tree with at most ℓ leaves. Let $\mathscr{W}'$ be a T'-witness structure of D'. Let v^*_{q-1} be the vertex obtained after contracting the arc $v_{q-1}v_q$. Let P^* be the path from v_0 to v_{q+1} in graph D'. In other words, P^* is a path obtained from P by contracting edge $v_{q-1}v_q$. Since P^* is a path of size $k+2$, by Lemma 6.2.1, no edge in F' is incident on vertices in P^*. This implies that if $W(t^*)$ is the

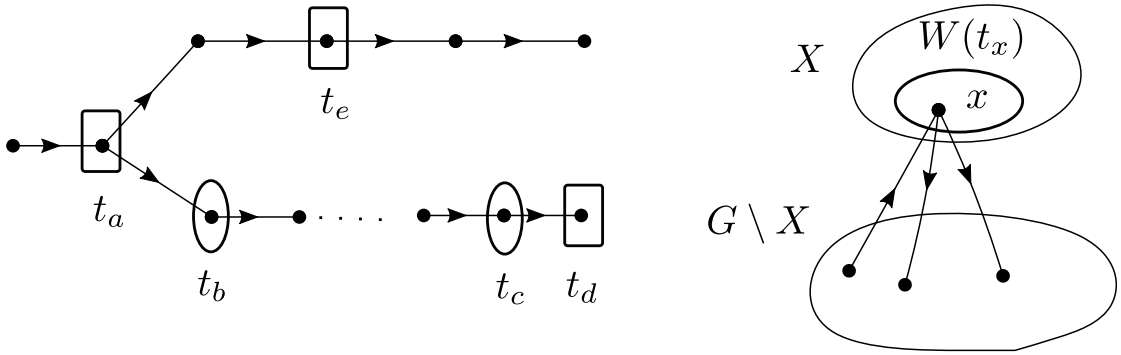


Figure 6.2: For left figure, please refer to Lemma 6.3.2. Vertices t_a, t_d are marked as they are part of $T_1 \cup T_3$. Vertices t_c, t_d are marked because they are end-points of a path. Vertex t_e marked as $W(t_e)$ is a big witness set. For figure on right, please refer to Lemma 6.3.3.

witness set in $\mathscr{W}'$ which contains v^*_{q-1} then $W(t^*)$ is a singleton witness set. Moreover, every vertex in $V(P) \setminus \{v_{q-1}, v_q\}$ is in singlton witness set in $\mathscr{W}'$. Let t_1, t_2 be two vertices in T' which are in-neighor and out-neighor, respectively, of t^*.

Consider a witness structure $\mathscr{W}$ obtained from $\mathscr{W}'$ by removing $\{v^*_{q-1}\}$ and adding two sets $\{v_{q-1}\}, \{v_q\}$. Formally, $\mathscr{W} = (\mathscr{W}' \setminus \{v^*_{q-1}\}) \cup \{\{v_{q-1}\}, \{v_q\}\}$. Note that $\mathscr{W}$ partitions $V(D)$ and for each $W \in \mathscr{W}$, $D[W]$ is connected. Let T be the digraph for which $\mathscr{W}$ is a T-witness structure of D. We argue that T is an out-tree with at most ℓ edges. Note that T can be obtained from T' by subdividing edge t^*t_2. By Observation 6.2.2, T is an out-tree with at most ℓ leaves. This completes the proof of the lemma. □

For simplicity, by (D, k, ℓ) we denote an instance of BOUNDED OTC on which the Reduction Rule 6.3.1 is not applicable.

Lemma 6.3.2. *Let (D, k, ℓ) be a* YES *instance of* BOUNDED OTC *on which Reduction Rule 6.3.1 is not applicable. Then, D has at most $\mathcal{O}(k^2 + k\ell)$ vertices.*

Proof. Let (D, k, ℓ) be a YES instance and $F \subseteq A(D)$ be a solution such that $T = D/F$ is an out-tree with at most ℓ leaves. Let $\mathscr{W}$ be a T-witness structure of a digraph D. For counting the number of vertices in D, we first count the vertices in T. Towards this we employ a miarking scheme. By M we denote the set of vertices in T that have been marked by our

scheme. Let X be the set of vertices in T which corresponds to big witness sets in $\mathscr{W}$. We mark all the vertices in X. Let T_1, T_3 denote the set of vertices in T which have total degree exactly one and at least three, respectively in T. We mark all the vertices in T_1 and T_3. Note that $|T_1| \leq \ell + 1$. Here, we have $|T_1| \leq \ell + 1$, rather than $|T_1| \leq \ell$, to take into account the case when the root of T has total degree 1. Also, $|X| \leq k$ and $|T_3| \leq |T_1|$. Therefore, it follows that currently the number of vertices in M is upper bounded by $k + 2\ell + 2$. See Figure 6.2.

Let $\mathscr{P}$ be the set of induced maximal (directed) paths in $T[V(T) \setminus M]$. Observe that, by viewing each path in $\mathscr{P}$ as an edge between vertices in M we get a tree on M. Thus, $|\mathscr{P}| \leq |M| - 1$. For each $P \in \mathscr{P}$, we additionally mark two of the endpoints in P. Clearly, this increases the size of M by at most $2|P|$. However, even now the size of $|M| = \mathscr{O}(k + \ell)$. Note that each of the unmarked vertices have in-degree and out-degree exactly one. Since Reduction Rule 6.3.1 is not applicable, therefore length of each of the maximal paths comprising of unmarked vertices is bounded by $\mathscr{O}(k)$. But then, the number of vertices in T is bounded by $\mathscr{O}(k^2 + k\ell)$. As T is obtained using at most k edge contractions from digraph D, it follows from Observation 6.2.1 that $|V(D)| \leq |V(T)| + k$. Since $|V(T)| = \mathscr{O}(k^2 + k\ell)$, this implies that $|V(D)| = \mathscr{O}(k^2 + k\ell)$. □

Lemma 6.3.3. *Let (D, k, ℓ) be a* YES *instance of* BOUNDED OTC *on which Reduction Rule 6.3.1 is not applicable. Then, D has at most $\mathscr{O}(k^2 + k\ell)$ arcs.*

Proof. Let (D, k, ℓ) be a YES instance and $F \subseteq A(D)$ be a set of edges such that $T = D/F$ is an out-tree with at most ℓ leaves. Let $\mathscr{W}$ be a T-witness structure of a digraph D. Let X be the set of vertices in D to which an edge in F is incident to. Notice that $|X| \leq 2k$. The number of arcs with both endpoints in X is bounded by $\mathscr{O}(k^2)$. Observe that the underlying undirected graph of $D - X$ is a forest with at most $\mathscr{O}(k^2 + k\ell)$ vertices. This implies the number of arcs in D that have both endpoints in $D - X$ is bounded by $\mathscr{O}(k^2 + k\ell)$. The only arcs that remain to be counted are those with one endpoint in $D - X$ and other in X. For a vertex $x \in X$, let t_x be the vertex in $V(T)$ such that $x \in W(t_x)$. Also let $\hat{N}$ be the neighbors

of t_x in T. Observe that $|\hat{N}| \leq \ell + 1$. This together with Observation 6.2.1 implies that $|\cup_{t \in \hat{N}} W(t)|$ is bounded by $2k + \ell + 1$. Therefore, the maximum degree of a vertex in X is bounded by $\mathcal{O}(k + \ell)$. This implies that the number of arcs with one end point in X and other in $D - X$ is bounded by $\mathcal{O}(k^2 + k\ell)$. We have counted all types of arcs in D and hence, we conclude that the number of arcs in D is bounded by $\mathcal{O}(k^2 + k\ell)$. □

We are now ready to prove the main theorem of this section.

Theorem 6.3.1. BOUNDED OTC *admits a kernel of size* $\mathcal{O}(k^2 + k\ell)$.

Proof. Given an instance (D, k, ℓ), the algorithm repeatedly applies Reduction Rule 6.3.1, if applicable. By Lemma 6.3.1, we know that Reduction Rule 6.3.1 is safe and can be applied in polynomial time. Each application of reduction rule decreases the number of arcs and thus it can be applied only $|A(D)|$ times. If Reduction Rule 6.3.1 is not applicable then either the size of the instance is bounded by $\mathcal{O}(k^2 + k\ell)$, in which case we return a kernel of desired size. Otherwise, the algorithm correctly concludes that the instance is a NO instance of BOUNDED OTC. The correctness of this step follows by Lemmas 6.3.2 and 6.3.3. □

6.4 Kernel Lower Bound for BOUNDED OUT-TREE CONTRACTION

In this section we present a parameter preserving reduction from given an instance (G, R, B, k) of RBDS to an instance (D', k', ℓ') of BOUNDED OUT-TREE CONTRACTION. This reduction is same as the one presented in Section 3.4. We use this reduction to prove three things. First, we show that OUT-TREE CONTRACTION is NP-Hard. Second, OUT-TREE CONTRACTION parameterized by solution size k does not admit a polynomial kernel assuming NP $\not\subseteq$ coNP/poly. Third, the kernel presented for BOUNDED OTC in Section 6.3 is optimal under the same assumption.

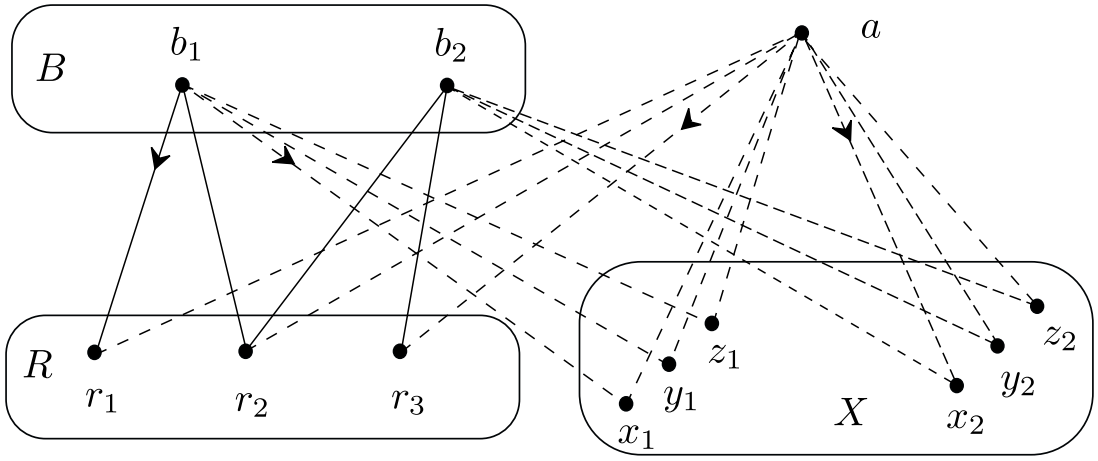


Figure 6.3: Kernel lower bound for BOUNDED OTC. For the sake of clarity, figure does not show directions for all arcs.

Reduction. Let (G,R,B,k) be an instance of RBDS. We construct graph G' in the following way. See Figure 6.3. Initialize $V(G') = V(G)$ and $E(G') = \{br \mid b \in B, r \in R \text{ and } br \in E(G)\}$. Add a vertex a in $V(G')$ and for every vertex r in R, add an edge ar to $E(G')$. For every vertex b_i in B, add three new vertices x_i, y_i, z_i to $V(G')$ and arcs b_ix_i, b_iy_i, b_iz_i to $E(G')$. Define set $X := \{x_i, y_i, z_i \mid b_i \in B\}$. We construct diagraph D' from G' by adding directions to edegs. For every vertex x in X, add an edge ax to $E(G')$. For every edge incident on a, add direction from a to other end point. Similarly, for any end incident on vertices in B, add direction from vertex in B to other end point. Set $k' = |B| + k$ and $\ell' = |R| + 3|B| - k$.

In the following lemma, we prove some structural properties of a solution to instance (D', k', ℓ').

Lemma 6.4.1. *Let (D', k', ℓ') be a* YES *instance of* BOUNDED OUT-TREE CONTRACTION. *There exists a solution $F^* \subseteq E(D')$ of size at most k' such that for each b_i in B one of the following holds.*

1. *b_i is in $W(t_a)$ or*
2. *x_i, y_i, z_i are in $W(t_a)$.*

Here, $W(t_a)$ is the witness set containing a in (D'/F^*)-witness structure of D'.

Proof. Let F be a set of arcs of size at most k in D' such that D'/F is an out-tree with at most ℓ leaves. Let $\mathscr{W}$ be a T-witness structure of D' where $T = D'/F$. Recall that T_G denotes the underlying undirected graph of T. Since T is an out-tree, T_G is a tree. Let t_a be the vertex in $V(T)$ such that $W(t_a)$ contains a. For a vertex b_i in B, if b_i is in $W(t_a)$ then the lemma holds. Consider a case when b_i is not in $W(t_a)$. There exists a vertex t_b, different from t_a, such that b_i is contained in $W(t_b)$. Similarly, consider vertices t_x, t_y and t_z such that x_i, y_i and z_i are contained in $W(t_x), W(t_y)$ and $W(t_z)$, respectively.

If neither of t_a or t_b is contained in set $\{t_x, t_y, t_z\}$, then no two of $\{t_x, t_y, t_z\}$ can be same as only neighbors of x_i, y_i, z_i are a and b_i, and by definition, a witness set needs to be connected. But then, by construction, $T_G[\{t_a, t_x, t_y, t_z, t_b\}]$ is a cycle, contradicting the fact that T_G is a tree. Therefore, at least one of $\{t_x, t_y, t_z\}$ is same as t_a or t_b. Without loss of generality, let $t_s \in \{t_a, t_b\}$. This implies there is an edge $t_a t_b$ in T_G. If t_y or t_z is not equal to t_a or t_b then there exist a cycle contradicting that T_G is a tree. Suppose, all t_x, t_y, t_z are same as t_a, then the second condition of the lemma is satisfied. Consider a case when at least one of t_x, t_y, t_z, say t_x, is not same as t_a, which implies $t_x = t_b$. By construction, the only arcs incident to x_i in D' are ax_i and bx_i. This implies that $bx_i \in F$ and $W(t'_b) = W(t_b) \setminus \{x_i\}$ is connected. Since $ax_i \in A(D')$, set $W(t'_a) = W(t_a) \cup \{x_i\}$ is connected. Thus, replacing $W(t_b)$ by $W(t'_b)$ and $W(t_a)$ by $W(t'_a)$ in $\mathscr{W}$ yields another T-witness structure of D'. Furthermore, the spanning forest of the new witness structure, $F' = (F \setminus \{bx_i\}) \cup \{ax_i\}$ has same cardinality as that of F. A similar swap can be carried out if $t_y = t_b$ or $t_z = t_b$. This concludes the proof. □

In the following lemma, we argue that the reduction is safe.

Lemma 6.4.2. *(G, R, B, k) is a* YES *instance of* RBDS *if and only if (D', k', ℓ') is a* YES *instance of* BOUNDED OTC.

Proof. Let (G, R, B, k) be a YES instance of RBDS and S be a subset of R of size k such

that S dominates every vertex in B. If S contains less than k vertices, then we take any of its superset of size exactly k. For each vertex b in B, we fix a vertex r_b in S such that b is neighbor of r_b in G. If there are multiple options for selecting r_b then we arbitrarily choose one of them. Let $F = \{br_b \mid b \in B\} \cup \{ar \mid r \in S\}$. Note that $|F| = |B| + k = k'$ and $D'[V(F)]$ is connected. Let T be the digraph obtained from D' by contracting edges in F. Let $\mathscr{W}$ be a T-witness structure of D'. Consider a vertex t_a such that a is in $W(t_a)$. Since all the edges in F are contracted to one vertex, set $S \cup B$ is also contained in $W(t_a)$. Recall that $R \cup X$ is an independent set in $G_{D'}$. No vertex in $(R \cup X) \setminus S$ is incident on edge which has been contracted. In other words, these vertices form singleton witness sets in $\mathscr{W}$. Since $R \cup X$ is an independent set in $G_{D'}$, it follows that set $T_{RS} = \{t_v \mid v \in (R \cup X) \setminus S\}$ is an independent set in G_T of size $|R| + 3|B| - k = \ell'$. Moreover, for all v in X', arc av is present in $A(T)$. Therefore, T is a out-tree with ℓ' leaves. This implies that F is a solution to (D', k', ℓ').

In the reverse direction, let (D', k', ℓ') be a YES instance of BOUNDED OUT-TREE CONTRACTION. By Lemma 3.4.1, there exists a solution F^* of size at most k' such that for every b_i in B, either b_i is in $W(t_a)$ or all of x_i, y_i, z_i are in $W(t_a)$. Here, $\mathscr{W}$ is the D'/F^*-witness structure of D' and t_a in $V(D'/F^*)$ such that vertex a is contained in witness set $W(t_a)$ in $\mathscr{W}$.

We partition vertices of B into two parts depending on whether they belong to $W(t_a)$ or not. Define set $B_g = \{b_i \in B \mid b_i \in W(t_a)\}$. Let $R_a = R \cap W(t_a)$. Partition B_g into B_1 and B_2, depending on whether or not they have a neighbor in R_a. Formally, $B_1 = \{b_i \in B_g \mid N(b_i) \cap R_a \neq \emptyset\}$ and $B_2 = B_g \setminus B_1$. For a vertex b_i in B_2 at least one of x_i, y_i, z_i is present in $W(t_a)$ as there is no arc between b_i and a. Note that, by construction, x_i, y_i, z_i is not adjacent with b_j for $i \neq j$. This implies there exists a separate vertex for each b_i in B_2 which provides connectivity between a and b_i. Let XB_2 be set of vertices in $X \cap W(t_a)$ which provides adjacency between a and b_i for some b_i in B_2. For every b_i which is in $B \setminus B_g$, by Lemma 6.4.1, x_i, y_i, z_i are present in $W(t_a)$.

We can partition $W(t_a) \setminus \{a\}$ into following four parts: vertices in B (captured by B_g); vertices in R (captured by R_a); vertices in X which are present because corresponding b_i is not present (captured by $B \setminus B_g$); and vertices in X which are present because they are needed to provide connectivity between b_i and a (captured by XB_2). This implies $|B_g| + 3|B \setminus B_g| + |R_a| + |XB_2| + |\{a\}| \leq |W(t_a)|$.

We construct a solution S for RBDS by taking vertices in R_a and two more sets S_g and S_w. Informally, S_g dominates vertices in B_2 and S_w dominates vertices in $B \setminus B_g$. We construct S_g in following way. For every vertex b_i in B_2, arbitrary pick one of its neighbor in R and add it to S_g. Note that $|S_g| \leq |XB_2|$. We create another set S_w in the following way. Initialize S_w to an empty set. For each b in $B \setminus B_g$, we add an arbitrary neighbor of b in R to S_w. This implies $|S_w| \leq |B \setminus B_g|$. As cardinality of F^* is at most $k + |B|$, size of $W(t_a)$ is at most $|W(t_a)| \leq k + |B| + 1$.

Putting all inequalities together, we get $|R_a| + |S_g| + |S_w| \leq k$ and every vertex in B is dominated some vertex in $R_a \cup S_g \cup S_w$. This concludes the proof. □

RED BLUE DOMINATING SET is NP-Complete [44] and it does not have a polynomial kernel when parameterized by $(|B|, k)$ [32]. This implies that classical problem of OUT-TREE CONTRACTION is NP-Complete. The existence of the polynomial parameter transformation described above and Proposition 2.3.1 implies following theorem.

Theorem 6.4.1. OUT-TREE CONTRACTION *does not have a polynomial kernel unless* NP $\subseteq$ coNP*/poly.*

We now argue that kernel presented for BOUNDED OTC is optimal.

Theorem 6.4.2. BOUNDED OUT-TREE CONTRACTION *does not admit a compression of size* $\mathcal{O}((k^2 + k\ell)^{1-\varepsilon})$*, for any* $\varepsilon > 0$ *unless* NP $\subseteq$ *coNP/poly.*

Proof. Assuming a contradiction, suppose BOUNDED OUT-TREE CONTRACTION admits a compression into $\Pi \subseteq \Sigma^*$ with bitsize in $\mathcal{O}((k^2 + k\ell)^{1-\varepsilon})$, for some $\varepsilon > 0$. This implies

that there exists an algorithm $\mathscr{A}$ which takes an instance $I = (G,k,\ell)$ of BOUNDED OUT-TREE CONTRACTION and in polynomial time returns an equivalent instance I' of Π with $|I'| \in \mathscr{O}((k^2+k\ell)^{1-\varepsilon})$.

Let (G,R,B,k) be an instance of RBDS, where G is a graph on n vertices. Using the reduction described, we create an instance (G,k',ℓ') of BOUNDED OUT-TREE CONTRACTION with $|V(G'_D)| \in \mathscr{O}(n)$, $|E(G'_D)| \in \mathscr{O}(n^2)$, $k' = k \leq |R| \in \mathscr{O}(n)$ and $\ell' = |B| + k \in \mathscr{O}(n)$. On the instance (G,k',ℓ') we run the algorithm $\mathscr{A}$ to obtain an instance I of Π such that $|I| \in \mathscr{O}((k'^2+k'\ell')^{1-\varepsilon})$. But then we have obtained a compression of size $\mathscr{O}(n^{2-\varepsilon})$ for RBDS, contradicting Proposition 2.3.2. □

Corollary 6.4.1. BOUNDED OUT-TREE CONTRACTION *does not admit a kernel of size* $\mathscr{O}((k^2+k\ell)^{1-\varepsilon})$, *for any* $\varepsilon > 0$ *unless* NP $\subseteq$ *coNP/poly.*

6.5 Lossy Kernel for OUT-TREE CONTRACTION

In this section, we describe a PSAKS for OUT-TREE CONTRACTION. We define parameterized minimization version of OUT-TREE CONTRACTION problem in the following way.

$$\mathrm{OTC}(D,k,F) = \begin{cases} \infty & \text{if D/F is not an out-tree} \\ \min\{|F|, k+1\} & \text{otherwise} \end{cases}$$

We note that the simplifying assumptions in TREE CONTRACTION, such as working with 2-connected components or the fact that leaves correspond to singleton witness sets, do not hold anymore. At this place, our treatment of directed and un-directed graph differs.

If D has at most $k+3$ vertices then we already have a kernel of desired side. We assume that input digraph has at least $k+3$ vertices. By definition of optimization problem, for a set of arcs F, if D/F is an out-tree then maximum value of $\mathrm{OTC}(D,F)$ is $k+1$. Hence any spanning tree of D is a solution of size $k+1$. We call it a *trivial solution* for given

instance. We denote a directed cycle on four vertices by C_4. One need to contract at least two edge to get an out-tree from C_4. We call $(C_4,1)$ as *trivial instance* of OUT-TREE CONTRACTION. If $\mathrm{OPT}(D,k) = k+1$ then we can return $(C_4,1)$ as its α-lossy kernel. Note that for any c-factor solution of $(C_4,1)$, solution lifting algorithm can return a trivial solution for (G,k) which is of size $k+1$. If underlying undirected graph of input digraph is not connected then we can not obtain an out-tree by edge contraction operations only. We assume underlying undirected input graph is connected.

First reduction rule states that it is safe to remove vertices which has one in-neighbor and zero out-neighbors. After applying this rule exhaustively, if digraph is contracted to an out-tree then for each leaf, either it or its unique neighbor correspond to a big witness set.

Reduction Rule 6.5.1. *If there is a vertex $v \in V(D)$ with $d^-(v) = 1$ and $d^+(v) = 0$ then delete v. The resulting instance is (D',k') where $D' = D - \{v\}$ and $k' = k$.*

This reduction rule can be applied in time polynomial time. Correctness of this reduction rule is based on the observation that the digraph obtained from an out-tree by adding a new vertex as an out-neighbor of any vertex is an out-tree.

Lemma 6.5.1. *Reduction Rule 6.5.1 is* 1*-safe.*

Proof. Consider a set $F' \subseteq A(D')$ such that $T' = D'/F'$ is an out-tree. If $|F'| \geq k'+1$, then the solution lifting algorithm returns a spanning tree F of D, otherwise it returns $F = F'$. If $|F'| \geq k'+1$ then $\mathrm{OTC}(D,k,A(D)) \leq k+1 = \mathrm{OTC}(D',k',F')$. In case $|F'| \leq k+1$, let $\mathscr{W}'$ denote the corresponding T'-witness structure of D'. There exists a vertex $t_i \in V(T')$ such that the unique neighbor of v in D is in $W(t_i)$. Define the partition of $V(D)$ as $\mathscr{W} = \mathscr{W}' \cup \{\{v\}\}$. No vertex in any set $W(t) \in \mathscr{W}'$ with $t \neq t_i$ contains a vertex that is adjacent to v. Thus $\mathscr{W}$ is a D/F-witness structure of D, where D/F is the out-tree obtained from T' by adding a new vertex, say t_v, as out-neighbor of t_i where $W(t_v) = \{v\}$. Hence, $\mathrm{OTC}(D,k,F) \leq \mathrm{OTC}(D',k',F')$.

Consider an optimum solution F^* to (D,k). If $|F^*| \geq k+1$, then $\mathrm{OPT}(D,k) = k+1$ and by definition, $\mathrm{OPT}(D',k') \leq k'+1 = k+1 = \mathrm{OPT}(D,k)$. Consider the case when $|F^*| \leq k$. Let $\mathscr{W}^*$ denote the corresponding T-witness structure of D where $T = D/F^*$. There exists $t \in V(T)$ such that $v \in W(t)$. If $W(t)$ is a singleton set, then F^* is also a solution to (D',k') and $\mathrm{OPT}(D',k') \leq \mathrm{OPT}(D,k)$. If $W(t)$ is not a singleton set, there exists an arc, say e in F which is incident on v. As v is a vertex of degree 1, the underlying undirected subgraph of $D[W(t) \setminus \{v\}]$ is connected. Since $d^-(v) = 1$ and $d^+(v) = 0$, $F^* \setminus e$ is also a solution to (D,k) contradicting the fact that F^* is an optimal solution. Hence the second case does not occur. This implies $\mathrm{OPT}(D',k') \leq \mathrm{OPT}(D,k)$

Putting together two inequalities, we get $\frac{\mathrm{OTC}(D,k,F)}{\mathrm{OPT}(D,k)} \leq \frac{\mathrm{OTC}(D',k',F')}{\mathrm{OPT}(D',k')}$ which concludes the proof. □

Second reduction rule states if there exists a long induced path in digraph then we can find an edge which can be safely contracted.

Reduction Rule 6.5.2. *If D has a directed path $P = (v_0, v_1, \ldots, v_q, v_{q+1})$ with $q > k+2$ and $d^-(v) = d^+(v) = 1$ for each $i \in [q]$, then contract edge $v_{q-1}u_q$. The resulting instance is (D',k') where $D' = D/\{v_{q-1}v_q\}$ and $k' = k$.*

This rule can be applied in polynomial time by searching for a path in the subgraph induced on the vertices of degree two.

Lemma 6.5.2. *Reduction Rule 6.5.2 is 1-safe.*

Proof. Consider a minimal set $F' \subseteq A(D')$ such that $T' = D'/F'$ is an out-tree. If $|F'| \geq k'+1$, then the solution lifting algorithm returns a spanning tree F of D, otherwise it returns $F = F'$. If $|F'| \geq k'+1$ then $\mathrm{OTC}(D,k,F) \leq k+1 = \mathrm{OTC}(D',k,F')$. Otherwise, let $\mathscr{W}'$ be a T'-witness structure of D'. Let v^*_{q-1} be the new vertex added while contracting edge $v_{q-1}v_q$. Let P' be the path obtained from P by this contraction. By Lemma 6.2.2, F' has no arc incident on $V(P') \setminus \{v_0, v_{q+1}\}$. Therefore, every vertex in $V(P') \setminus \{v_0, v_{q+1}\}$ is

in a singleton set of $\mathscr{W}'$. Let $\mathscr{W}$ be a witness structure obtained from $\mathscr{W}'$ by removing $\{v^*_{q-1}\}$ and adding $\{v_{q-1}\},\{v_q\}$. If $\mathscr{W}$ is a T-witness structure of D then T is obtained from T' by subdividing one of its edges, namely $t_{q-1}t_{q+1}$ where $W(t_{q-1}) = \{v^*_{q-1}\}$ and $W(t_{q+1}) = \{v_{q+1}\}$. By Observation 6.2.2, T is an out-tree. Therefore, $\mathrm{OTC}(D,k,F) \leq \mathrm{OTC}(D',k',F')$.

Next, consider a minimal optimum solution F^* to (D,k). If $|F^*| \geq k+1$ then $\mathrm{OPT}(D,k) = k+1$ and by definition, $\mathrm{OPT}(D',k') \leq k'+1 = k+1 = \mathrm{OPT}(D,k)$. Otherwise, $|F^*| \leq k$ and let $T = D/F^*$. Let $\mathscr{W}$ denote the corresponding T-witness structure of D. By Lemma 6.2.2, F^* has no edge incident on $V(P) \setminus \{v_0, v_{q+1}\}$. Therefore, every vertex in $V(P) \setminus \{v_0, v_{q+1}\}$ is a singleton set of $\mathscr{W}$. Let v^*_{q-1} be the new vertex added while contracting edge $v_{q-1}v_q$. Define $\mathscr{W}'$ be a witness set obtained from $\mathscr{W}$ by removing $\{v_{q-1}\},\{v_q\}$ and adding $\{v^*_{q-1}\}$. If $\mathscr{W}'$ is a T'-witness structure of D then T' can be obtained from T by contracting an edge $t_{q-1}t_q$ where $W(t_{q-1}) = \{v_{q-1}\}$ and $W(t_q) = \{v_q\}$. By Observation 6.2.2, T' is an out-tree. Thus, $\mathrm{OPT}(D',k') \leq \mathrm{OPT}(D,k)$.

Combining two inequalities, we get $\frac{\mathrm{OTC}(D,k,F)}{\mathrm{OPT}(D,k)} \leq \frac{\mathrm{OTC}(D',k',F')}{\mathrm{OPT}(D',k')}$ which concludes the proof. □

Before applying following reduction rules, we partition $V(D)$ into two parts. Consider a set of vertices with out-degree zero, denoted by I. Note that there is no arc with both end points in I. Let $H = V(G) \setminus I$. See Figure 6.4. Formally,

$$I = \{v \in V(D) \mid d^+(v) = 0\}$$

$$H = V(D) \setminus I$$

We argue that for an instance (D,k), if D is k-contractible to an out-tree T and none of the reduction rules mentioned above are applicable, then the number of vertices in D is bounded by some function of k. If Reduction Rule 6.5.1 is not applicable then for every

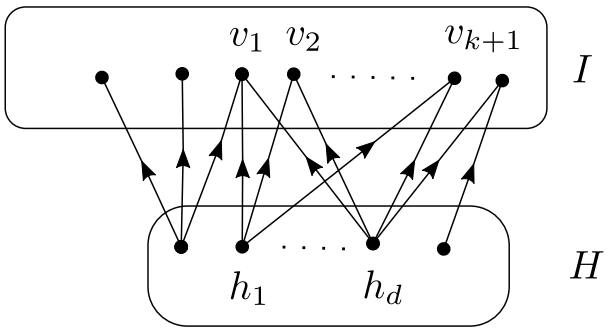


Figure 6.4: Partition of Digraph D. See Reduction Rule 6.5.4.

leaf t in T, either t or its neighbor corresponds to big witness set. If Reduction Rule 6.5.2 is not applicable then for every path of length at least $k+1$ in T there exists a vertex in this path which corresponds to a big witness set.

Lemma 6.5.3. *Let (D,k) be an instance of* OUT-TREE CONTRACTION *where Reduction Rules 6.5.1 and 6.5.2 are not applicable. If D is k-contractible to an out-tree then the number of vertices in H is at most $\mathcal{O}(k^2)$.*

Proof. We first sketch proof of the Lemma. Suppose D is contractible to an out-tree T. Let L denote the set of leaves in T. Define L_S as set leaves corresponding to singleton witness sets. By construction, every vertex in H has out-degree at least one and hence no vertex in H is a singleton witness set corresponding to a leaf in L_S. All vertices in H (and few from I) are contained in witness sets $W(t)$ such that t is in $V(T) \setminus L_S$. We bound number of vertices in H by upper bounding the cardinality of $V(T) \setminus L_s$. Consider the tree T' obtained from T by removing L_s. All leaves in T' corresponds to big witness sets. There are at most k many big-witness sets. This implies upper bound on number of leaves in T' and subsequently on vertices of degree at least 3 in T'. Every vertex t of degree two in T' is contained in path between t_i, t_j such that degrees of t_i, t_j are one or at least three. We argue that these number of such paths paths and length of each path is not large. This implies upper bound on vertices of degree two in T'. We use these upper bounds on $V(T')$ to bound the vertices in H.

We partition vertices of an out-tree T in three parts depending on their degrees. Vertices with degree one, two and at least three are contained in $V_1(T), V_2(T)$ and $V_3(T)$, respectively. Formally, $V_1(T) = \{t \in V(T) \mid d(t) = 1\}$, $V_2(T) = \{t \in V(T) \mid d(t) = 2\}$ and $V_3(T) = \{t \in V(T) \mid d(t) \geq 3\}$. Suppose D is contractible to an out-tree T. Let $\mathscr{W}$ be a T-witness structure. If a vertex t is L and t' is its neighbour then either $|W(t)| > 1$ or $|W(t')| > 1$ as otherwise Reduction Rule 6.5.1 would have been applied. Let L_s denote a set of leaves of T that correspond to singleton witness sets in $\mathscr{W}$. Formally, $L_S = \{t \in L \mid |W(t)| = 1\}$. By definition, out-degree of every vertex $v \in H$ is at least one. Since for any $t \in L_s$, $|W(t)| = 1$ and out-degree of t is zero, no vertex in H intersects $W(t)$. Consider a tree T' obtained from T by deleting all vetices in L_s. Let H' be set of vertices in $V(D)$ which are containd in $W(t)$ for some t in $V(T')$. In other words, $H' = \{v \in V(D) \mid \exists t \in V(T'), v \in W(t)\}$. It follows that $H \subseteq H'$. We bound cardinality of set H by bounding cardinality of set H'. Recall that there are at most k big witness set in T-witness structure of G. As $|W(t)| > 1$ for every leaf t in T', and thus $|V_1(T')| \leq k$. As the number of vertices of at least three in a tree is upper bounded by the number of leaves, we have $|V_3(T')| \leq k$.

Let $V_2 = \{t \in V_2(T') \mid |W(t)| = 1\}$. Every vertex in $V_2(T') \setminus V_2$ corresponds with a big witness set and hence $|V_2(T') \setminus V_2| \leq k$. Let $U = V_1(T') \cup V_3(T') \cup (V_2(T') \setminus V_2)$. The number of vertices in H' which are contained in a witness set $W(t)$ such that t is U is at most $\mathscr{O}(k)$. We now bound cardinality of V_2. Every vertex $t \in V_2$ is either the root or an internal vertex of a path between two vertices in $V_1(T') \cup V_3(T') \cup (V_2(T') \setminus V_2)$. The number of such paths is at most $\mathscr{O}(k)$. The interval vertices of these paths have degree two in digraph D. As the Reduction Rule 6.5.2 is not applicable, lengths of each paths are at most $k+2$ which implies that $|V_2|$ is $\mathscr{O}(k^2)$. Summarizing these bounds, the number of vertices in H' and hence in H is upper bounded by $\mathscr{O}(k^2)$. □

Using Lemma 6.5.3, we can identify digraphs which are not k-contractible to an out-tree.

Reduction Rule 6.5.3. *Given an instance (D,k), partition $V(D)$ into (I,H) such that*

$I = \{v \in V(D) \mid d^+(v) = 0\}$ *and* $H = V(D) \setminus I$. *If size of* H *is greater than* $\mathcal{O}(k^2)$ *then return the trivial instance* $(C_4, 1)$.

Lemma 6.5.4. *Reduction Rule 6.5.3 is* 1*-safe.*

Proof. Let (D,k) be an instance such that Reduction Rule 6.5.3 returns $(C_4,1)$ when applied on it. Solution lifting algorithm returns a spanning tree F of D.

Note that for a set of edges F', if C_4/F' is an out-tree then F' contains at least two edges. This implies $\mathrm{OTC}(C_4,1,F') = 2$ and $\mathrm{OPT}(C_4,1) = 2$.

Lemma 6.5.3, if D is k-contractible to an out-tree than size of H is at most $\mathcal{O}(k^2)$. Hence for any set of edges F^* if D/F^* is an out-tree than size of F^* is at least $k+1$. This implies $\mathrm{OPT}(D,k) = k+1$. For a spanning tree F of D, $\mathrm{OTC}(D,k,F) = k+1$.

Combining these values, we get $\frac{\mathrm{OTC}(D,k,F)}{\mathrm{OPT}(D,k)} = \frac{k+1}{k+1} = \frac{2}{2} = \frac{\mathrm{OTC}(C_4,1,F')}{\mathrm{OPT}(C_4,1)}$. This implies if F' is c-factor approximate solution for $(C_4,1)$ then F is 1-factor approximate solution for (D,k). This concludes the proof. □

We argue that vertex v and its $k+1$ false twins in I *forces* some vertices to be in one witness set. By applying Reduction Rule 6.5.4, we ensure that we store just enough vertices which enforces such condition. Following reduction rule states that we can delete all but $k+1$ vertices in D which has identical neighborhood.

Reduction Rule 6.5.4. *If there are vertices* $v, v_1, v_2, \ldots, v_{k+1} \in I$ *such that* $N^-(v) = N^-(v_1) = \cdots = N^-(v_{k+1})$*, then delete* v*. The resulting instance is* (D',k') *where* $D' = D - \{v\}$ *and* $k' = k$.

This reduction rule can be applied in polynomial time.

Lemma 6.5.5. *Reduction Rule 6.5.4 is* 1*-safe.*

Proof. Consider a set $F' \subseteq A(D')$ such that $T' = D'/F'$ is an out-tree. If $|F'| \geq k'+1$, then the solution lifting algorithm returns a spanning tree F of D, otherwise it returns $F = F'$.

If $|F'| \geq k'+1$ then $\mathrm{OTC}(D,k,F) \leq k+1 = \mathrm{OTC}(D',k',F')$. Otherwise, let $\mathscr{W}'$ denote a T'-witness structure of D'. As $|F'| \leq k$ and an arc in F' can be incident on at most one vertex in I, there exists a vertex v_i which is a singleton witness set in $\mathscr{W}'$ for some i in $[k+1]$. Let $t_i \in V(T)$ be a vertex in T such that $W(t_i) = \{v_i\}$. As v_j has no out-neighbor in D', t_i is a leaf in T. If t_j is the unique neighbor of t_i in T then $N^-(v_i) \subseteq W(t_j)$. Let $\mathscr{W}$ be a witness structure of D obtained from $\mathscr{W}$ by adding a singlton witness set $\{v\}$. If $\mathscr{W}$ is a T-witness structure of D then T can be obtained from T' by adding an out-neighbor t_v to t_i. Since T' is an out-tree, T is also an out-tree. Hence, $\mathrm{OTC}(D,k,F) \leq \mathrm{OTC}(D',k',F')$.

Consider an optimal solution F^* of (D,k). If $|F^*| \geq k+1$ then $\mathrm{OPT}(D,k) = k+1$ and by definition, $\mathrm{OPT}(D',k') \leq k'+1 = k+1 = \mathrm{OPT}(D,k)$. Otherwise, let $T = G/F^*$ and let $\mathscr{W}^*$ be a T-witness structure of D. There exists a vertex $t \in V(T)$ such that $v \in W(t)$. If t is a leaf and $W(t)$ is a singleton set, then F^* is also a solution to (D',k') where D'/F^* is an out-tree obtained from D/F^* by deleting one of its leaf. This implies $\mathrm{OPT}(D',k') \leq \mathrm{OPT}(D,k)$. Otherwise, as there are at least $k+1$ vertices with the same neighborhood as v, there exists a vertex v_i which is a singlton witness set in $\mathscr{W}$ for some i in $[k+1]$. Let t' be a vertex in $V(T)$ such that $W(t') = \{v_i\}$. As v_i has no out-neighbors, t' is a leaf. Let $\mathscr{W}'$ is a witness structure of D obtained from $\mathscr{W}^*$ by swapping v_i and v. This defines a set of arcs F' obtained from F by replacing the arc xv with the arc xv_i for each x. Since v_i and v have identical open neighborhood, if $\mathscr{W}'$ is a T'-witness structure of D then $T' = T$. Since F' is an optimum solution for (D,k) and there exists a leaf in D/F' which is singleton witness set containing v, it is a solution for (D',k). Therefore, $\mathrm{OPT}(D',k) \leq \mathrm{OPT}(D,k)$.

Putting two inequalities together, we get $\frac{\mathrm{OTC}(D,k,F)}{\mathrm{OPT}(D,k)} \leq \frac{\mathrm{OTC}(D',k',F')}{\mathrm{OPT}(D',k')}$ which concludes the proof. □

We describe the final reduction rule. Given $\alpha > 1$, let d be the minimum integer such that $\frac{d}{d-1} \leq \alpha$. In other words, fix $d = \lceil \frac{\alpha}{\alpha-1} \rceil$. Following reduction rules state that we can contract d vertices in H if all of them sees $k+1$ vertices in I.

Reduction Rule 6.5.5. *If there are vertices $v_1, v_2, \ldots, v_{k+1} \in I$ and $h_1, h_2, \ldots, h_d \in H$ such that $\{h_1, \ldots, h_d\} \subseteq N^-(v_i)$ for each $i \in [k+1]$, then contract arcs in $\tilde{A} = \{(h_i v_1) \mid i \in [d]\}$ and reduce the parameter by $d-1$. The resulting instance is (D', k') where $D = D/\tilde{A}$ and $k' = k-(d-1)$).*

This reduction rule can be applied in time $|H|^d \cdot n^{\mathcal{O}(1)}$.

Lemma 6.5.6. *Reduction Rule 6.5.5 is α-safe.*

Proof. Let w denote the vertex in $V(D') \setminus V(D)$ obtained by contracting $\tilde{A}$ in D. Consider a solution F' to the reduced instance (D', k'). If $|F'| \geq k'+1$, then the solution lifting algorithm returns a spanning tree F of D, otherwise it returns $F = F' \cup \tilde{A}$. If $|F'| \geq k'+1$ then $\mathrm{OTC}(D', k', F') = k'+1 = k-d$. As F is a spanning tree, $\mathrm{OTC}(D, k, F) = k+1 = k'+d = \mathrm{OTC}(D', k', F') + d - 1$. Consider the case when $|F'| \leq k'$ and let $\mathscr{W}'$ be a D'/F'-witness structure of digraph D'. Let $W'(t_1)$ be a witness set in $\mathscr{W}$ such that $w \in W'(t_1)$. Define $\mathscr{W} = (\mathscr{W}' \cup \{W_1\}) \setminus \{W'(t_1)\}$ where $W_1 = (W'(t_1) \cup \{v_1, h_1, h_2, \ldots, h_d\}) \setminus \{w\}$. Note that $V(D) \setminus \{v_1, h_1, h_2, \ldots, h_d\} = V(D') \setminus \{w\}$ and hence $\mathscr{W}$ is a partition of $V(D)$. Further, $G_D[W_1]$ is connected as $G'_D[W'(t_1)]$ is connected. A spanning tree of the latter along with edges $\{v_1 h_i \mid \forall i \in [d]\}$ is a spanning tree of the former. Also, $|W_1| = |W'(t_1)| + d$ and any vertex which is adjacent to w in D' is adjacent to at least one vertex in $\{v_1, h_1, h_2, \ldots, h_d\}$ in D. Thus, $\mathscr{W}$ is a D/F-witness structure of D where D/F is an out-tree. In this case, $\mathrm{OTC}(D, k, F) \leq \mathrm{OTC}(D', k', F') + d$. Therefore, we know $\mathrm{OTC}(D, k, F) \leq \mathrm{OTC}(D', k', F') + d$.

Let F^* be an optimum solution for (D, k) and $T = D/F^*$. Let $\mathscr{W}$ be a T-witness structure of D. If $|F^*| \geq k+1$, then $\mathrm{OPT}(D, k) = k+1 = k'+d \geq \mathrm{OPT}(D', k') + d - 1$. If $|F^*| \leq k$ then there is at least one vertex, say v_q in $\{v_1, v_2, \ldots, v_{k+1}\}$ which is not in $V(F^*)$. Consider the vertex $t_i \in V(D/F^*)$ such that $W(t_i) = \{v_q\}$. Vertex t_i is a leaf as v_q has no out-neighbors and there is no other vertex in $W(t_i)$. This implies that there exists a witness set, say $W(t_j)$, that contains all vertices in $N(v_q)$. Hence $\{h_1, h_2, \ldots, h_d\}$ are contained

in $W(t_j)$. We consider two cases based on whether v_1 is in $W(t_j)$ or not. Suppose $v_1 \in W(t_j)$. Let $\tilde{A} = \{v_1 h_i | \forall\, i \in [d]\}$. Then, $F' = F^* \setminus \tilde{A}$ is a solution to (D', k') and hence $\mathrm{OPT}(D', k') \leq |F'| = |F^*| - d = \mathrm{OPT}(D, k) - d$. Otherwise, $v_1 \notin W(t_j)$ and then there exists a vertex $t_1 \in V(T)$ adjacent to t_j such that $v_1 \in W(t_1)$. Define another partition $\mathscr{W}' = (\mathscr{W} \cup \{W(t_{j1})\}) \setminus \{W(t_j), W(t_1)\}$ of $V(D)$ where $W(t_{j1}) = W(t_j) \cup W(t_1)$. Graph $G_D[W(t_{j1})]$ is connected as both $G_D[W(t_j)]$ and $G_D[W(t_1)]$ are connected and there is an edge with one end point in $W(t_j)$ and another in $W(t_1)$. Thus, $\mathscr{W}'$ is a D/F-witness structure of D where $|F| = |F^*| + 1$ as $|W(t_i)| - 1 + |W(t_j)| - 1 = (|W(t_{ij})| - 1) - 1$. In other words, we needed one more edge than $|F^*|$ to contract all witness sets in $\mathscr{W}'$. Further F can be assumed to contain $\tilde{A}$ and $F' = F \setminus \tilde{A}$ is solution to (D', k') leading to $\mathrm{OPT}(D', k') \leq |F'| = |F^*| + 1 - d = \mathrm{OPT}(D, k) - d + 1$.

Combining these bounds, we have $\frac{\mathrm{OTC}(D,k,F)}{\mathrm{OPT}(D,k)} \leq \frac{\mathrm{OTC}(D',k',F')+d}{\mathrm{OPT}(D',k')+(d-1)} \leq \max\left\{\frac{\mathrm{OTC}(D',k',F')}{\mathrm{OPT}(D',k')}, \alpha\right\}$. This concludes the proof. □

We now argue that if digraph D is k-contractible to an out-tree and none of reduction rule mentioned so far is applicable on (D, k) then the number of vertices in D is bounded.

Lemma 6.5.7. *Let (D, k) be an instance of* Out-Tree Contraction *where none of Reduction Rules 6.5.1; 6.5.2; 6.5.3; 6.5.4; and 6.5.5 are applicable. If D is k-contractible to an out-tree then the number of vertices in H is at most $\mathcal{O}(k^{2d+1} + k^2)$.*

Proof. Recall the partition I, H of $V(D)$ defined before stating Reduction Rule 6.5.3. Set I is a collection of vertices which has no in-degree and H is set of remaining vertices in D. If Reduction Rule 6.5.3 return a trivial instance then statement is vacuously true. Otherwise, cardinality of H is at most $\mathcal{O}(k^2)$. Using this upper bound and the fact that Reduction Rules 6.5.4 and 6.5.5 are not applicable, we bound vertices in I.

For every set $H'' \subseteq H$ of cardinality less than d, there are at most $k+1$ vertices in I which have H'' as their neighborhood. Otherwise, Reduction Rule 6.5.4 would have been applicable. Hence, there are at most $(k+1) \cdot \binom{2k}{d-1}$ vertices in I which have degree less

than d. Every vertex in I of degree at least d is adjacent to all vertices in at least one d-sized subset of H. For such a subset H'' of H, there are at most $k+1$ vertices in I which contain H'' in their neighborhood. Otherwise, Reduction Rule 6.5.5 would have been applied. Thus, there $\mathcal{O}((k+1)\binom{k^2}{d})$ vertices in I of degree at least d. Hence, $|I|$ is upper bounded by $\mathcal{O}(k^{2d+1})$. This concludes the proof. □

We now present main result in this session.

Theorem 6.5.1. OUT-TREE CONTRACTION *admits a PSAKS with* $\mathcal{O}(k^{2\lceil\frac{\alpha}{\alpha-1}\rceil+1}+k^2)$ *vertices.*

Proof. For given $\alpha > 1$, kernelization algorithm fix $d = \lceil\frac{\alpha}{\alpha-1}\rceil$ and apply Reduction Rules 6.5.1; 6.5.2; 6.5.3; 6.5.4; and 6.5.5 exhaustively on given instance (D,k). Kernelization algorithm applies least indexed applicable reduction rule. If Reduction Rule 6.5.3 returns a trivial instance then statement is true. Otherwise there exists a partition (I,H) of $V(D)$ such that I is set of all vertices with in-degree zero; $H = V(D) \setminus I$ and number of vertices in H is at most $\mathcal{O}(k^2)$. Let (D,k) be a reduced instance after applying all reduction rules exhaustively. If the number of vertices in D is at most $\mathcal{O}(k^{2d+1}+k^2)$ by Lemma 6.5.7, $\mathrm{OPT}(D,k)$ is at least $k+1$ and the algorithm outputs $(C_4,1)$ as a reduced instance. Note that Reduction Rule 6.5.5 is applied only when the number of vertices in digraph as more than $\mathcal{O}(k^{2k+1}+k^2)$ and hence it can be applied in $n^{\mathcal{O}(1)}$ time. Hence all reduction rules can be applied in time $n^{\mathcal{O}(1)}$. The correctness of this algorithm follows from Lemma 6.5.1; Lemma 6.5.2; Lemma 6.5.4; Lemma 6.5.5; and Lemma 6.5.6. □

6.6 Conclusion

In this chapter, we study contraction problem on directed graphs. We consider OUT-TREE CONTRACTION problem. We show that this problem does not have a polynomial kernel when parameterized by solution size. To complement this negative result, we present

a polynomial kernel when parameterized by solution size and the number of leaves in resultant out-tree. We also present a lossy kernel of polynomial size when parameterized by solution size. We note that treatment of OUT-TREE CONTRACTION is different than that of TREE CONTRACTION. Many simplifying assumptions which work in undirected settings do not work when input graph is directed.

Chapter 7

Clique Contraction

7.1 Introduction

In this chapter, we study a problem of contraction to cliques. A graph is called *complete* if any two vertices are adjacent with each other. A set of vertices is said to be clique if it induces a complete graph.

CLIQUE CONTRACTION	**Parameter:** k
Input: A graph G and an integer k	
Question: Is it possible to obtain a clique from G with at most k edge contractions?	

Cai and Guo presented an FPT algorithm running in time $2^{\mathcal{O}(k\log k)} \cdot n^{\mathcal{O}(1)}$ [16]. Independent works of Lokshtanov et al. [73] and Cai and Guo [16] implies that there is no polynomial kernel for this problem when parameterized by k. In full version of the paper Lokshtanov et al. explicitly mentioned kernel with $\mathcal{O}(4^k k)$ vertices. In this chapter, we design an α-lossy kernel of polynomial size for this problem. We prove that given a graph G on n vertices, an integer k and an approximation parameter $\alpha > 1$, there is an algorithm that runs in $n^{\mathcal{O}(1)}$ time and outputs a graph G' on $\mathcal{O}(k^{d+1})$ vertices and an integer k' such that for every $c > 1$, a c-approximate solution for (G', k') can be turned into a

$(c\alpha)$-approximate solution for (G,k) in $n^{\mathcal{O}(1)}$. Here $d = \lceil \sqrt{\alpha}/(\sqrt{\alpha}-1) \rceil$.

We mention parametric dual of this problem which is known as HADWIGER NUMBER and has enjoyed more attention in literature.

HADWIGER NUMBER	**Parameter:** ℓ
Input: A graph G and an integer ℓ	
Question: Is it possible to obtain a clique with ℓ vertices from G by contracting edges?	

Hadwiger's conjecture states that chromatic number of graph is at least its Hadwiger number. This conjecture makes determining Hadwiger number is well studied problem in graph theory. Surprisingly, complexity of this problem, and hence of CLIQUE CONTRACTION, was not resolved until 2009 when Eppisten proved that this problem is NP-Complete [37]. We say graph H is a *minor* of graph G if H can be obtained from G by series of vertex deletions, edge deletions and edge contractions. If clique K_ℓ is minor of connected graph G then we can obtain K_ℓ from G without deleting any edge. Similarly, instead of deleting a vertex, we can contract it to one of its neighbors. This implies that one can obtain clique K_ℓ from graph G by series of edge contraction if and only if K_ℓ is a minor of graph G. By deep result of Robertson and Seymour [88], there exists an algorithm running in time $f(\ell) \cdot n^{\mathcal{O}(1)}$ to check whether K_ℓ is a minor of G. This implies that there exist an FPT algorithm for HADWIGER NUMBER. This relation was first noted by Alon et al. [4]. We mention, without proof, that there exists a simple *cross composition* of unparameterized version of HADWIGER NUMBER into its parameterized version. This cross composition, along with results in [43], state that HADWIGER NUMBER does not have a polynomial kernel unless NP $\subseteq$ coNP/poly. See [25, Theorem 15] for quick reference.

We study generalization of CLIQUE CONTRACTION problem called s-CLUB CONTRACTION. A graph is called s-club if the diameter of graph is at most s. We formally define the problem as follows.

s-CLUB CONTRACTION	**Parameter:** k
Input: A graph G and integers k, s	
Question: Is it possible to obtain a graph of diameter at most s from G with at most k edge contractions?	

Clearly, 1-CLUB CONTRACTION problem is identical to that of CLIQUE CONTRACTION. We prove that s-CLUB CONTRACTION does not admit a lossy kernel of polynomial size unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$ even when $s = 2$ and input is restricted to a split graph.

7.2 Preliminaries

We start with following observation which is useful to find large induced clique in input graph. Note that we can assume that input graph is connected as otherwise we know it can not be converted into a clique by edge contraction only.

Observation 7.2.1. *If graph G is k-contractible to a clique then G can be converted into a clique by deleting at most $2k$ vertices.*

Proof. Let F be a set of edges of size at most k such that G/F is a clique. Let $\mathscr{W}$ be a G/F-witness structure of G. Let X be set of all vertices which are contained in big witness sets in $\mathscr{W}$. Since every vertex in X is incident on some edge in F, size of X is at most $2k$. Any any two vertices in $V(G) \setminus X$, are adjacent with each other as these vertices form a singleton sets which are adjacent in G/F. Hence G can be converted into a clique by deleting vertices in X. □

For a given graph G, its compliment graph, denoted by $\bar{G}$, is defined on same set of vertices and edge uv is present in $\bar{G}$ if vertices u, v are not adjacent in graph G.

Observation 7.2.2. *There exists a 2-factor approximation algorithm to compute set of vertices whose deletion results in a clique.*

Proof. For a given graph G, consider its complement graph $\bar{G}$. For a set of vertices X in G, $G-X$ is a complete graph if and only if X is a vertex cover of $\bar{G}$. As there exists a 2-factor approximation algorithm to compute a vertex cover of given graph, we get a 2-factor approximation algorithm to compute set of vertices whose deletion results in clique. □

Consider a connected graph G which is k-contractible to clique K_ℓ. Let $\mathscr{W}$ be a K_ℓ-witness structure of G. Following observation proves that given a graph with its witness structure, we can merge two witness set and delete a vertex to obtain a witness structure for smaller graph.

Observation 7.2.3. *Let $\mathscr{W}$ be a clique witness structure of a graph G. If there exists two different witness sets $W(t_1), W(t_2)$ in $\mathscr{W}$ and a vertex v in $W(t_1)$ such that set $W(t) = (W(t_1) \cup W(t_2)) \setminus \{v\}$ is a connected set in $G-\{v\}$ then $\mathscr{W}'$ is a clique witness structure of $G-\{v\}$ where $\mathscr{W}'$ is obtained from $\mathscr{W}$ be removing $W(t_1), W(t_2)$ and adding $W(t)$.*

Proof. Let $G' = G-\{v\}$. Note that $\mathscr{W}'$ is a partition of vertices in G'. Any set in $\mathscr{W}' \setminus \{W(t)\}$ is a witness set in $\mathscr{W}$ and does not contain v. Hence it is a connected set in G'. Since $G'[W(t)]$ is connected, all sets in $\mathscr{W}'$ are connected in G'.

Consider any two witness sets $W(t'), W(t'')$ in $\mathscr{W}'$. If none of these two is equal to $W(t)$ then both of these sets are present in $\mathscr{W}$. Since none of these witness sets contains vertex v, they are adjacent with each other in G'. Now, consider a case when one of them, say $W(t'')$, is equal to $W(t)$. As witness sets $W(t')$ and $W(t_2)$ are present in $\mathscr{W}$, there exists an edge with one end point in $W(t')$ and another in $W(t_2)$. The same edge is present in graph G' as it is not incident on v. Since $W(t_2)$ is subset of $W(t)$, sets $W(t')$ and $W(t)$ are adjacent in G'. Hence any two witness sets in $\mathscr{W}'$ are adjacent with each other. This implies that $\mathscr{W}'$ is a clique witness structure of graph $G-\{v\}$. □

7.3 Lossy Kernel for CLIQUE CONTRACTION

In this section, we present a lossy kernel for CLIQUE CONTRACTION. As mentioned earlier, as assume that input graph G is connected. We define optimization problem in the following way.

$$\text{CLC}(G,k,F) = \begin{cases} \infty & \text{if G/F is not a clique} \\ \min\{|F|, k+1\} & \text{otherwise} \end{cases}$$

If number of vertices in input graph is at most $k+3$ then we can return same instance as kernel for given problem. We only consider inputs which has at least $k+3$ vertices. By the definition of optimization problem, for any set of edges F if G/F is a clique then the maximum value of $\text{CLC}(G,k,F)$ is $k+1$. Hence any spanning tree of G is a solution of cost $k+1$. We call it a *trivial solution* for given instance. Consider an instance $(P_4,1)$ where P_4 is a path on four vertices. One need to contract at least two edges to convert P_4 into a clique. We say $(P_4,1)$ as *trivial instance* for this problem.

We start with first reduction rule which says if the minimum number of vertices that needs to be deleted from input graph is large then we can return trivial instance as lossy kernel.

Reduction Rule 7.3.1. *Given an instance (G,k), apply the algorithm mentioned in Observation 7.2.2 to find set X such that such that $G-X$ is a complete graph. If size of X is greater than $4k$ then return $(P_4,1)$.*

Lemma 7.3.1. *Reduction Rule 7.3.1 is 1-safe.*

Proof. Let (G,k) be an instance such that Reduction Rule 7.3.1 returns $(P_4,1)$ when applied on it. Solution lifting algorithm returns a spanning tree F of G.

Note that for a set of edges F', if P_4/F' is a clique then F' contains at least two edges. This implies $\text{CLC}(P_4,1,F') = 2$ and $\text{OPT}(P_4,1) = 2$.

Since a 2-factor approximation algorithm returns a set of size strictly more than $4k$, for any set X' of size at most $2k$, $G-X'$ is not a complete graph. But by Observation 7.2.1, if G is k-contractible to a clique then G can be converted into a clique by deleting at most $2k$ vertices. Hence for any set of edges F^* if G/F^* is a clique than size of F^* is at least $k+1$. This implies $\mathrm{OPT}(G,k) = k+1$. For a spanning tree F of G, $\textsc{Clc}(G,k,F) = k+1$.

Combining these values, we get $\frac{\textsc{Clc}(G,k,F)}{\mathrm{OPT}(G,k)} = \frac{k+1}{k+1} = \frac{2}{2} = \frac{\textsc{Clc}(P_4,1,F')}{\mathrm{OPT}(P_4,1)}$. This implies if F' is c-factor approximate solution for $(P_4,1)$ then F is 1-factor approximate solution for (G,k). This concludes the proof. □

For a given graph G, let (X,Y) be a partition of $V(G)$ such that $G-X = G[Y]$ is a complete graph. For $\alpha > 1$, let $\beta^2 = \alpha$. Find a smallest integer d such that $\frac{d+1}{d} \leq \beta$. Or in other words, fix $d = \lceil \frac{\beta}{\beta-1} \rceil$. Given an instance (G,k) and partition (X,Y) of G, we deploy following two marking schemes.

Marking Scheme 7.3.1. *For a subset A of X, let $M_1(A)$ be a collection of vertices in Y whose neighborhood contains A. For every subset A of X which is of size at most d, mark a vertex in $M_1(A)$.*

Formally, $M_1(A) = \{y |\ y \in Y$ such that $A \subseteq N(y)\}$. If $M_1(A)$ is empty then marking scheme does not mark any vertex. If it is not empty then marking scheme arbitrary chooses a vertex and marks it.

Marking Scheme 7.3.2. *For a subset A of X, let $M_2(A)$ be a collection of vertices in Y whose neighborhood does not intersect A. For every subset A of X which is of size at most d, mark $2k+1$ vertex in $M_2(A)$.*

Formally, $M_2(A) = \{y |\ y \in Y$ such that $N(y) \cap A = \emptyset\}$. If the number of vertices in $M_2(A)$ is at most $2k+1$ then marking scheme marks all vertices in $M_2(A)$. If it is larger than $2k+1$ then it arbitrary chooses $2k+1$ vertices and marks them.

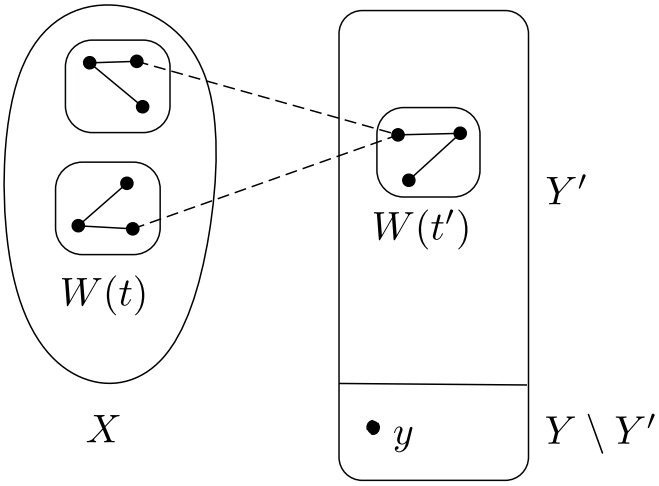


Figure 7.1: Straight lines (Ex. within $W(t)$) represent edges in original solution F. Dashed lines (Ex. across $W(t)$ and $W(t')$) represents extra edges added to solution F. Please refer to Lemma 7.3.2.

Reduction Rule 7.3.2. *For a given instance (G,k), and partition (X,Y) of $V(G)$, applying Marking Scheme 7.3.1 and 7.3.2. Let G' be the graph obtained from G by deleting unmarked vertices in Y. Return instance (G',k).*

Above reduction rule can be applied in time $|X|^d \cdot n^{\mathcal{O}(1)}$. Let Y' be a subset of Y which has been marked in Marking Scheme 7.3.1 or 7.3.2. Note that G' is an induced subgraph of G. In the following lemma, we argue that given a solution for (G',k) we can construct a solution of almost the same size for (G,k).

Lemma 7.3.2. *Let (G',k) be the instance returned by Reduction Rule 7.3.2 when applied on an instance (G,k). If there exists a set of edges of size at most k, say F', such that G'/F' is a clique then there exists a set of edges F such that G/F is a clique and cardinality of F is at most $\beta \cdot |F'|$.*

Proof. If no vertex in Y has been deleted than G' and G are identical graphs and statement is true. We assume that at least one vertex in Y has been deleted. Let Y' be a set of vertices in Y which has been marked. Sets X, Y' forms a partition of $V(G')$ such that $G' - X$ is a complete graph and Y' is a proper subset of Y. Let $\mathscr{W}'$ be a G'/F'-witness structure of

G'. We construct a clique witness structure $\mathscr{W}$ of G from $\mathscr{W}'$ by adding singleton witness set $\{y\}$ for every vertex y in $Y \setminus Y'$. Since $G[Y \setminus Y']$ is a clique in G, any two newly added witness sets are adjacent with each other. Moreover, any witness set in $\mathscr{W}'$ which intersects Y' is adjacent with newly added witness sets. We now consider witness sets in $\mathscr{W}'$ which do not intersect Y'.

Let $\mathscr{W}^*$ be a collection of witness set $W(t)$ in $\mathscr{W}'$ such that $W(t)$ is contained in X and there exists a vertex y in $Y \setminus Y'$ whose neighborhood does not intersects $W(t)$. See Figure 7.1. We argue that every witness set in $\mathscr{W}^*$ has at least $d+1$ vertices. For the sake of contradiction, assume that there exists a witness set $W(t)$ in $\mathscr{W}^*$ which contains at most d vertices. Since Marking Scheme 7.3.2 iterated over all sets of size at most d, it also considered $W(t)$ while marking. Note that vertex y belongs to set $M_2(W(t))$. Since y is unmarked, there are $2k+1$ vertices in $M_2(W(t))$ which has been marked. All these marked vertices are in G'. Since cardinality of F is at most k, the number of vertices in $V(F)$ is at most $2k$. This implies at least one marked vertex in $M_2(W(t))$ is a singleton witness set in $\mathscr{W}'$. But there is no edge between this singleton witness set and $W(t)$. This contradicts the fact that any two witness sets in $\mathscr{W}'$ are adjacent with each other in G'. Hence our assumption is wrong and $W(t)$ has at least $d+1$ many vertices.

Fix a witness set, say $W(t')$, in G'/F'-witness structure which intersects Y'. Because of Marking Scheme 7.3.1, we have at least one such witness set.

We note that $W(t')$ is adjacent with every vertex in $Y \setminus Y'$. Let $W(t)$ be a witness set in $\mathscr{W}^*$. Since $W(t')$ and $W(t)$ are two witness sets in G'/F'-witness structure, there exists an edge with one end point in $W(t')$ and another in $W(t)$. Set $W(t') \cup W(t)$ is adjacent with every other witness set in $\mathscr{W}$.

We now describe how to obtain F from F'. We initialize $F = F'$. For every witness set $W(t)$ in $\mathscr{W}^*$ add an edge between $W(t)$ and $W(t')$ to set F'. Equivalently, we construct a new witness set by taking a union of $W(t')$ and all witness sets $W(t)$ in $\mathscr{W}^*$. This witness set is adjacent with every vertex in $Y \setminus Y'$ and hence G/F is a clique. We now argue the

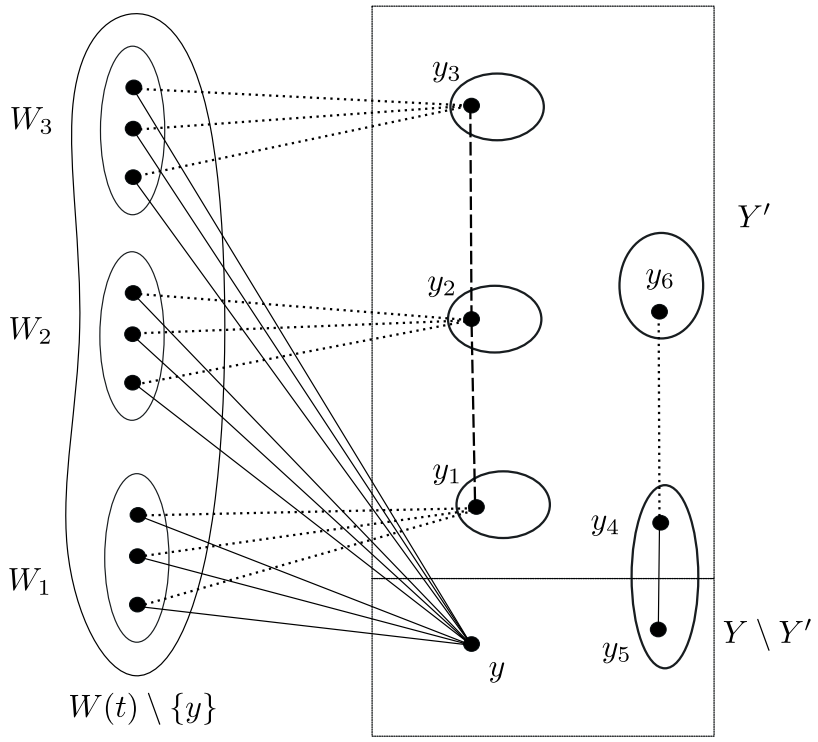


Figure 7.2: Straight lines (Ex. y_4y_5) represent edges in original solution F. Dotted lines (Ex. y_4y_6) represents edges which are replaced for some edges in F. Dashed lines (Ex. y_1y_2) represents extra edges added to solution F. Please refer to Lemma 7.3.3.

size bound on F. Note that we have added one extra edge for every witness set $W(t)$ in $\mathscr{W}^*$. We know that every such witness set has at least $d+1$ vertices. Hence we have added one extra edge for at least d edges in solution F. Moreover, since witness sets in $\mathscr{W}^*$ are vertex disjoint, no edge in F can be part of two witness sets. This implies number of edges in F is at most $\frac{d+1}{d}|F| \leq \beta|F|$. □

In the following lemma, we argue that value of optimum solution for reduced instance can be upper bounded by value of optimum solution for original instance.

Lemma 7.3.3. *Let (G',k) be the instance returned by Reduction Rule 7.3.2 when applied on an instance (G,k). If $\mathrm{OPT}(G,k) \leq k$ then $\mathrm{OPT}(G',k) \leq \beta \cdot \mathrm{OPT}(G,k)$.*

Proof. Let F be a set of at most k edges in G such that $\mathrm{OPT}(G,k) = \textsc{Clc}(G,k,F)$ and $\mathscr{W}$ be a G/F-witness structure of G. Since we are working with minimization problem,

to prove the lemma it is sufficient to find a solution for G' which is of size $\beta \cdot |F|$. Recall that (X,Y) is a partition of $V(G)$ such that $G-X = G[Y]$ is a complete graph. Set of vertices marked by either of marking schemes is denoted by Y'. In other words, (X,Y') is a partition of G' such that $G'-X = G'[Y]$ is a complete graph. We proceed as follows. At each step, we construct graph G^* from G by deleting one or more vertices in $Y \setminus Y'$. We also construct a set of edges F^* from F by replacing existing edges and/or adding extra edges to F. At any intermediate state, we ensure that G^*/F^* is a clique and the number of edges in F^* is at most $\beta \cdot |F|$. Let $F^\circ = F$ is an optimum solution for input instance (G,k).

To obtain G^* and F^*, we delete witness sets which are subsets of $Y \setminus Y'$ (Condition (1)) and modify the ones which intersect with $Y \setminus Y'$. Every witness set of later type intersects with Y' or X or both. We partition these big witness sets in $\mathscr{W}$ into two groups depending on whether they intersects X (Condition (2)) or not (Condition (3)). We modify witness sets which satisfy least indexed condition. If there exist no witness set which satisfy either of these three conditions then $Y \setminus Y'$ is an empty set and the lemma is vacuously true.

Condition (1): There exists a witness set $W(t)$ in $\mathscr{W}$ which is a subset of $Y \setminus Y'$.

Construct G^* from G by deleting witness sets $W(t)$ in $\mathscr{W}$. Let $F^* = F$. Since class of complete graphs is closed under vertex deletion, G^*/F^* is a clique. We repeat this process until there exists a witness set which satisfy Condition (1).

At this stage we rename G^* to G and F^* to F for notational convenience.

Condition (2): There exists a witness set $W(t)$ in $\mathscr{W}$ which contains vertices from $Y \setminus Y'$ but does not intersects X.

Note that $W(t)$ must intersects with Y'. See Figure 7.2. Let y_4 and y_5 be vertices in $W(t) \cap Y'$ and $W(t) \cap (Y \setminus Y')$. Let $W(t_1)$ be a witness set which intersects Y'. Let y_6 be a vertex in set $W(t_1) \cap Y'$. Consider witness sets $W(t), W(t_1)$ and vertex y_5 in $W(t)$ in graph G. These satisfies the premise of Observation 7.2.3. This implies $\mathscr{W}^*$ is a clique witness structure of $G - \{y_5\}$ where $\mathscr{W}^*$ is obtained from $\mathscr{W}$ be removing $W(t), W(t_1)$ and adding

$(W(t) \cup W(t_1)) \setminus \{y_5\}$. This corresponding to replacing an edge in F which was incident on y_5 with the one across $W(t)$ and $W(t_1)$. For example, in Figure 7.2, we replace edge y_4y_5 in set F with an edge y_4y_6 to obtain a solution for $G - \{y_5\}$. An edge in F has been replaced with another edge and one vertex in $Y \setminus Y'$ is deleted. The size of F^* is same as that of F and G^*/F^* is a clique. We repeat this process until there exist a witness set which satisfy Condition (2).

At this stage we rename G^* to G and F^* to F for notational convenience.

Condition (3): There exists a witness set $W(t)$ in $\mathscr{W}$ which contains vertices from $Y \setminus Y'$ and intersects X.

Let y be a vertex in $W(t) \cap (Y \setminus Y')$ and X_t be vertices in $W(t) \cap X$ which are adjacent with y via edges in F. We find a *substitute* for y in $Y \setminus Y'$. We consider two cases based on cardinality of X_t.

Condition $(3.a)$: There exists a witness set $W(t)$ which contains y from $Y \setminus Y'$ whose neighborhood via edges in F in $W(t) \cap X$ is of size at most d.

In this case, X_t have been considered by Marking Scheme 7.3.1. Since y is adjacent with every vertex in X_t, set $M_1(X_t)$ is not empty. As y is in $Y \setminus Y'$, and hence unmarked, there is another vertex, say y_1, in $M_1(X_1)$ which has been marked. Let $W(t_1)$ be the witness set containing y_1. Witness sets $W(t), W(t_1)$ and y in $W(t)$ satisfies the premise of Observation 7.2.3. This implies $\mathscr{W}^*$ is a clique witness structure of $G - \{y_5\}$ where $\mathscr{W}^*$ is obtained from $\mathscr{W}$ be removing $W(t), W(t_1)$ and adding $(W(t) \cup W(t_1)) \setminus \{y\}$. This corresponding to replacing edge xy in F by xy_1 for every x in X_t. A set of edges in F has been replaced with another set of edges of same size and a vertex in $Y \setminus Y'$ is deleted. The size of F^* is same as that of F and G^*/F^* is a clique. We repeat this process until there exists a witness set which satisfy Condition $(3.a)$.

At this stage we rename G^* to G and F^* to F for notational convenience.

Condition $(3.b)$: There exists a witness set $W(t)$ which contains y from $Y \setminus Y'$ whose

neighborhood via edges in F in $W(t) \cap X$ is of size at least $d+1$.

Since Marking Scheme 7.3.1 iterated over subset of X of cardinality at most d, it may not have marked any vertex in $M_1(X_t)$. In this case, we partition $W(t) \setminus \{y\}$ into sets $W_1, W_2, \ldots, W_p$ such that the number of vertices in W_i for i in $[p-1]$ is exactly d and the number of vertices in W_p is at most d. See Figure 7.2. Since y is adjacent with every vertex in X_t, and hence every vertex in W_i, set $M_1(W_i)$ is not empty for any W_i. Since y is in $Y \setminus Y'$ and hence unmarked, there is a vertex in $M_1(W_i)$, say y_i, different from y which has been marked for each i in $[p]$. We assume that all vertices in $\{y_1, y_2, \ldots, y_p\}$ are different to obtain the upper bound. We construct F^* from F by following operation: For every i in $[p-1]$, replace an edge xy in F by an edge xy_i and for every i in $[p-2]$ add an edge $y_i y_{i+1}$. We first argue about the cardinality of F^*. Note that we have added an extra edge corresponding to W_i for each i in $[p-1]$. These sets are of size d. We did not add an extra edge corresponding to W_p whose cardinality may be smaller than d. This implies that we have added an extra edge for d edges in F. Moreover, since W_is are pairwise disjoints, no edge in F can be part of two sets of edges corresponding to which new edge has been added. Hence size of F^* is at most $\frac{d+1}{d}|F| = \beta \cdot |F|$. We now argue that if G^* is obtained from G by deleting y then G^*/F^* is a clique. For every i in $[p]$, let $W(y_i)$ be the witness set containing y_i. Let Z be the union of $W(t) \setminus \{y\}$ and $W(y_i)$ for all i in $[p]$. Let $\mathscr{W}^*$ be a witness structure of G^* obtained from $\mathscr{W}$ by removing $W(t), W(y_1), \ldots, W(y_t)$ and adding Z. Since all other witness sets remains same and we only replaced or added edges incident on vertices in $Z \cup \{y\}$, union of all spanning trees of witness sets in $\mathscr{W}^*$ is contained in F^*. Any two witness sets in $\mathscr{W}^*$ which are part of $\mathscr{W}$ are adjacent with each other. As Z contains $W(y_1)$, any witness set in $\mathscr{W}^*$ which is not contained in Z is adjacent with Z. Hence any two witness sets in $\mathscr{W}^*$ are adjacent with each other. This implies that G^*/F^* is a clique. We repeat this process until there exists a witness set which satisfy Condition $(3.b)$. We argue that $|F^*| \leq \beta \cdot |F^\circ|$ even after repeating this process. Consider a witness set $W(t)$ in $\mathscr{W}$ which satisfy Condition $(3.b)$ and which has been replaced by set Z. If Z does not intersect $Y \setminus Y'$ then it does not satisfy any condition and hence never

been modified again. If it intersects $Y \setminus Y'$ then it also intersects Y' and hence satisfy Condition (2). This implies that any witness set in $\mathscr{W}$ is replaced by this process at most once. In other words, if an edge xy in F° which has been replaced with edge xy_i before adding extra edge y_iy_{i+1} for some i in $[p-1]$ then edge xy is never considered by the process again.

Any vertex in $Y \setminus Y'$ must be part of some some witness set in $\mathscr{W}$ and any witness set in $\mathscr{W}$ satisfies at least one of the conditions mentioned above. If there is no witness sets which satisfy any condition then $Y \setminus Y'$ is empty. This implies $G^* = G'$ and there exists a solution F^* of size $\beta \cdot |F^\circ|$. This concludes the proof of the lemma. □

We are now in position to prove following lemma.

Lemma 7.3.4. *Reduction Rule 7.3.2, along with a solution lifting algorithm, is an α-reduction rule.*

Proof. Let (G',k) be the instance returned by Reduction Rule 7.3.2 when applied on an instance (G,k). We present a solution lifting algorithm. For a solution F' for (G,k) if $\textsc{Clc}(G',k,F') = k+1$ then solution lifting algorithm returns a spanning tree F of G (a trivial solution) as solution for (G,k). In this case, $\textsc{Clc}(G,k,F) = \textsc{Clc}(G',k,F')$. If $\textsc{Clc}(G',k,F') \leq k$ then size of F' is at most k and G'/F' is a clique. Solution lifting algorithm uses Lemma 7.3.2 to construct a solution F for (G,k) such that cardinality of F is at most $\beta \cdot |F'|$. In this case, $\textsc{Clc}(G',k,F') \leq \beta \cdot \textsc{Clc}(G,k,F)$. Hence there exists a solution lifting algorithm which given a solution F' for (G',k') returns a solution F for (G,k) such that $\textsc{Clc}(G,k,F) \leq \beta \cdot \textsc{Clc}(G,k,F)$.

If $\mathrm{OPT}(G,k) \leq k$ then there exists a set of edges of cardinality at most k, say F^*, such that G/F^* is a clique. By Lemma 7.3.3 we know that $\mathrm{OPT}(G',k) \leq \beta \cdot \mathrm{OPT}(G,k)$. If $\mathrm{OPT}(G,k) = k+1$ then $\mathrm{OPT}(G',k) \leq k+1 = \mathrm{OPT}(G,k)$. Hence in either case, $\mathrm{OPT}(G',k) \leq \beta \cdot \mathrm{OPT}(G,k)$.

Combining two inequalities, we get $\frac{\text{CLC}(G,k,F)}{\beta \cdot \text{OPT}(G,k)} \leq \frac{\beta \cdot \text{CLC}(G',k,F')}{\text{OPT}(G',k)}$. This implies if F' is c-factor approximate solution for (G',k) then F is $(c \cdot \beta^2)$-factor approximate solution for (G,k). As $\alpha = \beta^2$, this concludes the proof. □

We present main result of the chapter.

Theorem 7.3.1. CLIQUE CONTRACTION *parameterized by size of solution k, admits a time efficient PSAKS with* $\mathcal{O}(k^{d+1})$ *vertices, where* $d = \lceil \frac{\sqrt{\alpha}}{\sqrt{\alpha}-1} \rceil$.

Proof. For a given instance (G,k), a kernelization algorithm applies Reduction Rule 7.3.1. If it returns a trivial instance than statement is vacuously true. If it does not return a trivial instance then the algorithm partition $V(G)$ in two sets (X,Y) such that $G-X = G[Y]$ is a complete graph and size of X is at most $4k$. The algorithm apply Reduction Rule 7.3.2 on instance (G,k) with partition (X,Y). The algorithm returns the reduced instance as α-lossy kernel for (G,k).

The correctness of the algorithm follows from Lemma 7.3.1, and Lemma 7.3.4 combined with the fact that Reduction Rule 7.3.2 is applied at most once. By Observation 7.2.2, Reduction Rule 7.3.1 can be applied in polynomial time. The size of instance returned by Reduction Rule 7.3.2 is at most $\mathcal{O}((4k)^d \cdot (2k+1)+4k) = \mathcal{O}(k^{d+1})$. Reduction Rule 7.3.2 can be applied in time $n^{\mathcal{O}(1)}$ if number of the vertices in (G,k) is more $\mathcal{O}(k^{d+1})$. □

7.4 (No) Lossy Kernel for s-CLUB CONTRACTION

In this section, we argue that there is no lossy kernel for s-CLUB CONTRACTION. We can safely assume that input graph G is connected. We define optimization problem in the following way.

$$s\text{-CLUBC}(G,k,F) = \begin{cases} \min\{|F|, k+1\} & \text{if diameter of G/F is at most } s \\ \infty & \text{otherwise} \end{cases}$$

To prove that s-CLUBC(G,k,F) does not have a lossy kernel of polynomial size, it is sufficient to prove that there is no α-appt from a problem for which similar kind of results are known. In [47], Golovach et al. presented a reduction from an instance of HITTING SET to an instance of s-CLUBC(G,k,F). They proved that for any $s \geq 2$, the s-CLUB CONTRACTION problem on chordal graphs is NP-Hard as well as W[2]-Hard when parameterized by k. Moreover, 2-CLUB CONTRACTION is NP-Hard and W[2]-Hard when parameterized by k even on split graphs. We use similar ideas to prove a 1-appt from SET COVER$/k$ to 2-CLUB CONTRACTION. We present following lemma for $s = 2$ and briefly mention how to generalize the lemma for any fixed s.

Lemma 7.4.1. *There exists an* 1-appt *from* SET COVER$/k$ *to* 2-CLUB CONTRACTION *even input graph is restricted to a split graph.*

Proof. To prove the lemma, present a reduction algorithm, say $R_{\mathscr{A}}$, which given an instance $((U,\mathscr{S}),k)$ of SC$/k$ outputs an instance (G,k') of s-CLUBC. We also present a solution lifting algorithm that takes as input an instance $((U,\mathscr{S}),k)$ of SC$/k$, the output instance $(G,k') = R_{\mathscr{A}}((U,\mathscr{S}),k)$ of s-CLUBC, and a solution F to the instance (G,k') and outputs a solution $\mathscr{F}$ to $((U,\mathscr{S}),k)$ such that SC$/k((U,\mathscr{S}),k,\mathscr{F}) = s$-CLUBC$(G,k,F)$.

We first present a reduction algorithm.

Reduction Algorithm : Given an instance $((U,\mathscr{S}),k)$ of the SET COVER problem with $U = \{u_1,\ldots,u_n\}$ and $\mathscr{S} = \{S_1,\ldots,S_m\}$, we create a split graph G as follows. Create a vertex s_i for each $S_i \in \mathscr{S}$. Let $V_{\mathscr{S}}$ be set of all vertices corresponding to some S in $\mathscr{S}$. Add edges in G to convert $V_{\mathscr{S}}$ into a clique. For every $u_j \in U$, we create $k+1$ vertices $u_j^1,\ldots,u_j^{k+1}$ that are made adjacent to vertex s_i if and only if $u_j \in S_i$. Add a vertex a and make it adjacent with every vertex in $V_{\mathscr{S}}$. Add $k+1$ vertices $b^1,b^2,\ldots,b^{k+1}$ and make them adjacent with a only. This completes the construction of G. See Figure 7.3.

Note that the vertex set of G can be partitioned into a clique $\mathscr{S} \cup \{x\}$ and an independent set $V(G) \setminus (V_{\mathscr{S}} \cup \{a\})$, so G is a split graph. Also observe that the diameter of G is 3.

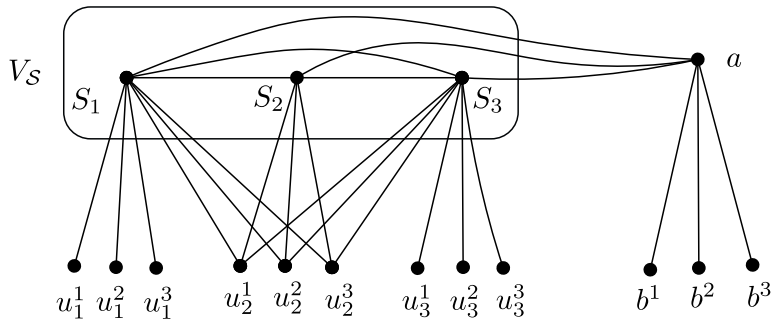


Figure 7.3: Reduction from a set cover instance $((U,\mathscr{S}),k)$ to an instance of s-CLUB CONTRACTION. Here $U=\{u_1,u_2,u_3\}$;$\mathscr{S}=\{S_1,S_2,S_3\}$ where $S_1=\{u_1,u_2\},S_2=\{u_1,u_2,u_3\},S_3=\{u_2,u_3\}$ and $k=2$.

Informally speaking, all the paths of length of 3 are between one of the vertices of type b^q and u_j^q for some $q\in\{1,\ldots,k\}$. To shorten all these paths with at most k edge contraction, one need to find a set cover of original instance of size k and vice versa.

Solution Lifting Algorithm : Let F a solution for (G,k). If $|F|\geq k+1$ then return $\mathscr{F}=\mathscr{S}$ as a solution. Otherwise, let $W(t_a)$ be the witness set in (G/F)-witness structure of G which contains a. Let $\mathscr{F}$ be the collection of set S_i in $\mathscr{S}$ for all s_i in $W(t_a)$. Return $\mathscr{F}$.

For any subset $\mathscr{F}$ of $\mathscr{S}$, let $F_{\mathscr{F}}$ be the set of edges in G which are incident on a and s_i for some s in $\mathscr{F}$.

Claim 1: If $\mathscr{F}$ is a set cover of instance $(U,\mathscr{S},k)$ then the diameter of $G/F_{\mathscr{F}}$ is at most 2.
Proof : Let T be the graph obtained from G by contracting all edges in $F_{\mathscr{F}}$. Let t_a be the vertex in T which corresponds to the unique big witness set. Note that $W(t_a)$ contains a and all vertices corresponding to sets in $\mathscr{F}$. The fact that $\mathscr{F}$ is a set cover implies that in T, vertex u_j^q is adjacent to some vertex in $W(t_a)$ for every $j\in\{1,\ldots,n\}$ and $q\in\{1,\ldots,k+1\}$. Hence t_a is a universal vertex in T, implying that T has diameter at most 2. $\diamond$

Let F be a set of edges in G such that G/F has diameter at most 2. Let $W(t_a)$ be the witness set in T-witness structure of G which contains a, where $T=G/F$. Let $\mathscr{F}_F$ be a collection of set S_i in $\mathscr{S}$ such that s_i is in $W(t_a)$.

Claim 2: If F has size at most k then $\mathscr{F}_F$ is a set cover of $((U,\mathscr{S}),k)$.

Proof : For any element u_j of universe, vertices $u_j^1,\ldots,u_j^{k+1}$ forms an independent set in G. Since size of F is at most k, edges in F can be incident on at most k vertices in this set. Without loss of generality, let u_j^1 be a vertex such that there is no solution edge is incident on it. By same arguments for set $b^1,\ldots,b^{k+1}$, we can assume that there is no edge incident on b^1. Consider a shortest path from u_j^1 to b^1 in graph G. Every such path is of the form (u_j^1,s_i,a,b^1) where S_i is a set which contains u_j. Since the diameter of T is at most 2, for every vertex u_j there exists a path from u_j to b^1 which has been shortened by an edge contraction. Since no edge in F is incident on u_1^j or on b^1, edge s_ia has been contracted for some S_i which contains u^j. Hence for every vertex u_j^1 there exists a vertex s_i in $W(t_a)$ which is adjacent to u_j^1. Since F has at most k edges, the number of vertices in $W(t_a)$ apart from a is at most k. Hence $\mathscr{F}_F$ has at most k sets and for every vertex in U there exists a set in $\mathscr{F}_F$ which contains that element. This implies that $\mathscr{F}_F$ is a set cover for $(U,\mathscr{S},k)$. $\diamond$

By Claim 1 and 2, $\mathrm{OPT}_{\mathrm{SC}/k}((U,\mathscr{S}),k) = \mathrm{OPT}_{s\text{-CLUBC}}(G,k)$. Moreover, if $|F| \geq k+1$ then solution lifting algorithm return $\mathscr{F} = \mathscr{S}$ and in this case, s-CLUBC$(G,k,F) = k+1 = \mathrm{SC}/k((U,\mathscr{S}),k,\mathscr{S})$. If $|F| \leq k$ then by Claim 2, s-CLUBC$(G,k,F) = \mathrm{SC}/k((U,\mathscr{S}),k,\mathscr{F})$. This implies that there exists a 1-`appt` from SET COVER$/k$ to 2-CLUB CONTRACTION even input graph is restricted to a split graph. □

Arguments to generalise this lemma to higher value of s is identical to those presented in [47]. We present them here briefly for the sake of completeness. To show above lemma holds for 3-CLUB CONTRACTION, we modify the reduction algorithm as follows. Instead of adding $b^1,\ldots,b^{k+1}$ adjacent to a, we crate $k+1$ vertices $z^1,\ldots,z^{k+1}$ and make the set $\{z^1,\ldots,z^{k+1},a\}$ into a clique. Now we construct $b^1,\ldots,b^{k+1}$ and make b^i adjacent to z^i for $i \in \{1,\ldots,k+1\}$. By similar arguments, we can show that $\mathrm{OPT}_{\mathrm{SC}/k}((U,\mathscr{S}),k) = \mathrm{OPT}_{s\text{-CLUBC}}(G,k)$. Moreover, s-CLUBC$(G,k,F) = \mathrm{SC}/k((U,\mathscr{S}),k,\mathscr{F}_F)$. For $s \geq 4$, consider a graph G and denote by G' the graph obtained from G by adding $k+1$ pendant vertices adjacent to v for each vertex v of G. It is straightforward to observe that G' is

k-contractible to a graph of diameter at most s if and only if G is k-contractible to a graph of diameter at most $s-2$. As we have proved that the lemma holds for $s \in \{2,3\}$, this observation immediately implies the lemma for every fixed $s \geq 2$. By Corollary 2.4.1 and Lemma 7.4.1, we get following result.

Theorem 7.4.1. *s-CLUBC(G,k,F) does not have a polynomial size α-approximate compression for any $\alpha \geq 1$, unless NP $\subseteq$ coNP/poly.*

7.5 Conclusion

In this chapter we present a lossy kernel of polynomial size for CLIQUE CONTRACTION when parameterized by solution size. This compliments the known results that the problem does not have a polynomial (classical) kernel. Our kernelization algorithm depends on the fact that in a large instance solution edges affect very few vertices. Remaining set of vertices are adjacent with each other and most of affected vertices. If large number of unaffected vertices have same neighbors and non-neighbors in affected vertices then we can delete one of these vertices with slight loss of accuracy. It is interesting to see whether these methods can be generalized to get lossy kernel for SPLIT CONTRACTION.

Chapter 8

Path Contraction

8.1 Introduction

Any connected graph can be contracted to an edge which is a path on two vertices. In this chapter, we address a question of determining the largest integer ℓ for given graph such that it can be contracted to P_ℓ, path on ℓ vertices. Formally, we study following problem.

PATH CONTRACTION
Input: Graph G
Output: Largest integer ℓ such that G can be contracted to P_ℓ

Early paper of Brouwer and Veldman states that we can determine whether a given graph can be contracted to P_3 or not in polynomial time but it is NP-Hard to determine whether it can be contracted to P_4 or not [14]. This implies that we can not expect an algorithm for the problem which runs in time $\mathcal{O}(n^{f(\ell)})$. However, there is a simple algorithm running in time $\mathcal{O}^\star(2^n)$ * algorithm (See Observation 8.2.3). Algorithm with better running time are known for special case. Cygan et al. [26] observed that P_4-CONTRACTION is same as partitioning given graph into two disjoint connected subgraphs which contain specified terminals. They called it 2-DISJOINT CONNECTED SUBGRAPHS problem and gave an

* $\mathcal{O}^\star$ notation hides factors which are polynomial in size of input.

algorithm running in time $\mathcal{O}(1.933^n)$. Telle and Villanger [90] presented an algorithm to solve the same problem in time $\mathcal{O}(1.7804^n)$.

We generalize the approach presented by Telle and Villanger [90] to solve 3-DISJOINT CONNECTED SUBGRAPHS problem (precise definition follows) which is idential to P_5-CONTRACTION problem (Section 8.4). We present an algorithm running in time $\mathcal{O}(1.877^n)$. We use this and Telle and Villanger's algorithm as subroutine in our main algorithm. The main algorithm, presented in Section 8.5, is based on four different methods to attack the problem. We argue that for any graph, at least one of these methods returns an optimum value. In one methods, key component is to enumerate all connected supersets of given set of vertices which are of size at most a and boundary b, where a, b are two fixed integers. We present an algorithm to enumerate all such connected sets in Section 8.3 which may be of independent interest.

We mention that parameterized version of this problem, with number of edges, k, allowed to contract to obtain a path as parameter has been studied by Hergerners et al. They presented an algorithm running in time $2^{k+o(k)}n^{\mathcal{O}(1)}$ and kernel of size $5k+3$ [55].

8.2 Preliminaries

In this chapter, we slightly abuse the notation of set brackets when writing a P_t-witness structure of graph. When we say $\mathscr{W} = \{W_1, W_2, \ldots, W_t\}$ is a P_t-witness structure of graph G, we treat $\mathscr{W}$ as *ordered set*. In other words, we assume that one end point in path P_t is designated as first vertex and witness sets $W_1, W_2, \ldots$ corresponds to first, second, and so on vertices in P_t. We start with few simple observations.

Observation 8.2.1. *Any connected graph can be contracted to P_2.*

Observation 8.2.2. *Consider a graph G which can be contracted to P_t. There exists a P_t-witness structure $\mathscr{W} = \{W_1, \ldots, W_t\}$ of G such that W_1, W_t are singleton sets.*

Proof. Let $\{W_1', \ldots, W_t'\}$ be a P_t-witness structure of G. We modify witness sets W_1', W_t' to ensure that they satisfy desired property. There exists an edge, say u_1u_2 in graph G where u_1, u_2 are contained in sets W_1', W_2'. Assume that witness set W_1' is not a singleton set. Fix a spanning tree of graph $G[W_1']$ which is rooted at u_1 and let v be one of its leaf. Since v is leaf in a spanning tree of $G[W_1']$, set $W_1' \setminus \{v\}$ is connected. Moreover, set $(W_1' \cup W_2') \setminus \{v\}$ is also connected. It is easy to check that $\{\{v\}, (W_1' \cup W_2') \setminus \{v\}, \ldots, W_t'\}$ is also a P_t-witness structure of G. Applying similar arguments on witness set W_t', we obtain a P_t-witness structure in which both end points are singleton sets. □

We present a simple algorithm for PATH CONTRACTION.

Observation 8.2.3. *There exists an algorithm that solves* PATH CONTRACTION *problem in* $\mathcal{O}^\star(2^n)$ *time where n is the number of vertices in an input graph.*

Proof. Algorithm $\mathscr{A}$ starts with initialising an integer t to 2. For a given graph G, the algorithm runs over all possible 2-colorings of vertices of G. For every coloring, the algorithm contracts each monochromatic connected component of the coloring to a vertex. If the resulting graph is a path then the algorithm updates value of t to length of this path. Algorithm returns value of t after iterating over all 2-colorings.

Running time of the algorithm is $\mathcal{O}^\star(2^n)$ as contracting edges and checking whether a graph is a path or not is polynomial time process. Any connected graph can be contracted to P_2.The algorithm returns an integer t which is strictly greater then two only if had found a P_t-witness structure of G. It remains to argue that if ℓ is the largest integer such that G can be contracted to P_ℓ then algorithm returns ℓ. Let $\mathscr{W} = \{W_1, W_2, \ldots, W_\ell\}$ be a P_ℓ-witness structure of G. Since algorithm $\mathscr{A}$ iterates over all 2-coloring of $V(G)$, it also consider a coloring where all vertices in odd indexed witness sets in $\mathscr{W}$ are colored with one color and all vertices in even indexed witness sets are colored with another color. For this particular coloring, $W_1, W_2, \ldots, W_\ell$ are monochromatic connected components of G. Contracting each of them to a vertex results in path of length ℓ. Hence algorithm returns a value which is at

least ℓ. $\square$

We end this section with an observation which is used to bound the number of subsets of universal set U which are of size size at most $\delta|U|$ for a fixed fraction δ. We start with following inequality for integers n and k such that $k \leq n$.

$$\binom{n}{k} \leq \left[\left(\frac{k}{n}\right)^{-\frac{k}{n}} \cdot \left(1-\frac{k}{n}\right)^{\frac{k}{n}-1}\right]^n$$

Using above inequality we get following upper bound on summand for $k < n/2$.

$$\sum_{i=1}^{k} \binom{n}{i} \leq k \cdot \left[\left(\frac{k}{n}\right)^{-\frac{k}{n}} \cdot \left(1-\frac{k}{n}\right)^{\frac{k}{n}-1}\right]^n$$

For a positive constant $\delta < 1/2$, assume that δn is an integer for the sake of clarity. Above inequalities can be written as:

$$\sum_{i=1}^{\delta n} \binom{n}{i} \leq \delta n \cdot \left[\delta^{-\delta} \cdot (1-\delta)^{\delta-1}\right]^n$$

$$\sum_{i=1}^{\delta n} \binom{n}{i} \leq \delta n \cdot \left[\frac{1}{\delta^{\delta}} \cdot \frac{1}{(1-\delta)^{1-\delta}}\right]^n = \delta n[g(\delta)]^n$$

where function $g(\delta)$ is defined as:

$$g(\delta) = \frac{1}{\delta^{\delta} \cdot (1-\delta)^{(1-\delta)}}$$

Following observation is implied by above inequalities.

Observation 8.2.4. *For a universe U of n elements and a constant $\delta < 1/2$, the number of subsets of U of size at most δn is $\mathcal{O}^{\star}([g(\delta)]^n)$ and all these subsets can be enumerated in same time.*

Algorithm 8.3.1: Enum-Conn-Sets: Enumeration Algorithm for (Q,a,b)-connected sets

Input: A graph G, a non-empty set $Q \subseteq V(G)$, and integers $a,b \in \mathbb{N}$.
Output: The set of all (Q,a,b)-connected sets in G.

```
if |Q| > a or |N[Q]| > a + b then
    return ∅
if |Q| = a and G[Q] is connected then
    return {Q}
if |Q| = a and G[Q] is not connected then
    return ∅
Consider a vertex v ∈ N(Q);
return
  Enum-Conn-Sets(G, Q ∪ {v}, a − 1, b) ∪ Enum-Conn-Sets(G − {v}, Q, a, b − 1)
```

8.3 Enumeration of Connected Sets

A graph is called *connected* if there is a path between every pair of vertices. A maximal connected subgraph is called a *connected component* or a *component* in a graph. A set $A \subseteq V(G)$ is a *connected set* in G if $G[A]$ is a connected graph. For a graph G, a non-empty set $Q \subseteq V(G)$, and integers $a,b \in \mathbb{N}$, a connected set A in G is a (Q,a,b)-*connected set* if $Q \subseteq A$, $|A| \leq a$, and $|N(A)| \leq b$.

Lemma 8.3.1. *For a graph G, a non-empty set $Q \subseteq V(G)$, and integers $a,b \in \mathbb{N}$ the number of (Q,a,b)-connected sets in G is at most $2^{a+b-|Q|}$. Moreover, we can enumerate all (Q,a,b)-connected sets in G in time $\mathcal{O}(2^{a+b-|Q|} \cdot n^c)$.*

Proof. We give an algorithmic (constructive) proof for the lemma. The algorithm Enum-Conn-Sets, for enumerating all (Q,a,b)-connected sets in G is given in Algorithm 8.3.1. Next, we prove the correctess and the desired running time bound for the algorithm Enum-Conn-Sets.

Correctness. Let $I = (G,Q,a,b)$ be an instance for the algorithm Enum-Conn-Sets. The objective is to show that the algorithm outputs all the (Q,a,b)-connected sets in G. We proof the correctness by induction on $\mu = \mu(I) = a - |Q| + b$.

- **Base Cases:** The base case occurs when one of the following conditions hold.

  1. $|Q| > a$ or $|N[Q]| > a+b$. In this case, Step 1 of the algorithm returns $\emptyset$ as the output. If $|Q| > a$, then there is no (Q,a,b)-connected set in G, and hence the output of the algorithm is correct. Otherwise, we have $|N[Q]| > a+b$. Consider a (Q,a,b)-connected set A (if it exists) in G. Notice that for a vertex $v \in N(Q)$ is either in A or in $N(A)$. Moreover, $N[A]$ has size at most $a+b$. This imples that such a connected set cannot exist. Therefore, the output of the algorithm is correct.
  2. $|Q| = a$ and $G[Q]$ is connected. Since Base Case 1 is not applicable, we have that $|N(Q)| \leq b$. Furthermore, Step 3 of the algorithm is applicable, which output $\{Q\}$ as the set of all (Q,a,b)-connected sets in G. Since $|Q| = a$, and any (Q,a,b)-connected set in G must contain all the vertices in Q, therefore Q is the only potential candiate for a (Q,a,b)-connected set in G. Moreover, Q is a (Q,a,b)-connected sets in G as it satisfies all the conditions of the definition (including $|N(Q)| \leq b$). Hence, the output of the algorithm is correct.
  3. $|Q| = a$ and $G[Q]$ is not connected. Similar to the previous case we have that $|N(Q)| \leq b$. Furthermore, Step 5 of the algorithm is applicable, which output $\emptyset$ as the set of of all (Q,a,b)-connected sets in G. Since $|Q| = a$, and any (Q,a,b)-connected set in G must contain all the vertices in Q, therefore Q is the only potential candiate for a (Q,a,b)-connected set in G. But, $G[Q]$ is not connected, and therefore, Q cannot be a connected set. Hence, the output of the algorithm is correct.
  4. $\mu = 0$, which occurs when $|Q| = a$ and $b = 0$. In this case, one of previous base cases must be applicable. Therefore, from item 1 and 2 of the base cases it follows that the output of the algorithm is correct.

- **Induction Hypothesis:** We assume that the output of the algorithm is correct for all $\mu \leq t$, where $t \in \mathbb{N}$. Next, we show that the output of the algorithm is correct when $\mu = t+1$. Since the base cases are not applicable, we have $|Q| < a$ and

$|N[Q]| \leq a+b$. Consider a vertex $v \in N(Q)$. We can partition the set $\mathscr{S}$, of all (Q,a,b)-connected sets in G into two sets $\mathscr{S}_1^v$ and $\mathscr{S}_2^v$, where $\mathscr{S}_1^v = \{A \in \mathscr{S} \mid v \in A\}$ and $\mathscr{S}_2^v = \mathscr{S} \setminus \mathscr{S}_1^v$. Notice that $\mathscr{S}_1^v$ is the set of all $(Q \cup \{v\},a,b)$-connected sets in G and $\mathscr{S}_2^v$ is the set of all $(Q,a,b-1)$-connected sets in $G-\{v\}$. By induction hypothesis we correctly obtain the set $\mathscr{S}_1^v$, of all $(Q \cup \{v\},a,b)$-connected sets in G for the input instance $I_1 = (G, Q \cup \{v\}, a-1, b)$ to the algorithm. This follows from the fact that $\mu(I_1) = a-(|Q|+1)+b \leq t < \mu$. Similarly, we correctly obtain the set $\mathscr{S}_2^v$, of all $(Q,a,b-1)$-connected sets in $G-\{v\}$ for the input instance $I_2 = (G-\{v\}, Q, a, b-1)$ to the algorithm, which follows from the fact that $\mu(I_2) = a-|Q|+(b-1) \leq t < \mu$. Hence, the output $\mathscr{S} = \mathscr{S}_1^v \cup \mathscr{S}_2^v$ of the algorithm is correct.

Number of (Q,a,b)-connected sets. Let $I = (G,Q,a,b)$ be an instance for the algorithm Enum-Conn-Sets. We use the measure $\mu = \mu(I) = a-|Q|+b$ for counting the number of (Q,a,b)-connected sets in G. Observe that Step 1 to Step 7 (in total) output at most 1 set. At Step 8, we make two recursive calls to the algorithm. Let $K(G,a-|Q|,b)$ denote the number of (Q,a,b)-connected sets in G. The recurrence for the number of the connected sets is given by the following recurrence.

$$K(G,a-|Q|,b) \leq K(G,a-|Q|-1,b) + K(G-\{v\},a-|Q|,b-1)$$

Solving the above recurrence we obtain that the number of (Q,a,b)-connected sets in a graph is bounded by $2^{a+b-|Q|}$.

Runtime Analysis. Let $I = (G,Q,a,b)$ be an instance for the algorithm Enum-Conn-Sets. We use the measure $\mu = \mu(I) = a-|Q|+b$ for analysing the running time of the algorithm. Observe that Step 1 to Step 7 of the algorithm can be executed in polynomial time. At Step 8, we make two recursive calls to the algorithm. Let $T(n,a-|Q|,b)$ denote the running time required for an instance where the graph comprises of n vertices. The recurrence for

the runtime of the algorithm is as follows.

$$T(n,a-|Q|,b) \leq T(n,a-|Q|-1,b)+T(n,a,b-1)+n^c$$

We note that in the above recurrence c is some (fixed) constant. Solving the recurrence we obtain that the running time of the algorithm is bounded by $2^{a+b-|Q|}n^c$. □

For a graph G and integers $a,b \in \mathbb{N}$, a connected set A in G is a (a,b)-*connected set* if $|A| \leq a$, and $|N(A)| \leq b$.

Lemma 8.3.2. *For a graph G and integers $a,b \in \mathbb{N}$ the number of (a,b)-connected sets in G is at most 2^{a+b}. Moreover, we can enumerate all (a,b)-connected sets in G in time $\mathcal{O}(2^{a+b} \cdot n^c)$.*

Proof. Note that every non empty (a,b)-connected set is $(\{v\},a,b)$-connected sets for somve vertex v in G. Hence proof of this lemma follows from Lemma 8.3.1. □

8.4 3-DISJOINT CONNECTED SUBGRAPH

In this section, we define a generalization of 2-DISJOINT CONNECTED SUBGRAPHS (2-DCS), called 3-DISJOINT CONNECTED SUBGRAPHS (3-DCS), and present an exact algorithm to solve it. This algorithm runs in time $\mathcal{O}^\star(1.877^n)$, where n is number of vetices in input graph. Apart from using this algorithm as subroutine in an algorithm for PATH CONTRACTION we use this algorithm to solve P_5-CONTRACTION. We start with formal definition of 2-DISJOINT CONNECTED SUBGRAPHS (2-DCS) problem.

2-DISJOINT CONNECTED SUBGRAPHS (2-DCS)

Input: Connected graph G and two disjoint terminal sets Z_1 and Z_2

Question: Do there exist two subsets V_1, V_2 of $V(G)$ which satisfies following properties?

1. Tuple (V_1, V_2) is a partition of $V(G)$.
2. Sets V_1, V_2 are supersets of Z_1, Z_2, respectively.
3. Graphs $G[V_1]$ and $G[V_2]$ are connected.

In 3-DCS problem, an input is same but we are intersected in tri-partition of $V(G)$. The third part separates two subgraphs containing terminal sets. We formally define it as follows.

3-DISJOINT CONNECTED SUBGRAPHS (3-DCS)

Input: Connected graph G and two disjoint terminal sets Z_1 and Z_2

Question: Doe there exist three subsets V_1, U, V_2 of $V(G)$ which satisfies following properties?

1. Tuple (V_1, U, V_2) is a partition of $V(G)$.
2. Sets V_1, V_2 are supersets of Z_1, Z_2, respectively.
3. Graphs $G[V_1], G[V_2]$ and $G[U]$ are connected.
4. Graph $G - U$ has exactly two connected components viz V_1, V_2.

Any tri-partition (V_1, U, V_2) which satisfies these condition is called a *solution* tri-partition. To solve this problem efficiently, we try to find special kind of tri-partition called *immovable tri-partition*. Informally, this is a solution partition in which no vertex in V_1 or V_2 can be *moved* to U. We use notion of *terminal-separator* to formally define these special partitions. For a given graph G and set of terminals Z, a vertex v is called *Z-separator* if Z intersects with at least two connected components of $G - v$.

Definition 8.4.1 (Immovable Tri-partition)**.** *A solution tri-partition (V_1, U, V_2) is said to be immovable tri-partition if any vertex v in $V_1 \setminus Z_1$ (resp. $V_2 \setminus Z_2$) which has at least one neighbor in U is a Z_1-separator (resp. Z_2-separator) in graph $G[V_1]$ (resp. in $G[V_2]$).*

Following claim guarantees existence of such tri-partition for a YES instances.

Claim 8.4.1. *If instance (G, Z_1, Z_2) is a* YES *instance of* 3-DCS *then there exists an immovable tri-partition of G.*

Proof. Let (V_1, U, V_2) be a solution tri-partition of $V(G)$. If this is an immovable tri-partition then we are done. Otherwise, without loss of generality, assume that there exists a vertex in $V_1 \setminus Z_1$ which has neighbors in U and is not a Z_1-separator in graph $G[V_1]$. Let $C_1, C_2, \ldots, C_d$ be connected components of $G[V_1] - v$. Note that d can be equal to 1. Since v is not a Z_1-separator, we know that Z_1 is contained in one of the connected components. Let C_1 be the connected component which contains Z_1. Consider tri-partition (V_1', U', V_2) of $V(G)$ where $V_1' = C_1 = V_1 \setminus (\{v_1\} \cup C_2 \cup \cdots \cup C_d)$ and $U' = U \cup \{v_1\} \cup C_2 \cup \cdots \cup C_d$. This tri-partition is also a solution partition as both $V_1' = C_1$ and U' are connected and V_1' contains Z_1. For a given tri-partition we can either find a vertex to move from $V_1 \cup V_2$ to U or conclude that it is an immovable tri-partition Since every step reduces the number of vertices in $V_1 \cup V_2$ and we never add any vertex in $V_1 \cup V_2$ this process terminates in at most n steps and returns an immovable tri-partition. □

We soon see that one can compute immovable tri-partitions in graph G using *minimal terminal connectors* in the graph obtained by adding a specific edge in G.

Definition 8.4.2 (Minimal Z-connector)**.** *Given a graph G and a set of terminal vertices Z, a superset S of Z is called Z-connector if S induces a connected graph. Moreover, if no strict subset of S is Z-connector then it is called minimal Z-connector.*

We use terms *Z-connecting* and *Z-connector* interchangeably.

Claim 8.4.2. *Let (G, Z_1, Z_2) be a* YES *instance of* 3-DCS *and (V_1, U, V_2) be an immovable tri-partition of $V(G)$. Consider a minimal Z_1-connecting set S_1 in graph $G[V_1]$ and a minimal Z_2-connecting set S_2 in graph $G[V_2]$. Then, no connected component of $G[V_1] - S_1$ or $G[V_2] - S_2$ is adjacent with U.*

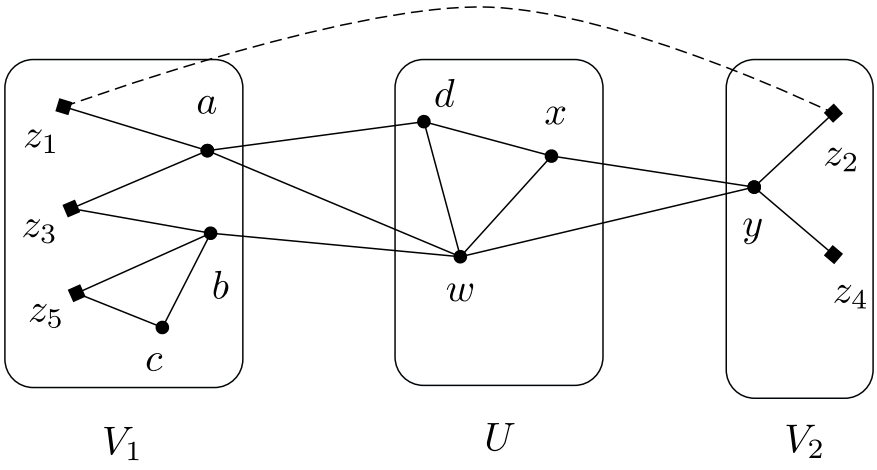


Figure 8.1: Consider an instance (G, Z_1, Z_2) with $Z_1 = \{z_1, z_3, z_5\}$ and $Z_2 = \{z_2, z_4\}$. Dotted line is added in graph G to obtain G'. Tuple (V_1, U, V_2) is an immovable tri-partition of $V(G)$. Set $S_1 = Z_1 \cup \{a, b\}$ and $S_2 = Z_2 \cup \{y\}$ are minimal Z_1-connector and Z_2-connector in $G[V_1]$ and $G[V_2]$, respectively. Set $S = S_1 \cup S_2$ is minimal $(Z_1 \cup Z_2)$-connector in graph G'. Please refer to Claim 8.4.3.

Proof. Assume that there exists a connected component C of $G[V_1] - S_1$ which is adjacent with U. Let v be a vertex in C which has neighbor in U. Note that since S_1 is a connected and v is outside S_1, vertex v is not Z_1-separator in $G[V_1]$. This contradicts the fact that (V_1, U, V_2) is an immovable partition. Hence no vertex in any connected component of $G[V_1] - S_1$ has neighbors in U. Similar argument holds for any connected component of $G[V_2] - S$ leads to the same contradiction. Hence no connected component of $G[V_1] - S_1$ or $G[V_2] - S_2$ is adjacent with U. □

Note that above claim implies that in graph $G - S_1$, there exists a unique connected component which is $U \cup V_2$ and all other connected components of same as that of $G[V_1] - S_1$.

For given instance (G, Z_1, Z_2), we assume that there does not exists an edge with one end point in Z_1 and another in Z_2 as otherwise it is a NO instance. Fix vertices z_1 and z_2 in set Z_1 and Z_2, respectively. Let G' be the graph obtained by adding an edge $z_1 z_2$ in G. In the following claim, we relate an immovable tri-partition of G with minimal separators in G'. See Figure 8.1.

Claim 8.4.3. *Let (G, Z_1, Z_2) be a* YES *instance of* 3-DCS *and (V_1, U, V_2) be an immovable tri-partition of $V(G)$. Consider a minimal Z_1-connecting set S_1 in graph $G[V_1]$ and a minimal Z_2-connecting set S_2 in graph $G[V_2]$. Then, set $S = S_1 \cup S_2$ is a minimal $(Z_1 \cup Z_2)$-connecting set in graph G'.*

Proof. Let $cc(G)$ denotes the number of connected components in graph G. Consider any two sets X_1 and X_2 which are subsets of V_1 and V_2 respectively. Since we have added only one edge between V_1 and V_2 while constructing graph G', we get $cc(G'[X_1 \cup X_2]) \geq cc(G[X_1]) + cc(G[X_2]) - 1$.

We first argue that S is a $(Z_1 \cup Z_2)$-connector in G'. As $G'[S_1]$ and $G'[S_2]$ are connected and there is an edge $z_1 z_2$ with one end point in S_1 and another in S_2, graph $G'[S]$ is connected. Since S contains $Z_1 \cup Z_2$, it is a $(Z_1 \cup Z_2)$-connector.

It remains to argue that no proper subset of S is a $(Z_1 \cup Z_2)$-connector. For the sake of contradiction, assume that there exists a proper subset of S, say S', which is a $(Z_1 \cup Z_2)$-connector in graph G'. Let $S'_1 = S' \cap V_1$ and $S'_2 = S' \cap V_2$. Consider a case when S' does not contain all vertices in $S_1 \setminus Z_1$. In other words, S'_1 is a proper subset of S_1. Recall that S_1 is a minimal Z_1-connector in $G[V_1]$ and S'_1 contains Z_1. By minimality of S_1, graph $G[S'_1]$ is not connected and hence $cc(G[S'_1]) \geq 2$. This implies $G'[S'] = G'[S'_1 \cup S'_2] \geq G[S'_1] + G[S'_1] - 1 \geq 2$ as $cc(G[S'_2]) \geq 1$. This contradicts the fact that $G'[S']$ is a connected graph. By symmetric arguments, assuming S'_2 is a proper subset of S_2 leads to same the contradiction. Hence our assumption is wrong and no proper subset of S is a $(Z_1 \cup Z_2)$-connector. □

We say a minimal $(Z_1 \cup Z_2)$-connector S in graph G' is *realized* by an immovable tri-partition (V_1, U, V_2) of $V(G)$ if S can be partitioned into S_1, S_2 such that S_1 is a minimal Z_1-connector in $G[V_1]$ and S_2 is a minimal Z_2-connector in $G[V_2]$. Claim 8.4.3 implies that every immovable tri-partition of $V(G)$ realizes at least one minimal $(Z_1 \cup Z_2)$-connector in G'.

Given a minimal $(Z_1 \cup Z_2)$-separator S of G', we want to construct an immovable tri-partition of $V(G)$, if exits, which realizes it. Note that if S is realized by some immovable tri-partition of $V(G)$ then $G[S]$ has two connected components containing Z_1 and Z_2. If this is not the case then we can conclude that this minimal separator is not realized by any immovable tri-partition. Let S_1, S_2 are two connected components of $G[S]$ which contains Z_1 and Z_2 respctively. If S is realized by an immovable tri-partition (V_1, U, V_2) then, by Claim 8.4.2, every connected component of $G - S_1$ which is not $U \cup V_2$ is also a connected component of $G[V_1] - S_1$. We do not know $U \cup V_2$ in advace. But connected component of $G - S_1$ which is $U \cup V_2$ contains S_2. Hence, V_1 consists of S_1 together with all connected components of $G - S_1$ which do not intersect S_2.

We illutrate this idea with an example. Consider the graph drawn in Figure 8.1. Set $S = Z_1 \cup \{a,b\} \cup Z_2 \cup \{y\}$ is a minimal $(Z_1 \cup Z_2)$-connector in G'. Set S can be partitioned into two sets, $S_1 = Z_1 \cup \{a,b\}$ and $S_2 = Z_2 \cup \{y\}$ such that both $G[S_1]$ and $G[S_2]$ are connected and they contain Z_1 and Z_2. Note that connected component of $G - S_1$ which does not intersects S_2, vertex c, is contained in V_1.

We enumerate all minimal $(Z_1 \cup Z_2)$-connector S in G' and try to construct a tri-partition (V_1, U, V_2) of $V(G)$ as described above for every S. If we find such tri-partition we return it as solution or conclude that no such tri-partition exists. We use following result to enumerate all minimal $(Z_1 \cup Z_2)$-connecting subsets in G'.

Proposition 8.4.1 ([90]). *For an n vertex graph G and a terminal set $T \subset V(G)$ where $|T| \leq n/3$ there are at most $\binom{n-|T|}{|T|-2} \cdot 3^{(n-|T|)/3}$ minimal T-connecting vertex sets and those can be enumerated in time $\mathcal{O}^\star(\binom{n-|T|}{|T|-2} \cdot 3^{(n-|T|)/3})$.*

We now state the following lemma which solves 3-DCS when number of terminals are *small* as compare to number of vertices in graph. The reason to choose specific value of δ will be clear in Theorem 8.4.1.

Lemma 8.4.1. *There exists an algorithm that solves the* 3-DISJOINT CONNECTED SUBGRAPHS *problem in $\mathcal{O}^\star(1.877^n)$ time if the number of terminals is at most δn. Here n is*

number of vertices in input graph and $\delta = 0.092$.

Proof. Let (G, Z_1, Z_2) be an input instance where G is graph on n vertices and $|Z_1 \cup Z_2| \leq \delta n$. The algorithm arbitrarily fixes terminals z_1, z_2 in Z_1, Z_2, respectively and adds an edge $z_1 z_2$ to obtain graph G'. It enumerates all minimal $(Z_1 \cup Z_2)$-connector in G'. For every minimal connector S, the algorithm checks whether there are exactly two connected component of $G[S]$, say S_1, S_2, containing Z_1 and Z_2, respectively. If such connected component exists then the algorithm construct a tri-partition of $V(G)$ in the following way. Initialize sets V_1, U, V_2 to $S_1, \emptyset, S_2$, respectively. Any connected component of $G - S_1$ which does not contain V_2 is added to V_1. The algorithm expands V_2 in similar way. All vertices which are not added in V_1 or V_2 are added to U. If (V_1, U, V_2) is a solution tri-partition then the algorithm returns it and terminates otherwise moves on to next minimal $(Z_1 \cup Z_2)$-connector of G'. The algorithm concludes that solution exists if it can not find a solution tri-partition for any minimal $(Z_1 \cup Z_2)$-connector in graph G'.

We argue correctness of the algorithm. Note that the algorithm returns a tri-partition only if it has found one. By Claim 8.4.1, if (G, Z_1, Z_2) is a YES instance then there exists an immovable tri-partition (V_1, U, V_2) of $V(G)$. By Claim 8.4.3, there exists a minimal $(Z_1 \cup Z_2)$-connector in G' which is realized by (V_1, U, V_2). Since algorithm considers all minimal $(Z_1 \cup Z_2)$-connector, it also considers the one realized by (V_1, U, V_2). By Claim 8.4.2, if S is a minimal $(Z_1 \cup Z_2)$-connector realized (V_1, U, V_2), then V_1 is union of S_1 with connected components of $G - S_1$ which do not contain S_2. Similar statement holds for V_2. Hence if given instance is a YES instance, the algorithm considers the minimal terminal connector realized by an immovable tri-partition and constructs the tri-partition associated with it.

Note that $\delta = 0.092 < 1/3$ and hence we can use Proposition 8.4.1 to enumerate all minimal $(Z_1 \cup Z_2)$-connector in time $\mathcal{O}^\star(\binom{n-|Z_1 \cup Z_2|}{|Z_1 \cup Z_2|-2} \cdot 3^{(n-|Z_1 \cup Z_2|)/3})$ which is $\mathcal{O}^\star(\binom{(1-\delta)n}{\delta n} \cdot 3^{(1-\delta)n/3})$. Using $\mu = 1 - \delta$ and Stirling approximation, we can bound $\binom{\mu n}{\delta n}$ by $\mathcal{O}^\star((\frac{\mu^\mu}{\delta^\delta \cdot (\mu-\delta)^{(\mu-\delta)}})^n)$ or $\mathcal{O}^\star((\frac{(1-\delta)^{(1-\delta)}}{\delta^\delta \cdot (1-2\delta)^{(1-2\delta)}})^n)$. Using computer we can verify that maximum value of $((\frac{(1-\delta)^{(1-\delta)}}{\delta^\delta \cdot (1-2\delta)^{(1-2\delta)}}) \cdot$

$3^{1/3}$) for $0 < \delta \leq 0.092$ occurs when $\delta = 0.092$ and it is 1.877. This implies the mentioned running time of the algorithm. □

We are now in position to present main theorem of this section.

Theorem 8.4.1. *There exists an algorithm that solves the* 3-DISJOINT CONNECTED SUBGRAPHS *problem in $\mathcal{O}^\star(1.877^n)$ time where n is number of vertices in input graph.*

Proof. Let (G, Z_1, Z_2) be an input instance. We consider two cases depending on number of terminals. If $|Z_1 \cup Z_2| > \delta n$ where $\delta = 0.092$ then enumerate all subsets in $V(G) \setminus (Z_1 \cup Z_2)$ to determine middle portion of tri-partition. For every set U of size $(1-\delta)n$, we check in polynomial time whether $G(U)$ is connected and $G - U$ has exactly two connected components, say V_1, V_2 which contain Z_1 and Z_2, respectively. If there exists such set then we return (V_1, U, V_2) as tri-partition. The correctness of algorithm follows from the fact that we are doing exhaustive search in this process. Total time to complete the process is $\mathcal{O}^\star(2^{(1-\delta)n}) = \mathcal{O}^\star(2^{(1-0.092)n}) = \mathcal{O}^\star(1.877^n)$. If $|Z_1 \cup Z_2| \leq \delta n$ then, by Lemma 8.4.1, there exists an algorithm running in time $\mathcal{O}^\star(1.877^n)$. □

P_5-Contraction

We use Theorem 8.4.1 to check whether given graph G can be contracted to P_5 or not. By Observation 8.2.2, if graph G is contractible to P_5 then there exists a P_5-witness structure $\mathscr{W} = \{W_1, \ldots, W_5\}$ of G such that W_1, W_5 are singleton sets. We guess the pair of vertices which are in these singleton witness set. There are at most $\mathcal{O}(n^2)$ many choices for such pairs. Let $\{x\}, \{y\}$ be guess for W_1, W_5 respectively. Sets $N(x), N(y)$ must be contained in witness set $W(t_2), W(t_4)$ respectively. In graph $G - \{x, y\}$, we use $N(x), N(y)$ as set of terminals to find a tri-partition (V_1, U, V_2). If exists, these three sets can work as witness structures corresponding to W_2, W_3, W_4 respectively. This simple algorithm implies following corollary of Theorem 8.4.1.

Corollary 8.4.1. *Given a graph on n vertices, one can decide whether it can be contracted to P_5 or not in time $\mathcal{O}^\star(1.877^n)$.*

8.5 Exact Algorithm for Path Contraction

We start with overview of the algorithm for PATH CONTRACTION which consists of four methods. We elaborate on each method in separate subsections and present entire algorithm with proof of correctness in Subsection 8.5.5. Let α, β, γ be positive constants which are strictly less than one.

• In Subsection 8.5.1, we enumerate all subsets of size less than $n\beta/2$ and for every subset check whether is it a union of all odd or even indexed witness sets for some witness structure corresponding to a path.

• In Subsection 8.5.2, we use dynamic programming to build *partial* witness structure for some subgraphs of given graph. To do this, we store the the maximum length of path that can obtain from a subgraph with additional constraint that all *boundary vertices* of the subgraph are in one of end bags. Once we have this value, we enumerate all possible sets which can be added as next bag to this partially contracted graph. We perform these operation until cardinality of closed neighborhood of subgraph under consideration is at most αn.

• In Subsection 8.5.3, we iterate over all set of vertices of size at most $(1-\gamma)n$. If a graph induced on a subset has exactly two connected components, say C_1, C_2, and graph obtained by removing this set has exactly one connected component, say C, then we consider this set for further checks. We use dynamic programming to get path structure for graph induced on C_1 and C_2. We use the algorithm for 2-DISJOINT CONNECT SUBGRAPHS problem to check whether we can partition C into two parts with desired properties.

• In Subsection 8.5.4, we iterate over all subsets of size εn where $\varepsilon = 1 - \beta/2 - \gamma/2$

and check whether a set can be union of *almost all* odd or even indexed sets. For each connected component of graph obtained by removing the set, we check whether connected component can be partitioned into three parts with desired properties. To do this, we use the algorithm to solve 3-DISJOINT CONNECT SUBGRAPHS mentioned in Section 8.4.

Each subsections contains pseudo-code for an algorithm, its proof of correctness and time require to complete it. For each method, we argue that for a given graph if there exists a witness structure which satisfies certain conditions then PATH CONTRACTION can be solved in time better than $\mathcal{O}(2^n)$ using this method. Clearly, we do not know any witness structure for a given graph corresponding to maximum length of path to which it can be contracted. All these conditions on witness structure are existential. We specify the conditions in forms which are most useful in Subsection 8.5.5 and hence a priory it may not be obvious that these conditions are exhaustive.

Let ℓ be the largest integer such that given graph can be contracted to P_ℓ. In Subsection 8.5.5, we argue that there exists a P_ℓ-witness structure of graph which satisfy at least one of the conditions mentioned in subsections. We now present a brief overview of proof of correctness. Since any connected graph can be contracted to a path of length two, such integer exists. If there exists a P_ℓ-witness structure for given graph in which number of vertices in odd or even indexed witness sets is at most $\beta n/2$ then method in Subsection 8.5.1 correctly identifies this witness structure. We now consider the case when number of vertices are almost equally divided into odd and even numbered witness sets for *all* P_ℓ-witness structures. We subdivide this case based on number of *large* witness sets in witness structure. We quantify large in such a way that there are at most two large bags in any P_ℓ-witness structure. If there exists a witness structure which does not contain any large witness set then we can build a P_ℓ-witness structure using dynamic programming mentioned in Sub-section 8.5.2. If exactly one large bag then we argue that either earlier step returns an optimum solution or we can solve the problem using method mentioned in Sub-section 8.5.4. Consider a case when there are exactly two large bags. If these two bags

Algorithm 8.5.1: Solving PATH CONTRACTION by enumerating subsets

Input: Connected graph G and a positive fraction β
Output: An integer t such that G can be contracted to P_t

```
Initialize t = 2;
𝔹 ← Collection of all subsets of V(G) which are of size at most βn/2 − 1;
for each S in 𝔹 do
    𝒲 ← witness structure obtained by consider each connected component of
      G[S] and G − S as a witness set;
    G′ ← graph obtained from G by contracting witness sets in 𝒲;
    if G′ is a path then
        t = max{t, length of path G′};
return t;
```

are adjacent then we obtain a witness structure by method mentioned in Sub-section 8.5.3. If these two large bags are not adjacent then we get optimum solution from method in Subsection 8.5.2.

8.5.1 Method Using Enumeration of Subsets

In this sub-section we explain the method of enumerating subsets and specify the criteria in which this method returns an optimum solution.

Lemma 8.5.1. *For a given connected graph G on n vertices and a positive constant β, Algorithm 8.5.1 returns an integer t such that G can be contracted to P_t and it terminates in time $\mathcal{O}^\star(c^n)$ where $c = g(\beta/2)$.*

Proof. If algorithm returns 2 then it is correct by Observation 8.2.1. If algorithm returns an integer which is greater than 2 then there exists a set S such that connected components of $G[S]$ and $G-S$ are witness sets in a P_t-witness structure. The running time of algorithm follows from Observation 8.2.4 and the fact that for a given set S, algorithm can obtained graph G' and check whether it is a path or not in polynomial time. □

For a P_t-witness structure $\mathcal{W} = \{W_1, W_2, \ldots, W_t\}$, we define *odd sets* (OS) and *even sets*

(ES) as union of odd and even indexed sets respectively. Formally, these sets are defined as:

$$\texttt{OS} = \bigcup_{x=0}^{\lfloor t/2 \rfloor} W_{2x+1} \text{ and } \texttt{ES} = \bigcup_{x=1}^{\lfloor t/2 \rfloor} W_{2x}$$

.

Definition 8.5.1 (β-Equally Partitioned)**.** *For a positive constant β, a P_t-witness structure $\mathscr{W} = \{W_1, W_2, \ldots, W_t\}$ is said to be β-equally partitioned if the number of vertices in both of sets OS and ES are greater than or equal to $\beta n/2$.*

We note that $\{\texttt{OS}, \texttt{ES}\}$ is a partition of $V(G)$. Lower bound on sizes of both these sets also implies upper bound of $(1-\beta/2)n$ on their sizes. Following lemma states that for a given graph if there exists a witness structure such that size of one of sets OS or ES is at most $\beta n/2$ then Algorithm 8.5.1 is effective on this graph.

Lemma 8.5.2. *For a given connected graph G and a positive constant β, if ℓ is the largest integer such that G can be contracted to P_ℓ and there exists a P_ℓ-witness structure of G which is* not *β-equally partitioned then Algorithm 8.5.1 returns ℓ.*

Proof. Consider a case when size of OS is strictly less than $\beta n/2$. Since Algorithm 8.5.1 enumerates all subsets of $V(G)$ of size strictly less than $\beta n/2$, set OS is also enumerated by it. For this set, algorithm obtains a graph G' which is a path on ℓ vertices. Similar arguement holds when size of ES is strictly less than $\beta n/2$. Hence algorithm returns value which is greater than or equal to ℓ. □

8.5.2 Method Using Dynamic Programming

In this sub-section, we explain a method to build *partial witness structures* for given graph using dynamic programming. Our aim is to construct a witness structure of a connected set corresponding to a path that can be *extended* in remaining graph. For set S of $V(G)$,

we define $\delta(S)$ as the set of vertices in S which have at least one neighbor outside S. Formally, $\delta(S) = \{v|\ v \in S \text{ and } N(v) \setminus S \neq \emptyset\}$. For every connected set S, let $\mathscr{T}[S]$ denotes the largest integer q such that $G[S]$ can be contracted to P_q with property that $\delta(S)$ is contained in an end bag in P_q-witness structure. To compute $\mathscr{T}[S]$, we iterate over all possible sets which can potentially be last bag containing $\delta(S)$. Note that if set B is the witness set corresponding with one of the end bags in path contraction of $G[S]$ then both $G[B]$ and $G[S \setminus B]$ are connected. For a given set S, we enumerate all sets which are possible candidates for B. For a given set S, let $\mathscr{Z}[S]$ denotes collection of sets which can be witness sets corresponding with one of end bags. Formally,

$$\mathscr{Z}[S] = \{B \subseteq S|\ \delta(S) \subseteq B \text{ and } G[B], G[S \setminus B] \text{ are connected}\}$$

We initialize $\mathscr{T}[\{u\}] = 1$ for all vertex u in $V(G)$ which is correct by definition. We compute $\mathscr{T}[S]$ using following recurrence.

$$\mathscr{T}[S] = \max_{B \in \mathscr{Z}[S]} \{\mathscr{T}[S \setminus B]\} + 1 \tag{8.1}$$

We now prove that the above recurrence is correct.

Claim 8.5.1. *Recurrence 8.1 correctly computes $\mathscr{T}[S]$ for every connected subset S of $V(G)$.*

Proof. For a subset S, let B' be the set in $\mathscr{Z}[S]$ where maximum value for right hand side of the equation is achieved. Notice that since B' is in $\mathscr{Z}[S]$, graph $G[S \setminus B']$ is connected and hence $\mathscr{T}[S \setminus B']$ is well defined. Let $\{W_1, W_2, \ldots, W_q\}$ be a P_q-witness structure of $G[S \setminus B']$ associated with $\mathscr{T}[S \setminus B']$. Set $W_{q+1} = B'$. We argue that $\{W_1, W_2, \ldots, W_q, W_{q+1}\}$ is one of candidate witness structure of $G[S]$. By the property of $\mathscr{T}[S \setminus B']$, set $\delta(S \setminus B')$ is contained in W_q and hence there is no edge between W_i and W_{q+1} for any i in $[q-1]$. Moreover, there is at least one edge between W_q and W_{q+1} since S is connected. As $B' = W_{q+1}$ is in $\mathscr{Z}[S]$,

it is connected and contains $\delta(S)$. Hence $\mathscr{T}[S] \geq \mathscr{T}[S \setminus B'] + 1$.

Let $\{W_1, W_2, \ldots, W_{q+1}\}$ be a P_q-witness structure of $G[S]$ corresponding to the value $\mathscr{T}[S]$. We argue that if there is only one witness set in witness structure, i.e. $q = 0$, then every vertex s in S has either one of the following two properties: it is a cut vertex in $G[S]$ or it is in $\delta(S)$. If there exists a vertex in S which does not satisfy either of these two properties then we can construct another witness structure by creating separate witness set containing only this vertex. This contradicts maximality of $\mathscr{T}[S]$. This implies that any non-empty proper subset B of S, either $G[B]$ is not connected or B does not contain all boundary vertices. Hence $\mathscr{Z}[S]$ is empty and recurrence is true. For $q \geq 1$, let $B = W_{q+1}$. To prove the recurrence, we need to argue that B is in $\mathscr{Z}[S]$. By the property of witness structure, $G[B]$ and $G[S \setminus B]$ are connected. By definition of $\mathscr{T}[S]$, set $\delta(S)$ is contained in B. Since $\{W_1, W_2, \ldots, W_q\}$ is a P_q-witness structure of $G[S \setminus B]$, set $\delta(S \setminus B)$ is contained in W_q. Hence $\{W_1, W_2, \ldots, W_q\}$ is one of candidate witness structures corresponding to value of $\mathscr{T}[S \setminus B]$. This implies $\mathscr{T}[S] - 1 \leq \mathscr{T}[S \setminus B]$ for some B in $\mathscr{Z}$. This completes the proof. □

Notice that $\mathscr{T}[V(G)]$ is equal to the largest integer ℓ such that G can be contracted to P_ℓ. We can use the Recurrence 8.1, exactly as stated above, to compute the value of $\mathscr{T}[V(G)]$. The running time of such algorithm is $\mathscr{O}^\star(3^n)$. To avoid this, we compute $\mathscr{T}[S]$ only for connected sets S whose closed neighborhood has at most αn vertices for some constant fraction α. We calculate values corresponding to entries in *bottom-up* fashion but instead of *looking backward* while computing the values, we *look forward* and update values in the table. At each table entry S, we do not iterate over all its proper subsets to update the value of $\mathscr{T}[S]$. Instead we assume that the optimum value for $\mathscr{T}[S]$ is known and update values of some of its supersets. Since we are interested in values of $\mathscr{T}[S]$ only for set S which are connected and for which $N[S]$ is at most αn, we only consider super sets of S which are of size at most αn. We prove that the number of such sets is smaller than $2^{n-|S|}$. These savings at each iterations results in overall running time of $\mathscr{O}^\star(2^{\alpha n})$.

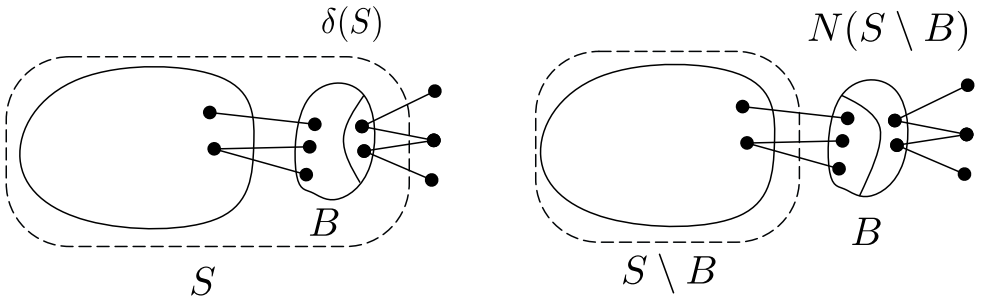


Figure 8.2: Dotted border denotes the set under consideration while updating value in dynamic programming table. In first figure, algorithm consider B as element in $\mathscr{Z}[S]$. In second figure, algorithm consider B as an element in $\mathscr{A}_{a,b}[S \setminus B]$ where $a = |B|$ and $b = |N(S)|$. See Claim 8.5.2

For a connected set S, let $\mathscr{A}[S]$ be a collection of all potential witness sets in $G - S$ which can be appended to contracted path corresponding to $\mathscr{T}[S]$. At each table entry S, we update value of $\mathscr{T}[S \cup A]$ for every A in $\mathscr{A}[S]$. For tight upper bounds, we define $\mathscr{A}_{a,b}[S]$ where a, b are two fixed integers. Set $\mathscr{A}_{a,b}[S]$ is a collection of connected set of size exactly a in $G - S$ which is superset of $N(S)$ and size of neighbors of A in $G - S$ is at most b. In other words, size of neighbors of $A \cup S$ in G is at most b. Formally,

$$\mathscr{A}_{a,b}[S] = \{A |\ N(S) \subseteq A; |A| = a, |N(A) \setminus S| = b \text{ and set } A \text{ is connected in } G - S\}$$

In the follow claim, we argue that instead of computing $\mathscr{Z}[S]$, it is sufficient to compute $\mathscr{A}_{a,b}[S \setminus B]$ for some subset B of S and specific values of a and b.

Claim 8.5.2. *For a connected set S and its non empty subset B, let $a = |B|$ and $b = |N(S)|$. Set B is in $\mathscr{Z}[S]$ if and only if $G[S \setminus B]$ is connected and B is in $\mathscr{A}_{a,b}[S \setminus B]$.*

Proof. ($\Rightarrow$) Definition of $\mathscr{Z}[S]$ implies that $G[B]$, $G[S \setminus B]$ are connected and $\delta(S)$ is subset of B or $N(S \setminus B)$ is contained in B. Since $G[S \setminus B]$ is connected, $\mathscr{A}_{a,b}[S \setminus B]$ is well defined for two integers a, b. It is easy to verify that B is connected set in $G - (S \setminus B)$ and hence B is in $\mathscr{A}_{a,b}[S \setminus B]$ for $a = |B|$ and $b = |N(S)|$.

($\Leftarrow$) Definition of $\mathscr{A}_{a,b}[S \setminus B]$ implies that cardinality of set B is a; $G[B] - (S \setminus B)$ is

Algorithm 8.5.2: Solving PATH CONTRACTION using Dynamic Programming

Input: Connected graph G and a positive fraction α
Output: An integer t such that G can be contracted to P_t

1 Initialize $t = 2$;
2 $\mathbb{S}_\alpha \leftarrow$ Set of all connected subset S of $V(G)$ such that $|N[S]| \leq \alpha n$;
/* $\mathscr{T}[S]$ denotes maximum number of bags in path contraction of $G[S]$ which contains $\delta(S)$ in an end bag. */
3 **for** S *in* $\mathbb{S}_\alpha$ **do**
4 $\quad \mathscr{T}[S] = 1$;
5 **for** S *in* $\mathbb{S}_\alpha$ **do**
6 $\quad x = |N(S)|$; $y = |S|$;
7 $\quad$ **for** *every pair* (a,b) *of positive integers s.t.* $y+a+b \leq \alpha n$ *and* $x \leq b$ **do**
8 $\quad\quad$ Compute $\mathscr{A}_{a,b}[S]$ using Lemma 8.3.1;
$\quad\quad$ /* $\mathscr{A}_{a,b}[S]$ is a collection of connected set A in $G-S$ such that $N(S) \subseteq A$, $|A| = a$ and $|N(A) \setminus (G-S)| \leq b$. */
9 $\quad\quad$ **for** A *in* $\mathscr{A}_{a,b}[S]$ **do**
10 $\quad\quad\quad \mathscr{T}[S \cup A] = \max\{\mathscr{T}[S \cup A], \mathscr{T}[S]+1\}$;
11 **for** S *in* $\mathbb{S}_\alpha$ **do**
12 $\quad$ **if** $V(G) \setminus S$ *is also in* $\mathbb{S}_\alpha$ **then**
13 $\quad\quad t = \max\{t, \mathscr{T}[S] + \mathscr{T}[V(G) \setminus S]\}$;
14 **return** t;

connected and $\delta(S \setminus B)$ is contained in B. This implies that set $N(S)$ is identical to $N(B) \setminus S$ or in other words, $\delta(S)$ is a subset of B. This together with fact that $G[B]$, $G[S \setminus B]$ are connected implies that B is in $\mathscr{Z}[S]$. □

We use Claim 8.5.2 to obtain improvement in running time while computing values in $\mathscr{T}$.

Lemma 8.5.3. *For a given connected graph G on n vertices and a positive constant α, Algorithm 8.5.2 returns an integer t such that G can be contracted to P_t and it terminates in time $\mathscr{O}^\star(c^n)$ where $c = 2^\alpha$.*

Proof. The correctness of the recurrence used in Step 10 of the algorithm is implied by Claim 8.5.1 and Claim 8.5.2. Condition $y+a+b \leq \alpha n$ ensures that we only consider sets S,A such that $|N[S \cup A]|$ is at most αn. Hence $\mathscr{T}[S \cup A]$ is well defined. If algorithm returns 2 then it is correct by Observation 8.2.1. If algorithm returns an integer which

is larger than 2 then there exists a set S such that sets S and $V(G) \setminus S$ are connected and their closed neighborhood is at most αn. Let $\{W_1, W_2, \ldots, W_j\}$ be a P_j-witness structure corresponding to value $\mathscr{T}[S]$ with $\delta(S) \subseteq W_j$ and $\{W'_1, W'_2, \ldots, W'_k\}$ is the witness structure corresponding to $\mathscr{T}[V(G) \setminus S]$ with $\delta(V(G) \setminus S)$ is contained in W'_1. It is easy to see that $\{W_1, \ldots, W_j, W'_1, \ldots, W'_k\}$ is $P_{j+k'}$-witness structure of G. Hence algorithm returns an integer whose value is more than two only if it had found a P_t-witness structure.

We now argue about the running time of this algorithm. Recall that $y = |S|$ and $x = |N(S)|$. At table entry corresponding to a set S, algorithm computes $\mathscr{A}_{a,b}[S]$ in time $\mathscr{O}^\star(2^{a+b-x})$ (by Lemma 8.3.1). The number of sets in $\mathscr{A}_{a,b}[S]$ is upper bounded by $\mathscr{O}^\star(2^{a+b-x})$. For each set A in $\mathscr{A}_{a,b}[S]$, algorithm updates the value of $\mathscr{T}[S \cup A]$ in polynomial time. Hence total time spent at each entry is at most $\mathscr{O}^\star(2^{a+b-x})$. By Lemma 8.3.2, the number of connected sets of size y whose neighborhood is of size x is upper bounded by $\mathscr{O}^\star(2^{y+x})$ and all of those can be enumerated in same time. This implies there $\mathscr{O}^\star(2^{y+x})$ entries where set S is of size y and $N(S)$ is of size x. Hence total time to compute the table is $\mathscr{O}^\star(2^{x+y} \cdot 2^{a+b-x}) = \mathscr{O}^\star(2^{y+a+b}) = \mathscr{O}^\star(2^{\alpha n})$. □

For a given P_t-witness structure $\mathscr{W} = \{W_1, W_2, \ldots W_\ell\}$ of a graph we define Q_i, R_i as : $Q_i = \bigcup_{x=1}^{i} W_x$ and $R_i = \bigcup_{x=i}^{\ell} W_x$ for all i in $[t]$. Note that (Q_i, R_{i+1}) is a partition of $V(G)$ for all i in $[t-1]$. Moreover, sets Q_i, R_i are connected for all i.

Definition 8.5.2 (α-Balanced Bi-partition)**.** *For a positive constant α, a P_t-witness structure $\mathscr{W} = \{W_1, W_2, \ldots, W_t\}$ is said to be α-balanced bi-partitioned if there exists an integer i in $[t-1]$ such that cardinality of sets Q_{i+1} and R_i are less than or equal to αn.*

We now specifies the types of graphs on which Algorithm 8.5.2 is effective.

Lemma 8.5.4. *For a given connected graph G and a positive constants α, if ℓ is the largest integer such that G can be contracted to P_ℓ and there exists a P_ℓ-witness structure of G which is α-balanced bi-partitioned then Algorithm 8.5.2 returns ℓ.*

Proof. Let $\mathscr{W} = \{W_1, W_2, \ldots, W_\ell\}$ be a α-balanced bi-partitioned P_ℓ-witness structure of G. By definition, there exists i in $[t-1]$ such that cardinality of Q_{i+1} and R_i are less than or equal to αn. Note that, $N[Q_i] \subseteq Q_{i+1}$ and $N[R_{i+1}] \subseteq R_i$. Since Algorithm 8.5.2 computes $\mathscr{T}$ value for all connected sets S whose closed neighborhood is at most αn, it computes values for Q_i and R_{i+1}. Moreover, $\mathscr{T}[Q_i]$ and $\mathscr{T}[R_{i+1}]$ is at least i and $\ell - i$, respectively. Hence in this case, the integer returned by Algorithm 8.5.2 is greater than or equal to ℓ. □

8.5.3 Method using an algorithm for 2-DISJOINT CONNECTED SUBGRAPHS

Recall that an input of 2-DISJOINT CONNECTED SUBGRAPHS (2-DCS) consists of a connected graph H and two disjoint terminal sets Z_1, Z_2. The task is to check whether is it possible to partition $V(H)$ into V_1, V_2 such that Z_1, Z_2 are contained in V_1, V_2, respectively, and both $H[V_1]$, $H[V_2]$ are connected. We use the algorithm presented by Telle and Villanger [90] as black-box in our algorithm.

Proposition 8.5.1 ([90] Theorem 3)**.** *There exists an algorithm that solves* 2-DISJOINT CONNECTED SUBGRAPHS *problem in* $\mathscr{O}^\star(1.7804^n)$ *time where n is number of vertices in input graph.*

Algorithm 8.5.3 divides input graph into three parts with middle one containing bulk of vertices. See Figure 8.3. Corner parts contains at most $(1-\gamma)n$ many vertices and hence algorithm can afford to guess it. For every such guess of corner parts, algorithm finds a suitable path contraction using method specified in Sub-Section 8.5.2. For the middle part, algorithm checks whether it can be partitioned into two connected subgraphs using Proposition 8.5.1.

Lemma 8.5.5. *For a given connected graph G on n vertices and a positive constant* γ*, Algorithm 8.5.3 returns an integer t such that G can be contracted to* P_t *and it terminates in time* $\mathscr{O}^\star(2^{(1-\gamma/2)n} + c^n)$ *where* $c = \max_{\gamma \leq \delta \leq 1} \left\{1.7804^\delta \cdot g(1-\delta)\right\}$.

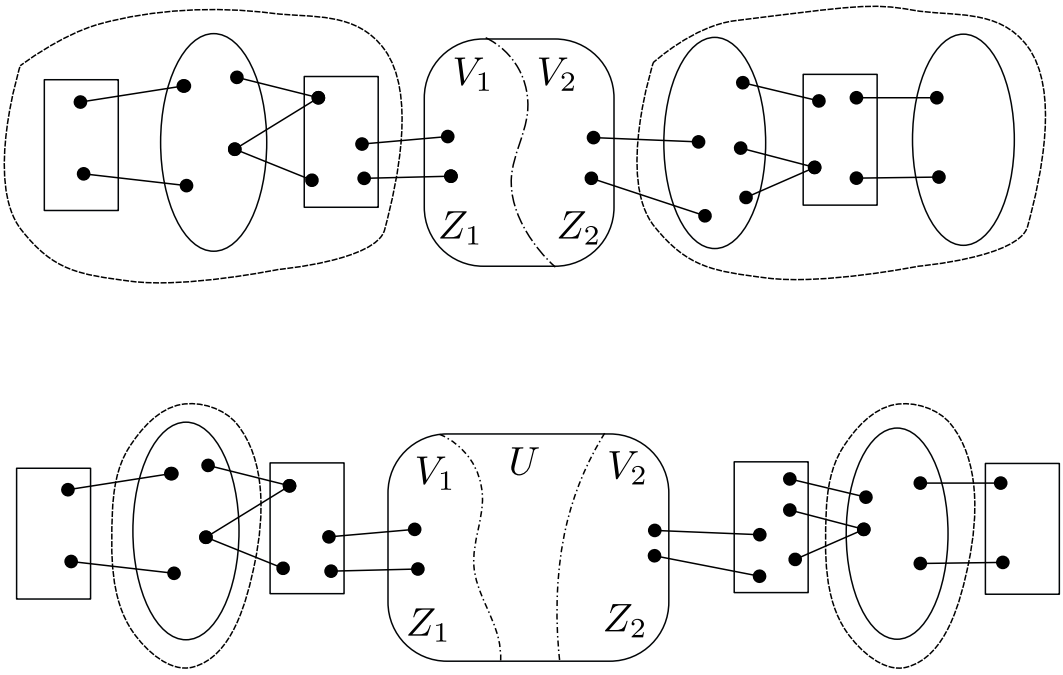


Figure 8.3: Guessing vertices in Methods described in Subsections 8.5.3 and 8.5.4. Dotted region denotes S, set of vertices guessed by algorithms. In Subsection 8.5.3, middle part into divided into two witness sets while in Subsection 8.5.4, we partition it into three witness sets.

Proof. The algorithm computes $\mathscr{T}[S]$ for all connected set S in G whose closed neighborhood is of size at most $(1-\gamma/2)n$. It enumerates all subsets of size at most $(1-\gamma)n$. Out of these sets, the algorithm considers set S which satisfies following four properties.

1. Graph $G-S$ is connected,
2. Graph $G[S]$ has exactly two connected components, say S_1, S_2,
3. Cardinality of closed neighborhoods of sets S_1, S_2 are at most $(1-\gamma/2)n$, and
4. Instance $((G-S); N(S_1) \cap C; N(S_2) \cap C)$ is a YES instance of 2-DCS.

Algorithm returns maximum of $\mathscr{T}[S_1] + \mathscr{T}[S_2] + 2$ over all sets S which satisfies above properties.

We argue the correctness of algorithm. If algorithm returns 2 then it is correct by Observation 8.2.1. If algorithm returns an integer which is larger than 2 then there exists a set S which satisfies above conditions. Let $\{W_1'', \ldots, W_j''\}$ be a P_j-witness structure correspond-

Algorithm 8.5.3: Solving PATH CONTRACTION using the algorithm for 2-DISJOINT CONNECT SUBGRAPH (2-DCS)

Input: Graph G and a positive constant γ
Output: An integer t such that G can be contracted to P_t

Initialize $t = 2$;
Run Algorithm 8.5.2 on input $(G, 1-\gamma/2)$ to compute table $\mathscr{T}$;
$\mathbb{C} \leftarrow$ all subsets of size at most $(1-\gamma)n$;
for S *in* $\mathbb{C}$ **do**
    $S_1, S_2 \leftarrow$ connected components of $G[S]$;
    `/* Continue if G[S] do not have two connected components */`
    `/* Continue if |N[S1]| or |N[S2]| are not at most (1-γ/2)n */`
    if $(G-(S_1 \cup S_2); N(S_1); N(S_2))$ *is a* YES *instance of* 2-DCS **then**
        $t = \max\{t, \mathscr{T}[S_1] + \mathscr{T}[S_2] + 2\}$
return t;

ing to $\mathscr{T}[S_1]$ such that $\delta(S_1)$ is a subset of W_j'' and $\{W_1', \ldots, W_k'\}$ be a P_k-witness structure corresponding to $\mathscr{T}[S_2]$ such that $\delta(S_2)$ is a subset of W_1'. Let (V_1, V_2) be a partition of $G - S$ such that terminal sets $N(S_1), N(S_2)$ are contained in V_1, V_2, respectively, and both V_1, V_2 are connected sets in $G - S$. It is easy to see that $\{W_1'', \ldots, W_j'', V_1, V_2, W_1', \ldots, W_k'\}$ is a P_{j+k+2}-witness structure of graph G.

By Lemma 8.5.3, the algorithm can compute the table in time $\mathscr{O}^\star(2^{(1-\gamma/2)n})$. For every set S, conditions $(1), (2)$ and (3) can be checked in polynomial time and condition (4) can be checked in time $\mathscr{O}^\star(1.7804^{n-|S|})$ using Proposition 8.5.1. By Observation 8.2.4, the number of sets of cardinality at most $(1-\delta)n$ are upper bounded by $\mathscr{O}^\star([g(1-\delta)]^n)$ which can be enumerated in same time. Hence the running time of algorithm is $\mathscr{O}^\star(2^{(1-\gamma/2)n} + c^n)$ where $c = \max_{\gamma \leq \delta \leq 1} \left\{1.7804^\delta \cdot g(1-\delta)\right\}$. □

We present a lemma which specifies the types of graphs on which Algorithm 8.5.3 is effective. To specify such graph, we need to define certain kind of witness structure.

Definition 8.5.3 (γ-Bi-large)**.** *For a positive constant γ, a P_t-witness structure $\mathscr{W} = \{W_1, W_2, \ldots, W_t\}$ is said to be γ-bi-large if there exists an integer i in $[t-1]$ such that cardinality of sets W_i and W_{i+1} are greater than or equal to $\gamma n/2$.*

Before stating the lemma, we recall that for a given P_t-witness structure $\mathscr{W}$ we defined Q_i, R_i as : $Q_i = \bigcup_{x=1}^{i} W_x$ and $R_i = \bigcup_{x=i}^{\ell} W_x$ for all i in $[t]$.

Lemma 8.5.6. *For a given connected graph G and a positive constant γ, if ℓ is the largest integer such that G can be contracted to P_ℓ and there exists a P_ℓ-witness structure of G which is γ-bi-large then Algorithm 8.5.3 returns ℓ.*

Proof. Let $\mathscr{W} = \{W_1, W_2, \ldots, W_\ell\}$ be a γ-bi-large P_ℓ-witness structure of G. Consider two adjacent witness sets W_i, W_{i+1} such that $|W_i|, |W_{i+1}|$ are greater than or equal to $\gamma n/2$. To prove that the algorithm returns the optimum value, we argue that $Q_{i-1} \cup R_{i+2}$ is one of the sets considered by algorithm while updating value of t. Since $(Q_{i-1}, W_i, W_{i+1}, R_{i+2})$ is a partition of $V(G)$, we have $|Q_{i-1}| + |W_i| + |W_{i+1}| + |R_{i+2}| = n$ which implies $|Q_{i-1} \cup R_{i+1}| \leq (1-\gamma)n$. Hence $Q_{i-1} \cup R_{i+j}$ is one of sets considered while enumerating all vertex sets of size at most $(1-\gamma)n$. We argue that it satisfies all four conditions mentioned in proof of Lemma 8.5.5. By the properties of P_ℓ-witness structure, set $Q_{i-1} \cup R_{i+1}$ satisfies conditions (1) and (2). To see that it satisfies (3), notice that $N[Q_{i-1}]$ does not intersects with W_{i+1} and hence cardinality of $N[Q_{i-1}]$ is upper bounded by $n - |W_{i+1}| \leq (1-\gamma/2)n$. By similar argument, size of $N[R_{i+1}]$ is at most $(1-\gamma/2)n$. By the property of P_ℓ-witness structure, $N(Q_{i-1}), N(R_{i+2})$ are two disjoint vertex sets. Let G' is the subgraph of G induced on $W_i \cup W_{i+1}$. Notice that $(G', N(Q_{i-1}), N(R_{j+1}))$ is a YES instance of 2-DCS as $V(G')$ can be partitioned into W_i, W_{i+1} such that these sets contains $N(Q_{i-1})$ and $N(R_{j+2})$ respectively and both are connected sets in graph G'. This implies that Algorithm 8.5.3 returns an integer which is greater than or equal to ℓ. □

8.5.4 Method using an algorithm for 3-DISJOINT CONNECTED SUBGRAPH

In this sub-section, we present a method used to solve PATH CONTRACTION using the algorithm for 3-DCS presented in Section 8.4. In the method mentioned in Sub-section 8.5.3,

Algorithm 8.5.4: Solving PATH CONTRACTION using the algorithm for 3-DISJOINT CONNECT SUBGRAPH (3-DCS)

Input: Connected graph G and a positive constant ε
Output: An integer t such that G can be contracted to P_t

Initialize $t = 2$;
$\mathbb{C} \leftarrow$ all subsets of size at most εn;
for S *in* $\mathbb{C}$ **do**
    $\mathscr{W} \leftarrow$ witness structure obtained by consider each connected component of $G[S]$ and $G - S$ as a witness set;
    $G' \leftarrow$ graph obtained from G by contracting witness sets in $\mathscr{W}$;
    if G' *is* not *a path* **then**
        Continue with next set in $\mathbb{C}$;
    for C *in connected component of* $G - S$ **do**
        $C_1, C_2 \leftarrow$ Connected components of $G[S]$ which are adjacent with C;
        if $(G[C]; N(C_1) \cap C; N(C_2) \cap C)$ *is a* YES *instance of* 3-DCS **then**
            $t = \max\{t,$ length of path $G' + 2\}$
return t;

algorithm divides input graph into three parts by guessing all vertices in corner parts. Algorithm then checks whether middle part can be partitioned into two bags and corner bags can be contracted to a path using Algorithm 8.5.2. In this method, instead of guessing *all* vertices in corner parts, algorithm guesses *some* vertices in corner parts which partition it different witness sets. See Figure 8.3. We consider guess S for which connected components of $G[S]$ and $G - S$ corresponds to a witness structure corresponding to a path. For each connected component of $G - S$, we check whether it can be tri-partitioned to get two more witness sets. Following lemma asserts the correctness of Algorithm 8.5.4.

Lemma 8.5.7. *For a given connected graph G on n vertices and a positive constant ε, Algorithm 8.5.4 returns an integer t such that G can be contracted to P_t and it terminates in time $\mathcal{O}^\star(c^n)$ where $c = \max_{0 \leq \delta \leq \varepsilon} \left\{1.877^{(1-\delta)} \cdot g(\delta)\right\}$.*

Proof. If algorithm returns 2 then it is correct by Observation 8.2.1. If algorithm returns an integer which is greater than 2 then there exists a set S which satisfies following conditions: Graph obtained by contracting connected components of $G[S]$ and $G - S$ to vertices is a path. There exists connected components C of $G - S$, and C_1, C_2 of $G[S]$ such that C is ad-

jacent with C_1, C_2 and instance $(G[C]; N(C_1) \cap C; N(C_2) \cap C)$ is a YES instance of 3-DCS. Let (V_1, U, V_2) be a partition of $V(C)$ such that vertices of $N(C_1) \cap C$ and $N(C_2) \cap C$ are contained in V_1, V_2 respectively, sets V_1, U, V_2 are connected sets in $G[C]$ and V_1, V_2 are the only two connected components of $G[C] - U$. If $\{W_1, \ldots, W_{j-1}, C_1, C, C_2, W'_1, \ldots, W'_{k-1}\}$ is a P_{j+k+1}-witness structure of a graph G then $\{W_1, \ldots, W_{j-1}, C_1, V_1, U, V_2, C_2, W'_1, \ldots, W'_{k-1}\}$ is a P_{j+k+3}-witness structure of G. Hence, the algorithm returns an integer greater than 2 only if it has found a witness structure of path of that length.

We now argue the running time of the algorithm. By Observation 8.2.4, the number of sets of cardinality at most εn are upper bounded by $\mathcal{O}^\star([g(\varepsilon)]^n)$ which can be enumerated in same time. For every set S, conditions mentioned in previous paragraph can be checked in time $\mathcal{O}^\star(1.877^{n-|S|})$ using Theorem 8.4.1. Hence the running time algorithm is $\mathcal{O}^\star(c^n)$ where $c = \max_{0 \leq \delta \leq \varepsilon} \left\{1.877^{(1-\delta)} \cdot g(\delta)\right\}$. □

We define a type of witness structure used to specify graphs for which this method is effective.

Definition 8.5.4 (ε-Partition Concentrated)**.** *For a positive constant ε, a P_t-witness structure $\mathcal{W} = \{W_1, W_2, \ldots, W_t\}$ is said to be ε-partition concentrated if there exists an integer i in $\{2, \ldots, t-1\}$ such that cardinality of either $\mathtt{OS} \setminus W_i$ or $\mathtt{ES} \setminus W_i$ is at most εn (depending on whether i is odd or even integer).*

In above definition, we insist that witness set W_i is does not corresponds with end point of P_t to avoid dealing with corner cases in following lemma. In Subsection 8.5.5, we argue that these corner cases do not occur.

Lemma 8.5.8. *For a given connected graph G and a positive constant ε, if ℓ is the largest integer such that G can be contracted to P_ℓ and there exists a P_ℓ-witness structure of G which is ε-partition concentrated then Algorithm 8.5.4 returns ℓ.*

Proof. Let $\mathcal{W} = \{W_1, W_2, \ldots, W_\ell\}$ be a P_ℓ-witness structure of G which is ε-partition concentrated. Without loss of generality, we can assume that OS is a concentrated partition.

In other words, there exists an odd integer i in $\{2,\ldots,t-1\}$ such that $|\texttt{OS} \setminus W_i|$ is upper bounded by εn. We argue that $\texttt{OS} \setminus W_i$ is one of sets considered by algorithm while updating the value of t. Note that connected components of $G[\texttt{OS} \setminus W_i]$ and $G-(\texttt{OS} \setminus W_i)$ are $W_1,\ldots,W_{i-2},C,W_{i+2},\ldots,W_\ell$ where $C = W_{i-1} \cup W_i \cup W_{i+1}$. Together these sets forms a $P_{\ell-2}$-witness structure of G as $\mathscr{W}$ is a P_ℓ-witness structure of G. Moreover, $(G[C]; N(W_{i-2}) \cap C; N(W_{i+2}) \cap C)$ is a YES instance of 3-DCS as C can be partitioned into (W_{i-1}, W_i, W_{i+1}) which satisfies the desired properties. Hence the value of t has been updated to ℓ when considering set $\texttt{OS} \setminus W_i$ by the algorithm. This implies that Algorithm 8.5.4 returns an integer which is greater than or equal to ℓ. □

8.5.5 Algorithm for PATH CONTRACTION

We fix values of α, β, γ such that they satisfies following inequalities: (1) $2+\gamma/2-\beta/2 \leq 2\alpha$ (used in Case 1); (2) $1-\gamma \leq \beta/2$ (used in Case 3); (3) $1-\gamma/2 \leq \alpha$ (used in Case 3). Following theorem is the main result of this paper.

Theorem 8.5.1. *There exists an algorithm that solves* PATH CONTRACTION *problem in* $\mathscr{O}^\star(1.99987^n)$ *time where n is the number of vertices in an input graph.*

Proof. Let G be input graph. Fix $\alpha = 0.9996; \beta = 0.9885; \gamma = 0.9864$. Main algorithm runs Algorithm 8.5.1, Algorithm 8.5.2, Algorithm 8.5.3, Algorithm 8.5.4 with input (G,β), (G,α), (G,γ) and $(G, 1-\beta/2-\gamma/2)$, respectively. It returns the maximum among values obtained by these four algorithms. By Lemma 8.5.1, Lemma 8.5.3, Lemma 8.5.5 and Lemma 8.5.7 the running time for these algorithms for specified values of α, β, γ are $\mathscr{O}^\star(1.99987^n)$, $\mathscr{O}^\star(1.9994^n)$, $\mathscr{O}^\star(1.8983^n)$ and $\mathscr{O}^\star(1.9921^n)$ respectively. These lemmas also implies that if algorithm returns an integer t then input graph can be contracted to P_t. To argue the correctness of main algorithm, it remains to argue that if ℓ is the largest integer such that G can be contracted to P_ℓ than there exists a P_ℓ-witness structure of G which satisfies premises of either one of Lemma 8.5.2, Lemma 8.5.4, Lemma 8.5.6, or

Lemma 8.5.8.

Recall that for a P_t-witness structure $\{W_1, W_2, \ldots, W_t\}$ of a graph we have defined set ES (resp. OS) as collection of vertices in G which are present in even (resp. odd) numbered witness sets. For all i in $[t]$, set Q_i (resp. R_i) is union all witness sets indexed less than or equal (resp. greater than or equal) to i. Formally these sets are defined as follows.

$$\texttt{OS} = \bigcup_{x=0}^{\lfloor t/2 \rfloor} W_{2x+1} \; ; \; \texttt{ES} = \bigcup_{x=1}^{\lfloor t/2 \rfloor} W_{2x} \; ; \; Q_i = \bigcup_{x=1}^{i} W_x \; ; \; R_i = \bigcup_{x=i}^{t} W_x$$

It is clear from the definition that $N(Q_i)$ (resp. $N(R_i)$) is contained in Q_{i+1} (resp. $N(R_{i-1})$). We use this observation frequently in the remaining proof.

If there exists a P_ℓ-witness structure of G which is not β-equally partitioned (Definition 8.5.1) then premise of Lemma 8.5.2 is satisfied and hence the algorithm returns optimum value. For rest of the proof we assume that *all P_ℓ-witness structures of G are β-equally partitioned*. In other words, for any P_ℓ-witness structure, cardinalities of both sets OS and ES are strictly greater than $\beta n/2$. This lower bound also implies upper bound of $(1-\beta/2)n$ on cardinalities both these sets. Any P_ℓ-witness structure of G contains at most two witness sets of size greater than or equal to $\gamma n/2$. We consider three cases based on existence of witness structure containing certain number of witness sets of size greater than or equal to $\gamma n/2$.

Case 1: There exists a P_ℓ-witness structure, say $\mathscr{W} = \{W_1, W_2, \ldots, W_\ell\}$, which contains no witness set of size greater than or equal to $\gamma n/2$.

We prove that $\mathscr{W}$ is α-balanced bi-partitioned (Definition 8.5.2) and hence premise of Lemma 8.5.4 is satisfied and main algorithm returns optimum value.

Let j be the largest integer such that the cardinality of Q_j is at most αn. By Observation 8.2.2, integer j is not equal to 1 or ℓ. As j is largest such integer, cardinality of Q_{j+1} is strictly greater than αn and hence $|Q_j| + |W_{j+1}| > \alpha n$ which can be written as $|Q_j| > \alpha n -$

$|W_{j+1}|$. As (Q_j, R_{j+1}) is a partition of $V(G)$, we have $|Q_j| + |R_{j+1}| = n$. This implies cardinality of R_{j+1} is strictly less than $n - \alpha n + |W_{j+1}|$. We use this to obtain upper bound on cardinality of R_{j-1}. By definition, $|R_{j-1}| = |W_{j-1}| + |W_j| + |R_{j+1}|$ and hence cardinality of R_{j-1} is strictly less than $|W_{j-1}| + |W_j| + n - \alpha n + |W_{j+1}| = n - \alpha n + |W_{j-1}| + |W_j| + |W_{j+1}|$. Since $j-1, j+1$ has same parity and are disjoint with each other, $|W_{j-1}| + |W_{j+1}|$ is at most $(1 - \beta/2)n$. There is no witness set of size strictly greater than $\gamma n/2$ and hence cardinality of W_j is at most $\gamma n/2$. Plugging these upper bounds we get that cardinality of R_{j-1} is upper bounded by $n - \alpha n + (1 - \beta/2)n + \gamma/2n = (2 - \alpha - \beta/2 + \gamma/2)n \leq \alpha n$ (by Equation 1). This implies cardinalities of set Q_j and R_{j-1} are upper bounded by αn and hence $\mathscr{W}$ is a α-balanced bi-partitioned witness structure.

Case 2: There exists a P_ℓ-witness structure, say $\mathscr{W} = \{W_1, W_2, \ldots, W_\ell\}$, which contains exactly one witness set of size greater than or equal to $\gamma n/2$.

We prove that either $\mathscr{W}$ is α-balanced bi-partitioned or it is $(1 - \beta/2 - \gamma/2)$-partition concentrated (Definition 8.5.4) witness structure. In first case, proof of correctness is similar to that of previous case. In second case, premise of Lemma 8.5.8 is satisfied and hence the algorithm returns optimum value.

Let W_k be the unique witness whose cardinality is strictly greater than $\gamma n/2$. Let j be the largest integer such that cardinality of Q_j is at most αn. We know that, as in previous case, $|W_{j-1}| + |W_{j+1}|$ is at most $(1 - \beta/2)n$. If $j \neq k$ then upper bound on cardinality of W_j is still valid and arguments are similar as in previous case. We now consider a case when $j = k$. Without loss of generality, assume that k is an odd integer. Since cardinality of $\mathtt{OS}$ is upper bounded by $(1 - \beta/2)n$ and that of W_j is lower bounded by $\gamma n/2$, we have $|\mathtt{OS} \setminus W_j| \leq (1 - \beta/2 - \gamma/2)n$. By Observation 8.2.2, we know k is not equal to 1 or ℓ which implies $\mathscr{W}$ is a $(1 - \beta/2 - \gamma/2)$-partition concentrated set.

Case 3: There exists a P_ℓ-witness structure, say $\mathscr{W} = \{W_1, W_2, \ldots, W_\ell\}$, which contains exactly two witness sets of size greater than or equal to $\gamma n/2$.

We first prove that these two witness sets are of different parity. If these two large witness sets are adjacent then $\mathscr{W}$ is γ-bi-large witness structure (Definition 8.5.3). If they are not adjacent then we argue that $\mathscr{W}$ is a α-balanced bi-partitioned. In first case, premise of Lemma 8.5.6 is satisfied and algorithm returns optimum value. In later case, proof of correctness is similar to that of Case 1.

Let W_j, W_k be two witness sets whose cardinality is strictly greater than $\gamma n/2$. Without loss of generality, assume that $j < k$. Since W_j, W_k are disjoint, if j,k has same parity then $\gamma n < |W_j \cup W_k| \leq (1-\beta/2)n$ which implies $\gamma < 1-\beta/2$ contradicting Equation 2. This implies j,k are of different parity. If $k = j+1$ then two large witness sets are adjacent and $\mathscr{W}$ is γ-bi-large witness structure. We now handle the case when $k \geq j+3$. We argue that cardinalities of both Q_{j+2} and R_{j+1} are bounded above by αn. Since $k \geq j+3$, set W_k does not intersect with Q_{j+2}. This implies that the cardinality of Q_{j+2} is at most $n - \gamma n/2 \leq \alpha n$ (by Equation 3). By symmetric argument, cardinality of R_{j+1} is upper bounded by αn. This implies that $\mathscr{W}$ is a α-balanced bi-partitioned witness structure.

Since $\gamma = 0.9864$, no P_ℓ-witness structure can have more than two witness sets of size $\gamma/2$. Hence above three cases are exhaustive. This completes the proof of the theorem. □

8.6 Conclusion

In this chapter we presented an algorithm which given a graph G and a connected set Q, enumerates all connected supersets of Q which are of size at most a and has at most b neighbors. We generalized 2-DISJOINT CONNECTED SUBGRAPHS problem and gave an exact exponential algorithm to solve it. We use both these techniques, along with others, to give an algorithm for PATH CONTRACTION which breaks $\mathcal{O}^\star(2^n)$ barrier.

Bibliography

[1] Akanksha Agrawal, Lawqueen Kanesh, Saket Saurabh, and Prafullkumar Tale. Paths to trees and cacti. In *10th International Conference on Algorithms and Complexity (CIAC 2017)*, pages 31–42, 2017.

[2] Akanksha Agrawal, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. Split contraction: The untold story. In *STACS*, volume 66 of *LIPIcs*, pages 5:1–5:14, 2017.

[3] Akanksha Agrawal, Saket Saurabh, and Prafullkumar Tale. On the parameterized complexity of contraction to generalization of trees. In *12th International Symposium on Parameterized and Exact Computation, IPEC 2017*, pages 1:1–1:12, 2017.

[4] Noga Alon, Andrzej Lingas, and Martin Wahlen. Approximating the maximum clique minor and some subgraph homeomorphism problems. *Theoretical Computer Science*, 374(1-3):149–158, 2007.

[5] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995.

[6] Takao Asano and Tomio Hirata. Edge-Contraction Problems. *Journal of Computer and System Sciences*, 26(2):197–208, 1983.

[7] Rémy Belmonte, Petr A Golovach, Pinar Heggernes, Pim van't Hof, Marcin Kamiński, and Daniël Paulusma. Finding contractions and induced minors in chordal graphs via disjoint paths. In *International Symposium on Algorithms and Computation*, pages 110–119. Springer, 2011.

[8] Rémy Belmonte, Petr A. Golovach, Pim van't Hof, and Daniël Paulusma. Parameterized complexity of three edge contraction problems with degree constraints. *Acta Informatica*, 51(7):473–497, 2014.

[9] Rémy Belmonte, Pinar Heggernes, and Pim van't Hof. Edge contractions in subclasses of chordal graphs. In *International Conference on Theory and Applications of Models of Computation*, pages 528–539. Springer, 2011.

[10] Ivan Bliznets, Fedor V. Fomin, Marcin Pilipczuk, and Michal Pilipczuk. A subexponential parameterized algorithm for proper interval completion. In *ESA*, pages 173–184, 2014.

[11] Ivan Bliznets, Fedor V. Fomin, Marcin Pilipczuk, and Michal Pilipczuk. A subexponential parameterized algorithm for interval completion. In *SODA*, pages 1116–1131, 2016.

[12] Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009.

[13] Hans L Bodlaender, Stéphan Thomassé, and Anders Yeo. Kernel bounds for disjoint cycles and disjoint paths. *Theoretical Computer Science*, 412(35):4570–4578, 2011.

[14] Andries Evert Brouwer and Hendrik Jan Veldman. Contractibility and NP-completeness. *Journal of Graph Theory*, 11(1):71–79, 1987.

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

[16] Leizhen Cai and Chengwei Guo. Contracting few edges to remove forbidden induced subgraphs. In *Parameterized and Exact Computation: 8th International Symposium, IPEC*, pages 97–109. Springer International Publishing, 2013.

[17] Yixin Cao. Unit interval editing is fixed-parameter tractable. In *ICALP*, pages 306–317, 2015.

[18] Yixin Cao. Linear recognition of almost interval graphs. In *SODA*, pages 1096–1115, 2016.

[19] Yixin Cao, Jianer Chen, and Yang Liu. On feedback vertex set: New measure and new structures. *Algorithmica*, 73(1):63–86, 2015.

[20] Yixin Cao and Dániel Marx. Chordal editing is fixed-parameter tractable. In *STACS*, pages 214–225, 2014.

[21] Yixin Cao and Dániel Marx. Interval deletion is fixed-parameter tractable. *ACM Transactions on Algorithms (TALG)*, 11(3):21, 2015.

[22] Jianer Chen, Iyad A Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 411(40-42):3736–3756, 2010.

[23] Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990.

[24] Marek Cygan. Deterministic parameterized connected vertex cover. In *Scandinavian Workshop on Algorithm Theory*, pages 95–106. Springer, 2012.

[25] Marek Cygan, Fedor V Fomin, $\mathscr{L}$ukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*, volume 4. Springer, 2015.

[26] Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. Solving the 2-disjoint connected subgraphs problem faster than 2^n. *Algorithmica*, 70(2):195–207, 2014.

[27] Konrad K Dabrowski and Daniël Paulusma. Contracting bipartite graphs to paths and cycles. *Information Processing Letters*, 127:37–42, 2017.

[28] Holger Dell and Dieter van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *Journal of the ACM*, 61(4):23:1–23:27, 2014.

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

[30] Reinhard Diestel, Tommy R Jensen, Konstantin Yu Gorbunov, and Carsten Thomassen. Highly connected sets and the excluded grid theorem. *Journal of Combinatorial Theory, Series B*, 75(1):61–73, 1999.

[31] Öznur Yaşar Diner, Daniël Paulusma, Christophe Picouleau, and Bernard Ries. Contraction blockers for graphs with forbidden induced paths. In *International Conference on Algorithms and Complexity*, pages 194–207. Springer, 2015.

[32] Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Kernelization lower bounds through colors and IDs. *ACM Transactions on Algorithms (TALG)*, 11(2):13, 2014.

[33] Rodney G Downey and Michael Ralph Fellows. *Parameterized complexity*. Springer Science & Business Media, 2012.

[34] P G Drange and M Pilipczuk. A polynomial kernel for trivially perfect editing. In *ESA*, pages 424–436, 2015.

[35] Pål Grønås Drange, Markus Sortland Dregi, Daniel Lokshtanov, and Blair D Sullivan. On the threshold of intractability. In *Algorithms-ESA 2015*, pages 411–423. Springer, 2015.

[36] Pål Grønås Drange, Fedor V Fomin, Michał Pilipczuk, and Yngve Villanger. Exploring Subexponential Parameterized Complexity of Completion Problems. In *31st International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 25 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 288–299. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014.

[37] David Eppstein. Finding large clique minors is hard. *J. Graph Algorithms Appl.*, 13(2):197–204, 2009.

[38] Jiří Fiala, Marcin Kamiński, and Daniël Paulusma. A note on contracting claw-free graphs. *Discrete Mathematics and Theoretical Computer Science*, 15(2):223–232, 2013.

[39] Jörg Flum and Martin Grohe. Parameterized complexity theory, 2006.

[40] Fedor V. Fomin, Stefan Kratsch, Marcin Pilipczuk, Michał Pilipczuk, and Yngve Villanger. Tight bounds for parameterized complexity of cluster editing with a small number of clusters. *Journal of Computer and System Sciences*, 80(7):1430–1447, 2014.

[41] Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar F-deletion: Approximation, kernelization and optimal FPT algorithms. In *FOCS*, 2012.

[42] Fedor V. Fomin and Yngve Villanger. Subexponential parameterized algorithm for minimum fill-in. *SIAM Journal on Computing*, 42(6):2197–2216, 2013.

[43] Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *Journal of Computer and System Sciences*, 77(1):91–106, 2011.

[44] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H.Freeman and Company, 1979.

[45] Esha Ghosh, Sudeshna Kolay, Mrinal Kumar, Pranabendu Misra, Fahad Panolan, Ashutosh Rai, and M. S. Ramanujan. Faster parameterized algorithms for deletion to split graphs. *Algorithmica*, 71(4):989–1006, 2015.

[46] Esha Ghosh, Sudeshna Kolay, Mrinal Kumar, Pranabendu Misra, Fahad Panolan, Ashutosh Rai, and MS Ramanujan. Faster parameterized algorithms for deletion to split graphs. *Algorithmica*, 71(4):989–1006, 2015.

[47] Petr A Golovach, Pinar Heggernes, Pim van't Hof, and Christophe Paul. Hadwiger number of graphs with small chordality. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 201–213. Springer, 2014.

[48] Petr A Golovach, Marcin Kamiński, and Daniël Paulusma. Contracting a chordal graph to a split graph or a tree. In *International Symposium on Mathematical Foundations of Computer Science*, pages 339–350. Springer, 2011.

[49] Petr A Golovach, Marcin Kamiński, Daniël Paulusma, and Dimitrios M Thilikos. Increasing the minimum degree of a graph by contractions. *Theoretical computer science*, 481:74–84, 2013.

[50] Petr A Golovach, Pim van't Hof, and Daniël Paulusma. Obtaining planarity by contracting few edges. *Theoretical Computer Science*, 476:38–46, 2013.

[51] Sylvain Guillemot and Dániel Marx. A faster FPT algorithm for bipartite contraction. *Information Processing Letters*, 113(22-24):906–912, 2013.

[52] Chengwei Guo and Leizhen Cai. Obtaining split graphs by edge contraction. *Theoretical Computer Science*, 607:60–67, 2015.

[53] Pinar Heggernes, van't Hof, Benjamin Lévêque, and Christophe Paul. Contracting chordal graphs and bipartite graphs to paths and trees. *Discrete Applied Mathematics*, 164:444–449, 2014.

[54] Pinar Heggernes, Pim van't Hof, Bart M. P. Jansen, Stefan Kratsch, and Yngve Villanger. Parameterized complexity of vertex deletion into perfect graph classes. In *FCT*, pages 240–251, 2011.

[55] Pinar Heggernes, Pim van't Hof, Benjamin Lévêque, Daniel Lokshtanov, and Christophe Paul. Contracting graphs to paths and trees. *Algorithmica*, 68(1):109–132, 2014.

[56] Pinar Heggernes, Pim van't Hof, Daniel Lokshtanov, and Christophe Paul. Obtaining a bipartite graph by contracting few edges. *SIAM Journal on Discrete Mathematics*, 27(4):2143–2156, 2013.

[57] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[58] Yoichi Iwata. Linear-time kernelization for feedback vertex set. In *44th International Colloquium on Automata, Languages, and Programming, ICALP*, pages 68:1–68:14, 2017.

[59] Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. A near-optimal planarization algorithm. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1802–1811, 2014.

[60] Bart M. P. Jansen and Astrid Pieterse. Sparsification upper and lower bounds for graphs problems and not-all-equal SAT. In *10th International Symposium on Parameterized and Exact Computation, IPEC*, pages 163–174, 2015.

[61] Marcin Kamiński, Daniël Paulusma, and Dimitrios M Thilikos. Contractions of planar graphs in polynomial time. In *European Symposium on Algorithms*, pages 122–133. Springer, 2010.

[62] Marcin Kaminski and Dimitrios M Thilikos. Contraction checking in graphs on surfaces. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 14. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2012.

[63] Ken-ichi Kawarabayashi. Planarity allowing few error vertices in linear time. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 639–648. IEEE, 2009.

[64] Walter Kern and Daniel Paulusma. Contracting to a longest path in H-free graphs. *arXiv preprint arXiv:1810.01542*, 2018.

[65] Eun Jung Kim, Alexander Langer, Christophe Paul, Felix Reidl, Peter Rossmanith, Ignasi Sau, and Somnath Sikdar. Linear kernels and single-exponential algorithms via protrusion decompositions. In *Automata, Languages, and Programming - 40th International Colloquium, ICALP*, pages 613–624, 2013.

[66] Tomasz Kociumaka and Marcin Pilipczuk. Faster deterministic feedback vertex set. *Information Processing Letters*, 114(10):556–560, 2014.

[67] Sudeshna Kolay, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Quick but Odd Growth of Cacti. In *10th International Symposium on Parameterized and Exact Computation, IPEC*, pages 258–269, 2015.

[68] Stefan Kratsch. *Kernelization, Preprocessing for Treewidth*. Encyclopedia of Algorithms, 2016.

[69] R. Krithika, Pranabendu Misra, Ashutosh Rai, and Prafullkumar Tale. Lossy kernels for graph contraction problems. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016*, pages 23:1–23:14, 2016.

[70] R Krithika, Pranabendu Misra, and Prafullkumar Tale. An FPT algorithm for contraction to cactus. In *International Computing and Combinatorics Conference*, pages 341–352. Springer, 2018.

[71] Asaf Levin, Daniel Paulusma, and Gerhard J Woeginger. The computational complexity of graph contractions i: Polynomially solvable and np-complete cases. *Networks: An International Journal*, 51(3):178–189, 2008.

[72] Wenjun Li, Qilong Feng, Jianer Chen, and Shuai Hu. Improved kernel results for some FPT problems based on simple observations. *Theor. Comput. Sci.*, 657:20–27, 2017.

[73] Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. On the hardness of eliminating small induced subgraphs by contracting edges. In *IPEC*, pages 243–254, 2013.

[74] Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. On the hardness of eliminating small induced subgraphs by contracting edges. In *International Symposium on Parameterized and Exact Computation*, pages 243–254. Springer, 2013.

[75] Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 224–237, 2017.

[76] Barnaby Martin and Daniël Paulusma. The computational complexity of disconnected cut and 2k2-partition. *Journal of combinatorial theory, series B*, 111:17–37, 2015.

[77] Dániel Marx. Parameterized graph separation problems. *Theoretical Computer Science*, 351(3):394–406, 2006.

[78] Dániel Marx. Chordal deletion is fixed-parameter tractable. *Algorithmica*, 57(4):747–768, 2010.

[79] Dániel Marx, Barry O'sullivan, and Igor Razgon. Finding small separators in linear time via treewidth reduction. *ACM Transactions on Algorithms (TALG)*, 9(4):30, 2013.

[80] Dániel Marx and Ildikó Schlotter. Obtaining a planar graph by vertex deletion. *Algorithmica*, 62(3-4):807–822, 2012.

[81] Moni Naor, Leonard J Schulman, and Aravind Srinivasan. Splitters and near-optimal derandomization. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 182–191. IEEE, 1995.

[82] Jesper Nederlof. Fast polynomial-space algorithms using inclusion-exclusion. *Algorithmica*, 65(4):868–884, 2013.

[83] Rolf Niedermeier. *Invitation to fixed-parameter algorithms*. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.

[84] Daniël Paulusma, Christophe Picouleau, and Bernard Ries. Reducing the clique and chromatic number via edge contractions and vertex deletions. In *International Symposium on Combinatorial Optimization*, pages 38–49. Springer, 2016.

[85] Pim Pim van't Hof, Daniël Paulusma, and Gerhard J Woeginger. Partitioning graphs into connected parts. *Theoretical Computer Science*, 410(47-49):4834–4843, 2009.

[86] Bruce Reed, Kaleigh Smith, and Adrian Vetta. Finding odd cycle transversals. *Operations Research Letters*, 32(4):299–301, 2004.

[87] Neil Robertson and Paul Seymour. Graph minors. xxii. irrelevant vertices in linkage problems. *Journal of Combinatorial Theory, Series B*, 102(2):530–563, 2012.

[88] Neil Robertson and Paul D Seymour. Graph minors. XIII. the disjoint paths problem. *Journal of combinatorial theory, Series B*, 63(1):65–110, 1995.

[89] Carla Savage. Depth-first search and the vertex cover problem. *Information Processing Letters*, 14(5):233–235, 1982.

[90] Jan Arne Telle and Yngve Villanger. Connecting terminals and 2-disjoint connected subgraphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 418–428. Springer, 2013.

[91] Stéphan Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Transactions on Algorithms*, 6(2), 2010.

[92] Pim van't Hof, Marcin Kamiński, Daniël Paulusma, Stefan Szeider, and Dimitrios M Thilikos. On contracting graphs to fixed pattern graphs. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 503–514. Springer, 2010.

[93] Toshimasa Watanabe, Tadashi Ae, and Akira Nakamura. On the removal of forbidden graphs by edge-deletion or by edge-contraction. *Discrete Applied Mathematics*, 3(2):151–153, 1981.

[94] Toshimasa Watanabe, Tadashi Ae, and Akira Nakamura. On the NP-hardness of edge-deletion and contraction problems. *Discrete Applied Mathematics*, 6(1):63–78, 1983.