ABOVE AND BELOW GUARANTEE PARAMETERIZATIONS FOR COMBINATORIAL OPTIMISATION PROBLEMS

A thesis submitted to Royal Holloway, University of London for the degree of Doctor of Philosophy in the Faculty of Science

Mark Jones
Department of Computer Science
February 2013

Declaration

I Mark Jones hereby declare that this thesis and the work presented in it is entirely my own. Where the work is a result of collaborative research, or the work of others has been used, this is clearly stated.

Abstract

Parameterized complexity can be viewed as a two-dimensional approach to traditional complexity theory. Rather than measuring the complexity of a problem purely in terms of the input size, we also consider some structural parameter of the input instance, and measure how the complexity of the problem changes as this parameter changes.

The analogue of polynomial time tractability is *fixed-parameter tractability*, in which the problem can be solved efficiently as long as the value of the parameter remains small. It turns out that for many practical applications of traditionally NP-complete problems, the problem is fixed-parameter tractable for some choice of parameter which in practice is often small. Parameterized complexity thus provides a theoretically rigorous framework for explaining why so many applications of NP-complete problems are in practice solvable efficiently.

It is natural to ask how the complexity of a problem changes with the size of the solution sought. Above- and below-guarantee parameterizations are a way to model this in parameterized complexity theory. Given an optimisation problem with a known lower (upper) bound on the value of the objective function, we ask how the complexity of the problem changes as the desired solution value gets further above (below) the bound.

In the first part of this thesis, we consider above-bound parameterizations for a family of graph theory problems related to Max-Cut, in which we are given a graph and asked to find a maximum bipartite subgraph. The Edwards-Erdős bound gives a lower bound on the size of such a subgraph, and the parameterization of Max-Cut above this bound was an open problem for 15 years. We show that this problem is fixed-parameter extendible, and extend this result to the weighted case for a family of problems that generalizes Max-Cut. For the non-weighted case, we also prove polynomial kernel results (in which an instance can be reduced to an equivalent instance with size bounded by a polynomial in the parameter) for the corresponding parameterizations of Balanced Subgraph (a problem that generalizes Max-Cut) and Max Acyclic Subgraph on oriented graphs.

In the second part of this thesis, we investigate various below-guarantee parameterizations for two hypergraph problems, Hitting Set and Test-Cover. We also use our results for the former problem to show positive results for other parameterized problems, including an above-guarantee parameterization of Max-SAT.

Acknowledgements

First and foremost, I would like to thank my supervisors, Gregory Gutin and Anders Yeo. I could not ask for supervisors more wise or more dedicated to the well-being and development of their students. Thanks for helping me take my first faltering steps into academia.

Secondly, thank you to all those I've had the pleasure to collaborate with during the course of my PhD, including Manu Basavaraju, Dave Cohen, Jason Crampton, Robert Crowston, Marek Cygan, Mike Fellows, Matthew Francis, Gregory Gutin, Eun Jung Kim, Daniel Lokshtanov, Matthias Mnich, Gabriele Muciaccia, Geevarghese Philip, Michał Pilipczuk, Ashutosh Rai, Venkatesh Raman, M.S. Ramanujan, Igor Razgon, Frances Rosamond, Imre Rusza, Saket Saurabh, Dominik Scheder, Ondřa Suchý, Stéphan Thomassé, and Anders Yeo. Particular thanks to those I shared the PhD experience with at Royal Holloway: Eun Jung Kim, Gabriele Muciaccia, and my Phd-sibling Robert Crowston.

Beyond the academic, I would like to thank all my friends and family, too many to name, in Egham, London, Manchester, Oxford, Bexley, Aberystwyth and places farther afield. Particular thanks are owed to my housemates Ira and John, for putting up with me at my worst while working on this thesis. To Ceri, who has grown from being my annoying little sister to being one of my best friends; I am so proud of you.

The greatest thanks of all go to my parents, for everything, from teaching me my times tables, to letting me know I'd be supported in whatever I chose to do with my life, to buying me that book of logic puzzles. None of this work would exist without you.

Finally, this thesis is dedicated to the memory of my grandfathers, Gwynne Lloyd Jones and William Leonard Vine, and to my first cousin once removed Jessica Ruth Lloyd, who was born while this thesis was being written.

Contents

Chapter 1

Introduction

This thesis studies a number of problems in *above and below guarantee parameterizations*, which is a sub-field of *parameterized complexity theory*. In order to explain above and below guarantee parameterizations, it is first necessary to give an overview of parameterized complexity. Parameterized complexity itself is best introduced by comparison to classical complexity theory.

Thus, in Section 1.1, we give a brief overview of the basics of classical complexity theory. In Section 1.2 we outline the central ideas of parameterized complexity, and in Section 1.3 we focus on above and below guarantee parameterizations. In Section 1.4 we outline our main results, and the structure of the rest of the thesis. Section 1.5 gives the terminology and notation used in this thesis.

1.1 Classical Complexity Theory

We assume the reader is familiar with the basics of classical complexity theory and only give a brief overview of the definitions we require. For a detailed introduction to classical complexity theory, we refer the reader to *Computers and Intractability: A Guide to the Theory of NP-Completeness* by Garey and Johnson [32].

In what follows let Σ denote some finite alphabet, and Σ^* the set of all finite strings over Σ. A *(classical) problem* is a set $\mathcal{Q} \subseteq \Sigma^*$. A classical problem will normally be presented in the following way:

Instance: $X \in \Sigma^*$
Question: Is $X \in \mathcal{Q}$?

As an example (and by way of early introduction) we present the problem Max-Cut, which is the central problem of Part I:

MAX-CUT
Instance: A graph G, an integer p.
Question: Does G contain a bipartite graph with at least p edges?

We say $\mathcal{Q}$ is *polynomial-time solvable* if for some constant c there exists an algorithm $\mathbb{A}$ which takes as an argument a string in Σ^*, such that $\mathbb{A}(X)$ takes at most $|X|^c$ time, and $\mathbb{A}(X)$ returns YES if $X \in \mathcal{Q}$, and NO otherwise. The class of polynomial-time solvable problems is denoted P.

We say $\mathcal{Q}$ is *polynomial-time verifiable* if there is an algorithm $\mathbb{A}$ which takes as an argument two strings in Σ^*, such that $\mathbb{A}(X, Y)$ takes at most $(|X| + |Y|)^c$ time, and $X \in \mathcal{Q}$ if and only if there exists Y such that $|Y| \leq |X|^c$ and $\mathbb{A}(X, Y)$ returns YES. Thus, Y is a 'certificate' that is used to prove that $X \in \mathcal{Q}$. The class of polynomial-time verifiable problems is denoted NP.

It is easy to show[1] that P $\subseteq$ NP. A problem $\mathcal{Q}$ is NP-*hard* if for any $\mathcal{Q}' \in$ NP, $\mathcal{Q} \in$ P implies $\mathcal{Q}' \in$ P. A problem is NP-*complete* if it is NP-hard and in NP.

It is an open question whether any NP-complete problem is polynomial-time solvable, and thus whether P=NP. One NP-complete problem being polynomial-time solvable would imply that all NP-complete problems are polynomial-time solvable, but these problems have been studied by many people with no polynomial-time algorithm ever being discovered. So there is some empirical evidence to suggest that P$\neq$NP. This is unfortunate, because many NP-complete problems have applications in the real world, where we would prefer algorithms that run quickly.

1.2 Parameterized Complexity

In practice, many real-world applications of NP-complete problems are solvable in a reasonable running time. In some cases this is because in practice we are happy with inexact solutions to a problem, as when we use approximation algorithms. But often it is because a real-world application of an NP-complete problem does not cover all the possible instances of that problem. To put it another way, the specifics of a real-world application of a problem often impose some further structure on that problem, and this structure can make the difference between polynomial-time solvability and intractability. Parameterized complexity provides a precise framework for analysing how such restrictions affect the complexity of a problem.

Parameterized complexity has its roots in research by Fellows and Langston [49]. They observed that not all NP-complete problems are created equal; some structural

[1] For any $\mathcal{Q} \in$ P, let $\mathbb{A}'$ be an algorithm that decides whether $X \in \mathcal{Q}$ in polynomial time. Then let $\mathbb{A}(X, Y)$ be the algorithm that returns $\mathbb{A}'(X)$.

restrictions will make one problem easy while another remains NP-hard. For example, the problem k-Coloring problems asks whether a given graph has a proper vertex coloring with at most k colors, while the problem k-Vertex Cover asks whether a graph has an vertex cover set with at most k vertices. If k is fixed, k-Vertex Cover is polynomial-time solvable, whereas k-Coloring remains NP-hard even when $k = 3$. Investigating this way of distinguishing NP-complete problems led to the foundation of parameterized complexity theory, which first appeared in its current form in *Fixed parameter tractability and completeness* by Downey and Fellows [25].

Parameterized complexity is a framework that examines how the structural properties of problem instances affect the complexity of a problem. It can be viewed as a two-dimensional version of traditional complexity theory: rather than measuring the running time of an algorithm in terms of just the input size, we measure it in terms of the size of the input, and another property called the *parameter*. The analogue of P in parameterized complexity is the class of *fixed-parameter tractable* problems, denoted FPT. If a problem is fixed-parameter tractable, this means that the problem can still be solved efficiently as long as the parameter stays relatively small.

Many NP-complete problems turn out to be fixed-parameter tractable with respect to parameters which are often small in practical applications. For these sorts of problems, parameterized complexity explains why they are easy in practice. Moreover, the tools developed in parameterized complexity can be used to produce new, efficient algorithms for real-world problems that were previously dismissed as NP-complete. From a more theoretical viewpoint, parameterized complexity can give us much more insight into what makes a problem hard - if an NP-complete is FPT with respect to some parameter, then we know that the 'hardness' of that problem is in a sense dependent on that parameter.

We now make these notions precise. For a more detailed introduction, we refer the reader to [26, 30, 58].

A *parameterized problem* is a set $\mathcal{L} \subseteq \Sigma^* \times \Sigma^*$. (Often, we have $\mathcal{L} \subseteq \Sigma^* \times \mathbb{N}$.) For $(X, k) \in \Sigma^* \times \Sigma^*$, we say that k is the *parameter* of the instance (X, k).

A *parameterization* is a function $\kappa : \Sigma^* \to \Sigma^*$ (often $\kappa : \Sigma^* \to \mathbb{N}$). Given a classical problem $\mathcal{Q}$ and a parameterization κ, we define the parameterized problem $\mathcal{Q}$ *parameterized by* κ as $\mathcal{L} = \{(X, k) : X \in \mathcal{L}, k = \kappa(X)\}$. All the parameterized problems we consider in this thesis are equivalent to parameterizations of NP-complete problems.

Parameterized problems are normally introduced by describing a classical problem together with a parameter, in the following way:

Instance: $X \in \Sigma^*$.
Parameter: A function $\kappa : \Sigma^* \to \Sigma^*$.
Question: Is $X \in \mathcal{Q}$?

For example, this is Max-Cut parameterized by the solution size:

p-Max-Cut
Instance: A graph G, an integer p.
Parameter: p.
Question: Does G contain a bipartite subgraph with at least p edges?

On the other hand, this is Max-Cut parameterized by treewidth[2]:

tw-Max-Cut
Instance: A graph G, an integer p.
Parameter: The treewidth of G.
Question: Does G contain a bipartite subgraph with at least p edges?

We can now formally define the notion of fixed-parameter tractability.

Definition 1. *A parameterized problem $\mathcal{L} \subseteq \Sigma^* \times \mathbb{N}$ is* fixed-parameter tractable (FPT) *if there exists an algorithm which, for any $(X, k) \in \Sigma^* \times \mathbb{N}$, decides whether $(X, k) \in \mathcal{L}$ in at most $f(k)|X|^c$ steps, for a constant c and a computable function f that depends only on k.*

We also introduce the notion of a *parameterized reduction*, which is of particular use in proving hardness results.

Definition 2. *Let L and Q be parameterized problems. We say a function $f : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ is a* parameterized reduction *from L to Q if there exists a constant c and computable functions $g, h : \mathbb{N} \to \mathbb{N}$ such that for any $(x, k) \in \Sigma^* \times \mathbb{N}$,*

- *$f(X, k) = (X', k')$ can be calculated in time $g(k)|X|^c$.*
- *$(X, k) \in L$ if and only if $(X', k') \in Q$.*
- *$k' \leq h(k)$.*

Note that in a parameterized reduction, $|X'|$ may be larger than $|X|$ and k' may be larger than k, just as long as the value of k' only depends on k'. It can be seen that if Q is fixed-parameter tractable and there is a parameterized reduction from L to Q, then L is also fixed-parameter tractable.

[2]See Section 1.5.1 for the definition of treewidth

1.2.1 Kernelization

Just as important as fixed-parameter tractability is the notion of *kernelization*. This is a process by which an instance of a parameterized problem can be reduced to an equivalent instance (the *kernel*) with size bounded by the parameter. A fixed-parameter tractable problem has its complexity dependent on the value of the parameter, and a kernelization shows this directly by making the size of the whole instance dependent on the parameter. Kernelizations are of interest because they allow for interaction with other approaches - we can apply a kernelization to replace a large instance with a much smaller one, which then makes the problem much easier to solve using whatever other methods are available.

Formally:

Definition 3. *[26] We say a parameterized problem $\mathcal{L}$ has a* kernel *if there exists a polynomial -time transformation $\rho : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ which, for any instance (X, k), produces an instance $(X', k') = \rho(X, k)$ satisfying the following conditions:*

1. *$(X, k) \in \mathcal{L}$ if and only if $(X', k') \in \mathcal{L}$.*
2. *$|X'| \leq f(k)$*
3. *$k' \leq f(k)$*

For some computable function function f depending on k only.

We call ρ a *kernelization* of $\mathcal{L}$. We say that $f(k)$ is the *size* of the kernel. Thus, for example, we would say a problem has a kernel of size k^2 if any instance (X, k) can be transformed into an equivalent instance (X', k') in which $|X'|$ and k' are bounded by k^2. We say $\mathcal{L}$ has a *polynomial kernel* if it has a kernel of size at most k^c for some constant c.

It is well-known that a parameterized problem which is decidable has a kernelization if and only if it is fixed-parameter tractable [58]. It is clear that if the problem has a kernelization, then we can simply solve the kernel using any brute-force method and the time taken to do this will depend only on the parameter. In the other direction, suppose that $\mathcal{L}$ has an algorithm that decides any instance (X, k) in time $f(k)|X|^c$. Then if $|X| \leq f(k)$, (X, k) is already a kernel. Otherwise, the algorithm to decide $(|X|, k)$ takes time at most $|X|^{c+1}$. Thus, the trivial polynomial-time kernelization is to return (X, k) if $|X| \leq f(k)$ and solve the problem in time $|X|^{c+1}$ otherwise, replacing (X, k) with any small Yes- or No-instance as appropriate.

In practice, the trivial kernelization derived from an FPT algorithm is rarely of interest, because they tend to be unmanageably large. In fact, what is most desirable

is a polynomial-size kernel for a problem. Thus, for a problem which is known to be fixed-parameter tractable, an important further question is whether it has a kernel of polynomial size or not.

There are many parameterized problems which are fixed-parameter tractable but have no polynomial kernel, under some reasonable complexity assumptions. Bodlaender et. al. [6] were the first (in 2009) to prove non-existence of polynomial kernels for certain parameterized problems, under the assumption that $coNP \not\subseteq NP/poly$. (This is a reasonable assumption, as without it the polynomial hierarchy collapses to the third level). The approach developed by Bodlaender et. al. involves showing that a problem is *OR-compositional*, which roughly means that multiple instances of the same problem can be transformed into an instance which is a YES-instance if and only if one the original instances is a YES-instance, in such a way that the new instance has a parameter not much larger than the parameter of any one of the original instances. It was shown in [6] that a parameterized problem which is OR-compositional has no polynomial kernel, unless $coNP \subseteq NP/poly$. In 2011, Bodlaender et. al. [7] improved this result by replacing the OR-compositional condition with that of being *cross-compositional*, a generalisation of OR-composition that makes it easier to show a problem has no polynomial kernel.

We will not use OR-composition or cross-composition in this thesis, but we will use the notion of *polynomial parameter transformations* developed in [8, 23].

Definition 4. *Let L and Q be parameterized problems. We say a polynomial time computable function $f : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ is a* polynomial parameter transformation *from L to Q if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ such that for any $(X, k) \in \Sigma^* \times \mathbb{N}, (X, k) \in L$ if and only if $f(X, k) = (X', k') \in Q$, and $k' \leq p(k)$.*

Lemma 5. [8, Theorem 3] *Let L and Q be parameterized problems, and suppose that L^c and Q^c are the derived classical problems. Suppose that L^c is NP-complete, and $Q^c \in NP$. Suppose that f is a polynomial parameter transformation from L to Q. If Q has a polynomial-size kernel, then L has a polynomial-size kernel.*

For an overview of other techniques in kernelization, we refer the reader to [50].

1.2.2 Fixed-Parameter Intractability and the W-Hierarchy

In classical complexity theory, we can show that a problem is unlikely to be polynomial-time solvable by showing that it is hard for the class of problems NP. In a similar way, in parameterized complexity we can show that a parameterized problem is unlikely to be fixed-parameter tractable by showing that it is hard for a certain class

of problems. Just as in classical complexity theory, we do not know for sure that these classes are distinct from FPT, but if any problem in this class can be reduced to a parameterized problem L, that is strong evidence for the claim that L is no fixed-parameter tractable.

A parameterized problem is said to be in the class $W[t]$, for any integer t, if it can be reduced by a parameterized reduction to the problem of finding a satisfying assignment to a circuit with depth t, which assigns 1 to at most a parameter k number of inputs. (The depth of a circuit can be thought of as the maximum number of nodes with unbounded input on any path from an input node to the output node.) In particular, the class $W[1]$ is the class of all problems which can be reduced to the problem WEIGHTED 3-CNF SAT (given a 3-CNF formula and a parameter k, the task is to determine whether there is a solution to the formula in which k variables are set to true?) [24].

The class XP is the class of all parameterized problems for which any instance (X, k) can be solved in time $|X|^{f(k)}$ for some computable function f. Letting FPT denote the class of all fixed-parameter tractable problems, the classes FPT, XP and $W[t]$ for all t have the following relationship:

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq XP.$$

We use the notions of *hardness* and *completeness* in a similar way to classical complexity theory. Thus, a problem is $W[1]$-*hard* if there is a parameterized reduction to that problem from any problem in $W[1]$; a problem is $W[1]$-complete if it is both in $W[1]$ and $W[1]$-hard.

Finally, the class para-NP is the class of parameterized problems for which any instance (X, k) be decided by a nondeterministic Turing machine in time $O(f(k)|X|^c)$, for some computable function f and constant c. In particular, every parameterization of a problem in NP is in para-NP. It is well-known that a parameterized problem Ł belonging to para-NP is para-NP-complete if it is NP hard even for fixed values of the parameter - that is, for some constant c, the problem of deciding whether $(X, c) \in \mathcal{L}$ is NP-hard [30]. Note that if a problem is para-NP-hard then it is also XP-hard.

For a more detailed discussion of these classes, we refer the reader to *Parameterized Complexity Theory*, by Flum and Grohe [30].

1.3 Combinatorial Optimisation problems and above and below guarantee parameterizations

Parameterizations above a tight bound, as a systematic way of parameterizing NP-complete problems, were first introduced by Mahajan and Raman [53]. Given an optimisation problem for which there is a known lower bound, we ask if there is a solution which achieves that bound plus a parameter. We now make this notion more precise and provide some motivation.

In a combinatorial optimisation problem, we are given a structure together with a characterization of feasible solutions, and asked to find a solution that maximizes or minimizes a particular function. In general, given a function $\Phi : \Sigma^* \times \Sigma^* \to \mathbb{R}$, we could define a *maximization problem* as follows:

MAX-Φ
Instance: $X \in \Sigma^*$, an integer p.
Question: Does there exist $Y \in \Sigma^*$ such that $\Phi(X, Y) \geq p$?

Or we could define a *minimization problem*:

MIN-Φ
Instance: $X \in \Sigma^*$, an integer p.
Question: Does there exist $Y \in \Sigma^*$ such that $\Phi(X, Y) \leq p$?

For example, in MAX CUT, X would be a graph, and $\Phi(X, Y)$ would be the number of edges in Y if Y is a bipartite subgraph of G, and 0 otherwise.

For such problems, the *natural parameterization* is to let the parameter be p. Thus, the following parameterized problem is the natural parameterization of MAX-Φ.

MAX-Φ
Instance: $X \in \Sigma^*$, an integer p.
Parameter: p.
Question: Does there exist $Y \in \Sigma^*$ such that $\Phi(X, Y) \geq p$?

The natural parameterization is defined similarly for minimization problems.

Many of the most famous and well-studied parameterized problems are natural parameterizations of NP-complete problems. Indeed, in the context of parameterized complexity, the name of an unparameterized problem is often taken as shorthand for the natural parameterization of that problem. In particular, the parameterized problems VERTEX COVER (given a graph, find a vertex cover set with at most k

vertices, where k is the parameter) is a natural parameterizations of a minimization problem, and INDEPENDENT SET (given a graph, find an independent set with at least k vertices, where k is the parameter) is a natural parameterization of a maximization problem.

For optimization problems, the natural parameterization is appealing as it simple, and allows us to investigate how the complexity of a problem changes with the value of the solution we are seeking. However, there are cases in which the natural parameterization is not a useful one. Suppose that for some maximization problem MAX-Φ, there exists a function $\gamma : \Sigma^* \to \mathbb{R}$ such that for all $X \in \Sigma^*$, there exists $Y \in \Sigma^*$ such that $\Phi(X, Y) \geq \gamma(X)$. Suppose furthermore γ is increasing in $|X|$. Then we can show that the natural parameterization of MAX-Φ has a kernel and is thus fixed-parameter tractable.

Let $f : \mathbb{R} \to \mathbb{R}$ be any increasing function such that $f(\gamma(X)) \geq |X|$ for any $X \in \Sigma^*$. Then for any instance (X, p) of MAX-Φ, if $\gamma(X) \geq p$ then we know that (X, p) is a YES-instance. Otherwise, we have that $|X| \leq f(\gamma(X)) \leq f(p)$. Thus, (X, p) itself is a kernel of size $\max(f(p), p)$.

As an example, in the case of the problem MAX-CUT, the Edwards-Erdős bound [27, 28] says that for any connected graph G with n vertices, m edges, there exists a bipartite subgraph of G with at least $\frac{m}{2} + \frac{n-1}{4}$ edges. Then if $\frac{m}{2} + \frac{n-1}{4} \geq k$, we know that (G, k) is a YES-instance. Otherwise, we have a kernel with at most $4k+4$ vertices and at most $2k$ edges.

However, in this case our FPT result is almost trivial, and not particularly useful. For fixed-parameter tractable algorithms are of most use in cases when the parameter is small. However, when p is small, then (G, p) is already known to be a YES-instance, unless G itself itself is small as well. Thus, either we have no need of an algorithm, or any brute-force algorithm will do.

We would, however, still like to analyse how the complexity of the problem changes with the desired solution value. Mahajan and Raman argued that for maximization problems, one should parameterize *above tight bounds* [53]. Thus, given a maximization problem MAX-Φ with a known lower bound γ, we would consider MAX-Φ parametrized above γ:

MAX-Φ-ABOVE-γ
Instance: $X \in \Sigma^*$, an integer k.
Parameter: k.
Question: Does there exist $Y \in \Sigma^*$ such that $\Phi(X, Y) \geq \gamma(X) + k$?

Thus, for example, this is MAX-CUT parameterized above the Edwards-Erdős

bound:

MAX-CUT-ABOVE-EDWARDS-ERDŐS
Instance: A graph G with n vertices, m edges, an integer k.
Parameter: k.
Question: Does G contain a bipartite subgraph with at least $\frac{m}{2} + \frac{n-1}{4} + k$ edges?

Similarly, given a minimization problem MIN-Φ, if there is a known upper bound γ such that for any $X \in \Sigma^*$ there exists $Y \in \Sigma^*$ such that $\Phi(X, Y) \leq \gamma(X)$, we can parameterize below γ:

MIN-Φ-BELOW-γ
Instance: $X \in \Sigma^*$, an integer k.
Parameter: k.
Question: Does there exist $Y \in \Sigma^*$ such that $\Phi(X, Y) \leq \gamma(X) - k$?

In Part I of this thesis we consider MAX-CUT-ABOVE-EDWARDS-ERDŐS and a number of other maximization problems parameterized above a lower bound. In Part II we consider a number of minimization problems parameterized below an upper bound.

It is is also possible to consider maximization problems parameterized below an upper bound. That is, suppose for a maximization problem MAX-Φ there exists a function $\gamma : \mathbb{R} \to \mathbb{R}$, such that for any $Y \in \Sigma^*$, $\Phi(X, Y) \leq \gamma(X)$. Thus, $\gamma(X)$ is an upper bound on the best possible solution. Then we can consider the following problem:

MAX-Φ-BELOW-γ
Instance: $X \in \Sigma^*$, an integer k.
Parameter: k.
Question: Does there exist $Y \in \Sigma^*$ such that $\Phi(X, Y) \geq \gamma(X) - k$?

Similarly, we can consider minimization problems parameterized above a lower bound. Suppose for a minimization problem MIN-Φ there exists a function $\gamma : \mathbb{R} \to \mathbb{R}$, such that for any $Y \in \Sigma^*$, $\Phi(X, Y) \geq \gamma(X)$. Then we can consider the following problem:

MIN-Φ-ABOVE-γ
Instance: $X \in \Sigma^*$, an integer k.
Parameter: k.
Question: Does there exist $Y \in \Sigma^*$ such that $\Phi(X, Y) \leq \gamma(X) + k$?

We refer to the sort of parameterized problems defined above as *above/below guarantee parameterizations*.

Most of the problems considered in this thesis are above-guarantee parameterizations for maximization problems, or below-guarantee parameterizations for minimization problems. However, in the last part of Chapter 8 we do consider an above-guarantee parameterization of the minimization problem Test-r-Cover.

While the problems in this thesis are mainly concerned with graphs and hypergraphs, there has also been much investigation into above or below guarantee parameterizations of constraint satisfaction. In particular, the problem Max Sat, in which we are given a CNF formula and asked to find a truth assignment maximizing the number of satisfying clauses, has been the subject of much study. Mahajan and Raman [53] showed that some above guarantee versions of Max-Cut and Max Sat are FPT. In a later paper, Mahajan et al. [54] published a paper with several new results and open problems around parameterizations beyond guaranteed lower and upper bounds. In a breakthrough paper Gutin et al. [35] developed a probabilistic approach to problems parameterized above or below tight bounds, for cases when the bound is the expected value of a random assignment. Alon et al. [3] combined this approach with a method from Fourier analysis to obtain an FPT algorithm for parameterized Max r-SAT beyond the guaranteed lower bound (in which the clauses of the CNF formula may contain at most r literals). In the same paper a quadratic kernel was also given for Max r-SAT. It was finally shown in [16] that Max Sat parameterized above the guaranteed lower bound is para-NP complete.

Other significant results in this direction include quadratic kernels for ternary permutation constraint satisfaction problems parameterized above average, and results on systems of linear equations over the field of two elements [14, 15, 36].

Recently, Cygan et al. [19] introduced a novel form of above guarantee parameterization, in which, for a problem which can be modelled as an Integer Programming problem, the lower bound is the optimal solution to the Linear Programming relaxation of the problem. This approach was further developed in [51] and [57], where the above LP-relaxation parameterization of Vertex Cover was used to obtain improved algorithms for a number of parameterized problems.

1.4 Main Results and Structure of Thesis

In Part I, we consider a family of above-guarantee parameterized problems concerning λ-extendible graph properties. The notion of a λ-extendible property is a notion which includes the properties of being a bipartite graph and being an acyclic

oriented graph. For any λ-extendible property Π, the Poljak-Turzík bound on a weighted graph G bound is a lower bound on the maximum weight of a subgraph of G with property Π. For a λ-extendible property Π, the parameterized problem Π-SAPT is the problem, given a graph G and parameter k, of finding a Π-subgraph of G with weight at least the Poljak-Turzík bound plus k.

In Chapter 2, we give the definitions related to λ-extendible properties and the Poljak-Turzík bound. We then give a necessary condition for Π-SAPT to be fixed-parameter tractable, for any λ-extendible property Π. This extends the results of Mnich et. al. [55], which dealt only with the unweighted version of Π-SAPT.

In Chapters 3 and 4, we consider two specific λ-extendible properties. In Chapter 3, we consider the problem of finding the maximum balanced subgraph of a signed graph, parameterized above the Poljak-Turzík bound. We show that this problem is fixed-parameter tractable and, in the unweighted case, has a polynomial kernel. This problem generalises the problem Max-Cut parameterized above the Edwards-Erdős bound, and a such, the results in this chapter answer an open question of Mahajan and Raman from 1999 [53].

In Chapter 4 we consider the problem of finding the maximum acyclic subgraph of an oriented graph, parameterized above the Poljak-Turzík bound. We show that this problem is fixed-parameter tractable and, in the unweighted case, has a polynomial kernel, answering an open question of Mahajan *et al.* [54] and Raman and Saurabh [63].

In Part II we consider mainly below-guarantee parameterizations. In Chapter 5 we consider the problem Hitting Set: given a hypergraph with n vertices and m edges, find a minimum *hitting set*, that is, a set of vertices which intersects every edge. Note that n and m are both tight upper bounds on on the size of a minimum hitting set. We consider Hitting Set parameterized below m, denoted $(m-k)$HitSet: Given a hypergraph H and parameter k, does k contain a hitting set with at most $(m-k)$ vertices? Similarly, we consider Hitting Set parameterized below n, denoted $(n-k)$HitSet. We show that $(m-k)$HitSet is fixed-parameter tractable but is unlikely to have a polynomial kernel. We show that $(n-k)$HitSet is $W[1]$-complete, but becomes fixed-parameter tractable with a polynomial kernel if we add the degeneracy of H to the parameter.

In Chapter 6, we give some applications of the results of Chapter 5. In particular, we use the FPT result of $(m-k)$HitSet to prove fixed-parameter tractability of an above-guarantee parameterization of Max Sat: given a CNF formula F and an integer k, decide if there is an assignment that satisfies $\nu(F)+k$ clauses of F, where

$\nu(F)$ is the matching number of F. We also use the results on $(n-k)$HitSet to prove a polynomial kernel for the parameterized problem Directed Nonblocker.

In Chapter 7, we consider the problem Test Cover: given a hypergraph H with n vertices and m edges, find a minimum *test cover*, that is, a set of edges such that every pair of vertices is separated by some edge in the set. Again, n and m are tight upper bounds on the size of a minimum test cover; we consider Test Cover parameterized below these bounds, denoted $(n-k)$Test Cover and $(m-k)$Test Cover, respectively. We show that $(m-k)$Test Cover is $W[1]$-hard, while $(n-k)$Test Cover is fixed-parameter tractable; however, the running time we get for $(n-k)$Test Cover is too large to be practical.

In light of the results of Chapter 7, in Chapter 8 we consider a restriction of Test Cover when the size of all edges is bounded by a constant r, denoted Test-r-Cover. Again, we consider parameterizations of this problem below n and m, denoted $(n-k)$Test-r-Cover and $(m-k)$Test-r-Cover, respectively. We show that both problems are fixed-parameter tractable with a polynomial kernel. Finally, independent of the other results, we prove a new lower bound for Test-r-Cover of $\frac{2(n-1)}{r+1}+k$. We consider Test-r-Cover parameterized above this bound, and show that it is para-NP complete.

1.4.1 Bibliographic note

Much of the work in this thesis is based on work from the following papers:

- R. Crowston, G. Gutin and M. Jones, Directed Acyclic Subgraph Problem Parameterized above the Poljak-Turzík Bound. In *FSTTCS 2012*, LIPICS 18, 400–411, 2012.

- R. Crowston, G. Gutin, M. Jones, and G. Muciaccia, Maximum Balanced Subgraph Problem Parameterized Above Lower Bound, http://arxiv.org/abs/1212.6848 (2012)

- R. Crowston, G. Gutin, M. Jones, G. Muciaccia, and A. Yeo, Parameterizations of Test Cover with Bounded Test Sizes, http://arxiv.org/abs/1209.6528 (2012)

- R. Crowston, G. Gutin, M. Jones, V. Raman, S. Saurabh, and A. Yeo, Fixed-Parameter Tractability of Satisfying beyond the Number of Variables, In *SAT 2012*, 355-368

- R. Crowston, G. Gutin, M. Jones, S. Saurabh, and A. Yeo, Parameterized Study of the Test Cover Problem. In *MFCS 2012*, Lect. Notes Comput. Sci. 7464 (2012), 283-295.

- R. Crowston, M. Jones, and M. Mnich, Max-Cut Parameterized above the Edwards-Erdős Bound, In *ICALP 2012*, Lect. Notes Comput. Sci. 7391 (2012) 242–253.

- G. Gutin, M. Jones, and A. Yeo, Kernels for below-upper-bound parameterizations of the hitting set and directed dominating set problems, *Theor. Comput. Sci.*, 412(41):5744–5751, 2011.

In addition, Chapter 2 is based heavily on discussions with Robert Crowston, Geevarghese Philip, Ashutosh Rai and Saket Saurabh in late 2012.

1.5 Terminology and Notation

1.5.1 Graphs

For most standard graph theory terms, our terminology and notation is based on *Modern Graph Theory* by B. Boolobás [9].

A *simple undirected graph* is an ordered pair $G = (V, E)$, where E is a set of unordered pairs of elements in V. In what follows, when we say graph we mean a simple undirected graph, unless otherwise stated. The elements of V are the *vertices* of G and the elements of E are the *edges* of G. Two vertices $u, v \in V$ are *adjacent* if $u \neq v$ and $\{u, v\} \in E$. An edge $\{u, v\}$ is normally written uv for short; thus u, v are adjacent if and only if $uv \in E$.

A graph $H = (V', E')$ is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$. $H = (V', E')$ is an *induced subgraph* of G if $V' \subseteq V$ and $E' = E \cap (V')^2$ - that is, H contains all edges between vertices in V' that appear in G. For a set of vertices $X \subseteq V(G)$, the subgraph $G[X]$ is the induced subgraph of G with vertex set X. We denote by $G - X$ the subgraph $G[V(G) \setminus X]$.

Given a vertex v and a subgraph G' of G, we define $N_{G'}(v) = \{y \in V(G') : xy \in E(G')\}$ and $N_{G'}[v] = N_{G'}(v) \cup \{v\}$. For a set of vertices Y, we define $N_{G'}[Y] = \bigcup_{v \in Y} N_{G'}[v]$ and $N_{G'}(Y) = N_{G'}[Y] \setminus Y$.

For a vertex v in a graph G, we define $N(v) = N_G(v)$ and $N[v] = N_G[v]$, and call $N(v)$ the *open neighborhood* and $N[v]$ the *closed neighborhood* of v. For a set of vertices Y, the open neighborhood $N(Y)$ and closed neighborhood $N[Y]$ of Y are

defined similarly. When we refer to a neighborhood, we mean the open neighborhood unless otherwise specified. We define the *degree* of v to be the integer $d(v) = |N(v)|$.

We say that a graph G is *isomorphic* to a graph H if there exists a bijection $\phi : V(G) \to V(H)$ such that $uv \in E(G)$ if and only if $\phi(u)\phi(v) \in E(H)$, for any $u, v \in V(G)$.

We say that G *contains* H (or G contains a copy of H; G contains an H), if G has a subgraph isomorphic to H. we say that G contains an *induced* H if G contains an induced subgraph isomorphic to H. When X is the set of vertices such that $G[X]$ is isomorphic to H, we will refer to X as an induced H.

For any integer $r \geq 1$, the *complete graph on r vertices* is the graph K_r with $|V(K_r)| = r$ and $E(K_r) = \{uv : u, v \in V(K_r), u \neq v\}$.

A *clique* is a subgraph of G which is isomorphic to K_r for some r. We will often call a set of vertices X a *clique* if $G[X]$ is a clique.

For any integer $r \geq 0$, the *path of length r* is the graph P_{r+1} with $V(P_{r+1}) = \{v_1, \ldots, v_{r+1}\}$ and $E(P_{r+1}) = \{v_i v_{i+1} : 1 \leq i \leq r\}$.[3] For a pair of vertices $a, b \in V(G)$, we say G contains a *path between a and b of length r* if G contains a subgraph isomorphic to P_{r+1}, in which a is mapped to v_1 and b is mapped to v_{r+1}. In Section I, the notion of an *induced* P_3 will be important. As implied by the definitions already given, an induced P_3 in G is set of vertices $\{a, b, c\}$ (or the subgraph $G[\{a, b, c\}]$) such that $ab \in E(G), bc \in E(G), ac \notin E(G)$.

For any integer $r \geq 3$, the *cycle of length r* is the graph C_r with $V(C_r) = \{v_1, \ldots, v_r\}$ and $E(C_r) = \{v_i v_{i+1}\} \cup \{v_r v_1\}$. We say a graph *contains a cycle* if G contains a subgraph isomorphic to C_r for some $r \geq 3$.

We say $u, v \in V(G)$ are *connected* in G if there is a path between a and b in G. We say G is *connected* if there is a path between every pair of vertices in G. We say G is *disconnected* otherwise. In particular, we will count a graph as connected if it is an empty graph with no vertices or edges.

We say a set of vertices $X \subseteq V(G)$ *disconnects* G if $G - X$ is disconnected. If $x \in V(G)$ is a single vertex such that $G - \{x\}$ is disconnected, we say v is a *cut-vertex* in G. G is r-connected if $|V(G)| > r$ and $G - X$ is connected, for every $X \subseteq V$ with $|X| < r$.

A *component* of G is a maximal subgraph which is connected. That is, a subgraph G' is a component of G if there is no subgraph G'' of G such that G' is a subgraph of G'' and $G \neq G''$. Note that every component is an induced subgraph. For any

[3] Note that, in deference to modern convention, our notation here differs slightly from that in [9]. In that book, the path of length r is denoted P_r rather than P_{r+1}. Thus, in this thesis, the path with two edges and three vertices is denoted P_3, whereas in [9] it is denoted P_2.

graph G, we let $\mathcal{C}(G)$ denote the set of all components of G. We will sometimes refer to a set of vertices X as forming a component of G if $G[X]$ is a component of G.

A *forest* is a graph which contains no cycles. A *tree* is a connected forest. A subgraph T of G is a *spanning tree* of G if T is a tree and $V(T) = V(G)$.

Given a vertex $r \in V(G)$ and a set of vertices $X \subseteq V(G)$, if $r \notin X$ we define $R(r, X)$ to be the set of vertices connected to r in $G - X$. That is, $R(r, X)$ is the set of vertices in the component of $G - X$ containing r. If $r \in X$ then $R(r, X) = \emptyset$.

For two disjoint sets of vertices $X, Y \subseteq V(G)$, we denote by $E(X, Y)$ the set of all edges with one vertex in X and one vertex in Y. We define by $E(X)$ the set of edges with both vertices in X. Let V_1, V_2 be a partition of $V(G)$. Then we say that G is (V_1, V_2)-*bipartite* if $E(V_1, V_2) = E(G)$, that is, all edges in G have one vertex in V_1 and one vertex in V_2. For a graph $G = (V, E)$ a *matching* in G is a set of edges $M \subseteq E$ such that no two edges in M share a vertex. We say M is a *maximum matching* if there is no matching in G with more than $|M|$ edges. If G has a maximum matching M such that every vertex of G is contained in an edge in M, we say that M is a *perfect matching*

A *block* in G is a maximal connected subgraph $G[X]$ of G such that $G[X]$ has no cut-vertex. We will often refer to a set of vertices X as a *block* if $G[X]$ is a block. Note that a block may consist of a single vertex x, if x is an isolated vertex in G. A block may be an empty subgraph (with no vertices or edges) only if G itself is an empty graph. Observe that the blocks of a graph G have disjoint edges, and when G is connected, the vertices which appear in two or more blocks are exactly the cut-vertices of G. We say a block is a *leaf-block* if it contains exactly one cut-vertex of G. Given a block X in G and a vertex $v \in X$, we say v is an *internal vertex of* X if v is not a cut-vertex.

We will say graph G is a *forest of cliques* if every block in G is a clique. This will be a central concept in Part I. is A forest of cliques is a *tree of cliques* if it is connected.

A *tree decomposition* of an (undirected) graph G is a pair (U, T) where T is a tree whose vertices we will call *nodes* and $U = (\{U_i \mid i \in V(T)\})$ is a collection of subsets of $V(G)$ such that

1. $\bigcup_{i \in V(T)} U_i = V(G)$,
2. for each edge $vw \in E(G)$, there is an $i \in V(T)$ such that $v, w \in U_i$, and
3. for each $v \in V(G)$ the set $\{i : \ v \in U_i\}$ of nodes forms a subtree of T.

The U_i's are called *bags*. The *width* of a tree decomposition $(\{U_i : \ i \in V(T)\}, T)$

equals $\max_{i \in V(T)}\{|U_i| - 1\}$. The *treewidth* of a graph G is the minimum width over all tree decompositions of G. We use notation $tw(G)$ to denote the treewidth of a graph G.

1.5.2 Directed Graphs

A *directed graph* is an ordered pair $D = (V, A)$, where A is a set of ordered pairs of elements from V. The elements of V are the *vertices* of D and the elements of A are the *arcs* of D.

Similar to undirected graphs, we will, use $V(D)$ and $A(D)$ to denote the vertices and arcs of D, respectively.

We say an arc (u, v) is an arc *from u to v* and we normally write it uv.

The notions of subgraph and induced subgraph are defined analogously to the undirected case.

The *underlying* graph of D is the undirected graph G, where the vertices of G are the vertices of D, and there is an edge between two vertices u, v in G if and only if there is an arc from u to v or an arc from v to u in G.

The (*open, closed*) *neighborhood* of a vertex v in D is the neighborhood of v in the underlying graph, and the (open, closed) neighborhood of a set of vertices in D is defined similarly. The degree of a vertex v is the degree of v in the underlying graph. The *out-neighborhood* of v is $N^+(v) = \{u \in V : vu \in A\}$ and its *in-neighborhood* if $N^-(v) = \{u \in V : uv \in A\}$. The *out-degree* of v is $d^+(v) = |N^+(v)|$ and the *in-degree* if $d^-(v) = |N^-(v)|$. For a set S of vertices in D, $N^+(S) = (\bigcup_{v \in S} N^+(v)) \setminus S$ and $N^-(S) = (\bigcup_{v \in S} N^-(v)) \setminus S$.

We say there is a *directed path from a to b* in D if D contains a subgraph with vertices $\{v_1, \ldots v_{r+1}\}$ and arcs $v_i v_{i+1} : 1 \leq i \leq r\}$. We say D has a *directed cycle* if D contains a subgraph with vertices $\{v_1, \ldots v_{r+1}\}$ and arcs $v_i v_{i+1} : 1 \leq i \leq r\}$ and $v_{r+1} v_1$. A directed graph is *acyclic* if it does not have a directed cycle.

A directed graph is *connected* if its underlying graph is connected. A directed graph is *strongly connected* if for every pair of vertices u, v, there is a directed path from u to v.

A directed graph is *oriented* if for any $u, v \in V$, at most of uv, vu is in A. For the most part, we will be concerned with oriented graphs. A directed graph is a *tournament* if for every distinct pair of vertices u, v, exactly one of uv, vu is in A.

A spanning tree of D is a subgraph T such that the underlying graph of T is a tree and $V(T) = V(D)$.

A directed graph D is a *forest of cliques* if its underlying graph is a forest of

cliques.

1.5.3 Weights

A *weight function* on a graph G is a function $w : E(G) \to \mathbb{R}^+$. When no weight function is specified, we assume $w(e) = 1$ for all $e \in E(G)$. Similarly, for a directed graph D a *weight function* is a function $w : A(D) \to \mathbb{R}^+$.

For a set of edges $F \subseteq E(G)$, we define $w(F) = \sum_{e \in F} w(e)$. Similarly for a set of arcs $B \subseteq A(D)$, we define $w(B) = \sum_{a \in B} w(a)$. We define $w(G) = w(E(G))$ and $w(D) = w(A(D))$.

For a connected graph G, the *minimum weight spanning tree* of a graph G is a subgraph T such that T is a spanning tree of G and $w(T)$ is minimal for all spanning trees of G. We denote the weight $w(T)$ of a minimum weight spanning tree of G by $\tau(G)$.

We say a graph G is a *uniform forest of cliques* if G is a forest of cliques and within each block of G, all edges have the same weight.

1.5.4 Hypergraphs

A *hypergraph* $H = (V, \mathcal{F})$ consists of a nonempty set V of *vertices* and a family $\mathcal{F}$ of nonempty subsets of V called *edges* of H. Note that $\mathcal{F}$ may have *parallel* edges, i.e., copies of the same subset of V. An edge e is a *singleton* if $|e| = 1$.

For any vertex $v \in V$, and any $\mathcal{E} \subseteq \mathcal{F}$, $\mathcal{E}[v]$ is the set of edges in $\mathcal{E}$ containing v, $N[v]$ is the set of all vertices contained in edges of $\mathcal{F}[v]$, and the *degree* of v is $d(v) = |\mathcal{F}[v]|$. For a subset T of vertices, $\mathcal{F}[T] = \bigcup_{v \in T} \mathcal{F}[v]$.

Deleting an edge e from a *hypergraph* $H = (V, \mathcal{F})$ results in a new hypergraph $H - e$ with vertex set V and edge set $\mathcal{F} \setminus \{e\}$. *Deleting* a vertex v from a *hypergraph* $H = (V, \mathcal{F})$ results in a new hypergraph $H - v$ with vertex set $V \setminus \{v\}$ and edge set $\{e \setminus \{v\} : \ e \in \mathcal{F}\}$.

For a hypergraph $H = (V, \mathcal{F})$ and a set $X \subset V$, the subhypergraph $H \ominus X$ is obtained from H by deleting the set $\mathcal{E}$ of all edges hit by X and all vertices contained only in $\mathcal{E}$. A hypergraph $H = (V, \mathcal{F})$ is *d-degenerate* if, for all $X \subset V$, the subhypergraph $H \ominus X$ contains a vertex of degree at most d. The *degeneracy* $\deg(H)$ of a hypergraph H is the smallest d for which H is d-degenerate.

The degeneracy of a hypergraph can be calculated in linear time using the following algorithm. Pick a vertex v_1 in H of minimum degree d_1, and set $H := H \ominus \{v_1\}$. Pick a vertex v_2 of minimum degree d_2 and set $H := H \ominus \{v_2\}$, and so on. Then $d = \max\{d_i : i \in [n]\}$ is the degeneracy of H. (It is clear that the degeneracy of

H must be at least d; the equality follows by observing that for any $X \subset V$ the smallest numbered vertex $v_i \in V \setminus X$ has degree at most d_i in $H \ominus X$.)

Part I

Max-Cut and λ-extendible properties

In 1973, Edwards [27, 28] proved that every connected graph with n vertices and m edges contains a bipartite subgraph with at least $\frac{m}{2} + \frac{n-1}{4}$ edges, proving a conjecture of Erdős. This lower bound on the size of a bipartite subgraph is known as the *Edwards-Erdős bound*. The problem Max-Cut parameterized above the Edwards-Erdős bound is as follows: given a graph G and parameter k, decide whether it contains a bipartite subgraph with at least $\frac{m}{2} + \frac{n-1}{4} + k$ edges.[4] Mahajan and Raman [53], in their first paper on above-guarantee parameterizations, asked whether this problem is fixed-parameter tractable. As such, the problem was one of the first open problems in above-guarantee parameterizations.

The problem remained open until 2012, when Crowston, Jones and Mnich [18] proved that it is indeed fixed-parameter tractable and has a polynomial kernel. In that paper, we gave an algorithm which constructed a set of vertices S, such that if (G, k) is not a Yes-instance then $|S| < 12k$. Moreover, $G - S$ is a graph in which every block is a clique - that is, $G - S$ is a forest of cliques. This structure is then used as the basis of the FPT algorithm and polynomial kernel.

Edwards' result is generalised by a result of Poljak and Turzík [61], who proved a related bound for a more general class of problems. Poljak and Turzík introduced the concept of λ-extendible properties, where $0 < \lambda < 1$. A number of graph properties are λ-extendible for some λ; in particular, bipartiteness is a $\frac{1}{2}$-extendible property. Poljak and Turzík proved that for every λ-extendible property Π, a connected graph with n vertices and m edges contains a graph with property Π that has at least $\lambda m + \frac{1-\lambda}{2}n$ edges. The bound extends to weighted graphs: Recall that $\tau(G)$ denotes the minimum weight of a spanning tree of G. Then any graph G with weight function w contains a subgraph with property Π and weight $\lambda w(G) + \frac{1-\lambda}{2}\tau(G)$. In what follows, we assume that all weights are integer. Note that when $\lambda = \frac{1}{2}$ and all edges have weight 1, $\lambda w(G) + \frac{1-\lambda}{2}\tau(G)$ is exactly the Edwards-Erdős bound.

In the paper [18], the construction of the set S depends only on facts about bipartiteness which are common to all 1/2-extendible properties. Motivated by this observation, Mnich, Philip, Saurabh, and Suchý [55] considered a generalisation of the problem considered in [18], in which we search for a subgraph with some λ-extendible property rather than a bipartite subgraph, and in which the Edward-Erdős bound is replaced by the Poljak-Turzík bound. They proved that this problem is fixed-parameter tractable given two further conditions on Π. The first condition is that the problem is fixed-parameter on graphs which are close to being a forest of

[4] Note that when $\frac{m}{2} + \frac{n-1}{4}$ is not an integer, this is equivalent to asking if there is a bipartite subgraph with at least $\lceil \frac{m}{2} + \frac{n-1}{4} \rceil + k$ edges, as long as k is an integer.

cliques. This condition corresponds to the second part of the proof in [18]. The second condition is that Π is *strongly* λ-extendible; this is a slightly stronger condition than being λ-condition. As bipartiteness satisfies both these conditions, the result of Mnich et.al. generalises the FPT result of [18]. The same paper shows that a number of graph properties satisfy both conditions, including the acyclic subgraph property in oriented graphs, and the balanced subgraph property in signed graphs, (which generalises the bipartite subgraph property). The results of both [18] and [55] apply only to unweighted graphs.

In Chapter 2, we extend the results of [55] to weighted graphs. The underlying ideas are similar to those in [18] and [55]; however the methods used are complicated by the need to keep track of the minimum weight subtree of the graph. In place of a forest of cliques, we have a *uniform forest of cliques* - a graph in which each block is a clique, and the weights within each block are the same.

The fixed-parameter results given above do not imply any polynomial-size kernel. In Chapters 3 and 4, we consider the specific 1/2-extendible properties Maximum Balanced Subgraph and Maximum Acyclic Subgraph parameterized above the Poljak-Turzík bound. We first show that the results of Chapter 2 apply to these problems, and that they are therefore fixed-parameter tractable. We then prove that the unweighted versions of these problems have polynomial kernels.

We now give an outline of the methods used in [18], as similar methods will be used in the following chapters.[5]

Given a graph G, let $\beta(G)$ denoted the maximum number of edges of a bipartite subgraph of G, and let $\gamma(G)$ denote the Edwards-Erdős bound on G - i.e. $\gamma(G) = \frac{|E(G)|}{2} + \frac{|V(G)|-1}{4}$. Thus, for an instance (G, k), our aim is to determine whether $\beta(G) \geq \gamma(G) + k$,

Consider the path with three vertices and two edges, P_3. Observe that $\gamma(P_3) = \frac{3}{2}$ but, as P_3 itself is a bipartite graph, $\beta(P_3) = 2$. Now consider a connected graph G with a set of vertices X such that $G' = G - X$ is connected and $G[X] = P_3$. Let H' be a maximum bipartite subgraph of G'. Observe that we can create a bipartite subgraph H of G using the edges of H', the edges of $G[X]$, and at least half of the edges between X and $G - X$. (Indeed, let H' be a (V_1, V_2)-bipartite graph and $G[X]$ a (V_3, V_4)-bipartite graph. Then we can add either $E(V_1, V_3) \cup E(V_2, V_4)$ or $E(V_1, V_4) \cup E(V_2, V_3)$ to H' and $G[X]$, and either will produce a bipartite graph. But one of $E(V_1, V_3) \cup E(V_2, V_4)$ and $E(V_1, V_4) \cup E(V_2, V_3)$ contains at least half of

[5]Note that the results of [18] are extended in this thesis by the results of Chapter 3, which concerns a weighted version of a generalisation of Max-Cut.

the edges between H' and $G[X]$. It follows that if $|E(H')| = \gamma(G') + k'$, then G contains a bipartite subgraph H with $|E(H)| \geq \gamma(G') + k' + \frac{|E(X,V(G)\setminus X)|}{2} + 2$. But observe that $pt(G') = \gamma(G) - \frac{(|E(X,V(G)\setminus X)|+2)}{2} - \frac{3}{4}$. It then follows that $\beta(G) \geq |E(H)| \geq \gamma(G) + k' + \frac{1}{4}$.

The key idea (in [18] and the chapters in this part of the thesis) is as follows: Find small subgraphs $G[X]$ of G (such as an induced P_3 as described above) for which $G - X$ is connected and we can prove that $\beta(G) - \gamma(G) \geq \beta(G - X) - \gamma(G - X) + c$, for some constant c. If we find such a subgraph $G[X]$, then remove X and reduce k by c. If we can do this at least k/c times, then we end up with a subgraph G' such that $\beta(G) - \gamma(G) \geq \beta(G') - \gamma(G') + k \geq k$, and we have a YES-instance. Otherwise, let S be the set of vertices we deleted. Then we know that $|S| < dk/c$, where d is the maximum number of vertices in we removed in one go, and $G - S$ does not contain any of the subgraphs we were looking for. In the case of MAX-CUT, we have that $|S| < 12k$, and $G - S$ is a forest of cliques.

(We gloss over some details here. In [18], we may delete a large number of vertices at once but only a small number are added to S. In this thesis, we present the construction of S as a recursive algorithm which may call itself on several subgraphs of G at once (in particular when G contains a cut-vertex), rather than removing subgraphs one at a time. In both cases, the underlying idea remains the same).

For the chapters in this part of the thesis, we modify the construction of S, and the resulting structure of $G - S$, to best suit the problem at hand. In Chapter 2, we modify the algorithm to work for any λ-extendible property and add an extra case to handle edge weights. We find a set S such that $G - S$ is a forest of cliques and edges within the same block have the same weight. In Chapter 3, we modify the algorithm for signed graphs, and find a set S such that $G - S$ is a forest of cliques with all edges negative (see Chapter 3 for the definitions of these terms). In Chapter 4, we modify the algorithm for oriented graphs, and find a set S such that $G - S$ is a forest of cliques and all blocks have at most 3 vertices. In every case $|S|$ is bounded by a linear function of k, or else we can prove (G, K) is a YES-instance.

Once $|S|$ has been found, we have to take advantage of $G - S$ to produce an FPT algorithm. This is normally done by guessing the structure of the maximum required subgraph restricted to $G[S]$, and then using a polynomial-time algorithm to find a optimal extension of the subgraph to the rest of G. In the case of MAX-CUT, we guess a partition of S, and then in polynomial time we are able to find the optimal partition of $V(G) \setminus S$ to produce the maximum bipartite subgraph. For full details, see Section 3.4, which deals with a weighted generalisation of MAX-CUT. In Chapter

4, we guess an ordering of the vertices of S and then find an optimal extension of this ordering to $V(G) \setminus S$ in polynomial time. Chapter 2 gives a meta-result for general λ-extendible properties, and the condition of that result is that the problem is fixed-parameter tractable when given the set S.

Finally, it remains to find a polynomial kernel. The details of the kernelization depend on the particulars of the λ-extendible property in question. However, they are also based on the existence of a set S such that $|S|$ is bounded by a function of k and $G - S$ has some specific structure. We are only able to give polynomial kernel results for the unweighted versions of specific problems. In Chapters 3 and 4, we give polynomial kernel results for the unweighted versions of Balanced Subgraph (which generalises Max-Cut) and Acyclic Subgraph parameterized above the Edwards-Erdős bound.

Chapter 2

Fixed-Parameter Tractability of λ-extendible Properties Above the Poljak-Turzík Bound

2.1 λ-extendible properties

In what follows, we think of a graph property Π as a class of graphs, and we say a graph G is a Π-graph or has property Π if $G \in \Pi$.

Definition 6. *[62] Let $\mathcal{G}$ be a class of (oriented) graphs (possibly with some labelling on the edges(arcs)). Let $0 < \lambda < 1$. We say a property Π is* λ-extendible over $\mathcal{G}$ *if for every $G \in \mathcal{G}$, the following conditions hold:*

1. *If G is connected and $|V(G)| = 1$ or 2 then $G \in \Pi$.*
2. **Block additivity:** *G is in Π if and only if each of its blocks is in Π.*
3. **λ-edge extension:** *For any real-valued positive weight function w on the edges (arcs) of G, if $X \subseteq V(G)$ is such that $|X| = 2$, $G[X]$ is connected and $G - X \in \Pi$, then there exists $F \subseteq E(X, V(G) \setminus X)$ such that $w(F) \geq \lambda w(E(X, V(G) \setminus X))$, and the graph $(V(G), E(X) \cup E(G - X) \cup F)$ is in Π.*

An equivalent definition holds when $\mathcal{G}$ is a class of oriented graphs, except that we replace edges with arcs (in particular, we replace $E(X, V(G) \setminus X)$ with $A(X, V(G) \setminus X)$, where $A(X, V(G) \setminus X)$ is the set of arcs going between X and $V(G) \setminus X$ in either direction).

When the class $\mathcal{G}$ is clear from context, we will just say that Π is λ-extendible if Π is λ-extendible over $\mathcal{G}$.

For our results, we require the (possibly) less general concept of *strong λ-extendibility*. Note that the only difference between the two definitions is that in the third condition, $G[X]$ can be any connected graph in Π rather than specifically a connected graph with two vertices.[1]

Definition 7. *[55] Let $\mathcal{G}$ be a class of (oriented) graphs (possibly with some labelling on the edges(arcs)). Let $0 < \lambda < 1$. We say a property Π is* strongly λ-extendible over $\mathcal{G}$ *if for every $G \in \mathcal{G}$, the following conditions hold:*

1. *If G is connected and $|V(G)| = 1$ or 2 then $G \in \Pi$.*
2. **Block additivity:** *G is in Π if and only if each of its blocks is in Π.*
3. **Strong λ-subgraph extension:** *For any real-valued positive weight function w on the edges (arcs) of G, if $X \subseteq V(G)$ is such that $G[X]$ is connected and $G[X], G - X \in \Pi$, then there exists $F \subseteq E(X, V(G) \setminus X)$ such that $w(F) \geq \lambda w(E(X, V(G) \setminus X))$, and the graph $(V(G), E(X) \cup E(G - X) \cup F)$ is in Π.*

As with λ-extendibility, there is an equivalent definition of strong λ-extendibility over classes of oriented graphs.

We now give some examples of graph properties which are strongly λ-extendible for some λ.

Lemma 8. *Let Π be the class of all bipartite graphs. Then Π is $\frac{1}{2}$-extendible.*

Proof. We show that Π satisfies each of the conditions of strong $1/2$-extendibility in turn.

If $|V(G)| = 1$ or 2, then clearly G is a bipartite subgraph.

To see that Π satisfies block additivity, let $X_1, \ldots, X_l$ be the blocks of a graph G. First observe that if G is a (V_1, V_2)-bipartite graph, then $G[X_i]$ is a $(V_1 \cap X_i, V_2 \cap X_i)$-bipartite graph for each i. So now assume that $G[X_i]$ is a $(X_{i,1}, X_{i,2})$-bipartite subgraph for each i. Observe that if $v \in X_{i,h} \cap X_{j,l}$ for $i \neq j$, we may assume that $h = l$ as v is a cut-vertex. Then let $V_1 = \bigcup_i X_{i,1}$ and $V_2 = \bigcup_j X_{j,2}$, and observe that G is a (V_1, V_2)-bipartite graph.

It remains to show that Π satisfies strong $\frac{1}{2}$-subgraph extension. Suppose $G[X]$ is a (X_1, X_2)-bipartite graph and $G[Y]$ is a (Y_1, Y_2)-bipartite subgraph, where $Y = V \setminus X$. If $w(E(X_1, Y_2) \cup E(X_2, Y_1)) \geq w(E(X, Y))/2$, then let $F = E(X_1, Y_2) \cup$

[1] In fact, for the results in this chapter, it is enough for the third condition to allow $G[X]$ to be any connected graph with up to three vertices. However, we keep the stronger third condition as this it is how strong-λ-extendibility is defined in existing literature.

$E(X_2, Y_1)$. Then $w(F) \geq w(E(X,Y))/2$, and $(V, E(X) \cup E(Y) \cup F)$ is a $(X_1 \cup Y_1, X_2 \cup Y_2)$-bipartite graph. Otherwise, let $F = E(X_1, Y_1) \cup E(X_2, Y_2)$. Then $w(F) \geq w(E(X,Y))/2$, and $(V, E(X) \cup E(Y) \cup F)$ is a $(X_1 \cup Y_2, X_2 \cup Y_1)$-bipartite graph. □

Lemma 9. *Let Π be the class of all acyclic oriented graphs. Then Π is $\frac{1}{2}$-extendible over the class of oriented graphs.*

Proof. Again, we show that Π satisfies each of the conditions of strong $1/2$-extendibility in turn.

If $|V(G)| = 1$ or 2, then clearly G is acyclic.

To see that Π satisfies block additivity, observe that a directed cycle is a 2-connected subgraph in the underlying undirected graph, and therefore any directed cycle is contained within a block. Therefore an oriented graph D contains a directed cycle if and only if one of its blocks contains a directed cycle.

It remains to show that Π satisfies strong $\frac{1}{2}$-subgraph extension. Let G be an oriented graph with weight function w, and suppose that $G[X]$ and $G-X$ are acyclic subgraphs. If $w(A^+(X, V \setminus X) \geq w(A^-(X, V \setminus X))$ then let $F = A^+(X, V \setminus X)$ otherwise let $F = A^-(X, V \setminus X)$. In either case, $w(F) \geq w(A(X, V \setminus X))/2$ and $(V, A(X) \cup A(G - X) \cup F)$ is acyclic, as required. □

Lemma 10. *Given an integer $r \geq 2$, let Π be the class of all r-colorable graphs. Then Π is $\frac{1-r}{r}$-extendible.*

Proof. As $r \geq 2$, then if $|V(G)| = 1$ or 2, clearly G is r-colorable.

To see that Π satisfies block additivity, let $X_1, \ldots, X_l$ be the blocks of a graph G. First observe that if G is an r-colorable graph, then $G[X_i]$ is an r-colorable graph for each i. So now assume that $G[X_i]$ is an r-colorable subgraph for each i, and for each h, let $X_{i,h}$ denote the vertices of X_i which are assigned color h. Observe that if $v \in X_{i,h} \cap X_{j,l}$ for $i \neq j$, we may assume (by recoloring X_j) that $h = l$, as v is a cut-vertex. Thus, there is an r-coloring of G which is a proper r-coloring of each block, and is thus a proper r-coloring of G.

Finally, we show that Π satisfies strong $\frac{r-1}{r}$-subgraph extension. Suppose $G[X]$ and $G[Y]$ are r-colorable subgraphs, where $Y = V \setminus X$, and fix a proper r-coloring of X and a proper r-coloring of Y. Let X_i denote the vertices of X which are assigned color i, and similarly let Y_i denote the vertices of Y which are assigned color i. Now let σ be a random permutation of $[r]$, and replace the coloring on Y with one in which each vertex in Y_i is assigned color $\sigma(i)$. Observe that this is still a proper r-coloring of Y. Let F be the set of edges between X and Y which are properly

colored under this coloring. Thus, $(V, E(X) \cup E(Y) \cup F)$ is an r-colorable subgraph of G. Observe that an edge between $x \in X_i$ and $y \in Y_j$ is in F if and only $i \neq \sigma(j)$. Thus, the probability of an edge being in F is $\frac{r-1}{r}$. It follows that the expected value of $w(F)$ is $\frac{r-1}{r}w(E(X,Y))$, and therefore there is some choice of σ for which $w(F) \geq \frac{r-1}{r}w(E(X,Y))$, as required. □

Note that while we have been unable to prove that λ-extendibility and strong λ-extendibility are equivalent, we have also been unable to find a graph property Π such that Π is λ-extendible but not strongly λ-extendible, for some λ. Thus, it is an interesting open question whether λ-extendibility and strong λ-extendibility are equivalent.

2.1.1 Note on definitions

Recall that in the definition of strong λ-subgraph extension in Definition 7, we require that $G[S]$ is connected and in Π, but allow S to be of any size (whereas in Definition 6 we required that $|S| = 2$). However, in the results of this section, we only use strong λ-subgraph extension in cases where the set S has three vertices or fewer. Thus, we could replace Definition 6 with a more specific definition, in which strong λ-subgraph extension is only required when $|S| \leq 3$.

There are two reasons we do not do this. The first reason is for consistency with the paper [55] in which strong λ-extendibility is first defined. The second is that we believe that a polynomial kernel result will be possible for Π-SAPT, but that this will require the more general version of strong λ-subgraph extension. (Indeed, for the specific $\frac{1}{2}$-extendible properties considered in the next two chapters, we are able to prove polynomial kernel results but the proofs make use of the general version of strong $\frac{1}{2}$-extendibility.)

The requirement that $G[S]$ is connected is again something we keep to preserve consistency with [55]. Note however, that strong λ-extendibility implies a version in which $G[S]$ is not required to be connected. That is, for any strongly λ-extendible property Π, given a set $S \subseteq V$ such that $G[S]$ and $G - S$ are in Π, there exists $F \subseteq E(X, V(G) \setminus X)$ such that $w(F) \geq \lambda w(E(X, V(G) \setminus X))$, and the graph $(V(G), E(X) \cup E(G - X) \cup F)$ is in Π, even if $G[S]$ is not connected. This can be seen by letting $S_1, \ldots, S_l$ be the components of $G[S]$, letting $G_i = G - (\bigcup_{j>i} S_j)$, and applying strong λ-extendibility on G_i and S_i, for each $i \in [l]$ in turn.

2.2 The Poljak-Turzík bound and Π-SAPT

We are now ready to give the Poljak-Turzík bound, which provides a lower bound on the weight of a maximum Π-subgraph of a graph, for any λ-extendible property Π. Recall that for a connected weighted graph G, we denote the weight of a minimum weight spanning tree of G by $\tau(G)$.

Theorem 11 (Poljak-Turzík Bound)**.** *[61] Let $\mathcal{G}$ be a class of (oriented) graphs (possibly with some labelling on the edges(arcs)). Let Π be a λ-extendible property over $\mathcal{G}$ and G a connected (oriented) graph in $\mathcal{G}$, with w a real-valued positive weight function on the edges (arcs) of G. Then there exists a Π-subgraph H of G such that $w(H) \geq \lambda w(G) + \frac{1-\lambda}{2}\tau(G)$.*

Theorem 11 extends to non-connected graphs; we simply replace $\tau(G)$ with the weight of a minimum spanning *forest* of G. Throughout this part of the thesis we concern ourselves only with connected graphs, simply for convenience of some of the proofs. All of the results in this part can be extended to non-connected graphs simply by considering each component separately.

Definition 12. *Let Π be a property which is λ-extendible over a class $\mathcal{G}$ and let G be a connected graph in $\mathcal{G}$.*

Then we define

$$\beta(G) = \max\{w(H) : H \text{ is a } \Pi\text{-subgraph of } G\}$$

and

$$\gamma(G) = \lambda w(G) + \frac{1-\lambda}{2}\tau(G).$$

Note that $\gamma(G)$ is the Poljak-Turzík bound on G, and so $\beta(G) \geq \gamma(G)$.

Given a property Π which is λ-extendible over a graph class $\mathcal{G}$, we define the classical problem Π-SUBGRAPH as follows:

Π-SUBGRAPH
Instance: A connected graph $G \in \mathcal{G}$ with weight function $w : E \to \mathbb{N}$, an integer p
Question: Is $\beta(G) \geq \gamma(G) + p$?

Given a property Π which is λ-extendible over a class $\mathcal{G}$, we define the parameterized problem Π-SUBGRAPH ABOVE POLJAK-TURZÍK BOUND as follows:

Π-SUBGRAPH ABOVE POLJAK-TURZÍK BOUND (Π-SAPT)
Instance: A connected graph $G \in \mathcal{G}$ with weight function $w : E \to \mathbb{N}$, an integer k
Parameter: k.
Question: Is $$\beta(G) \geq \gamma(G) + k?$$

Again, the problems Π-SUBGRAPH and Π-SAPT have equivalent definitions when Π is a property of oriented graphs, with the only difference being that w is a weight function on the arcs and not the edges.

Note that in the definition of Π-SAPT we require all weights to be integers, although the Poljak-Turzík holds without any such restrictions. However, if we did not have this requirement then it is very unlikely we'd be able to get any FPT results. Indeed, let REALWEIGHTED-Π-SAPT be the problem Π-SAPT in which there is no requirement for the weight function to be integer. Then the following lemma shows that if Π-SUBGRAPH is NP-hard, REALWEIGHTED-Π-SAPT is not fixed-parameter tractable unless $P = NP$.

Lemma 13. *Let* Π *be a* λ*-extendible property such that* Π-SUBGRAPH *is NP-hard. Then* REALWEIGHTED-Π-SAPT *is para-NP hard.*

Proof. Assume for convenience that Π is λ-extendible over the class of undirected unlabelled graphs. We will give a reduction that transforms any instance of Π-SUBGRAPH into an instance of REALWEIGHTED-Π-SAPT with parameter 1. Consider an instance (G, p) of Π-SUBGRAPH in which we are given a graph G with weight function w, and asked whether there is a Π-subgraph with weight $\gamma(G) + p$. Now let G' be the graph G with weight function w', such that $w'(e) = w(e)/p$ for all $e \in E(G)$. Observe that $w'(G') = w(G)/p$ and $\tau(G') = \tau(G)/p$, and so $\gamma(G') = \gamma(G)$, and furthermore $\beta(G') = \beta(G)/k$. Thus, $\beta(G') \geq \gamma(G') + p$ if and only if $\beta(G) \geq \gamma(G) + 1$. □

In what follows, it will be useful to define the problem Π-SAPT-AFOC. We will go on to prove that Π-SAPT is fixed-parameter tractable for certain Π by reducing Π-SAPT to Π-SAPT-AFOC.

Π-subgraph above Poljak-Turzík Bound on Almost-Forests of Cliques(Π-SAPT-AFOC)
Instance: A connected graph $G \in \mathcal{G}$ with weight function $w : E \to \mathbb{N}$, an integer k, a set of vertices S such that $G - S$ is a uniform forest of cliques.
Parameter: $k + \lvert S \rvert$.
Question: Is $$\beta(G) \geq \gamma(G) + k?$$

Informally, we will say that a problem Π-SAPT is *FPT on almost-forests of cliques* if Π-SAPT-AFOC is FPT.

2.3 Meta-result for weighted Π-SAPT

For the remainder of this chapter, we will assume the property Π is strongly λ-extendible over a class $\mathcal{G}$, and that all graphs considered are members of $\mathcal{G}$. Our aim is to show that Π-SAPT is fixed-parameter tractable, provided that Π-SAPT is fixed-parameter on almost-forests of cliques.

For convenience we will assume that $G = (V, E)$ is an undirected graph. All results in this chapter also apply to oriented graphs; we can simply view the proofs and results as applying to the underlying undirected graph. For example, the conditions of Lemma 15 (on the next page) require that $xy, yz \in E(G)$, $w(xy) > w(yz)$ for some vertices x, y, z. In an oriented graph, this should taken to mean that there is an arc between x and y in some direction and and arc between y and z in some direction, and the weight of the arc between x and y is greater than the weight of the arc between y and z.

For a graph G, recall that $\mathcal{C}(G)$ denotes the set of all components of G.

Lemma 14. *Let G be a connected graph in $\mathcal{G}$ and v a cut-vertex of G. Then $\beta(G) = \sum_{X \in \mathcal{C}(G-v)} \beta(G[X \cup \{v\}])$.*

Proof. This follows from the block additivity property. For each $X \in \mathcal{C}(G - \{v\})$, let H_X be a maximum Π-subgraph of $G[X \cup \{v\}]$ and let $H = \bigcup_{X \in \mathcal{C}(G-\{v\})} H_X$. Note that every block in H is a block in H_X for some X. Then by block additivity, H is a Π-subgraph of G, and so $\beta(G) \geq \sum_{X \in \mathcal{C}(G-v)} \beta(G[X \cup \{v\}])$. Conversely, given a maximum Π-subgraph H of G, for each $X \in \mathcal{C}(G - \{v\})$ let $H_X = H[X \cup \{v\}]$, and observe that by block additivity, H_X is a Π-subgraph of $G[X \cup \{v\}]$. Therefore $\beta(G) \leq \sum_{X \in \mathcal{C}(G-v)} \beta(G[X \cup \{v\}])$, completing the proof. □

Lemma 15. *Suppose G contains vertices x, y, z such that $xy, yz \in E(G)$, $w(xy) > w(yz)$ and $G' = G - \{x, y\}$ is connected. Then $\beta(G) - \gamma(G) \geq \beta(G') - \gamma(G') + \frac{(1-\lambda)}{2}(w(xy) - w(yz))$.*

Proof. The graph $G[\{x, y\}]$ is in Π as it contains two vertices. Then by the strong λ-subgraph extension property, $\beta(G) \geq w(xy) + \beta(G') + \lambda w(E(\{x, y\}, V(G) \setminus \{x, y\}))$. Observe that we can form a spanning tree of G by taking a minimum weight spanning tree of G' and adding the edges xy, yz, and so $\tau(G) \leq \tau(G') + w(xy) + w(yz)$. Therefore $\gamma(G) \leq \gamma(G') + \lambda(w(xy) + w(E(\{x, y\}, V(X) \setminus \{x, y\})) + \frac{1-\lambda}{2}(w(xy) + w(yz))$. Then

$$\begin{aligned}
\beta(G) - \gamma(G) &\geq w(xy) + \beta(G') + \lambda w(E(\{x, y, \}, V(X) \setminus \{x, y\})) - \gamma(G') \\
&\quad -\lambda(w(xy) + w(E(\{x, y\}, V(X) \setminus \{x, y\})) - \frac{1-\lambda}{2}(w(xy) + w(yz)) \\
&= \beta(G') - \gamma(G') + (1-\lambda)w(xy) - \frac{1-\lambda}{2}(w(xy) + w(yz)) \\
&= \beta(G') - \gamma(G') + \frac{1-\lambda}{2}(w(xy) - w(yz)).
\end{aligned}$$

□

Recall that an *induced* P_3 is set of vertices $\{a, b, c\}$ such that $G[\{a, b, c\}]$ is isomorphic to P_3, i.e. $ab \in E(G), bc \in E(G), ac \notin E(G)$.

Lemma 16. *Suppose G contains an induced P_3 $\{a, b, c\}$ such that $G' = G - \{a, b, c\}$ is connected, $w(ab) = w(bc)$ and there exist $xz \in E(G)$ with $x \in \{a, b, c\}, z \notin \{a, b, c\}$ such that $w(xz) \leq w(ab)$. Then $\beta(G) - \gamma(G) \geq \beta(G') - \gamma(G') + \frac{(1-\lambda)}{2}w(ab)$.*

Proof. Let $\{a, b, c\}$ be an induced P_3 in G, such that $ab, bc \in E(G)$ and $ac \notin E(G)$. By the block additivity property and the fact that $G[a, b], G[b, c] \in \Pi$, we have that $G[\{a, b, c\}] \in \Pi$. Therefore $\beta(G) \geq 2w(ab) + \beta(G') + \lambda w(E(\{a, b, c\}, V(G) \setminus \{a, b, c\}))$. Observe that we can form a spanning tree of G by taking a minimum weight spanning tree of $G - \{a, b, c\}$ and adding the edges xz, ab, bc, and so $\tau(G) \leq \tau(G') + 3w(ab)$. Therefore $\gamma(G) \leq \gamma(G') + \lambda(2w(ab) + w(E(\{a, b, c\}, V(G) \setminus \{a, b, c\}))) + \frac{1-\lambda}{2}3w(ab)$.

Then

$$\begin{aligned}
\beta(G) - \gamma(G) &\geq \beta(G') + 2w(ab) + \lambda w(E(\{a, b, c\}, V(G) \setminus \{a, b, c\})) - \gamma(G') \\
&\quad -\lambda(2w(ab) + w(E(\{a, b, c\}, V(G) \setminus \{a, b, c\}))) - \frac{1-\lambda}{2}3w(ab) \\
&= \beta(G') - \gamma(G') + (1-\lambda)2w(ab) - \frac{1-\lambda}{2}3w(ab) \\
&= \beta(G') - \gamma(G') + \frac{1-\lambda}{2}w(ab).
\end{aligned}$$

□

We now show that in any connected graph G, either one of the conditions from Lemmas 14, 15 or 16 holds, or G is a clique with uniform weights.

Lemma 17. *Let G be a non-empty connected weighted graph. Then one of the following holds:*

1. *G contains a cut-vertex.*
2. *G contains vertices x, y, z such that $xy, yz \in E(G)$, $w(xy) > w(yz)$ and $G - \{x, y\}$ is connected.*
3. *G contains an induced P_3 $\{a, b, c\}$ such that $G = \{a, b, c\}$ is connected, and there exist $xz \in E(G)$ with $x \in \{a, b, c\}, z \notin \{a, b, c\}$ such that $w(xz) \leq w(ab)$.*
4. *G is a uniform clique.*

Proof. If the connectivity of G is 1, then Case 1 holds, so we may assume the connectivity of G is at least 2. Consider first the case when G has connectivity exactly 2. Let b, v be two vertices such that $G - \{b, v\}$ is disconnected, and observe that v is a cut-vertex for $G - b$. Therefore $G - v$ has at least two blocks, and in particular at least two leaf-blocks. Furthermore, every leaf-block must contain an internal vertex adjacent to b, as otherwise the root of that leaf-block is a cut-vertex for G. So now let a, c be vertices such that a and c are internal vertices of different leaf-blocks in $G - b$, and both a and c are adjacent to b. Then observe that $\{a, b, c\}$ is an induced P_3, and $G - \{a, b, c\}$ is connected.

If there exists an edge between $\{a, b, c\}$ and $G - \{a, b, c\}$ with weight at most $w(ab)$, then Case 3 applies. So now assume that $w(xz) > w(ab)$ for all $x \in \{a, b, c\}, z \notin \{a, b, c\}$. We will show that a has a neighbour $z \notin \{b, c\}$ such that $G - \{a, z\}$ is connected. Since $w(az) > w(ab)$, it follows that Case 2 holds.

Recall that $R(c, X)$ is the set of vertices connected to c in the graph $G - X$, for any set of vertices not containing c. Consider the partial order $\prec$ on $N(a) \setminus \{b, c\}$, such that $z \prec z'$ if $R(c, \{a, z\})$ is a strict subset of $R(c, \{a, z'\})$. We call z an *important neighbor* of a if z is a maximal element in this partial order. That is, z is an important neighbor of a if $az \in E(G), z \notin \{b, c\}$, and $R(c, \{a, z\}) \not\subset R(c, \{a, z'\})$ for any $z' \notin \{b, c\}$ with $az' \in E(G)$. The vertex a must have a neighbour in $V(G) \setminus \{a, b, c\}$ as otherwise b is a cut-vertex for G, a contradiction. So it follows that a has an important neighbour. Let z be an important neighbour of a. Note that z is adjacent to a vertex in $R(c, \{a, z\})$, as otherwise a is a cut-vertex for G. Suppose $G - \{a, z\}$ is not connected, and let X be a component of $G - \{a, z\}$ not containing c. If X contains a vertex z' adjacent to a, then z' is a neighbour of a not

in $\{a, b, c\}$ and $R(c, \{a, z'\}) \supseteq R(c, \{a, z\}) \cup \{z\}$, a contradiction as z is an important neighbour of a. On the other hand if X contains no vertices adjacent to a, then z is a cut-vertex of G, a contradiction. So we must have that $G - \{a, z\}$ is connected, as required.

Now we consider the case when G has connectivity at least 3. Since removing any pair of vertices leaves the graph connected, we may assume that every pair of edges that share a vertex have the same weight, as otherwise Case 2 holds. As G is connected, it follows that all edges in G have the same weight.

Observe that any connected graph which is not a clique contains an induced P_3 (consider the shortest path between any two vertices which are not adjacent, and observe that any three consecutive vertices in this path form an induced P_3). Since all edges have the same weight, it follows that either Case 4 applies or G contains an induced P_3. So now assume G contains an induced P_3. We will show that G contains an induced P_3 $\{a, b, c\}$ such that $G - \{a, b, c\}$ is connected.

Let r be an arbitrary vertex in $V(G)$. Consider the partial order $\prec$ on the induced P_3's in G, such that $\{a, b, c\} \prec \{a', b', c'\}$ if $R(r, \{a, b, c\})$ is a strict subset of $R(r, \{a', b', c'\})$. We call a set of vertices $\{a, b, c\}$ an *important* P_3 if $\{a, b, c\}$ is a maximal element in this partial order. That is, $\{a, b, c\}$ is an important P_3 if $\{a, b, c\}$ is an induced P_3 and $R(r, \{a, b, c\}) \not\subset R(r, \{a', b', c'\})$, for any induced P_3 $\{a', b', c'\}$. Let $\{a, b, c\}$ be an important P_3 with $ab, bc \in E(G), ac \notin E(G)$. If $r \in \{a, b, c\}$ then $G - r$ contains no induced P_3, and so $G - r$ must be a clique (as $G - r$ must be connected), and therefore $G - \{a, b, c\}$ is connected. So now assume $r \notin \{a, b, c\}$. Assume for a contradiction that $G - \{a, b, c\}$ is not connected and let X be a component of $G - \{a, b, c\}$ not containing r. Since G is 3-connected both $(r, \{a, b, c\})$ and X must have vertices adjacent to each of a, b, c. Therefore there must be a path $p_0, p_1, \ldots, p_l$, where $p_0 = a, p_l = c$, and $p_i \in X$ for all $1 \leq i < l$. By taking a shortest such path, and considering three consecutive vertices on this path, we have an induced P_3 $\{a', b', c'\}$ where $R(r, \{a', b', c'\}) \supseteq R(r, \{a, b, c\}) \cup \{b\}$, a contradiction as $\{a, b, c\}$ is an important P_3.

Thus $G - \{a, b, c\}$ is connected. As all weights are equal, Case 3 holds. □

We are now ready to describe the main algorithm, which will either help us to either decide that (G, k) is a YES-instance or find a set of at most $\frac{6k}{(1-\lambda)}$ vertices which when removed leave behind an almost-forest of cliques.

Consider algorithm ALG.
input : A connected graph G

output: A set $S \subseteq V(G)$; a non-negative real number t

Case 1: if *G contains a cut-vertex v* **then**
- **foreach** *Component X of $G - v$* **do**
  - Set $(S', t') = \text{ALG}(G[X \cup \{v\}])$;
  - Set $S = S \cup S'$;
  - Set $t = t + t'$;
- **end**

else
- **Case 2: if** *G contains vertices x, y, z such that $xy, yz \in E(G)$, $w(xy) > w(yz)$ and $G - \{x, y\}$ is connected* **then**
  - Set $G' = G - \{x, y\}$;
  - Set $(S', t') = \text{ALG}(G')$;
  - Set $S = S' \cup \{x, y\}$;
  - Set $t = t' + \frac{(1-\lambda)(w(xy)-w(yz))}{2}$;
- **else**
  - **Case 3: if** *G contains an induced P_3 $\{a, b, c\}$ such that $G - \{a, b, c\}$ is connected, and there exist $xz \in E(G)$ with $x \in \{a, b, c\}, z \notin \{a, b, c\}$ such that $w(xz) \leq w(ab)$* **then**
    - Set $G' = G - \{a, b, c\}$;
    - Set $(S', t') = \text{ALG}(G')$;
    - Set $S = S' \cup \{a, b, c\}$;
    - Set $t = t' + \frac{(1-\lambda)w(ab)}{2}$;
  - **else**
    - **Case 4:** Set $S = \emptyset$;
    - Set $t = 0$;
  - **end**
- **end**

end

return (S, t);

Algorithm 1: Algorithm ALG

Lemma 18. *Algorithm* ALG *can be run in polynomial time.*

Proof. Observe that each recursive call of ALG either calls ALG once on an instance with fewer vertices, or calls it on a set of at least two subgraphs which partition the edges of the graph. Therefore, for a graph with n vertices, m edges, there will be at most $2mn$ recursive calls of of ALG. Observe that a single iteration of ALG can be

run in $O(n^3)$ time. Therefore in total ALG(G) takes $O(mn^4)$ time. □

Observe that at each iteration of the algorithm, if we add any vertices to S, we add at most 3 vertices to S and increase t by at least $\frac{(1-\lambda)}{2}$. Therefore we have the following lemma.

Lemma 19. *Let* $(S,t) = \text{ALG}(G)$. *Then* $|S| \leq \frac{6t}{1-\lambda}$.

Lemma 20. *Let* $(S,t) = \text{ALG}(G)$. *Then* $\beta(G) \geq \gamma(G) + t$.

Proof. We prove this by induction on $|G|$. Observe that if $|G| = 0$ then $t = 0$, and so $\beta(G) \geq \gamma(G) + t$ is true by definition. So now suppose that $|G| = n$, and claim is true for smaller graphs. If Case 1, 2 or 3 of the algorithm holds, then the claim follows from Lemmas 14, 15 and 16 respectively. (Note that for Case 3, since Case 2 does not apply we may assume $w(ab) = w(bc)$, and as a result Lemma 16 applies.) Otherwise, $t = 0$, and so $\beta(G) \geq \gamma(G) + t$ is true by definition. □

Lemma 21. *Let* $(S,t) = \text{ALG}(G)$. *Then* $G - S$ *is a forest of uniform cliques.*

Proof. In what follows, it will be useful to note that if $G' - S'$ is a forest of cliques for some subgraph G' of G and $S' \subseteq S$, then $G' - S$ is also a forest of cliques.

We prove the lemma by induction on $|G|$. Observe that if $|G| = 0$ then $G - S$ is the empty graph and is therefore a forest of uniform cliques. So now suppose that $|G| = n$, and claim is true for smaller graphs. We consider the different cases of the algorithm separately.

Case 1: For each component X of $G - v$, let $G_X = G[X \cup \{v\}]$ and let $(S_X, t_X) = \text{ALG}(G_X)$. By the inductive hypothesis, $G_X - S_X$ is a forest of uniform cliques and so $G_X - S$ is also a forest of cliques. Since $G - S$ is formed either by taking the disjoint union of all $(G_X - S)$ (if $v \in S$), or by joining all $(G_X - S)$ at a single vertex (if $v \notin S$), it follows that $G - S$ is also a forest of uniform cliques.

Cases 2 and 3: Observe that since $S = S'$ together with all the vertices in G that are not in G', we have that $G - S = G' - S'$, which is a forest of uniform cliques by the inductive hypothesis.

Case 4: In this case none of Cases 1,2,3 apply. Then by Lemma 17, G is a uniform clique and so $G - S$ is also a uniform clique. □

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

Theorem 22. *Let* Π *be a property which is strongly* λ*-extendible over a class* $\mathcal{G}$*, such that* Π-SAPT-AFOC *is fixed-parameter tractable. Then* Π-SAPT *is fixed-parameter tractable.*

Proof. Assume that there is an algorithm that solves any instance (G, k, S) of Π-SAPT-AFOC in time $f(k, |S|)n^{O(1)}$. Given an instance (G, k) of Π-SAPT, in polynomial time run algorithm ALG to get $(S, t) = \text{ALG}(G)$. If $t \geq k$ then by Lemma 20, (G, k) is a YES-instance. Otherwise, by Lemma 19, $|S| < \frac{6k}{1-\lambda}$, and by Lemma 21, $G - S$ is a uniform forest of cliques. Then we can solve the Π-SAPT-instance (G, k) in time $f(k, \frac{6k}{1-\lambda})n^{O(1)}$ by solving the Π-SAPT-AFOC instance (G, k, S). $\square$

Chapter 3

Balanced Subgraph Problem Parameterized Above the Poljak-Turzík Bound

3.1 Definitions

A *signed graph* is an undirected graph in which every edge is labelled by + or −. An edge is *positive* (*negative*) if it is labelled by + (−). The labels + and − are the *signs* of the corresponding edges.

For the edge set E of a signed graph G, E^+ and E^- denote the set of positive and negative edges of F, respectively. For a signed graph $G = (V, E)$, the *dual* of G is the signed graph $\bar{G} = (V, \bar{E})$, where $\bar{E}^+ = E^-$ and $\bar{E}^- = E^+$. In other words, $\bar{G}$ is the graph G with all positive edges replaced with negative edges, and all negative edges replaced with positive edges.

For a set of vertices $W \subseteq V$, the *positive* neighbors of W are the neighbors of W in $G^+ = (V, E^+)$; the set of positive neighbors is denoted $N_G^+(W)$. The *negative* neighbors of W are defined similarly and denoted $N_G^-(W)$.

A cycle C in G is called *positive* (*negative*) if the number of negative edges in C is even (odd)[1]. A cycle in G is *dually positive* (*dually negative*) if the same cycle in $\bar{G}$ is positive (negative). [2]

[1] To obtain the sign of C simply compute the product of the signs of its edges.

[2] Note that a positive cycle may be either dually positive or dually negative. For example, a cycle of length 3 with all positive edges is positive and dually negative, while a cycle of length 4 with all positive edges is positive and dually positive. Similarly, a negative cycle may be dually positive or dually negative (we leave this as an exercise for the reader).

Definition 23. *Let $G = (V, E)$ be a signed graph and let $V = V_1 \cup V_2$ be a partition of V. Then G is* (V_1, V_2)-balanced *if an edge with both endpoints in V_1, or both endpoints in V_2 is positive, and an edge with one endpoint in V_1 and one endpoint in V_2 is negative; G is* balanced *if it is (V_1, V_2)-balanced for some partition V_1, V_2 of V (V_1 or V_2 may be empty).*

Observe that the negative edges of a balanced graph form a bipartite subgraph; thus, the problem of finding a bipartite subgraph is equivalent to the problem of finding a balanced subgraph when all edges are negative.

Signed graphs are well-studied due to their various applications and interesting theoretical properties, see, e.g., [12, 20, 33, 38, 41, 43, 70]. In some applications, we are interested in finding a maximum-size balanced subgraph of a signed graph [12, 20, 43, 70]. This is the problem Balanced Subgraph. Observe that Balanced Subgraph is equivalent to Max-Cut when all edges of G are negative. Thus Balanced Subgraph generalizes Max-Cut, and as such is NP-hard

Hüffner *et al.* [43] parameterized Balanced Subgraph below a tight upper bound: decide whether $G = (V, E)$ contains a balanced subgraph with at least $|E|-k$ edges, where k is the parameter. They showed that this problem is fixed-parameter tractable using a simple reduction to the Edge Bipartization Problem: decide whether an unsigned graph can be made bipartite by deleting at most k edges (where k is the parameter). Using this result and a number of heuristic reductions, Hüffner *et al.* [43] designed a nontrivial practical algorithm that allowed them to exactly solve several instances of Balanced Subgraph that were previously solved only approximately (under the name Undirected Labeling Problem) by DasGupta *et al.* [20].

In this chapter, we consider Balanced Subgraph parameterized above the Poljak-Turzík bound. First, we must show that the Poljak-Turzík bound applies for this problem.

Lemma 24. *Let Π be the class of all balanced graphs. Then Π is strongly $1/2$-extendible over the class of signed graphs.*

Proof. We show that Π satisfies each of the conditions of strong $1/2$-extendibility in turn.

If $|V(G)| = 1$ or 2, then clearly G is a balanced subgraph. (If $V(G) = 1$ then there are no edges so the graph is trivially balanced; if $V(G) = \{x, y\}$ and there is an edge between x and y, then G is $(\{v_1, v_2\}$-balanced if the edge is negative, and $(\{x, y\}, \emptyset)$-balanced if the edge is positive.)

To see that Π satisfies block additivity, let $X_1, \ldots, X_l$ be the blocks of a graph G. First observe that if G is a (V_1, V_2)-balanced graph, then $G[X_i]$ is a $(V_1 \cap X_i, V_2 \cap X_i)$-balanced graph for each i. So now assume that $G[X_i]$ is a $(X_{i,1}, X_{i,2})$-balanced subgraph for each i. Observe that if $v \in X_{i,h} \cap X_{j,l}$ for $i \neq j$, we may assume that $h = l$ as v is a cut-vertex. Then let $V_1 = \bigcup_i X_{i,1}$ and $V_2 = \bigcup_j X_{j,2}$, and observe that G is a (V_1, V_2)-balanced graph.

It remains to show that Π satisfies strong $\frac{1}{2}$-subgraph extension. Suppose $G[X]$ is a (X_1, X_2)-balanced graph and $G[Y]$ is a (Y_1, Y_2)-balanced subgraph, where $Y = V \setminus X$. If $w(E^+(X_1, Y_1) \cup E^+(X_2, Y_2) \cup E^-(X_1, Y_2) \cup E^-(X_2, Y_1)) \geq w(E(X, Y))/2$, then let $F = E^+(X_1, Y_1) \cup E^+(X_2, Y_2) \cup E^-(X_1, Y_2) \cup E^-(X_2, Y_1)$. Then $w(F) \geq w(E(X, Y))/2$, and $(V, E(X) \cup E(Y) \cup F)$ is a $(X_1 \cup Y_1, X_2 \cup Y_2)$-balanced graph. Otherwise, let $F = E^+(X_1, Y_2) \cup E^+(X_2, Y_1) \cup E^-(X_1, Y_1) \cup E^-(X_2, Y_2)$. Then $w(F) \geq w(E(X, Y))/2$, and $(V, E(X) \cup E(Y) \cup F)$ is a $(X_1 \cup Y_2, X_2 \cup Y_1)$-balanced graph. □

Lemma 24 and Theorem 11 give us the following theorem.

Theorem 25. *[62] Let $G = (V, E)$ be a connected signed graph with weight function $w : E \to \mathbb{R}^+$. Then there exists a balanced subgraph H of G such that $w(H) \geq \frac{w(G)}{2} + \frac{\tau(G)}{4}$.*

By considering only graphs with negative edges, we get the Edwards-Erdős bound, originally proved by Edwards [27] using probabilistic methods.

Theorem 26 (Edwards-Erdős Bound)**.** *[27] $G = (V, E)$ a connected graph with weight function $w : E \to \mathbb{R}^+$. Then there exists a bipartite subgraph H of G such that $w(H) \geq +\frac{w(G)}{2} + \frac{\tau(G)}{4}$.*

In this chapter, given a signed graph G, let $\beta(G)$ denote the maximum weight of a balanced subgraph of G, and let $\gamma(G)$ denote the lower bound $\frac{w(G)}{2} + \frac{\tau(G)}{4}$ on $\beta(G)$.

We consider the problem Π-SAPT when Π is the class of balanced graphs, and all input graphs are signed graphs.

Balanced Subgraph above Poljak-Turzík Bound (BSAPT)
Instance: A connected signed graph G with weight function $w : E \to \mathbb{N}^+$, an integer k
Parameter: k.
Question: Is $\beta(G) \geq \gamma(G) + k$?

Ideally, we would like to use Theorem 22 directly - that is, first prove that BSAPT is fixed-parameter tractable on almost-uniform forests of cliques, and then combine this with Theorem 22 to get fixed-parameter tractability of BSAPT as a corollary.

Unfortunately, the main motivation behind Theorem 22 - that almost-uniform forests of cliques should be much easier to deal with than the general case - does not quite apply here.

The reason is that, while a uniform forest of cliques has a very restricted structure in terms of the adjacency of vertices and the weights of edges, there is no restriction on the signs of edges. Thus, even a single clique with unit weighted edges may be difficult to reduce, as the clique could have any arrangement of positive and negative edges.

Instead of reducing a graph to an almost-uniform forest of cliques, we really want to reduce it to an almost-uniform forest of cliques in which all edges (within the forest of cliques) have the same sign. We can do this, but rather than using 22 directly, we will have to go back to the tools used to prove Theorem 22 and augment them a little.

several graph problems parameterized above a lower bound of Poljak and Turzík [62] are FPT under certain conditions.

3.2 Preliminaries

The following characterization of balanced graphs is well-known.

Theorem 27. *[41] A signed graph G is balanced if and only if every cycle in G is positive.*

Let $G = (V, E)$ be a signed graph. For a subset W of V, the *W-switch* of G is the signed graph G_W obtained from G by changing the signs of the edges between W and $V \setminus W$. That is, $G_W = (V, E_W)$, where $E^+ \cup \{uw : u \notin W, w \in W, uw \in E^-\} \setminus \{uw : u \notin W, w \in W, uw \in E^+\}$, and $E_W^- = E^- \cup \{uw : u \notin W, w \in W, uw \in E^+\} \setminus \{uw : u \notin W, w \in W, uw \in E^-\}$ Note that a signed graph G is balanced if and only if there exists a subset W of V (W may coincide with V) such that G_W has no negative edges. Indeed, if G_W has no negative edges, G is $(W, V \setminus W)$-balanced. If G is (V_1, V_2)-balanced, then G_{V_1} has no negative edges.

Deciding whether a signed graph is balanced is polynomial-time solvable.

Theorem 28. *[33] Let $G = (V, E)$ be a signed graph. Deciding whether G is balanced is polynomial-time solvable. Moreover, if G is balanced then, in polynomial time, we can find a subset W of V such that G_W has no negative edges.*

The following easy property will be very useful in later proofs. It follows from Theorem 27 by observing that for a signed graph the Poljak-Turzík bound does not depend on the signs of the edges and that, for any cycle in G, the sign of the cycle in G and in G_W is the same.

Corollary 29. *Let $G = (V, E)$ be a signed graph and let $W \subset V$. Then $\gamma(G_W) = \gamma(G)$ and $\beta(G_W) = \beta(G)$. Thus, (G, k) is a* YES*-instance of* BSAPT *if and only if (G_W, k) is a* YES*-instance of* BSAPT*.*

The next theorem is the 'dual' of Theorem 27, in the sense that it is its equivalent formulation on the dual of a graph.

Theorem 30. *Let $G = (V, E)$ be a signed graph. Then the dual graph $\bar{G}$ is balanced if and only if G does not contain a dually negative cycle.*

Note that a forest of cliques is a *chordal* graph, i.e., a graph in which every cycle of length ≥ 4 has a chord, that is an edge between two vertices which are not adjacent in the cycle. (This can be seen by observing that every cycle in a forest of cliques must be contained within a block, and each block is a clique.) The next lemma is a characterization of chordal graphs which have a balanced dual. A *triangle* is a cycle with three edges.

Corollary 31. *Let $G = (V, E)$ be a signed chordal graph. Then $\bar{G}$ is balanced if and only if G does not contain a positive triangle.*

Proof. If G contains a positive triangle, then, by Theorem 30, $\bar{G}$ is not balanced.

Now suppose that G is not balanced. By Theorem 30, G contains a dually negative cycle, i.e., a cycle with odd number of positive edges, but all triangles in G are negative by hypothesis. Let $C = v_1 v_2 \dots v_l v_1$ be a dually negative cycle of minimum length and note that $l > 3$ as a dually negative triangle is positive. Since the graph is chordal, we can find three consecutive vertices of C that form a triangle T. Suppose $T = v_1 v_2 v_3 v_1$. Recall that T is negative. So, if both $v_1 v_2$ and $v_2 v_3$ are positive edges (or negative edges), then $v_1 v_3$ must be a negative edge; otherwise if one of the two edges is positive and the other negative, then $v_1 v_3$ is a positive edge. In both cases, we conclude that C contains an odd number of positive edges if and only if $C' = v_1 v_3 v_4 \dots v_l v_1$ does, which is a contradiction since we supposed l to be the minimum length of a dually negative cycle. □

3.3 Main algorithm

We will make use of an algorithm similar to Algorithm ALG from the previous chapter with $\lambda = 1/2$. We need to make an adjustment to the algorithm so that it considers an extra case; this case is expressed by the following lemma.

Lemma 32. *Let G be a connected signed graph with weight w_e on all edges, and suppose that there exists $X \subseteq V(G)$ such that $G[X]$ is positive triangle, and $G' = G - X$ is connected. Then $\beta(G) - \gamma(G) \geq \beta(G') - \gamma(G') + 3w_e/4$.*

Proof. Observe that $G[X]$ either contains three positive edges or one positive edge and two negative edges, and in either case, $G[X]$ is a balanced graph. Therefore, by the strong 1/2-extension property $\beta(G) \geq \beta(G') + w(X) + \frac{w(E(X,V(G)\setminus X))}{2}$. Since all edges have weight w_e, $\tau(G) = w_e(|V(G)| - 1)$ and $\tau(G') = w_e(|V(G) - 4)$

Thus we have

$$\begin{aligned}
\beta(G) - \gamma(G) &= \beta(G) - \frac{w(G)}{2} - \frac{\tau(G)}{4} \\
&\geq \beta(G') + w(X) + \frac{w(E(X, V(G) \setminus X))}{2} - \frac{w(X) + w(G') + w(E(X, V(G) \setminus X))}{2} \\
&\quad - \frac{\tau(G') + 3w_e}{4} \\
&= \beta(G') + \frac{w(X)}{2} - \frac{w(G')}{2} - \frac{\tau(G')}{4} - \frac{3w_e}{4} \\
&= \beta(G') - \gamma(G') + \frac{6w_e - 3w_e}{4} \\
&= \beta(G') - \gamma(G') + \frac{3w_e}{4}
\end{aligned}$$

□

Using Lemma 32, we can now adjust the proof of Theorem 22 to work for balanced subgraphs.

Consider Algorithm BAL (given on the next page). This is the same as Algorithm ALG with $\lambda = 1/2$, except that there is an extra case added to handle positive triangles.

The proof of the following lemma is identical to the proof of Lemma 18

Lemma 33. *Algorithm BAL can be run in polynomial time.*

Observe that at each iteration of the algorithm, if we add any vertices to S, we add at most 3 vertices to S and increase t by at least 1/4. Therefore we have the following lemma.

Lemma 34. *Let $(S, t) = \text{BAL}(G)$. Then $|S| \leq 12t$.*

Lemma 35. *Let* $(S, t) = \textsc{Bal}(G)$. *Then* $\beta(G) \geq \gamma(G) + t$.

Proof. The proof is along similar lines to the proof of Lemma 20, with just one extra case to consider.

We prove the claim by induction on $|G|$. Observe that if $|G| = 0$ then $t = 0$, and so $\beta(G) \geq \gamma(G) + t$ is true by definition. So now suppose that $|G| = n$, and claim is true for smaller graphs. If Case 1, 2 or 3 of the algorithm holds, then the claim follows from Lemmas 14, 15 and 16 respectively. Otherwise, by Lemma 21, G is a uniform clique. So if Case 4 applies, then the claim follows from Lemma 32. Otherwise, $t = 0$, and so $\beta(G) \geq \gamma(G) + t$ is true by definition. □

input : A connected graph G

output: A set $S \subseteq V(G)$; a non-negative real number t

Case 1: if *G contains a cut-vertex v* **then**

  foreach *Component X of $G - v$* **do**

    Set $(S', t') = \textsc{Bal}(G[X \cup \{v\}])$;

    Set $S = S \cup S'$;

    Set $t = t + t'$;

  end

else

  Case 2: if *G contains vertices x, y, z such that $xy, yz \in E(G)$, $w(xy) > w(yz)$ and $G' = G - \{x, y\}$ is connected* **then**

    Set $(S', t') = \textsc{Bal}(G')$;

    Set $S = S' \cup \{x, y\}$;

    Set $t = t' + \frac{(w(xy) - w(yz))}{4}$;

  else

    Case 3: if *G contains an induced P_3 $\{a, b, c\}$ such that $G' = G - \{a, b, c\}$ is connected, and there exist $xz \in E(G)$ with $x \in \{a, b, c\}, z \notin \{a, b, c\}$ such that $w(xz) \leq w(ab)$* **then**

      Set $(S', t') = \textsc{Bal}(G')$;

      Set $S = S' \cup \{a, b, c\}$;

      Set $t = t' + \frac{w(ab)}{4}$;

    else

      Case 4: if *there exist vertices $\{a, b, c\} \subseteq V(G)$ such that $G[\{a, b, c\}]$ is positive triangle, and $G' = G - \{a, b, c\}$ is connected* **then**

        Set $(S', t') = \textsc{Bal}(G')$;

        Set $S = S' \cup \{a, b, c\}$;

        Set $t = t' + \frac{3w(ab)}{4}$;

      else

        Case 5: Set $S = \emptyset$;

        Set t=0;

      end

    end

  end

end

return (S, t);

Algorithm 2: Algorithm Bal

Lemma 36. *Let $(S,t) = \textsc{Bal}(G)$. Then $G - S$ is a forest of uniform cliques, and $G - S$ does not contain a positive triangle.*

Proof. The proof is similar to the proof of Lemma 21.

We prove the claim by induction on $|G|$. Observe that if $|G| = 0$ then $G-S$ is the empty graph and is therefore a forest of uniform cliques with no positive triangles. So now suppose that $|G| = n$, and claim is true for smaller graphs. We consider the different cases of the algorithm separately.

Case 1: For each component X of $G - v$, let $G_X = G[X \cup \{v\}]$ and let $(S_X, t_X) = \textsc{Alg}(G_X)$. By the inductive hypothesis, $G_X - S_X$ is a forest of uniform cliques with no positive triangles and so $G_X - S$ is also a forest of cliques with no positive triangles. Since $G - S$ is formed either by taking the disjoint union of all $(G_X - S)$ (if $v \in S$), or by joining all $(G_X - S)$ at a single vertex (if $v \notin S$), it follows that $G - S$ is also a forest of uniform cliques with no positive triangles.

Cases 2, 3 and 4: Observe that since $S = S'$ together with all the vertices in G that are not in G', we have that $G - S = G' - S'$, which is a forest of uniform cliques with no positive triangles by the inductive hypothesis.

Case 5: In this case none of Cases 1,2,3 apply. Then by Lemma 17, G is a uniform clique. Furthermore, since Case 4 does not apply, G does not contain any positive triangles. Therefore and so $G - S$ is also a uniform clique with no positive triangles. □

Putting Corollary 31 and Lemmas 34, 35 and 36, we have the following lemma.

Lemma 37. *Let $(G = (V, E), k)$ be an instance of* BSAPT. *In polynomial time, either we can conclude that (G, k) is a* Yes*-instance or we can find a set S of at most $12k$ vertices and a set of vertices W such that $G_W - S$ is a uniform forest of cliques without positive edges.*

Proof. Given an instance (G, k) of BSAPT, in polynomial time run algorithm Bal to get $(S, t) = \textsc{Bal}(G)$. If $t \geq k$ then by Lemma 35, (G, k) is a Yes-instance. Otherwise, by Lemma 34, $|S| < 12k$, and by Lemma 36, $G - S$ is a uniform forest of cliques with no positive triangles. Observe that $G - S$ is a chordal graph. Therefore, by Corollary 31, $\overline{G - S}$ is balanced and so, using Theorem 28, we can find a set of vertices W such that $(\overline{G - S})_W$ contains only positive edges. But then this implies that $G_W - S = (G - S)_W$ has only negative edges, as required. □

3.4 Fixed-parameter tractability of BSAPT

Finally, it is possible to prove that BSAPT is FPT. First we need to state the problem Max-Cut-with-Weighted-Vertices.

Max-Cut-with-Weighted-Vertices
Instance: A graph G with weight functions $w : E(G) \to \mathbb{N}$, $w_1 : V(G) \to \mathbb{N}_0$ and $w_2 : V(G) \to \mathbb{N}_0$, and an integer $t \in \mathbb{N}$.
Question: Does there exist an assignment $f : V(G) \to \{1, 2\}$ such that $\sum_{xy \in E} \lvert f(x) - f(y) \rvert w(xy) + \sum_{f(x)=1} w_1(x) + \sum_{f(x)=2} w_2(x) \geq t$?

Lemma 38. Max-Cut-with-Weighted-Vertices *can be solved in polynomial time when G is a forest of uniform cliques.*

Proof. We provide a polynomial-time transformation that replaces an instance (G, w, w_1, w_2, t) with an equivalent instance (G', w', w'_1, w'_2, t') such that G' has fewer vertices than G. By applying the transformation at most $|V(G)|$ times to get a trivial instance, we have a polynomial-time algorithm to decide (G, w, w_1, w_2, t).

We may assume that G is connected, as otherwise we can handle each component of G separately. Let $X \cup \{r\}$ be the vertices of a leaf-block in G, with r a cut-vertex of G (unless G consist of a single block, in which case let r be an arbitrary vertex and $X = V(G) - \{r\}$. Note that $X \cup \{r\}$ is a clique with uniform weights on the edges. Let w_e be the weight of every edge in this clique. For each possible assignment to r, we will in polynomial time calculate the optimal extension to the vertices in X. (This optimal extension depends only on the assignment to r, since no other vertices are adjacent to vertices in X.) We can then remove all the vertices in X, and change the values of $w_1(r)$ and $w_2(r)$ to reflect the optimal extension for each assignment.

Suppose we assign r the value 1. Let $\varepsilon(x) = w_1(x) - w_2(x)$ for each $x \in X$. Now arrange the vertices of X in order $x_1, x_2, \ldots x_{n'}$ (where $n' = |X|$), such that if $i < j$ then $\varepsilon(x_i) \geq \varepsilon(x_j)$. Observe that there is an optimal assignment for which x_i is assigned 1 for every $i \leq t$, and x_i is assigned 2 for every $i > t$, for some $0 \leq t \leq n'$. (Consider an assignment for which $f(x_i) = 2$ and $f(x_j) = 1$, for $i < j$, and observe that switching the assignments of x_i and x_j will increase $\sum_{f(x)=1} w_1(x) + \sum_{f(x)=2} w_2(x)$ by $\varepsilon(x_i) - \varepsilon(x_j)$, while $\sum_{xy \in E} |f(x) - f(y)| w(xy)$ will remain the same.) Therefore we only need to try $n' + 1$ different assignments to the vertices in X in order to find the optimal coloring when $f(r) = 1$. Let A be the value of this optimal assignment (over $X \cup \{r\}$).

By a similar method we can find the optimal assignment when r is assigned 2. Let the number of satisfied edges in this coloring be B. Now let $G' = G - X$ and $t' = t$, with w', w'_1, w'_2 the weight functions w, w_1, w_2 restricted to G, except that and $w'_1(r) = A$ and $w'_2(r) = B$. □

Theorem 39. BSAPT *can be solved in time* $2^{12k}n^{O(1)}$.

Proof. Let $(G = (V, E), k)$ be an instance of BSAPT. By Lemma 37, in polynomial time we can either decide that (G, k) is a YES-instance, or find a set S of at most $12k$ vertices and a set of vertices W such that $G_W - S$ is a uniform forest of cliques without positive edges. By Corollary 29, (G, k) is a YES-instance if and only if (G_W, k) is a YES-instance. So we may now replace G with G_W, and assume now that $G - S$ is a uniform forest of cliques without positive edges.

We now guess a partition (S_1, S_2) of S, corresponding to a (V_1, V_2)-balanced subgraph of $G[S]$.

Define the weight functions w_1, w_2 on the vertices of $G - S$ as follows. For each $x \in V \setminus S$, let $w_1(x)$ be the sum of the weights of positive edges between x and S_1, and negative edges between x and S_2. Similarly, let $w_2(x)$ be the sum of the weights of negative edges between x and S_1, and positive edges between x and S_2.

Now for any partition (U_1, U_2) of $V \setminus S$, observe that the weight of the maximum $(S_1 \cup U_1, S_2 \cup U_2)$-balanced subgraph of G is $w(E^+(U_1) \cup E^+(U_2) \cup E^-(U_1, U_2)) + w(E^+(U_1, S_1) \cup E^-(U_1, S_2) \cup E^-(U_2, S_1) \cup E^+(U_2, S_2)) + w(E^+(S_1) \cup E^+(S_2) \cup E^-(S_1, S_2))$. Observe that this simplifies to $w(E^-(U_1, U_2)) + \sum_{x \in U_1} w_1(x) + \sum_{x \in U_2} w_2(x) + w(E^+(S_1) \cup E^+(S_2) \cup E^-(S_1, S_2))$, by definition of $w_1(x)$ and $w_2(x)$ and the fact that $E^+(U_1) = E^+(U_2) = \emptyset$.

Now let $f(x) = 1$ if $x \in U_1$ and $f(x) = 2$ if $x \in U_2$. Then observe that the weight of the maximum $(S_1 \cup U_1, S_2 \cup U_2)$-balanced subgraph of G is $\sum_{xy \in E} |f(x) - f(y)| w(xy) + \sum_{f(x)=1} w_1(x) + \sum_{f(x)=2} w_2(x) + w(E^+(S_1) \cup E^+(S_2) \cup E^-(S_1, S_2))$. Thus, G contains a (V_1, V_2)-balanced subgraph with weight at least $\gamma(G) + k$ and $S_1 \subseteq V_1, S_2 \subseteq V_2$ if and only if $(G - S, w, w_1, w_2, t)$ is a YES-instance of MAX-CUT-WITH-WEIGHTED-VERTICES, with $t = \gamma(G) + k - w(E^+(S_1) \cup E^+(S_2) \cup E^-(S_1, S_2))$. But by Lemma 38 and the fact that $G - S$ is a forest of uniform cliques, this can be decided in polynomial time.

Therefore, to solve (G, k) it is enough to try every possible partition (S_1, S_2) of S, and solve the corresponding instance of MAX-CUT-WITH-WEIGHTED-VERTICES in polynomial time. Since the number of possible partitions of S is bounded by 2^{12k}, this gives an $2^{12k}n^{O(1)}$-time algorithm to solve SIGNED MAX CUT ATLB. □

3.5 Polynomial Kernel for BSAPT

In this section, we prove a polynomial kernel result for BSAPT. We are only able to show this for the unweighted case. Thus, in what follows we assume that all weights have weight 1. We show that BSAPT admits a kernel with $O(k^3)$ vertices under this assumption..

By Lemma 37, we may assume that we have a set S of at most $12k$ vertices such that $G - S$ is a forest of cliques, and a set of vertices such that $G_W - S$ contains no positive edges.

By Corollary 29, $\beta(G) \geq \gamma(G) + k$ if and only if $\beta(G_W) \geq \gamma(G_W)$. Thus we may consider the instance (G_W, k) instead of (G, k). For notational convenience, we will write G instead of G_W. Thus in what follows, we assume that $G - S$ is a forest of cliques with no positive edges.

The kernel is obtained via the application of a new set of reduction rules and using structural results that bound the size of No-instances (G, k). First, we need some additional terminology. For a block C in $G - S$, let $C_{\mathsf{int}} = \{x \in V(C) : N_{G-S}(x) \subseteq V(C)\}$ be the *interior* of C, and let $C_{\mathsf{ext}} = V(C) \setminus C_{\mathsf{int}}$ be the *exterior* of C. If a block C is such that $C_{\mathsf{int}} \cap N_G(S) \neq \emptyset$, C is a *special* block. We say a block C is a *path block* if $|V(C)| = 2 = |C_{\mathsf{ext}}|$. A *path vertex* is a vertex which is contained only in path blocks. A block C in $G - S$ is a *leaf block* if $|C_{\mathsf{ext}}| \leq 1$.

We now apply the following reduction rules. Note that in what follows we will treat k as a real number. In particular, the reduction rules will sometimes reduce k by a multiple of $1/4$, rather than a whole number. At all times we maintain the property that the reduced instance (G', k') has $\beta(G') \geq \gamma(G') + k'$ if and only if the original instance (G, k) has $\beta(G) \geq \gamma(G) + k$

Reduction Rule 1. *Let C be a block in $G - S$. If there exists $X \subseteq C_{\mathsf{int}}$ such that $|X| > \frac{|V(C)|+|N_G(X)\cap S|}{2} \geq 1$, $N_G^+(x) \cap S = N_G^+(X) \cap S$ and $N_G^-(x) \cap S = N_G^-(X) \cap S$ for all $x \in X$, then delete two arbitrary vertices $x_1, x_2 \in X$ and set $k' = k$.*

Reduction Rule 2. *Let C be a block in $G - S$. If $|V(C)|$ is even and there exists $X \subseteq C_{\mathsf{int}}$ such that $|X| = \frac{|V(C)|}{2}$ and $N_G(X) \cap S = \emptyset$, then delete a vertex $x \in X$ and set $k' = k - \frac{1}{4}$.*

Reduction Rule 3. *Let C be a block in $G - S$ with vertex set $\{x, y, u\}$, such that $N_G(u) = \{x, y\}$. If the edge xy is a bridge in $G - \{u\}$, delete C, add a new vertex z, positive edges $\{zv : v \in N^+_{G-u}(\{x, y\})\}$, negative edges $\{zv : v \in N^-_{G-u}(\{x, y\})\}$ and set $k' = k$. Otherwise, delete u and the edge xy and set $k' = k - \frac{1}{4}$.*

Reduction Rule 4. *Let T be a connected component of $G - S$ only adjacent to a vertex $s \in S$. Form a* Max-Cut-with-Weighted-Vertices *instance on T by defining $w_1(x) = 1$ if $x \in N_G^+(s) \cap T$ ($w_1(x) = 0$ otherwise) and $w_2(y) = 1$ if $y \in N_G^-(s) \cap T$ ($w_2(y) = 0$ otherwise). Let $\beta(G[V(T) \cup \{s\}]) = \gamma(G[V(T) \cup \{s\}]) + \frac{p}{4}$. Then delete T and set $k' = k - \frac{p}{4}$.*

Note that the value of p in Rule 4 can be found in polynomial time by solving Max-Cut-with-Weighted-Vertices on T.

A two-way reduction rule is *valid* if it transforms Yes-instances into Yes-instances and No-instances into No-instances.

To show that Rules 1-4 are valid, we first need the following lemmas:

The following lemma will be useful in the proofs that follow.

Lemma 40. *Let $G = (V, E)$ be a connected signed graph and let $V = U \cup W$ such that $U \cap W = \emptyset$, $U \neq \emptyset$ and $W \neq \emptyset$. Then $\beta(G) \geq \beta(G[U]) + \beta(G[W]) + \frac{1}{2}|E(U, W)|$. In addition, if $G[U]$ has c_1 components, $G[W]$ has c_2 components, $\beta(G[U]) \geq \gamma(G[U]) + k_1$ and $\beta(G[W]) \geq \gamma(G[W]) + k_2$, then $\beta(G) \geq \gamma(G) + k_1 + k_2 - \frac{(c_1+c_2-1)}{4}$.*

Proof. Let H (F) be a balanced subgraph of $G[U]$ ($G[W]$) with maximum number of edges and let H (F) be (U_1, U_2)-balanced ((W_1, W_2)-balanced). Let $E_1 = E^+(U_1, W_1) \cup E^+(U_2, W_2) \cup E^-(U_1, W_2) \cup E^-(U_2, W_1)$ and $E_2 = E(U, W) \setminus E_1$. Observe that both $E(H) \cup E(F) \cup E_1$ and $E(H) \cup E(F) \cup E_2$ induce balanced subgraphs of G and the largest of them has at least $\beta(G[U]) + \beta(G[W]) + \frac{1}{2}|E(U, W)|$ edges.

Now, observe that $\gamma(G) = \gamma(G[U]) + \gamma(G[W]) + \frac{1}{2}|E(U, W)| + \frac{c_1+c_2-1}{4}$. Hence $\beta(G) \geq \gamma(G) + k_1 + k_2 - \frac{(c_1+c_2-1)}{4}$. □

Lemma 41. *Let C be a block in $G - S$. If there exists $X \subseteq C_{int}$ such that $|X| \geq \frac{|V(C)|}{2}$, then there exists a (V_1, V_2)-balanced subgraph H of G with $\beta(G)$ edges such that at least one of the following inequalities holds:*

- $|V_2 \cap V(C)| \leq |V_1 \cap V(C)| \leq |N_G(X) \cap S| + |V_2 \cap V(C)|$;
- $|V_2 \cap V(C)| \leq |V_1 \cap V(C)| \leq |V_2 \cap V(C)| + 1$.

Proof. We may assume that $|V_1 \cap V(C)| \geq |V_2 \cap V(C)|$. Note that if $|V_1 \cap V(C)| > |V_2 \cap V(C)|$, then $X \cap V_1 \neq \emptyset$ (because $|X| \geq \frac{|V(C)|}{2}$).

First, if $N_G(X) \cap S = \emptyset$ and $|V_1 \cap V(C)| \geq |V_2 \cap V(C)| + 2$, then, for any $x \in X \cap V_1$, the subgraph induced by the partition $(V_1 \setminus \{x\}, V_2 \cup \{x\})$ has more edges than the subgraph induced by (V_1, V_2), which is a contradiction.

Now, suppose that $N_G(X) \cap S \neq \emptyset$ and suppose also that $|V_1 \cap V(C)| - |V_2 \cap V(C)|$ is minimal. If $|V_1 \cap V(C)| \leq |V_2 \cap V(C)| + 1$ we are done, so suppose $|V_1 \cap V(C)| \geq$

$|V_2 \cap V(C)|+2$. Consider the partition $V_1' = V_1 \setminus \{x\}$, $V_2' = V_2 \cup \{x\}$, where $x \in V_1 \cap X$, and the balanced subgraph H' induced by this partition. Then $|E(H')| \geq |E(H)| + |E(V_1 \setminus \{x\}, x)| - |E(V_2, x)| \geq |E(H)| + (|V_1 \cap V(C)| - 1 - |N_G(X) \cap S| - |V_2 \cap V(C)|)$. Since $|V_1' \cap V(C)| - |V_2' \cap V(C)| < |V_1 \cap V(C)| - |V_2 \cap V(C)|$, it holds that $|E(H')| \leq |E(H)| - 1$. Therefore, $|V_1 \cap V(C)| \leq |N_G(X) \cap S| + |V_2 \cap V(C)|$. □

Lemma 42. *Let C be a block in $G - S$. If there exists $X \subseteq C_{int}$ such that $|X| > \frac{|V(C)|+|N_G(X)\cap S|}{2}$, $N_G^+(x) \cap S = N_G^+(X) \cap S$ and $N_G^-(x) \cap S = N_G^-(X) \cap S$ for all $x \in X$, then, for any $x_1, x_2 \in X$, there exists a (V_1, V_2)-balanced subgraph H of G with $\beta(G)$ edges such that $x_1 \in V_1$ and $x_2 \in V_2$.*

Proof. First, we claim that there exist vertices $x_1, x_2 \in X$ for which the result holds. Let H be a (V_1, V_2)-balanced subgraph of G with $\beta(G)$ edges as given by Lemma 41.

Suppose $N_G(X) \cap S = \emptyset$. Then, by Lemma 41 it holds that $|V_2 \cap V(C)| \leq |V_1 \cap V(C)| \leq |V_2 \cap V(C)| + 1$; in addition, $|X| > \frac{|V(C)|}{2}$. Hence, either we can find x_1 and x_2 as required, or $X = V_1 \cap V(C)$ and $|V_1 \cap V(C)| = |V_2 \cap V(C)| + 1$. In the second case, pick a vertex $x \in V_1$ and form the partition $V_1' = V_1 \setminus \{x\}$ and $V_2' = V_2 \cup \{x\}$. Consider the balanced subgraph H' induced by this partition. Observe that $|E(H')| = |E(H)| - |E(x, V_2)| + |E(x, V_1 \setminus \{x\})| = |E(H)| - |V_2 \cap V(C)| + |V_1 \cap V(C)| - 1 = |E(H)|$, so H' is a maximum balanced subgraph for which we can find x_1 and x_2 as required.

Now, suppose $N_G(X) \cap S \neq \emptyset$. Then by Lemma 41 it holds that $|V_2 \cap V(C)| \leq |V_1 \cap V(C)| \leq |N_G(X) \cap S| + |V_2 \cap V(C)|$. For the sake of contradiction, suppose $X \subseteq V_1 \cap V(C)$ or $X \subseteq V_2 \cap V(C)$: in both cases, this means that $|X| \leq |V_1 \cap V(C)|$. Note that $|V(C)| = |V_1 \cap V(C)| + |V_2 \cap V(C)| = 2|V_2 \cap V(C)| + t$, where $t \leq |N_G(X) \cap S|$. Hence, $|V_1 \cap V(C)| \geq |X| > \frac{|V(C)|+|N_G(X)\cap S|}{2} = |V_2 \cap V(C)| + \frac{t}{2} + \frac{|N_G(X)\cap S|}{2} \geq |V_2 \cap V(C)| + t = |V_1 \cap V(C)|$, which is a contradiction.

To conclude the proof, notice that for a (V_1, V_2)-balanced subgraph H of G with $\beta(G)$ edges and vertices $x_1, x_2 \in X$ such that $x_1 \in V_1$ and $x_2 \in V_2$, we have $|E(H)| = |E(H')|$, where H' is a balanced subgraph induced by $V_1' = V_1 \setminus \{x_1\} \cup \{x_2\}$ and $V_2' = V_2 \setminus \{x_2\} \cup \{x_1\}$: this is true because $N_G^+(x_1) \cap S = N_G^+(x_2) \cap S$ and $N_G^-(x_1) \cap S = N_G^-(x_2) \cap S$. □

Theorem 43. *Rules 1-4 are valid.*

Proof. **Rule 1:** Let C, X be as in the description of Rule 1. Let $x_1, x_2 \in X$. By Lemma 42, there exists a (V_1, V_2)-balanced subgraph H of G with $\beta(G)$ edges such that $x_1 \in V_1$ and $x_2 \in V_2$. Now, let $G' = G - \{x_1, x_2\}$ and $H' = H - \{x_1, x_2\}$.

Since $N_G^+(x_1) \cap S = N_G^+(x_2) \cap S$ and $N_G^-(x_1) \cap S = N_G^-(x_2) \cap S$, it holds that $|E(H)| = |E(H')| + \frac{|E(G,\{x_1,x_2\})|}{2} + 1$, and so $\beta(G') + \frac{|E(G,\{x_1,x_2\})|}{2} + 1 \geq \beta(G)$. Conversely, by Lemma 40, $\beta(G) \geq \beta(G') + \frac{|E(G,\{x_1,x_2\})|}{2} + 1$. Finally, observe that $\gamma(G) = \gamma(G') + \frac{|E(G,\{x_1,x_2\})|}{2} + 1$, which implies that $\beta(G) - \gamma(G) = \beta(G') - \gamma(G')$. Hence, G admits a balanced subgraph of size $\gamma(G) + k$ if and only if G' admits a balanced subgraph of size $\gamma(G') + k$.

Rule 2: Let C, X and $x \in X$ be as in the description of Rule 2. By Lemma 41, there exists a (V_1, V_2)-balanced subgraph H of G with $\beta(G)$ edges, such that $|V_1 \cap V(C)| = |V_2 \cap V(C)|$. Consider the graph $G' = G - \{x\}$ formed by the application of the rule and the balanced subgraph $H' = H - \{x\}$. Then $|E(H)| = |E(H')| + \frac{|V(C)|}{2}$, and thus $\beta(G') \geq \beta(G) - \frac{|V(C)|}{2}$. Conversely, by Lemma 40, $\beta(G) \geq \beta(G') + \frac{|V(C)|}{2}$. However, $\gamma(G) = \gamma(G') + \frac{|V(C)|}{2} - \frac{1}{4}$. Hence, $\beta(G) - \gamma(G) = \beta(G') - \gamma(G') + \frac{1}{4}$. Therefore, G admits a balanced subgraph of size $\gamma(G) + k$ if and only if G' admits a balanced subgraph of size $\gamma(G') + k - \frac{1}{4}$.

Rule 3: Let C and $\{x, y, u\}$ be as in the description of Rule 3. Firstly consider the case when xy is a bridge in $G - \{u\}$. For any maximal balanced subgraph H of G, without loss of generality one may assume that $xu, yu \in E(H)$ and $xy \notin E(H)$. Suppose H is induced by a partition (V_1, V_2) and $x, y \in V_1$. Form a balanced subgraph of G' from $H - \{x, y, u\}$ by placing z in V_1. Therefore, $\beta(G) = \beta(G') + 2$. Since $\gamma(G) = \gamma(G') + \frac{3}{2} + \frac{2}{4} = \gamma(G') + 2$, it follows that $\beta(G) = \gamma(G) + k$ if and only if $\beta(G') = \gamma(G') + k$.

Now consider the case when xy is not a bridge in $G - \{u\}$. Then the graph G' formed by deleting the vertex u and the edge xy is connected. Furthermore, regardless of whether x and y are in the same partition that induces a balanced subgraph H' of G', H' can be extended to a balanced subgraph H of G such that $|E(H)| = |E(H')| + 2$. This means that, as before, $\beta(G) = \beta(G') + 2$. But in this case $\gamma(G) = \gamma(G') + \frac{7}{4}$ and thus $\beta(G) = \gamma(G) + k$ if and only if $\beta(G') = \gamma(G') + k - \frac{1}{4}$.

Rule 4: Let T and $s \in S$ be as in the description of Rule 4. Since $\beta(G[V(T) \cup \{s\}]) = \gamma(G[V(T) \cup \{s\}]) + \frac{p}{4}$, and s is a cut-vertex, by the block additivity property $\beta(G) = \beta(G - T) + \gamma(G[V(T) \cup \{s\}]) + \frac{p}{4}$. Also observe that $\gamma(G) = \gamma(G - T) + \gamma(G[V(T) \cup \{s\}])$. Hence $\beta(G) - \gamma(G) = \beta(G - T) - \gamma(G - T) + \frac{p}{4}$, which implies that G admits a balanced subgraph of size $\gamma(G) + k$ if and only if $G - T$ admits a balanced subgraph of size $\gamma(G - T) + k - \frac{p}{4}$. □

To show the existence of a kernel with $O(k^3)$ vertices, it is enough to give a bound on the number of non-path blocks, the number of vertices in these blocks and the number of path vertices. This is done by Corollaries 50 and 51 and Lemma 52.

While Lemma 52 applies to any graph reduced by Rule 1, the proofs of Corollaries 50 and 51 rely on Lemma 49, which gives a general structural result on forest of cliques with a bounded number of special blocks and bounded path length. Corollary 46 and Lemma 47 provide sufficient conditions for a reduced instance to be a YES-instance, thus producing a bound on the number of special blocks and the path length of NO-instances. Lastly, Theorem 53 puts the results together to show the existence of the kernel.

Henceforth, we assume that the instance (G, k) is such that G is reduced by Rules 1-4, $G - S$ is a forest of cliques which does not contain a positive edge and $|S| \leq 12k$.

Lemma 44. *Let T be a connected component of $G - S$. Then for every leaf block C of T, $N_G(C_{int}) \cap S \neq \emptyset$. Furthermore, if $|N_G(S) \cap V(T)| = 1$, then T consists of a single vertex.*

Proof. We start by proving the first claim. Note that if $T = C$ consists of a single vertex, then $N_G(C_{\mathsf{int}}) \cap S \neq \emptyset$ since G is connected. So assume that C has at least two vertices. Suppose that $N_G(C_{\mathsf{int}}) \cap S = \emptyset$ and let $X = C_{\mathsf{int}}$. Then if $|C_{\mathsf{int}}| > |C_{\mathsf{ext}}|$, Rule 1 applies. If $|C_{\mathsf{int}}| = |C_{\mathsf{ext}}|$ then Rule 2 applies. Otherwise, $|C_{\mathsf{int}}| < |C_{\mathsf{ext}}|$ and since $|C_{\mathsf{ext}}| \leq 1$ (as C is a leaf block), C has only one vertex, which contradicts our assumption above. For the second claim, first note that since $|N_G(S) \cap V(T)| = 1$, Q has one leaf block and so T consists of a single block. Let $N_G(S) \cap V(T) = \{v\}$ and $X = V(T) - \{v\}$. If $|X| > 1$, Rule 1 applies. If $|X| = 1$, Rule 2 applies. Hence $V(T) = \{v\}$. □

Let $\mathcal{B}$ be the set of non-path blocks.

Lemma 45. *If there exists a vertex $s \in S$ such that $\sum_{C \in \mathcal{B}} |N_G(C_{int}) \cap \{s\}| \geq 2(|S| - 1 + 4k)$, then (G, k) is a* YES*-instance.*

Proof. Form $T \subseteq N_G(s)$ by picking a vertex from each block C for which $|N_G(C_{\mathsf{int}}) \cap \{s\}| = 1$: if there exists a vertex $x \in C_{\mathsf{int}}$ such that $N_G(x) \cap S = \{s\}$, pick this, otherwise pick $x \in C_{\mathsf{int}}$ arbitrarily. Let $U = T \cup \{s\}$ and $W = V \setminus U$.

Observe that $G[U]$ is balanced by Theorem 27 as $G[U]$ is a tree. Thus $\beta(G[U]) = |T| = \frac{|T|}{2} + \frac{|T|}{4} + \frac{|T|}{4} = \gamma(G[U]) + \frac{|T|}{4}$.

Consider a connected component Q of $G - S$. By Rule 4, $|N_G(Q) \cap S| \geq 2$ and by Lemma 44, if $|N_G(S) \cap V(Q)| = 1$ then Q consists of a single vertex. Otherwise, either $(N_G(S) \setminus N_G(s)) \cap V(Q) \neq \emptyset$, or Q has at least two vertices in T. Moreover, note that the removal of interior vertices does not disconnect the component itself.

Hence $G[W]$ has at most $(|S|-1)+\frac{|T|}{2}$ connected components. Applying Lemma 40, $\beta(G) \geq \gamma(G) + \frac{|T|}{4} - \frac{(|S|-1)+\frac{|T|}{2}}{4} = \gamma(G) + \frac{|T|}{8} - \frac{|S|-1}{4}$. Hence if $|T| \geq 2(|S|-1+4k)$, then (G,k) is a YES-instance. □

Corollary 46. *If $\sum_{C\in\mathcal{B}} |N_G(C_{int}) \cap S| \geq |S|(2|S|-3+8k)+1$, the instance is a* YES*-instance. Otherwise, $\sum_{C\in\mathcal{B}} |N_G(C_{int}) \cap S| \leq 12k(32k-3)$.*

Proof. If $\sum_{C\in\mathcal{B}} |N_G(C_{\mathsf{int}}) \cap S| \geq |S|(2|S|-3+8k)+1$, then for some $s \in S$ we have $\sum_{C\in\mathcal{B}} |N_G(C_{\mathsf{int}}) \cap \{s\}| \geq 2|S|-3+8k+1/|S|$ and, since the sum is integral, $\sum_{C\in\mathcal{B}} |N_G(C_{\mathsf{int}}) \cap \{s\}| \geq 2(|S|-1+4k)$. Thus, (G,k), by Lemma 45, is a YES-instance. The second inequality of the corollary follows from the fact that $|S| \leq 12k$. □

Lemma 47. *If in $G-S$ there exist vertices $U = \{u_1, u_2, \ldots, u_p\}$ such that $N_{G-S}(u_i) = \{u_{i-1}, u_{i+1}\}$ for $2 \leq i \leq p-1$, and $p \geq |S|+4k+1$, then (G,k) is* YES*-instance. Otherwise, $p \leq 16k$.*

Proof. Observe that $G[U]$ is balanced by Theorem 27. Thus $\beta(G[U]) = p-1 = \gamma(G[U]) + \frac{p-1}{4}$. Let $W = V \setminus U$ and observe that $G[W]$ has at most $|S|$ components, since, by Lemma 44, every vertex in $G-U$ has a path to a vertex in S. Applying Lemma 40, $\beta(G) \geq \gamma(G) + \frac{p-1}{4} - \frac{|S|}{4}$. Hence if $p-1-|S| \geq 4k$, (G,k) is a YES-instance. □

Lemma 48. *A block C in $G-S$ such that $|C_{\mathsf{ext}}| = 2$ is either special or it is a path block.*

Proof. Suppose C is not special. If $|V(C)| \geq 5$, then Reduction Rule 1 would apply. If $|V(C)| = 4$, then Reduction Rule 2 would apply. If $|V(C)| = 3$, then Reduction Rule 3 would apply. Hence $|V(C)| = 2$ and it is a path block. □

In $G-S$, a *pure path* is a path consisting exclusively of path vertices. Note that every path vertex belongs to a unique pure path.

Lemma 49. *Suppose $G-S$ has at most l special blocks and the number of vertices in each pure path is bounded by p. Then $G-S$ contains at most $2l$ non-path blocks and $2pl$ path vertices.*

Proof. It suffices to prove that if every connected component T of $G-S$ has at most l_T special blocks, then T contains at most $2l_T$ non-path blocks and $2pl_T$ path vertices. So, we may assume that $T = G-S$ is connected. Pick an arbitrary non-path block C_R as the 'root' node. Define the distance $d(C_R, C)$ as the number of non-path

blocks different from C_R visited in a path from a vertex in C_R to a vertex in C. For every non-path block C in T, the *parent* block C' is the unique non-path block such that C' contains an edge of any path from C_R to C and $d(C_R, C) - d(C_R, C') = 1$. In addition, C_R is the parent of every block C such that $d(C_R, C) = 1$.

Consider the tree F that contains a vertex for every non-path block of T and such that there is an edge between two vertices if and only if one of the corresponding blocks is the parent of the other. Observe that given a vertex $v \in F$ which corresponds to a block C of T, it holds that $d_F(v) \geq |C_{\mathsf{ext}}|$. In addition, by Lemma 44, every leaf in F corresponds to a special block.

Now, we know that in a tree the number of vertices of degree greater or equal to three is bounded by the number of leaves. Moreover, by Lemma 48, if a block C is such that $|C_{\mathsf{ext}}| = 2$, then it is either special or a path block. Thus, the number of non-path blocks is bounded by $2l$.

Furthermore, note that the number of pure paths in T is bounded by the number of edges in F, which is bounded by $2l - 1$. Since every pure path contains at most p path vertices, the number of path vertices is bounded by $(2l - 1)p < 2pl$. □

Corollary 50. *$G - S$ contains at most $24k(32k-3)$ non-path blocks and $384k^2(32k-3)$ path vertices.*

Proof. By Corollary 46, $G - S$ contains at most $12k(32k - 3)$ special blocks and by Lemma 47, the length of every pure path is bounded by $16k$. Thus, Lemma 49 implies that $G - S$ contains at most $24k(8k - 3)$ non-path blocks and $16k(24k(8k - 3)) = 384k^2(32k - 3)$ path vertices. □

Corollary 51. *$G - S$ contains at most $48k(32k - 3)$ vertices in the exteriors of non-path blocks.*

Proof. For any component T of $G - S$, consider the tree F defined in the proof of Lemma 49. For any block C of T and any vertex v in C_{ext}, v corresponds to an edge of F. Furthermore, for any edge of F there are at most two exterior vertices in T that correspond to it. Therefore, $|\cup_{C \in \mathcal{B}} C_{\mathsf{ext}}| \leq 2|\mathcal{B}| \leq 48k(32k - 3)$. □

Lemma 52. *For a block C, if $|V(C)| \geq 2|C_{\mathsf{ext}}| + |N_G(C_{\mathsf{int}}) \cap S|(2|S| + 8k + 1)$, then (G, k) is a* Yes*-instance. Otherwise, $|V(C)| \leq 2|C_{\mathsf{ext}}| + |N_G(C_{\mathsf{int}}) \cap S|(32k + 1)$.*

Proof. Consider a fixed $s \in N_G(C_{\mathsf{int}}) \cap S$. We will show that we may assume that either $|N_G^+(s) \cap C_{\mathsf{int}}| \leq \frac{4k+|S|}{2}$ or $|N_G^+(s) \cap C_{\mathsf{int}}| \geq |C_{\mathsf{int}}| - \frac{4k+|S|}{2}$, because otherwise (G, k) is a Yes-instance.

Indeed, suppose $\lceil \frac{4k+|S|}{2} \rceil \leq |N_G^+(s) \cap C_{\mathsf{int}}| \leq |C_{\mathsf{int}}| - \lceil \frac{4k+|S|}{2} \rceil$. Let $U_1 \subseteq N_G^+(s) \cap C_{\mathsf{int}}$, $|U_1| = \lceil \frac{4k+|S|}{2} \rceil$, and let $U_2 \subseteq C_{\mathsf{int}} \setminus N_G^+(s)$, $|U_2| = \lceil \frac{4k+|S|}{2} \rceil$. Let $U = U_1 \cup U_2 \cup \{s\}$ and consider the subgraph H of $G[U]$ induced by the edges $E(U_1, U_2) \cup E(s, (U_1 \cap N_G^+(s))) \cup E(s, (U_2 \cap N_G^-(s)))$. Observe that H is $(U_1 \cup \{s\}, U_2)$-balanced and so $\beta(G[U]) \geq |U_1|^2 + |U_1 \cap N_G^+(s)| + |U_2 \cap N_G^-(s)|$. Furthermore, $\gamma(G[U]) = |U_1|^2 + \frac{|U_1 \cap N_G^+(s)|}{2} + \frac{|U_2 \cap N_G^-(s)|}{2}$, and hence $\beta(G[U]) \geq \gamma(G[U]) + \frac{|U_1 \cap N_G^+(s)| + |U_2 \cap N_G^-(s)|}{2} \geq \gamma(G[U]) + \frac{4k+|S|}{4}$.

Now consider $W = V \setminus U$. Any connected component of $G - S$ is connected to two vertices in S, hence $G[W]$ has at most $|S| - 1$ components adjacent to vertices in $S \setminus \{s\}$ and one component corresponding to the block C. Applying Lemma 40, $\beta(G) \geq \gamma(G) + \frac{(4k+|S|)-|S|}{4}$, which means that (G, k) is a Yes-instance.

Similarly, we can show that we may assume that either $|N_G^-(s) \cap C_{\mathsf{int}}| \leq \frac{4k+|S|}{2}$ or $|N_G^-(s) \cap C_{\mathsf{int}}| \geq |C_{\mathsf{int}}| - \frac{4k+|S|}{2}$, because otherwise (G, k) is a Yes-instance.

Let $S_1^+ = \{s \in S : 0 < |N_G^+(s) \cap C_{\mathsf{int}}| \leq \frac{4k+|S|}{2}\}$, $S_2^+ = (N_G^+(C_{\mathsf{int}}) \cap S) \setminus S_1^+$ and $X^+ = \{v \in C_{\mathsf{int}} \setminus N_G^+(S_1^+) : v \in N_G^+(s), \forall s \in S_2^+\}$. Observe that for all $s \in S_2^+$, $|N_G^+(s) \cap C_{\mathsf{int}}| \geq |C_{\mathsf{int}}| - \frac{4k+|S|}{2}$, which means that $|X^+| \geq |C_{\mathsf{int}} \setminus N_G^+(S_1^+)| - |S_2^+| \frac{4k+|S|}{2}$. In addition, $|N_G^+(S_1^+) \cap C_{\mathsf{int}}| \leq |S_1^+| \frac{4k+|S|}{2}$, hence $|C_{\mathsf{int}} \setminus N_G^+(S_1^+)| \geq |C_{\mathsf{int}}| - |S_1^+| \frac{4k+|S|}{2}$. Therefore, $|X^+| \geq |C_{\mathsf{int}}| - (|S_1^+| + |S_2^+|) \frac{4k+|S|}{2} = |C_{\mathsf{int}}| - |N_G^+(C_{\mathsf{int}}) \cap S| \frac{4k+|S|}{2} \geq |C_{\mathsf{int}}| - |N_G(C_{\mathsf{int}}) \cap S| \frac{4k+|S|}{2}$.

With similar definitions and the same argument we obtain $|X^-| \geq |C_{\mathsf{int}}| - |N_G(C_{\mathsf{int}}) \cap S| \frac{4k+|S|}{2}$. Now let $X = X^+ \cap X^-$ and observe that and $|X| \geq |C_{\mathsf{int}}| - |N_G(C_{\mathsf{int}}) \cap S|(4k + |S|)$.

However, by Rule 1, $|X| \leq \frac{|V(C)| + |N_G(C_{\mathsf{int}}) \cap S|}{2}$. So, $|C_{\mathsf{int}}| \leq |N_G(C_{\mathsf{int}}) \cap S|(|S| + 4k + \frac{1}{2}) + \frac{|V(C)|}{2}$, and so $|V(C)| \leq 2|C_{\mathsf{ext}}| + |N_G(C_{\mathsf{int}}) \cap S|(2|S| + 8k + 1)$ as claimed. □

Theorem 53. Signed Max Cut ATLB *has a kernel with* $O(k^3)$ *vertices.*

Proof. Let $(G = (V, E), k)$ be an instance of Signed Max Cut ATLB. Recall that by Lemma 37, we either have a Yes-instance or we may assume there exists $S \subseteq V$ such that $|S| \leq 12k$ and $G - S$ is a forest of cliques which does not contain a positive edge.

Now, apply Rules 1–4 exhaustively to (G, k) to obtain a new instance (G', k'). If $k' \leq 0$, then (G, k) is a Yes-instance since Rules 1–4 are valid. Now let $G = G', k = k'$. Check whether (G, k) is a Yes-instance due to Corollary 46, Lemma 47 or Lemma 52. If this is not the case, by Corollary 50, $G - S$ contains at most $24k(32k - 3)$ non-path blocks and $384k^2(32k - 3)$ path vertices. Hence, by Lemma

52, $|V(G)|$ is at most

$$|S|+384k^2(32k-3)+\sum_{C\in\mathcal{B}}|V(C)| \leq O(k^3)+2\sum_{C\in\mathcal{B}}|C_{\mathsf{ext}}|+(32k+1)\sum_{C\in\mathcal{B}}|N_G(C_{\mathsf{int}})\cap S|$$

Now, applying Corollary 46 and Corollary 51, we obtain:

$$|V(G)| \leq O(k^3) + 192k(32k-3) + 12k(32k-3)(32k+1) = O(k^3).$$

□

Chapter 4

Max Acyclic Subgraph Problem Parameterized Above the Poljak-Turzík Bound

The problem of finding the maximum acyclic subgraph in a directed graph is well-studied in the literature in graph theory, algorithms and their applications alongside its dual, the feedback arc set problem, see, e.g., Chapter 15 in [5] and references therein. This is true, in particular, in the area of parameterized complexity [11, 35, 37, 63].

Each directed graph D with m arcs has an acyclic subgraph with at least $m/2$ arcs. To obtain such a subgraph, order the vertices $x_1, \ldots, x_n$ of D arbitrarily and consider two spanning subgraphs of D: D' with arcs of the form $x_i x_j$, and D'' with arcs of the form $x_j x_i$, where $i < j$. One of D' and D'' has at least $m/2$ arcs. Moreover, $m/2$ is the largest size of an acyclic subgraph in every symmetric digraph S (in a symmetric digraph the existence of an arc xy implies the existence of an arc yx). Thus, it makes sense to consider the parameterization above the tight bound $m/2$: decide whether a digraph D contains an acyclic subgraph with at least $m/2+k$ arcs, where k is the parameter. Mahajan *et al.* [54] and Raman and Saurabh [63] asked what the complexity of this problem is. For the case of oriented graphs (i.e. directed graphs with no directed cycles of length 2), Raman and Saurabh [63] proved that the problem is fixed-parameter tractable. A generalization of this problem to integer-arc-weighted digraphs (where $m/2$ is replaced by the half of the total weight of D) was proved to be fixed-parameter tractable in [35].

For oriented graphs, $m/2$ is no longer a tight lower bound on the maximum size of an acyclic subgraph. Indeed we will see that the property of being acyclic

over oriented graphs is 1/2-extendible and thus every oriented graph G contains an acyclic subgraph of size $\frac{m}{2} + \frac{n-1}{4}$. (This was first proved by Poljak and Turzík [62].)

To see that the bound is indeed tight consider a directed path $x_1x_2 \dots x_{2t+1}$ and add to it arcs $x_3x_1, x_5x_3, \dots, x_{2t+1}x_{2t-1}$. This oriented graph H_t consists of t directed 3-cycles and has $2t+1$ vertices and $3t$ arcs. Thus, $\frac{m}{2} + \frac{n-1}{4} = 2t$ and $2t$ is the maximum size of an acyclic subgraph of H_t: we have to delete an arc from every directed 3-cycle as the cycles are arc-disjoint.

Raman and Saurabh [63] asked to determine the parameterized complexity of the following problem: decide whether a connected oriented graph D has an acyclic subgraph with at least $\frac{m}{2} + \frac{n-1}{4} + k$ arcs, where k is the parameter. Answering this question, we will prove that this problem is fixed-parameter tractable and admits a kernel with $O(k^2)$ vertices and $O(k^2)$ arcs.

Recall that we already proved that the property of being an acyclic oriented graph is $\frac{1}{2}$-extendible, in . For convenience, we repeat this result here.

Lemma 54. *Let* Π *be the class of all acyclic graphs. Then* Π *is strongly* 1/2*-extendible over the class of oriented graphs.*

Lemma 54 and the oriented version of Theorem 11 give us the following theorem.

Theorem 55. *[62] Let* $G = (V, A)$ *be an oriented graph with weight function* $w : A \to \mathbb{R}^+$*. Then there exists an acyclic subgraph* H *of* G *such that* $w(H) \geq \frac{w(G)}{2} + \frac{\tau(G)}{4}$.

In this chapter, given an oriented graph G, let $\beta(G)$ denote the maximum weight of an acyclic subgraph of G, and let $\gamma(G)$ denote the lower bound $\frac{w(G)}{2} + \frac{\tau(G)}{4}$ on $\beta(G)$.

We consider the problem Π-SAPT when Π is the class of acyclic graphs, and all input graphs are oriented graphs.

Acyclic Subgraph above Poljak-Turzík Bound (ASAPT)
Instance: An oriented connected graph G with n vertices and m arcs.
Parameter: k.
Question: Is $\beta(G) \geq \gamma(G) + k$?

Our results in this chapter are only for the unweighted version of ASAPT. That is, we assume that all arcs have weight 1. Thus, for an oriented graph with n vertices and m arcs the Poljak-Turzík bound is $\frac{m}{2} + \frac{n-1}{4}$.

Mnich et. al. [55] proved the following Lemma

Lemma 56 (Lemma 21, [55])**.** *The unweighted version of* ASAPT-AFOC *is fixed-parameter tractable.*

This, combined with Lemma 54 and Theorem 22, already gives us the following theorem:

Theorem 57. *[55] The unweighted version of* ASAPT *is fixed-parameter tractable.*

In this chapter we will give our own proof that ASAPT is fixed-parameter tractable. The reason is that to aid us in our kernel proof, we would like to reduce the graph to something more specific that an almost-forest of cliques. Similar to the situation with BSAPT, even if we have a uniform weighted tournament as our input graph, the problem is still not trivially easy because there could be any arrangement of orientations of the edges.

We will give an algorithm which either shows that (G, k) is a YES-instance, or finds a set S of at most $12k$ vertices such that $G - S$ is a forest of cliques in which each block is of size at most 3, and the blocks of size 3 are directed 3-cycles.

4.1 Main algorithm

In our arguments we use the following simple correspondence between acyclic digraphs and orderings of vertices in digraphs. Let H be an acyclic spanning subgraph of an oriented G. It is well-known [5] and easy to see that there is an ordering $x_1, \ldots, x_n$ of vertices of G such that if $x_i x_j$ is an arc of H then $i < j$. On the other hand, any ordering $x_1, \ldots, x_n$ of vertices of a digraph $G = (V, A)$ leads to an acyclic spanning subgraph of G: consider the subgraph induced by $\{x_i x_j : x_i x_j \in A, i < j\}$. As we study maximum-size acyclic subgraphs, we may restrict ourselves to acyclic spanning subgraphs. Thus, we may use interchangeably the notions of acyclic spanning subgraphs and vertex orderings.

There are some known lower bounds on $\beta(T)$ for tournaments T on n vertices, see, e.g., [65] and references therein. We show the following useful bound which we were unable to find in the literature [1].

Lemma 58. *Let T be a tournament on n vertices with $m = \binom{n}{2}$ arcs. We can, in polynomial time, find an acyclic subgraph with at least $\frac{m}{2} + \frac{3n}{4} - 1 = \gamma(G) + \frac{2n-3}{4}$ arcs, if n is even, or $\frac{m}{2} + \frac{3(n-1)}{4} - 1 = \gamma(G) + \frac{2n-6}{4}$ arcs, if n is odd.*

[1] Note, however, that it is shown in [65] that any tournament on n vertices and $m = \binom{n}{2}$ arcs has an acyclic subgraph with at least $\frac{m}{2} + cn^{3/2}$ arcs for a constant c, which is larger than the Poljak-Turzík bound for large enough n

Proof. We prove the lemma by induction. The claim can easily be checked for $n = 1$ and $n = 2$ and we may assume that $n \geq 3$.

Consider first the case when n is even. Suppose that there exists a vertex x such that $d^+(x) \geq \frac{n}{2} + 1$. Consider the tournament $T' = T - x$, with $m' = m - (n-1)$ arcs and $n' = n - 1$ vertices. By induction, there is an ordering on G' that produces an acyclic spanning subgraph H' of G' such that

$$|A(H')| \geq \frac{m'}{2} + \frac{3(n'-1)}{4} - 1 = \frac{m-(n-1)}{2} + \frac{3(n-2)}{4} - 1 = \frac{m}{2} + \frac{3n}{4} - \frac{n}{2} - 2.$$

Now add x to the beginning of this ordering. This produces an acyclic spanning subgraph H of G such that $|A(H)| \geq |A(H')| + \frac{n}{2} + 1 \geq \frac{m}{2} + \frac{3n}{4} - 1$.

If there is a vertex x such that $d^-(x) \geq \frac{n}{2} + 1$, the same argument applies, but x is added to the end of the ordering.

Otherwise, for every vertex x of G, $d^+(x) \in \{\frac{n}{2} - 1, \frac{n}{2}\}$. Moreover, by considering the sum of out-degrees, exactly half the vertices have out-degree $\frac{n}{2}$. Hence, if $n \geq 4$, there are at least two vertices with out-degree $\frac{n}{2}$. Let x and y be two such vertices, and suppose, without loss of generality, that there is an arc from x to y. Now consider $G' = G - \{x, y\}$ with $m' = m - (2n-3)$ edges and $n' = n - 2$ vertices. By induction, there is an ordering on the vertices of T' that produces an acyclic subgraph with at least $\frac{m'}{2} + \frac{3n'}{4} - 1 = \frac{m}{2} + \frac{3n}{4} - n - 1$ arcs. Place x and y at the beginning of this ordering, with x occurring before y. Then this will add all the arcs from x and y to the acyclic subgraph. Thus, $\beta(G) \geq \frac{m}{2} + \frac{3n}{4} - n - 1 + n = \frac{m}{2} + \frac{3n}{4} - 1$.

Now suppose that n is odd. Let x be any vertex in T, and let $T' = T - x$. By induction, there is an ordering on T' that produces an acyclic subgraph with at least $\frac{m'}{2} + \frac{3n'}{4} - 1$ arcs, where $n' = n-1$ is the number of vertices and $m' = m-(n-1)$ is the number of arcs in G'. By placing x either at the beginning or end of this ordering, we may add at least $(n-1)/2$ arcs. Thus, $\beta(G) \geq \frac{m-(n-1)}{2} + \frac{3(n-1)}{4} - 1 + \frac{n-1}{2} = \frac{m}{2} + \frac{3(n-1)}{4} - 1$. □

Lemma 59. *Let X be a nonempty set of vertices of an oriented graph G such that both $G - X$ and $G[X]$ are connected. If $\beta(G - X) \geq \gamma(G - X) + k'$ and $\beta(G[X]) \geq \gamma(G[X]) + k''$, then $\beta(G) \geq \gamma(G) + k' + k'' - \frac{1}{4} + \frac{|d^+(X) - d^-(X)|}{2}$. In particular, $\beta(G) \geq \gamma(G) + k' + k'' - \frac{1}{4}$ if $|E(X, V(G) \setminus X)|$ is even and $\beta(G) \geq \gamma(G) + k' + k'' + \frac{1}{4}$, if $|E(X, V(G) \setminus X)|$ is odd.*

Proof. Form an acyclic subgraph on G as follows. Assume without loss of generality that $d^+(X) \geq d^-(X)$. Pick the arcs leaving X together with the arcs of the acyclic subgraphs in $G - X$ and $G[X]$. This forms an acyclic subgraph H. Let $m = m' + m'' + \bar{m}$ and $n = n' + n''$, where $G - X$ has m' arcs and n' vertices, $G[X]$ has

m'' arcs and n'' vertices and $\bar{m} = d^+(X) + d^-(X)$. The acyclic subgraph H has at least $\gamma(G-X) + k' + \gamma(G[X]) + k'' + \frac{\bar{m}}{2} + \frac{d^+(X)-d^-(X)}{2} = \frac{m'+m''+\bar{m}}{2} + \frac{n'-1}{4} + \frac{n''-1}{4} + k' + k'' + \frac{d^+(X)-d^-(X)}{2} = \gamma(G) + k' + k'' - \frac{1}{4} + \frac{d^+(X)-d^-(X)}{2}$ arcs, as required. □

Lemma 60. *Let x be a vertex and X a set of two vertices such that $G[X]$ is a component of $G-x$ and $G[X \cup \{x\}]$ is a directed 3-cycle, and let $G' = G - X$. Then $\beta(G) - \gamma(G) = \beta(G') - \gamma(G')$.*

Proof. Observe that $\beta(G') = \beta(G) - 2$. Since $m' = m - 3$ and $n' = n - 2$, we have $\beta(G) \geq \frac{m}{2} + \frac{n-1}{4} + k' = \gamma(G) + k'$ if and only if $\beta(G') \geq \frac{m'}{2} + \frac{n'-1}{4} + k' = \gamma(G') + k'$. □

Lemma 61. *Let x be a vertex of G such that $d^+(x) \neq d^-(x)$, and $G' = G - \{x\}$ is connected. Then $\beta(G) - \gamma(G) \geq \beta(G') + \gamma(G') + \frac{2|d^+(x)-d^-(x)|-1}{4}$.*

Proof. Assume that $\beta(G) = \gamma(G) + k'$ for some k'. Then by Lemma 59 with $X = \{x\}$ and $k'' = 0$, $\beta(G) \geq \gamma(G) + k' - \frac{1}{4} + \frac{|d^+(X)-d^-(X)|}{2} = \gamma(G) + k' + \frac{2(|d^+(X)-d^-(X)|)-1}{4}$, as required. □

Lemma 62. *Let X be a set of vertices such that $G' = G - X$ is connected, $G[X]$ is a tournament, and $|X| \geq 4$. Then $\beta(G) - \gamma(G) \geq \beta(G') - \gamma(G') + \frac{2|X|-4}{4}$ if $|X|$ is even, and $\beta(G) - \gamma(G) \geq \beta(G') - \gamma(G') + \frac{2|X|-7}{4}$ if $|X|$ is odd.*

Proof. Suppose $|X|$ is even. By Lemma 58, $\beta(G[X]) \geq \gamma(G[X]) + \frac{2|X|-3}{4}$. By Lemma 59, if $\beta(G') = \gamma(G') + k'$, then $\beta(G) \geq \gamma(G) + k' + \frac{2|X|-3}{4} - \frac{1}{4} = \gamma(G) + k' + \frac{2|X|-4}{4}$, as required.

A similar argument applies in the case when $|X|$ is odd, except the bound from Lemma 58 is $\gamma(G[X]) + \frac{2|X|-6}{4}$ instead of $\gamma(G[X]) + \frac{2|X|-3}{4}$. □

Lemma 63. *Let X be a set of three vertices such that X is an induced P_3, and $G' = G - X$ is connected. Then $\beta(G) - \gamma(G) \geq \beta(G') - \gamma(G') + \frac{1}{4}$.*

Proof. Observe that $\beta(G[X]) = \gamma(G[X]) + \frac{1}{2}$. Hence, by Lemma 59, if $\beta(G') = \gamma(G') + k'$ for some k', then $\beta(G) \geq \gamma(G) + k' + \frac{1}{4}$, as required. □

We will give an algorithm that is built around the application of the four previous Lemmas. In order to show that the algorithm works, we will need the following structural lemma. This lemma is similar in spirit to Lemma 17, although the details of the proof are different.

Lemma 64. *Given any connected undirected graph H, at least one of the following properties holds:*

A *There exist $v \in V(H)$ and $X \subseteq V(H)$ such that X is a connected component of $H - v$ and X is a clique;*

B *There exist $a, b, c \in V(H)$ such that $H[\{a, b, c\}]$ is isomorphic to P_3 and $H - \{a, b, c\}$ is connected;*

C *There exist $x, y \in V(H)$ such that $\{x, y\} \notin E(H)$, $H - \{x, y\}$ is has two components X and Y, and $X \cup \{x\}$ and $X \cup \{y\}$ are cliques.*

Proof. Recall that a *leaf-block* is a block which contains at most one cut-vertex. Now let X be a leaf-block in H with cut-vertex r. (If H contains no cut-vertices, let r be an arbitrary vertex).

Recall that for any set of vertices Z, the set $R(r, Z)$ is the set of vertices reachable from r in $H - Z$. Consider the partial order $\prec$ on the induced P_3's in X, such that $\{a, b, c\} \prec \{a', b', c'\}$ if $R(r, \{a, b, c\})$ is a strict subset of $R(r, \{a', b', c'\})$. For $a, b, c \in X$, we call $\{a, b, c\}$ an *important* P_3 if $\{a, b, c\}$ is a maximal element in this partial order. That is, $\{a, b, c\}$ is an important P_3 if $\{a, b, c\}$ is an induced P_3 and there is no induced P_3 $\{a', b', c'\} \subseteq X$ such that $R(r, \{a, b, c\}) \subset R(r, \{a', b', c'\})$.

Observe that if there is no induced P_3 in X then X is a clique, and Case A applies (consider the shortest path between any two vertices in X which are not adjacent, and observe that any three consecutive vertices in this path form an induced P_3). Therefore we may assume there is an important P_3 in X. Let $\{a, b, c\}$ be an important P_3 with $ab, bc \in E(G), ac \notin E(G)$. If $r \in \{a, b, c\}$ then $X - r$ must have no induced P_3 and so $X - r$ is a clique, and Case A applies. Therefore we may assume $r \notin \{a, b, c\}$.

Let $Y = H - (R(r, \{a, b, c\}) \cup \{a, b, c\})$. If $Y = \emptyset$ then $H - \{a, b, c\}$ is connected and Case B applies. Therefore we may assume $Y \neq \emptyset$. Since X is 2-connected, at least two of a, b, c are adjacent to $R(r, \{a, b, c\})$. In particular, one of a and c is adjacent to $R(r, \{a, b, c\})$. Without loss of generality, assume c is adjacent to $R(r, \{a, b, c\})$.

We now prove some properties of Y, which will be used to show that Case C applies.

Property 1: For any $x \in Y$, $ax \in E(H)$ if and only if $bx \in E(H)$.

To see this, suppose there exists $x \in Y$ which is adjacent to a but not b. Then $\{x, a, b\}$ is an induced P_3, and $R(r, \{x, a, b\}) \supseteq R(r, \{a, b, c\}) \cup c$, a contradiction as $\{a, b, c\}$ is an important P_3. Similarly, if x is adjacent to b but not a, then $\{a, b, x\}$ is an induced P_3 and $R(r, \{a, b, x\}) \supseteq R(r, \{a, b, c\}) \cup c$.

Property 2: For each $s \in \{a, b, c\}$ and any $x, y \in Y$, if $sx \in E(H)$ and $xy \in E(H)$, then $sy \in E(H)$.

For suppose not. Then $\{s, x, y\}$ is an induced P_3, and since at least one vertex in $\{a, b, c\} \backslash s$ is adjacent to $R(r, \{a, b, c\})$, we have that $R(r, \{a, b, c\}) \subset R(r, \{s, x, y\})$, a contradiction as $\{a, b, c\}$ is an important P_3.

Property 3: For each $s \in \{a, b, c\}$ and any $x, y \in Y$, if $sx \in E(H)$ and $sy \in E(H)$, then $xy \in E(H)$.

For suppose not. Then $\{x, s, y\}$ is an induced P_3, and as with Property 2 we have a contradiction.

There are now two cases to consider. First consider the case when a is adjacent to $R(r, \{a, b, c\})$. Then by a similar argument to Property 1, we may show that for any $x \in Y$, $cx \in E(H)$ if and only if $bx \in E(H)$. This together with Property 1 implies that a, b, c have exactly the same neighbours in Y. Then by Properties 2 and 3, we have that Y is a clique and every vertex in y is adjacent to each of a, b, c. If b is adjacent to $R(r, \{a, b, c\})$ then for any $x \in Y$, $\{a, x, c\}$ is an induced P_3 and $R(r, \{a, x, c\}) \supseteq R(r, \{a, b, c\}) \cup b$, a contradiction as $\{a, b, c\}$ is an important P_3. So b is not adjacent to $R(r, \{a, b, c\})$. Then $Y \cup \{b\}$ and $R(r, \{a, b, c\})$ are the two connected components of $H - \{a, c\}$, and $(Y \cup \{b\}) \cup \{a\}$ and $(Y \cup \{b\}) \cup \{c\}$ are cliques. Therefore Case C applies.

Now consider the case when a is not adjacent to $R(r, \{a, b, c\})$. Then as X is 2-connected, b must be adjacent to $R(r, \{a, b, c\})$. Furthermore there must be a path from a to c in $H - b$, and the intermediate vertices in this path must be in Y. By Property 2, there exists $x \in Y$ adjacent to a and c. Then $\{a, x, c\}$ is an induced P_3 and $R(r, \{a, x, c\}) \supseteq R(r, \{a, b, c\}) \cup b$, a contradiction as $\{a, b, c\}$ is an important P_3. □

Lemma 65. *For any connected oriented graph G with at least one arc, one of the following cases applies:*

1. *There exists a vertex x and S a set of two vertices such that $G[S]$ is a component of $G - x$ and $G[S \cup \{x\}]$ is a directed 3-cycle.*
2. *There exists a vertex x such that $d^+(x) \neq d^-(x)$, and $G - \{x\}$ is connected.*
3. *There exists a set of vertices X such that $G - X$ is connected, $G[X]$ is a tournament, and $|X| \geq 4$.*
4. *There exists a set of vertices X such that X is an induced P_3 in the underlying graph of G, and $G - X$ is connected.*

Proof. We will consider the cases A,B,C from Lemma 64 applied to the underlying graph of G. If there is a vertex $x \in X$ such that $G-x$ is connected and $d^+(x) \neq d^-(x)$ (we will call such a case an *unbalanced case*), then Case 2 applies. Thus, assume that for each $x \in X$ such that $G - x$ is connected we have $d^+(x) = d^-(x)$.

Consider the case when property A holds. If $|X| \geq 4$, Case 3 applies on $S = X$. If $|X| = 3$, there has to be exactly one arc between X and v and $G[X]$ is a directed 3-cycle as otherwise we have an unbalanced case. Let $x \in X$ be the endpoint of this arc in X. Then Case 1 applies with $S = X \backslash \{x\}$. If $|X| = 2$, then $G[X \cup \{v\}]$ is a directed 3-cycle (as otherwise we have an unbalanced case) and so Case 1 applies. We cannot have $|X| = 1$ as this is an unbalanced case.

If property B holds, then Rule 4 can be applied to the path P_3 formed by a, b, c in the underlying graph of G.

Consider the case when property C holds. We may assume without loss of generality that the non-tournament component is adjacent to y.

Consider the subcase when $G - \{x, y\}$ has two connected components, X_1 and X_2, that are tournaments. Let $x_1 \in X_1$, $x_2 \in X_2$ and observe that the subgraph induced by x_1, x, x_2 forms a P_3 in the underlying graph of G and $G - \{x_1, x, x_2\}$ is connected, and so Case 4 applies.

Now consider the subcase when $G - \{x, y\}$ has only one connected component X that is a tournament. If $|X| \geq 3$, then $X \cup \{x\}$ is a tournament with least four vertices, and so Case 3 applies. If $|X| = 2$, then let $X = \{a, b\}$. Observe that a is adjacent to three vertices, b, x, y, and so we have an unbalanced case to which Case 2 applies. Finally, $X = \{a\}$ is a singleton, then observe that x, a, y form a P_3 in the underlying graph of G and $G - \{x, a, y\}$ is connected, and so Case 4 applies. □

Consider algorithm Acyc.
input : A connected oriented graph G

output: A set $S \subseteq V(G)$; a non-negative real number t

Set $S = \emptyset$;

Set t=0;

Case 1: if *There exists a vertex x and X a set of two vertices such that $G[X]$ is a component of $G - x$ and $G[X \cup \{x\}]$ is a directed 3-cycle.* **then**

  Set $G' = G - X$;

  Set $(S, t) = \text{Acyc}(G')$;

else

  Case 2: if *There exists a vertex x such that $d^+(x) \neq d^-(x)$, and $G' = G - \{x\}$ is connected.* **then**

    Set $(S', t') = \text{Acyc}(G')$;

    Set $S = S' \cup \{x\}$;

    Set $t = t' + \frac{2|d^+(x) - d^-(x)| - 1}{4}$;

  else

    Case 3: if *There exists a set of vertices X such that $G' = G - X$ is connected, $G[X]$ is a tournament, and $|X| \geq 4$.* **then**

      Set $(S', t') = \text{Acyc}(G')$;

      if *$|X|$ is even* **then**

        Set $t = t' + \frac{2|X| - 4}{4}$;

      else

        Set $t = t' + \frac{2|X| - 7}{4}$;

      end

    else

      Case 4: if *There exists a set of vertices X such that X is an induced P_3, and $G' = G - X$ is connected* **then**

        Set $(S', t') = \text{Acyc}(G')$;

        Set $S = S' \cup X$;

        Set $t = t' + \frac{1}{4}$;

      end

    end

  end

end

return (S, t);

Algorithm 3: Algorithm Acyc

Lemma 66. *Algorithm* Acyc *can be run in polynomial time.*

Proof. Observe that each recursive call of ACYC either calls ACYC once on an instance with fewer vertices. Therefore, for a graph with n vertices, there will be at most n recursive calls of of ACYC. Observe that a single iteration of ACYC can be run in $O(n^3)$ time. Therefore in total ACYC(G) takes $O(mn^4)$ time. □

Observe that at each iteration of the algorithm, if we add q vertices to S, we increase t by at least $q/12$ (using the fact that all weights are integer; the worst case is when we have an induced P_3, and add 3 vertices to S and possibly increase t by only 1/4.) Therefore we have the following lemma.

Lemma 67. *Let $(S,t) = \text{ACYC}(G)$. Then $|S| \leq 12t$.*

Lemma 68. *Let $(S,t) = \text{ACYC}(G)$. Then $\beta(G) \geq \gamma(G) + t$.*

Proof. We prove the claim by induction on $|G|$. Observe that if $|G| = 0$ then $t = 0$, and so $\beta(G) \geq \gamma(G) + t$ is true by definition. So now suppose that $|G| = n$, and claim is true for smaller graphs.

Observe that by Lemma 65, one of Cases 1, 2, 3 or 4 must apply. Then for each of these cases, the claim follows from Lemmas 60, 61, 62, and 63 respectively. □

Lemma 69. *Let $(S,t) = \text{ACYC}(G)$. Then $G - S$ is a forest of cliques with the following properties:*

1. *Every block in $G - S$ contains at most three vertices;*
2. *Every block X in $G - S$ with $|X| = 3$ induces a directed 3-cycle in G;*
3. *Every connected component in $G - S$ has at most one block X with $|X| = 2$ vertices;*
4. *There is at most one block in $G - S$ with one vertex (i.e., there is at most one isolated vertex in $G - S$).*

Proof. We prove the claim by induction on $|G|$. Observe that if $|G| = 0$ or 1 then $G - S$ satisfies all the properties of the claim. So now suppose that $|G| = n$, and claim is true for smaller graphs. Observe that by Lemma 65, one of Cases 1, 2, 3 or 4 must apply.

For Case 2,3,4, observe that $G - S = G' - S'$ which satisfies the claim by the inductive hypothesis.

Now consider Case 1. Observe that $S = S'$ and so $G' - S = G' - S'$, which satisfies the claim. If $x \in S'$ then $G - S$ consists of $G' - S$ with an extra component, consisting of the block X with two vertices. Thus $G - S$ also satisfies the claim. If

$x \notin S'$ then $G - S$ is the same as $G' - S$, except that one component has an extra block $X \cup \{x\}$ which is a directed 3-cycle. Thus, $G - S$ also satisfies the claim. □

4.2 Fixed-parameter tractability of ASAPT

To prove that unweighted ASAPT is fixed-parameter tractable, we use the algorithm described in the previous section, together with an algorithm for the following classical (in the sense of 'unparameterized') problem.

> Acyclic Subgraph Relative to S (ASR-S)
> *Instance:* An oriented graph G with vertex set $V = \{v_1, \dots v_n\}$, such that G is a forest of cliques with at most 3 vertices in each block, a separate set of vertices $S = \{s_0, s_1, \dots, s_l, s_{l+1}\}$, a series of function $w_j : V \to \mathbb{N}_0$ for each $j \in \{0, 1, \dots, l\}$, an integer t
> *Question:* Is there an ordering $\succ$ of $V \cup S$ in which $s_i \succ s_{i+1}$ for all $i \in [l-1]$, such that
> $$(|\{xy \in A(G) : x \succ y\}| + \sum_{s_j \succ x \succ s_{j+1}} w_j(x)) \geq t?$$

The main idea of our FPT algorithm for ASAPT is as follows. Recall that finding a maximum acyclic subgraph is equivalent to finding an ordering on the vertices which satisfies as many arcs as possible. We first run Acyc(G) to get a set of at most $12k$ vertices S such that $G - S$ is a block graph with at most 3 vertices in each block. We then guess an ordering $s_1, \dots, s_l$ of S, and solve an instance of ASR-S applied to $G - S$, in which for vertex x in $G - S$ and each $j \in [l-1]$, the value of $w_j(x)$ is the number of arcs between x and S that are satisfied if x is placed between s_j and s_{j+1} in some ordering.

Lemma 70. ASR-S *can be solved in time* $l^2 n^{O(1)}$

Proof. If G contains no arcs, then we can find an optimal ordering by putting each vertex x between s_{j_x} and s_{j_x+1}, where j_x is picked to maximise $w_{j_x}(x)$.

Otherwise, we provide a polynomial-time transformation that replaces an instance $(G, S, w_0, w_1, \dots, w_l, t)$ with an equivalent instance $(G', S, w'_0, w'_1, \dots, w'_l, t)$ such that G' has fewer vertices than G. By applying the transformation at most $|V(G)|$ times to get a trivial instance, we have a polynomial-time algorithm to decide $(G, S, w_0, w_1, \dots, w_l, t)$.

Pick a leaf-vertex X of G with cut-vertex v. If there are no cut-vertices, then let X be an arbitrary block with more than one vertex, and let v be an arbitrary

vertex in X. Now for each $i \in \{0, 1, \ldots, l\}$, choose an ordering of $S \cup X$ that preserves $s_j \succ s_{j+1}$ for each $j \in \{0, 1, \ldots, l\}$, assigns v to a position between s_i and s_{i+1}, and maximises $(|\{xy \in A(X) : x \succ y\}| + \sum_{s_j \succ x \succ s_{j+1}} w_j(x))$. Let W_i be the value of this sum under this ordering. Thus, W_i is the largest possible value of $(|\{xy \in A(X) : x \succ y\}| + \sum_{s_j \succ x \succ s_{j+1}} w_j(x))$ under the assumption that $s_i \succ v \succ s_{i+1}$. Since $X \cup S$ contains at most $l + 4$ vertices, and a relative ordering of $l + 2$ of them is already fixed, there are at most $(l + 2)(l + 3)$ possible orderings to consider in order to determine the value of W_i.

Now, let G' be the graph derived from G by removing all vertices of X except v, and for each $i \in \{0, 1, \ldots, l\}$, let w'_i be w_i restricted to $G - (X \setminus v)$, except that $w_i(x) = W_i$. Then observe that $(G, S, w_0, w_1, \ldots, w_l, t)$ is a YES-instance of ASR-S if and only if $(G', S, w'_0, w'_1, \ldots, w'_l, t)$ is a YES-instance of ASR-S, as required.

□

Theorem 71. *There is an algorithm for the unweighted version of* ASAPT *with running time* $(12k)!(12k)^2 n^{O(1)}$.

Proof. Given an instance (G, k) of ASPT, let $(S, t) = \text{ACYC}(G)$. If $t \geq k$, then by Lemma 68, (G, k) is a YES-instance. Otherwise, by Lemma 67, $|S| < 12k$. By Lemma 69, $G - S$ is a forest of cliques in which every block has at most 3 vertices.

Let $l = |S|$. Now for each possible ordering $s_1, \ldots, s_l$ of S, do the following.

Add two new isolated vertices s_0, s_{l+1} to S. So now we have an ordering $s_0, s_1, \ldots, s_l, s_{l+1}$ of S, such that there are no arcs between s_0 and x or s_{l+1} and x for any $x \in G - S$. (The point of introducing s_0 and s_{l+1} is simply so that we may assume there is an optimal solution in which $x \in G - S$ appears between s_j and s_{j+1} for some j.) Define the functions $w_0, w_1, \ldots w_l$ as follows. For each $i \in \{0, 1, \ldots, l\}$ and each vertex x of $G - S$, let $w_i(x) = |\{xs_j \in A(G) : j > i\}| + |\{s_j x \in A(G) : j \leq i\}|$. That is, $w_i(x)$ is the number of arcs between x and S that are satisfied if x is added to the ordering $s_0, s_1, \ldots, s_l, s_{l+1}$ between s_i and s_{i+1}.

Let t_S be the number of arcs $s_i s_j \in A(G)$ with $i < j$. Now observe that for any ordering $\succ$ of $V(G)$ in which $s_i \succ s_{i+1}$ for all $i \in \{0, 1, \ldots, l\}$, the number of arcs in G satisfied is exactly $t_S + (|\{xy \in A(G) : x \succ y\}| + \sum_{s_j \succ x \succ s_{j+1}} w_j(x))$. Therefore there is such an ordering that satisfies at leaf $\gamma(G) + k$ arcs if and only if $(G - S, S, w_0, w_1, \ldots, w_l, t)$ is a YES-instance of ASR-S, where $t = \gamma(G) + k - t_S$. By Lemma 70, this can be decided in $(12k)^2 n^{O(1)}$ time.

Since $|S \setminus \{s_0, s_{l+1}\}| \leq 12k$, there are at most $(12k)!$ orderings of S to consider, and one of these must correspond to an optimal ordering of $V(G)$. Thus, we can solve the problem in time $(12k)!(12k)^2 n^{O(1)}$. □

4.3 Polynomial Kernel for ASAPT

We now prove a polynomial kernel for unweighted ASAPT. We begin by applying the following reduction rules.

Rule 1. *Let x be a cut-vertex and X a set of two vertices such that $G[X]$ is a component of $G - x$ and $G[X \cup \{x\}]$ is a directed 3-cycle. Then replace G with $G' := G - X$.*

Lemma 72. *If (G', k) is an instance obtained from (G, k) by an application of Rule 1, then G' is connected, and (G', k) is a* YES*-instance of* ASAPT *if and only if (G, k) is a* YES*-instance of* ASAPT*.*

Proof. Any two components of $G' - x$ will be connected by x and so G' is connected. Since $\beta(G') = \beta(G) - 2$, $m' = m - 3$ and $n' = n - 2$, we have $\beta(G) \geq \frac{m}{2} + \frac{n-1}{4} + k$ if and only if $\beta(G') \geq \frac{m'}{2} + \frac{n'-1}{4} + k$. □

Rule 2. *Let a, b, c, d, e be five vertices in G such that $G[a, b, c]$ and $G[c, d, e]$ are directed 3-cycles, $G[a, b, c, d, e] = G[a, b, c] \cup G[c, d, e]$ and a, e are the only vertices in $\{a, b, c, d, e\}$ that are adjacent to a vertex in $G - \{a, b, c, d, e\}$. Then delete b, c and d, and add a new vertex x and three arcs such that $G[a, x, e]$ is a directed 3-cycle.*

Lemma 73. *If (G', k) is an instance obtained from (G, k) by an application of Rule 2, then G' is connected, and (G', k) is a* YES*-instance of* ASAPT *if and only if (G, k) is a* YES*-instance of* ASAPT*.*

Proof. Clearly, G' is connected. Note that $\beta(G') = \beta(G) - 2$, $m' = m - 3$ and $n' = n-2$. Thus, we have $\beta(G) \geq \frac{m}{2} + \frac{n-1}{4} + k$ if and only if $\beta(G') \geq \frac{m'}{2} + \frac{n'-1}{4} + k$. □

Rule 3. *Let x be a vertex such that $d^+(x) + d^-(x) = 1$. Then delete x from G and reduce k by $\frac{1}{4}$.*

Lemma 74. *If (G', k') is an instance obtained from (G, k) by an application of Rule 3, then G' is connected, and (G', k) is a* YES*-instance of* ASAPT *if and only if (G, k') is a* YES*-instance of* ASAPT*.*

Proof. Clearly, G' is connected. Note that $\beta(G') = \beta(G) - 1$, $m' = m - 1$ and $n' = n-1$. Thus, we have $\beta(G) \geq \frac{m}{2} + \frac{n-1}{4} + k$ if and only if $\beta(G') \geq \frac{m'}{2} + \frac{n'-1}{4} + k - \frac{1}{4}$. □

We now assume that (G, k) is reduced by Rules 1, 2 and 3. In what follows we may assume we have the set S derived from $\textsc{Acyc}(G)$ and $|S| \leq 12k$ (as otherwise by Lemmas 67 and 68 we have a YES-instance). Recall the properties of $G - S$ given

by Lemma 69. We say that a set $\{s, a, b\}$ of vertices is a *dangerous triangle* if $s \in S$, $G[a, b]$ is a block in $G - S$, and $G[s, a, b]$ is a directed 3-cycle.

Lemma 75. *Let T be a directed 3-cycle, with vertices labelled 0 or 1. Then there exists an acyclic subgraph of T with two arcs, such that there is no arc from a vertex labelled 1 to a vertex labelled 0.*

Proof. Let $V(T) = \{a, b, c\}$ and assume that a, b are labelled 0. Since T is a cycle, either the arc ac or bc exists. This arc, together with the arc between a and b, form the required acyclic subgraph. A similar argument holds when two vertices in T are labelled 1. □

Lemma 76. *For a vertex $s \in S$, let t_s denote the number of neighbors of s in $G - S$ which do not appear in a dangerous triangle containing s. If $t_s \geq 16k$, then we have a* YES*-instance.*

Proof. Let Y denote the subgraph of $G - S$ consisting of all components C of $G - S$ which have a neighbor of s. For each component C of Y, let $t_s(C)$ denote the number of neighbors of s in C which do not appear in a dangerous triangle containing s.

For each vertex $x \in G - S$, label it 0 if there exists an arc from x to s, or 1 if there is an arc from s to x. Recall from Lemma 69 each connected component in $G - S$ has at most one block $X = \{x, y\}$ with $|X| = 2$. If one vertex x is labelled, assign y the same label. Finally, assign label 1 to any remaining unlabelled vertices in $G - S$.

We will now construct an acyclic subgraph H' of $G - S$ such that there is no arc from a vertex labelled 1 to a vertex labelled 0. We then extend this to an acyclic subgraph H containing all the arcs between s and Y.

Consider each block X in $G - S$. If $|X| = 3$, and X is a directed 3-cycle, then by Lemma 75 there is an acyclic subgraph of X with two arcs. Add this to H'. Now suppose $|X| = 2$, and let a, b be the vertices of X with an arc from a to b. If $G[X \cup \{s\}]$ is a dangerous triangle, then a is labelled 1 and b is labelled 0. In this case we do not include the arc ab in H'. However, H will include the two arcs between X and s, which do not count towards $t_s(C)$. If $G[X \cup \{s\}]$ is not a dangerous triangle, then we include the arc ab in the acyclic subgraph H'. Finally, let H be the acyclic subgraph formed by adding all arcs between s and Y to H'.

Observe that for each component C of Y, if $G[C \cup \{s\}]$ contains no dangerous triangle then H contains at least $\gamma(C)$ arcs in $G[C]$ (by the construction of H') and $t_s(C)$ arcs between C and s (since all arcs between Y and s are in H), and $\gamma(C \cup \{s\}) := \gamma(G[C \cup \{s\}]) = \gamma(C) + \frac{t_s(C)}{2} + \frac{1}{4}$. So H contains at least $\gamma(C \cup \{s\}) +$

$\frac{t_s(C)}{2} - \frac{1}{4}$ arcs. Since $G[C \cup \{s\}]$ contains no dangerous triangle but C is adjacent to s, $t_s(C) \geq 1$, and so H contains at least $\gamma(C \cup \{s\}) + \frac{t_s(C)}{4}$ arcs.

If $G[C \cup \{s\}]$ contains a dangerous triangle then H contains at least $\gamma(C) - \frac{3}{4}$ arcs in $G[C]$ (this can be seen by contracting the arc in C appearing in the dangerous triangle, and observing that in the resulting component C', H has at least $\gamma(C')$ arcs) and $t_s(C) + 2$ arcs between C and s, and $\gamma(C \cup \{s\}) = \gamma(C) + \frac{t_s(C)+2}{2} + \frac{1}{4}$. Thus, H contains at least $\gamma(C \cup \{s\}) + \frac{t_s(C)}{2}$ arcs.

Let $C_1, C_2, \ldots, C_q$ be the components of Y. Observe that $\gamma(Y \cup \{s\}) = \sum_{i=1}^{q} \gamma(C_i \cup \{s\})$. Then by combining the acyclic subgraphs for each $G[C_i \cup \{s\}]$, we have that $\beta(G[Y \cup \{s\}]) \geq \sum_{i=1}^{q} (\gamma(C_i \cup \{s\}) + \frac{t_s(C_i)}{4}) = \gamma(Y \cup \{s\}) + \frac{t_s}{4}$.

Finally, observe $G - Y - s$ has at most $12k$ component, since each component must contain a vertex of S. By repeated application of Lemma 59, this implies there is an acyclic subgraph of G with at least $\gamma(G) + \frac{t_s - 12k}{4}$ arcs. Hence, if $t_s \geq 16k$, we have a YES-instance. □

Using the above lemma and the fact that $|S| \leq 12k$ (by Lemma 69), we have that unless (G, k) is a YES-instance, there are at most $12 \cdot 16k^2 = 192k^2$ vertices in $G - S$ that are adjacent to a vertex in S and do not appear in a dangerous triangle with that vertex.

Lemma 77. *Let t be the number of components in $G - S$ in which every neighbor of a vertex $s \in S$ appears in a dangerous triangle together with s. If $t \geq 4k$, we have a* YES*-instance.*

Proof. By Lemma 69 such a component C_i contains at most one block of size 2. Since only blocks of size 2 can have vertices in dangerous triangles, only the vertices from this block in C_i may be adjacent to a vertex in S. But since G is reduced by Rule 1, component C_i must consist of only this block. Let a_i, b_i be the vertices of C_i, $i = 1, \ldots, t$ and let $C = \cup_{i=1}^{t} \{a_i, b_i\}$. Let $a_i b_i$ be an arc for each $i = 1, \ldots, t$ and note that every arc of G containing a_i (b_i, respectively) is either $a_i b_i$ or is from S to a_i (from b_i to S, respectively). Let δ_i be the number of dangerous triangles containing a_i and b_i. Observe that if $\delta_i = 1$, then Rule 1 would apply with x being the unique neighbor of a_i and b_i in S. Therefore $\delta_i \geq 2$.

Observe that $G - C$ has at most $|S| \leq 12k$ components, and $G[C]$ has $2t$ vertices and there are $2\sum_{i=1}^{t} \delta_i + t$ arcs incident to vertices in C. Then by applying the Poljak-Turzík bound to each component of $G - C$ in turn, we have that $G - C$ has an acyclic subgraph H with at least $\frac{m - 2\sum_{i=1}^{t} \delta_i - t}{2} + \frac{n - 2t - 12k}{4}$ arcs. Observe that we can add to H all arcs entering each a_i and leaving each b_i, $i = 1, \ldots, t$, and obtain

an acyclic subgraph H^* of G. We will prove that H^* contains enough arcs to show that (G,k) is a YES-instance. Recall that each $\delta_i \geq 2$. Then, the number of arcs in H^* is at least

$$\begin{aligned}
|A(H)| + 2\sum_{i=1}^{s}\delta_i &\geq \frac{m - 2\sum_{i=1}^{t}\delta_i - t}{2} + \frac{n - 2t - 12k}{4} + 2\sum_{i=1}^{t}\delta_i \\
&\geq \gamma(G) + \sum_{i=1}^{t}\delta_i - t - \frac{12k}{4} \\
&\geq \gamma(G) + \sum_{i=1}^{t}(\delta 1) - 3k \geq \gamma(G) + t - 3k \geq \gamma(G) + k.
\end{aligned}$$

□

Let H be A forest of cliques, where each block contains at most three vertices. We denote the set of leaf-blocks of H by $\mathcal{L}(H)$. A block B of H is called a *path-block* if there is another block B' of H such that B and B' have a common vertex c which belongs only to these two blocks, at most one vertex of B belongs to a block other than B', and at most one vertex of B' belongs to a block other than B. We denote the set of path-blocks which are not leaf-blocks by $\mathcal{P}(H)$.

Lemma 78. *For a forest of cliques H, with each block of size at most three, if $l = |\mathcal{L}(H)|$ and $p = |\mathcal{P}(H)|$ then $|V(H)| \leq 8l + 2p$.*

Proof. We prove the claim by induction on the number of blocks in H. The case when H has only one block is trivial. Thus, we may assume that H has at least two blocks and H is connected. Let B be a leaf-block of H, and obtain subgraph H' by deleting the vertices of B not belonging to another block. Note that $|V(H)| \leq |V(H')| + 2$.

Assume that H' has a leaf-block B' which is not a leaf-block in H. Observe that $B' \in \mathcal{P}(H)$ and by induction $|V(H)| \leq 2 + 8l + 2(p-1) \leq 8l + 2p$.

Now assume that $|\mathcal{L}(H')| = l - 1$. Observe that removal of B from H may lead to a neighbour of B, B', becoming a path-block in H', together with at most two blocks neighbouring B'. Thus, at most three blocks may become path-blocks in H'. By the induction hypothesis, $|V(H')| \leq 8(l-1) + 2(p+3)$. Hence, $|V(H)| \leq 8(l-1) + 2(p+3) + 2 \leq 8l + 2p$. □

Theorem 79. ACYCLIC SUBGRAPH ABOVE POLJAK-TURZÍK BOUND (ASAPT) *has a kernel with $O(k^2)$ vertices and $O(k^2)$ arcs.*

Proof. Consider an instance of (G^*, k) of ASAPT. Apply Rules 1, 2 and 3 to obtain an instance (G,k) reduced by Rules 1, 2 and 3. Assume that (G,k) is a NO-instance.

Now let S be the set of vertices derived by $\textsc{Acyc}(G)$, and recall that we may assume $|S| \leq 12k$ as otherwise (G, k) is a YES-instance.

By Lemma 76, each $s \in S$ has at most $16k$ neighbors that do not appear in a dangerous triangle with s. By Lemma 77, there are at most $8k$ vertices in $G - S$ that appear in a dangerous triangle with every neighbor in S (there are at most $4k$ components, and each component has two vertices). Hence the number of neighbors in $G - S$ of vertices of S is at most $16k|S| + 8k = 192k^2 + 8k$.

Now we will adopt the terminology and notation of Lemma 78. Consider a leaf-block B in $G - S$. Since G is reduced by Rules 1 and 3, B must contain a vertex v adjacent to S, and furthermore, v is not contained in any other block. Hence, $|\mathcal{L}(G - S)| \leq 192k^2 + 8k$.

Next, we observe that Rule 2 implies there do not exist two adjacent 3-vertex blocks $B = \{a, b, c\}$, $B' = \{c, d, e\}$ such that only a and e belong to other blocks, unless one of b, c, d has a neighbor in S. Observe that each connected component of $G - S$ contains at most one 2-vertex block, so there are at most $192k^2 + 8k$ 2-vertex path blocks. Each 2-vertex path block is adjacent to at most two 3-vertex path blocks. Hence, $|\mathcal{P}(G - S)| \leq 6(192k^2 + 8k)$. So, by Lemma 78, $|V(G - S)| \leq 8(192k^2 + 8k) + 2 \cdot 6(192k^2 + 8k) = O(k^2)$, and so $|V(G)| \leq O(k^2) + 12k = O(k^2)$.

Finally, we show G has $O(k^2)$ arcs. There are at most $|S|^2$ arcs in S. Between $G - S$ and S there are at most $(16k + 8k)|S|$ arcs. Finally, observe that $G - S$ has at most $|V(G - S)| \leq 20(192k^2 + 8k)$ blocks, and each block contains at most 3 arcs. Hence, $|A(G)| \leq |S|^2 + 60(192k^2 + 8k) \leq 144k^2 + 60(192k^2 + 8k) = O(k^2)$.

Thus, either (G, k) is a YES-instance, or (G, k) forms a kernel with $O(k^2)$ vertices and $O(k^2)$ arcs. □

Part II

Below guarantee parameterizations

Chapter 5

Hitting Set below upper bounds

Given a hypergraph $H = (V, \mathcal{E})$, we say a vertex v *hits* an edge e if $v \in e$. A *hitting set* of H is a set $S \subseteq V$ such that $e \cap S \neq \emptyset$ for all $e \in \mathcal{E}$.

In this chapter, we define $\beta(H)$ as follows.

Definition 80. *Let H be a hypergraph. Then*

$$\beta(H) = \min\{|S| : S \text{ is a hitting set of } H\}$$

The Hitting Set problem is as follows:

Hitting Set
Instance: A hypergraph H, an integer p.
Question: Is $\beta(G) \leq p$?

Hitting Set is a well-known problem with various applications, e.g., in software testing [44], in computer networks [46] and in bioinformatics [64]. Hitting Set is equivalent to the Set Cover problem and several of its special cases are of importance (e.g., the Vertex Cover and Dominating Set problems). Hitting Set is NP-complete and its standard parameterization (when p is the parameter) is W[2]-complete.

Let $\mathcal{H}$ be a class of hypergraphs. Given a function $\Phi : \mathcal{H} \times \mathbb{N} \to \mathbb{N}$, define the parameterized problem $\Phi(H, k)$-Hitting Set as follows:

$\Phi(H,k)$-Hitting Set ($\Phi(H,k)$-HitSet)
Instance: A hypergraph $H \in \mathcal{H}$ with n vertices, m edges, an integer k.
Parameter: k.
Question: Is $\beta(G) \leq \Phi(H, k)$?

The function $\Phi(H, k)$ will be expressed in terms of the properties of H, including n and m, and the parameter k. Thus, for example, in the problem $(m-k)$HitSet, we are given a hypergraph H and a parameter k, and asked whether there H has a hitting set with at most $m-k$ vertices.

In this chapter, we consider the problems $(m-k)$HitSet and $(n-k)$HitSet. These are both parameterizations below a tight bound. It is clear that n and m are both upper bounds on the minimum size of a hitting set. To see that n and m are both tight, consider hypergraphs in which every vertex appears in a singleton, and hypergraphs in which all edges are disjoint, respectively.

Furthermore, note that m may be much greater or smaller than n. Consider a hypergraph with m edges and 2^m vertices, such that for every subset $\mathcal{F}$ of edges there is a vertex contained in exactly the edges of $\mathcal{F}$. This is a graph with $m = \log n$, in which there are no parallel edges and every vertex is contained in a different set of edges. At the other extreme, consider a hypergraph with n vertices and $2^n - 1$ edges, in which every possible subset of vertices except the empty set appear as edges. This is a graph with $m = 2^n - 1$, in which again there are no parallel edges and every vertex is contained in a different set of edges. This shows that both n and m are interesting upper bounds to consider for HitSet.

Aside from the standard parameterization of Hitting Set, the most well-known parameterization is to ask for a hitting set of size k and parameterize by $k + s$, where s is the maximum size of a set in $\mathcal{F}$. This parameterization is fixed-parameter tractable and has a kernel of size at most s^k (see Downey and Fellows [26]). Using the Sunflower Lemma, Flum and Grohe [30] obtained a kernel of size $O(sk^s s!)$. Abu-Khzam [1] recently proved that this parameterization has a kernel in which the number of elements in the ground set V is at most $(2s-1)k^{s-1} + k$. Dom et al. [23] proved that the problem does not admit a polynomial-size kernel unless coNP$\subseteq$NP/poly.

5.1 Hitting Set parameterized below m

In what follows, given a hypergraph $H = (V, \mathcal{E})$, let $n = |V|$ and $m = |\mathcal{E}|$. In this section we consider the problem $(m-k)$HitSet.

5.1.1 k-Mini Hitting Set

Key to solving $(m-k)$HitSet will be the notion of a *mini hitting set*. We give an outline of the main idea before proceeding to formal definitions. Observe that a

hitting set S of size at most $|\mathcal{E}(H)| - k$ is a set which hits at least $|S| + k$ edges. In fact, if we can find any set S' that hits at least $|S'| + k$ edges, it is easy to extend this to a hitting set of size at most $|\mathcal{E}(H)|$. For each edge e not hit by S', simply pick a vertex $v \in e$ and add v to S'. Then this gives us a hitting set S with size at most $|S'| + |\mathcal{E}(H)| - (|S'| + k) = |\mathcal{E}(H)| - k$. Thus, to solve $(m-k)$HitSet it is enough to decide whether there is a set S that hits at least $|S| + k$ edges. Furthermore, we will show that if such a set S exists, then there is such a set S which is relatively small. Thus, $(m-k)$HitSet now becomes a problem of finding a relatively small structure within the hypergraph.

Definition 81. *Let $H = (V, \mathcal{E})$ be a hypergraph and let k be an integer. Then a* k-mini hitting set *of H is a set $S_{\text{MINI}} \subseteq V$ such that $|S_{\text{MINI}}| \leq k$ and $|\mathcal{E}[S_{\text{MINI}}]| \geq |S_{\text{MINI}}| + k$.*

Lemma 82. *A hypergraph $H = (V, \mathcal{E})$ has a hitting set of size at most $|\mathcal{E}| - k$ if and only if it has a k-mini hitting set. Moreover,*

1. *Given a k-mini hitting set S_{MINI}, we can construct a hitting set S with $|S| \leq |\mathcal{E}| - k$ such that $S_{\text{MINI}} \subseteq S$ in polynomial time.*
2. *Given a hitting set S with $|S| \leq |\mathcal{E}| - k$, we can construct a k-mini hitting set S_{MINI} such that $S_{\text{MINI}} \subseteq S$ in polynomial time.*

Proof. 1. For each edge e not hit by S_{MINI}, pick one vertex in e and add it to S_{MINI}. The resulting set S contains at most $|\mathcal{E}| - k$ vertices and hits every edge of $\mathcal{E}$.

2. If $|S| \leq k$ then S itself is a k-mini hitting set.

   If $|S| > k$, construct S_{MINI} as follows. Let $S_0 = \emptyset$, and for every $0 \leq i \leq m-k-1$, let $S_{i+1} = S_i \cup \{v\}$, where $v \in S \backslash S_i$ is picked to maximise $|\mathcal{E}[v] \backslash \mathcal{E}[S_i]|$. Suppose for a contradiction that $|\mathcal{E}[S_k]| < |S_k| + k$. Then for some $j < k$, $|\mathcal{E}[S_{j+1}]| = |\mathcal{E}[S_j]| + 1$. Thus by construction, $|\mathcal{E}[S_{i+1}]| \leq |\mathcal{E}[S_i]| + 1$ for all $i \geq j$. It follows that $|\mathcal{E}[S]| = |\mathcal{E}[S_{m-k}]| < |S_{m-k}| + k = |\mathcal{E}|$, a contradiction. Therefore $|\mathcal{E}[S_k]| \geq |S_k| + k$, and thus S_k is the required S_{MINI}.

□

Now define the problem k-mini Hitting Set as follows.

k-mini Hitting Set (k-miniHitSet)
Instance: A hypergraph H, an integer k.
Parameter: k.
Question: Does H have a k-mini hitting set?

Corollary 83. *The problems* $(m-k)$HitSet *and* k-miniHitSet *are equivalent.*

Proof. By Lemma 82, for any hypergraph H and integer k, (H,k) is a Yes-instance of $(m-k)$HitSet if and only if (H,k) is a Yes-instance of k-miniHitSet. □

Thus, in the remainder of this section we will concentrate on k-miniHitSet.

5.1.2 Fixed-parameter tractability of k-miniHitSet

In this section, we give an algorithm to solve k-miniHitSet in time $c^k(m+n)^{O(1)}$, where c is a constant.

This is based on the technique of color-coding developed by Alon et. al. [4]

Next we give an algorithm that finds a k-mini hitting set S_{MINI} if it exists, in time $c^k(m+n)^{O(1)}$, where c is a constant. We first describe a randomized algorithm based on color-coding [4] and then derandomize it using hash functions. Let $\chi : E(H) \to [q]$ be a function. For a subset $S \subseteq V(H)$, $\chi(S)$ denotes the maximum subset $X \subseteq [q]$ such that for all $i \in X$ there exists an edge $e \in \mathcal{E}(H)$ with $\chi(e) = i$ and $e \cap S \neq \emptyset$. A subset $S \subseteq V(H)$ is called a *colorful hitting set* if $\chi(S) = [q]$. We now give a procedure that given a coloring function χ finds a minimum colorful hitting set, if it exists.

Lemma 84. *Given a hypergraph H and a coloring function $\chi : \mathcal{E}(H) \to [q]$, we can find a minimum colorful hitting set if there exists one in time $O(2^q q(m+n))$.*

Proof. We first check whether for every $i \in [q]$, $\chi^{-1}(i) \neq \emptyset$. If for any i we have that $\chi^{-1}(i) = \emptyset$, then we return that there is no colorful hitting set. So we may assume that for all $i \in [q]$, $\chi^{-1}(i) \neq \emptyset$. We will give an algorithm using dynamic programming over subsets of $[q]$. Let γ be an array of size 2^q indexed by the subsets of $[q]$. For a subset $X \subseteq [q]$, let $\gamma[X]$ denote the size of a smallest set $W \subseteq V(H)$ such that $X \subseteq \chi(W)$. It is clear that $\gamma[X] = 0$ if $X = \emptyset$. Otherwise, for a smallest set $W \subseteq V(H)$ such that $X \subseteq \chi(W)$, and for any $v \in W$, observe that $|W \setminus v| = \gamma[X \setminus \chi(v)]$, as otherwise we could replace $W \setminus v$ with a smaller set. Therefore we obtain the following recurrence for $\gamma[x]$:

$$\gamma[X] = \begin{cases} \min_{(v \in V(H), \chi(\{v\}) \cap X \neq \emptyset)} \{1 + \gamma[X \setminus \chi(\{v\})]\} & \text{if } |X| \geq 1, \\ 0 & \text{if } X = \emptyset. \end{cases}$$

The algorithm computes $\gamma[[q]]$ by filling the γ in the order of increasing set sizes. Clearly, each cell can be filled in time $O(q(n+m))$ and thus the whole array can be filled in time $O(2^q q(n+m))$. The size of a minimum colorful hitting set is given by $\gamma[[q]]$. We can obtain a minimum colorful hitting set by routine back-tracking. □

Theorem 85. *There exists an algorithm solving* $(m-k)$-Hitting Set *in time* $O((2e)^{2k+O(\log^2 k)}(m+n)^{O(1)})$.

Proof. Now we describe a randomized procedure to obtain a k-mini hitting set S_{MINI} in a hypergraph H, if there exists one. We do the following for each possible value p of $|S_{\text{MINI}}|$ (that is, for $1 \leq p \leq k$). Color $\mathcal{E}(H)$ uniformly at random with colors from $[p+k]$; we denote this random coloring by χ. Assume that there is a k-mini hitting set S_{MINI} of size p and some $p+k$ edges $e_1, \ldots, e_{p+k}$ such that for all $i \in [p+k]$, $e_i \cap S_{\text{MINI}} \neq \emptyset$. The probability that for all $1 \leq i < j \leq p+k$ we have that $\chi(e_i) \neq \chi(e_j)$ is $\frac{(p+k)!}{(p+k)^{p+k}} \geq e^{-(p+k)} \geq e^{-2k}$. Now, using Lemma 84 we can test in time $O(2^{p+k}(p+k)(m+n))$ whether there is a colorful hitting set of size at most p. Thus with probability at least e^{-2k} we can find a S_{MINI}, if there exits one. To boost the probability we repeat the procedure e^{2k} times and thus in time $O((2e)^{2k}2k(m+n)^{O(1)})$ we find a S_{MINI}, if there exists one, with probability at least $1-(1-\frac{1}{e^{2k}})^{e^{2k}} \geq \frac{1}{2}$. If we obtained S_{MINI} then using Lemma 82 we can construct a hitting set of H of size at most $m-k$.

To derandomize the procedure, we need to replace the first step of the procedure where we color the edges of $E(H)$ uniformly at random from the set $[p+k]$ to a deterministic one. This is done by making use of an $(m, p+k, p+k)$-*perfect hash family*. An $(m, p+k, p+k)$-perfect hash family, $\mathcal{H}$, is a set of functions from $[m]$ to $[p+k]$ such that for every subset $S \subseteq [m]$ of size $p+k$ there exists a function $f \in \mathcal{H}$ such that f is injective on S. That is, for all $i, j \in S$, $f(i) \neq f(j)$. There exists a construction of an $(m, p+k, p+k)$-perfect hash family of size $O(e^{p+k} \cdot k^{O(\log k)} \cdot \log m)$ and one can produce this family in time linear in the output size [66]. Using an $(m, p+k, p+k)$-perfect hash family $\mathcal{H}$ of size at most $O(e^{2k} \cdot k^{O(\log k)} \cdot \log m)$ rather than a random coloring we get the desired deterministic algorithm. To see this, it is enough to observe that if there is a subset $S_{\text{mini}} \subseteq V(H)$ such that $|\mathcal{E}[S_{\text{MINI}}]| \geq |S_{\text{MINI}}|+k$ then there exists a coloring $f \in \mathcal{H}$ such that the $p+k$ edges $e_1, \ldots, e_{p+k}$ that intersect S_{mini} are distinctly colored. So if we generate all colorings from $\mathcal{H}$ we must encounter the desired f. Hence for the given f, when we apply Lemma 84 we get the desired result. This concludes the description. The total time of the derandomized algorithm is $O(k2^{2k}(m+n)e^{2k} \cdot k^{O(\log k)} \cdot \log m) = O((2e)^{2k+O(\log^2 k)}(m+n)^{O(1)})$. □

5.1.3 Randomized FPT for $(m-k)$ Hitting Set

In this subsection we give a randomized algorithm for k-miniHitSet running in time $O(8^{k+O(\sqrt{k})}(m+n)^{O(1)})$. However, unlike the algorithm presented in the previous subsection we do not know how to derandomize this algorithm.

Towards this we introduce notions of a star-forest and a bush. For $l \geq 1$, we call $K_{1,\ell}$ a *star of size* ℓ; a vertex of degree ℓ in $K_{1,\ell}$ is a *central vertex* (thus, both vertices in $K_{1,1}$ are central). A *star-forest* is a forest consisting of stars. A star-forest F is said to have *dimension* $(a_1, a_2, \ldots, a_p)$ if F has p stars with sizes a_1, a_2, ..., a_p respectively. Given a star-forest F of dimension $(a_1, a_2, \ldots, a_p)$, we construct a graph, which we call a *bush of dimension* $(a_1, a_2, \ldots, a_p)$, by adding a triangle (x, y, z) and making y adjacent to a central vertex of in every star of F.

For a hypergraph $H = (V, \mathcal{E})$, the *incidence bipartite graph* B_H of H has partite sets V and $\mathcal{E}$, and there is an edge between $v \in V$ and $e \in \mathcal{E}$ in H if $v \in e$. Given B_H, we construct B_H^* by adding a triangle (x, y, z) and making y adjacent to every vertex in the V. The following lemma relates k-mini-hitting sets to bushes.

Lemma 86. *A hypergraph $H = (V, \mathcal{E})$ has a k-mini hitting set S_{MINI} if and only if there exists a tuple $(a_1, \ldots, a_p)$ such that*

(a) *$p \leq k$, $a_i \geq 1$ for all $i \in [p]$, and $\sum_{i=1}^{p} a_i = p + k$; and*

(b) *there exists a subgraph of B_H^* isomorphic to a bush of dimension $(a_1, \ldots, a_p)$.*

Proof. We first prove that the existence of a k-mini-hitting set in H implies the existence of a bush in B_H^* of dimension satisfying (a) and (b). Let $S_{\text{MINI}} = \{w_1, \ldots, w_q\}$ be a k-mini-hitting set and let $S_i = \{w_1, \ldots, w_i\}$. We know that $q \leq k$ and $|\mathcal{E}[S_{\text{MINI}}]| \geq |S_{\text{MINI}}| + k$. We define $\mathcal{E}_i := \mathcal{E}[S_i] \setminus \mathcal{E}[S_{i-1}]$ for every $i \geq 2$, and $\mathcal{E}_1 := \mathcal{E}[S_1]$. Let $\mathcal{E}_{s_1}, \ldots, \mathcal{E}_{s_r}$ be the subsequence of the sequence $\mathcal{E}_1, \ldots, \mathcal{E}_q$ consisting only of non-empty sets $\mathcal{E}_i$, and let $b_j = |\mathcal{E}_{s_j}|$ for each $j \in [r]$. Let p be the least integer from $[r]$ such that $\sum_{i=1}^{p} b_i \geq k + p$.

Observe that for every $j \in [p]$, the vertex w_{s_j} belongs to each edge of $\mathcal{E}_{s_j}$. Thus, the bipartite graph B_H contains a star-forest F of dimension $(b_1, \ldots, b_p)$, such that $p \leq k$, $b_j \geq 1$ for all $j \in [p]$, and $c := \sum_{j=1}^{p} b_j \geq p + k$. Moreover, each star in F has a central vertex in V. By the minimality of p, we have $\sum_{j=1}^{p-1} b_j < p - 1 + k$ and so $b_p \geq c + 1 - (p + k)$. Thus, the integers a_j defined as follows are positive: $a_j := b_j$ for every $j \in [p-1]$ and $a_p := b_p - c + (p + k)$. Hence, B_H contains a star-forest F' of dimension $(a_1, \ldots, a_p)$, such that each star in F' has a central vertex in V.

Thus, all central vertices are in V, $p \leq k$, $a_i \geq 1$ for all $i \in [p]$, and $\sum_{i=1}^{p} a_i = p + k$, which implies that B_H^* contains, as a subgraph, a bush with dimension $(a_1, \ldots, a_p)$ satisfying the conditions above.

The construction above relating a k-mini-hitting set of H with the required bush of B_H^* can be easily reversed in the following sense: the existence of a bush of dimension satisfying (a) and (b) in B_H^* implies the existence of a k-mini-hitting set

in H. Here the triangle ensures that the central vertices are in V. This completes the proof. □

Next we describe a fast randomized algorithm for deciding the existence of a k-mini-hitting set using the characterization obtained in Lemma 86. Towards this we will use a fast randomized algorithm for the Subgraph Isomorphism problem. In the Subgraph Isomorphism problem we are given two graphs F and G on k and n vertices, respectively, as an input, and the question is whether there exists a subgraph of G isomorphic to F. Recall that $tw(G)$ denotes the treewidth of a graph G. We will use the following result.

Theorem 87 (Fomin *et al.* [31])**.** *Let F and G be two graphs on q and n vertices respectively and $tw(F) \leq t$. Then, there is a randomized algorithm for the* Subgraph Isomorphism *problem that runs in expected time $O(2^q(nt)^{t+O(1)})$.*

Let $\mathcal{P}_\ell(s)$ be the set of all *unordered partitions* of an integer s into ℓ parts. Nijenhuis and Wilf [59] designed a polynomial delay generation algorithm for partitions of $\mathcal{P}_\ell(s)$. Let $p(s)$ be the partition function, i.e., the overall number of partitions of s. The asymptotic behaviour of $p(s)$ was first evaluated by Hardy and Ramanujan in the paper in which they develop the famous "circle method."

Theorem 88 (Hardy and Ramanujan [42])**.** *We have $p(s) \sim e^{\pi\sqrt{\frac{2s}{3}}}/(4s\sqrt{3})$, as $s \to \infty$.*

This theorem and the algorithm of Nijenhuis and Wilf [59] imply the following:

Proposition 89. *There is an algorithm of running time $2^{O(\sqrt{s})}$ for generating all partitions in $\mathcal{P}_\ell(s)$.*

Now we are ready to describe and analyse a fast randomized algorithm for deciding the existence of a k-mini-hitting set in a hypergraph H. By Lemma 86, it suffices to design and analyse a fast randomized algorithm for deciding the existence of a bush in B^*_H of dimension $(a_1, \ldots, a_p)$ satisfying conditions (a) and (b) of Lemma 86. Our algorithm starts by building B^*_H. Then it considers all possible values of p one by one ($p \in [k]$) and generates all partitions in $\mathcal{P}_p(p+k)$ using the algorithm of Proposition 89. For each such partition $(a_1, \ldots, a_p)$ that satisfies conditions (a) and (b) of Lemma 86, the algorithm of Fomin *et al.* [31] mentioned in Theorem 87 decides whether B^*_H contains a bush of dimension $(a_1, \ldots, a_p)$. If such a bush exists, we output Yes and we output No, otherwise.

To evaluate the running time of our algorithm, observe that the treewidth of any bush is 2 and any bush in Lemma 86 has at most $3k+3$ vertices. This observation, the algorithm above, Theorem 87 and Proposition 89 imply the following:

Theorem 90. *There exists a randomized algorithm solving* k-miniHitSet *in expected time* $O(8^{k+O(\sqrt{k})}(m+n)^{O(1)})$.

5.1.4 No Polynomial Kernel for $(m-k)$ Hitting Set

We now show that our exponential kernel for $(m-k)$HitSet cannot be improved to a polynomial size one, given certain complexity assumptions. We make use of a result of Dom et al. [23] who proved the following theorem. Recall that m denotes the number of edges in a hypergraph, and p denotes the size of the desired hitting set in an instance of HitSet.

Theorem 91. HitSet *parameterized by* $(m+p)$ *does not have a polynomial kernel, unless* $coNP \subseteq NP/poly$.

We may now prove the following theorem:

Theorem 92. $(m-k)$HitSet *does not have a polynomial kernel, unless* $coNP \subseteq NP/poly$.

Proof. Assume that $(m-k)$HitSet has a polynomial kernel. We will show that HitSet parameterized by $(m+p)$ has a polynomial kernel, a contradiction unless $coNP \subseteq NP/poly$.

Consider an instance (H,p) of HitSet. Let $k = m-p$. Observe that H has a hitting set of size p if and only if H has a hitting set of size $m-k$, and therefore (H,k) is a Yes-instance of $(m-k)$HitSet if and only if (H,p) is a Yes-instance of HitSet. By our assumption, there is a transformation which produces a hypergraph $H' = (V', \mathcal{E}')$ with $|V'| = n'$, $|\mathcal{E}'| = m'$ together with an integer k', such that H has a hitting set of size $m-k$ if and only if H' has a hitting set of size $m'-k'$. Furthermore, $m', n' \leq P(k)$ for some polynomial P, and the transformation takes time polynomial in n and m. We may assume without loss of generality that P is an increasing function.

Let $p' = m'-k'$, and observe that H has a hitting set of size p if and only if H' has a hitting set of size $m'+p'$. Therefore H' with parameter p' is an equivalent instance of HitSet$(p, m+p)$ which can be constructed in time polynomial in m and n, and $m', n' \leq P(k) = P(m-p) \leq P(m+p)$, i.e. the size of the instance is bounded by a polynomial in the original parameter. It remains to show that the

new parameter $m' + p'$ is also bounded by a function of the original parameter, but this follows from the fact that $p' \leq m'$. □

5.2 Hitting Set parameterized Below n

5.2.1 $W[1]$-completeness of $(n-k)$ Hitting Set

Unlike $(m-k)$HitSet, $(n-k)$HitSet is not fixed-parameter tractable unless $FPT = W[1]$.

Theorem 93. $(n-k)$HitSet *is* $W[1]$*-complete.*

Proof. To show hardness, we use a well-known reduction from independent set, in which we are given a graph $G = (V, E)$ and are asked whether it contains an independent set $V' \subseteq V$ set of size k, where k is the parameter. In our instance of $(n-k)$HitSet, we let H be G viewed as a hypergraph, that is $V(H) = V, \mathcal{E}(H) = E$. Then for any $V' \subseteq V$ with $|V'| = k$, V' is an independent set in the graph if and only if every edge contains a member of $V \backslash V'$, i.e. $V \backslash V'$ is a hitting set.

To show membership in $W[1]$, we reduce$(n-k)$HitSet to the problem p-WSat$(\Gamma^-_{2,1})$, described in Flum and Grohe [30]. $\Gamma^-_{2,1}$ is the class of CNF formulas which contain only negative literals. In the parameterized problem p-WSat$(\Gamma^-_{2,1})$, we are given a formula in $\Gamma^-_{2,1}$ and an integer parameter k, and we are asked whether the formula has a satisfying assignment in which exactly k variables are assigned True. It follows from Theorem 7.29 in Flum and Grohe [30] that p-WSat$(\Gamma^-_{2,1})$ is in $W[1]$ (a more general problem is in $W[1]$).

For an instance of $(n-k)$HitSet, let $V = \{v_1, \ldots v_n\}$ be the vertices and $\mathcal{E} = \{e_1, \ldots e_m\}$ the edges in H. For each edge $e \in \mathcal{E}$, we let the clause $C_e = \bigvee_{v_i \in e} \bar{x}_i$, and let our formula be $\bigwedge_{j \in [m]} C_{e_j}$. Then there is a hitting set of size $(n-k)$ if and only if the formula has a satisfying assignment in which exactly k variables are assigned True. This is precisely the problem p-WSat$(\Gamma^-_{2,1})$, and so we are done. □

5.2.2 Kernel for $(n-k)$ Hitting Set with bounded degeneracy

Note that in the hardness proof above, every set in the $(n-k)$HitSet instance was of size 2. This means that $(n-k)$HitSet is $W[1]$-hard even for the subcase where the edge size is bounded by r, for any $r \geq 2$. Therefore if we let the parameter be $k + \max_{e \in \mathcal{E}} |e|$, the problem is still $W[1]$-hard.

Another approach would be to consider the degree of the vertices as an additional parameter. Under this parameterization the problem does turn out to be fixed-

parameter tractable; in fact we prove a stronger result by showing that the problem is fixed-parameter tractable with respect to $k + d$, where d is the degeneracy of H. This is the problem $(n - k)$HitSet$(k + d)$.

We begin with the following simple result on the chromatic number of a d-degenerate hypergraph. For a hypergraph $H = (V, \mathcal{E})$, a mapping $c : V \to [t]$ is called a *proper t-coloring* if each edge e of H of cardinality at least two is not *monochromatic*, i.e., e has vertices u, v such that $c(u) \neq c(v)$. Here $c(u)$ is the *color* of u. The *chromatic number* $\chi(H)$ of a hypergraph H is the minimum integer t for which H has a proper t-coloring.

Lemma 94. *The chromatic number of a d-degenerate hypergraph is at most $d + 1$.*

Proof. The proof is by induction on the number n of vertices of H. For $n = 1$ no edge of H can be monochromatic, and so the chromatic number of H is 1. Now assume that $n \geq 2$. Let v be a vertex of minimum degree q in $H = (V, \mathcal{E})$. By the induction hypothesis and definition of a d-degenerate hypergraph, $\chi(H \ominus \{v\}) \leq d+1$. Consider a $(d+1)$-coloring of $H \ominus \{v\}$ and edges $e_1, \ldots, e_q$ of cardinality at least two containing v. Note that $q \leq d$ and form a set C of colors by picking one color used in each e_i (if any vertex in e_i is colored). If C is empty, add to it color 1. Clearly, $|C| \leq d$ and, thus, there is a color t not in C among colors in $[d+1]$. Assign v using color t and use one of the colors in C to color all other uncolored vertices. Observe that none of $e_1, \ldots, e_q$ is monochromatic. □

To get rid of edges of cardinality one, we use the following rule whose correctness is easy to see.

Rule 4. *If there exist $v \in V$, $e \in \mathcal{E}$ such that $e = \{v\}$, then replace $H = (V, \mathcal{E})$ by $H \ominus \{v\}$. Keep k the same.*

For a hypergraph H, a set S of vertices is *independent* if S does not contain any edge of H.

Theorem 95. *The problem $(n - k)$HitSet$(k + d)$ admits a kernel with less than $(d + 1)k$ vertices and $d(d + 1)k$ edges.*

Proof. Let H be a d-degenerate hypergraph. Using Rule 4 as long as possible, we reduce H to a d-degenerate hypergraph with no edge of cardinality 1. By Lemma 94, $\chi(H) \leq d + 1$. Consider a proper $\chi(H)$-coloring of H and a largest set S of vertices of H assigned the same color. Clearly, $|S| \geq |V|/(d + 1)$.

Now observe that T is a hitting set of $H = (V, \mathcal{E})$ if and only if $V \setminus T$ is an independent set. Thus, if $|V|/(d+1) \geq k$, the answer to $(n-k)$HitSet$(k+d)$ is Yes. Otherwise, $|V| < (d+1)k$.

To prove that $|\mathcal{E}| < d(d+1)k$, choose a vertex v of minimum degree and observe that $d(v) \leq d$. Now delete v from V and $\mathcal{E}[v]$ from $\mathcal{E}$, and choose a vertex v of minimum degree again, and observe that $d(v) \leq d$. Continuing this procedure we will delete all edges in $\mathcal{E}$ and thus $|\mathcal{E}| \leq d|V| < d(d+1)k$. □

Chapter 6

Applications of Hitting Set below upper bounds

6.1 $\nu(F) + k$ SAT

In this section we study a parameterization of MaxSat. We consider a CNF formula F as a multiset of clauses: $F = \{c_1, \ldots, c_m\}$. (We allow repetition of clauses.) We assume that no clause contains both a variable and its negation, and no clause is empty. The set of variables of F will be denoted by $V(F)$, and for a clause c, $V(c) = V(\{c\})$. A *truth assignment* is a function $\tau : V(F) \to \{\text{TRUE}, \text{FALSE}\}$. A truth assignment τ *satisfies* a clause C if there exists $x \in V(F)$ such that $x \in C$ and $\tau(x) = \text{TRUE}$, or $\bar{x} \in C$ and $\tau(x) = \text{FALSE}$. We will denote the number of clauses in F satisfied by τ as $\text{sat}_\tau(F)$ and the maximum value of $\text{sat}_\tau(F)$, over all τ, as $\text{sat}(F)$.

A function $\pi : U \to \{\text{TRUE}, \text{FALSE}\}$, where U is a subset of $V(F)$, is called a *partial truth assignment*. A partial truth assignment $\pi : U \to \{\text{TRUE}, \text{FALSE}\}$ is an *autarky* if π satisfies all clauses of F_U.

Let B_F denote the bipartite graph with partite sets $V(F)$ and F with an edge between $v \in V(F)$ and $c \in F$ if $v \in c$ or $\bar{v} \in c$. The *matching number* $\nu(F)$ of F is the size of a maximum matching in B_F. Clearly, $\text{sat}(F) \geq \nu(F)$ and this lower bound for $\text{sat}(F)$ is tight as there are formulas F for which $\text{sat}(F) = \nu(F)$.

In this section we study the following parameterized problem, where the parameterization is above a tight lower bound.

$(\nu(F) + k)$-SAT
Instance: A CNF formula F and a positive integer α.
Parameter: $k = \alpha - \nu(F)$.
Question: Is $\text{sat}(F) \geq \alpha$?

In our main result, we show that $(\nu(F)+k)$-SAT is fixed-parameter tractable by obtaining an algorithm with running time $O((2e)^{2k+O(\log^2 k)}(n+m)^{O(1)})$, where e is the base of the natural logarithm. We also develop a randomized algorithm for $(\nu(F)+k)$-SAT of expected running time $O(8^{k+O(\sqrt{k})}(m+n)^{O(1)})$.

The *deficiency* $\delta(F)$ of a formula F is $|F|-|V(F)|$; the *maximum deficiency* $\delta^*(F)=\max_{F'\subseteq F}\delta(F')$. A formula F is called *variable-matched* if $\nu(F)=|V(F)|$. Our main result implies fixed-parameter tractability of MaxSat parameterized by $\delta(F)$ for variable-matched formulas F.

There are two related results: Kullmann [47] obtained an $O(n^{O(\delta^*(F))})$-time algorithm for solving MaxSat for formulas F with n variables and Szeider [67] gave an $O(f(\delta^*(F))n^4)$-time algorithm for the problem, where f is a function depending on $\delta^*(F)$ only. Note that we cannot just drop the condition of being variable-matched from our result and expect a similar algorithm: it is not hard to see that the satisfiability problem remains NP-complete for formulas F with $\delta(F)=0$.

A formula F is *minimal unsatisfiable* if it is unsatisfiable but $F\setminus c$ is satisfiable for every clause $c\in F$. Papadimitriou and Wolfe [60] showed that recognition of minimal unsatisfiable CNF formulas is complete for the complexity class[1] D^P. Kleine Büning [10] conjectured that for a fixed integer k, it can be decided in polynomial time whether a formula F with $\delta(F)\leq k$ is minimal unsatisfiable. Independently, Kullmann [47] and Fleischner and Szeider [**?**] (see also [29]) resolved this conjecture by showing that minimal unsatisfiable formulas with n variables and $n+k$ clauses can be recognized in $n^{O(k)}$ time. Later, Szeider [67] showed that the problem is fixed-parameter tractable by obtaining an algorithm of running time $O(2^k n^4)$. Note that Szeider's results follow from his results mentioned in the previous paragraph and the well-known fact that $\delta^*(F)=\delta(F)$ holds for every minimal unsatisfiable formula F. Since every minimal unsatisfiable formula is variable-matched [2], our main result also implies fixed-parameter tractability of recognizing minimal unsatisfiable formula with n variables and $n+k$ clauses, parameterized by k.

The next lemma follows from Hall's matching theorem: add d vertices to V_2, each adjacent to every vertex in V_1.

Lemma 96. *Let $G=(V_1,V_2;E)$ be a bipartite graph, and suppose that for all subsets $X\subseteq V_1$, $|N(X)|\geq|X|-d$ for some $d\geq 0$. Then $\nu(G)\geq|V_1|-d$.*

We say that a bipartite graph $G=(A,B;E)$ is *q-expanding* if for all $A'\subseteq A$, $|N_G(A')|\geq|A'|+q$. Given a matching M, an *alternating path* is a path in which

[1] D^P is the class of problems that can be considered as the difference of two NP-problems; clearly D^P contains all NP and all co-NP problems

the edges belong alternatively to M and not to M.

For a subset X of the variables of CNF formula F, F_X denotes the subset of F consisting of all clauses c such that $V(c) \cap X \neq \emptyset$. A formula F is called *q-expanding* if $|X| + q \leq |F_X|$ for each $X \subseteq V(F)$. Note that, by Hall's matching theorem, a formula is variable-matched if and only if it is 0-expanding. Clearly, a formula F is q-expanding if and only if B_F is q-expanding.

For $x \in V(F)$, $n(x)$ and $n(\bar{x})$ denote the number of clauses containing x and the number of clauses containing $\bar{x}$, respectively.

6.1.1 Fixed-parameter tractability of $\nu(F) + k$ SAT

In this section we give preprocessing rules and their correctness.

Let F be the given CNF formula on n variables and m clauses with a maximum matching M on B_F, the variable-clause bipartite graph corresponding to F. Let α be a given integer and recall that our goal is to check whether $\text{sat}(F) \geq \alpha$. For each preprocessing rule below, we let (F', α') be the instance resulting by the application of the rule on (F, α). We say that a rule is *valid* if (F, α) is a YES instance if and only if (F', α') a YES instance.

Rule 5. *Let x be a variable such that $n(x) = 0$ (respectively $n(\bar{x}) = 0$). Set $x =$* FALSE *($x =$* TRUE*) and remove all the clauses that contain $\bar{x}$ (x). Reduce α by $n(\bar{x})$ (respectively $n(x)$).*

The proof of the following lemma is immediate.

Lemma 97. *If $n(x) = 0$ (respectively $n(\bar{x}) = 0$) then $\text{sat}(F) = \text{sat}(F') + n(\bar{x})$ (respectively $\text{sat}(F) = \text{sat}(F') + n(x)$), and so Rule 5 is valid.*

Rule 6. *Let $n(x) = n(\bar{x}) = 1$ and let c' and c'' be the two clauses containing x and $\bar{x}$, respectively. Let $c^* = (c' - x) \cup (c'' - \bar{x})$ and let F' be obtained from F be deleting c' and c'' and adding the clause c^*. Reduce α by 1.*

Lemma 98. *For F and F' in Reduction Rule 6, $\text{sat}(F) = \text{sat}(F') + 1$, and so Rule 6 is valid.*

Proof. Consider any assignment for F. If it satisfies both c' and c'', then the same assignment will satisfy c^*. So when restricted to variables of F', it will satisfy at least $sat(F) - 1$ clauses of F'. Thus $sat(F') \geq sat(F) - 1$ which is equivalent to $sat(F) \leq sat(F') + 1$. Similarly if an assignment γ to F' satisfies c^* then at least one of c', c'' is satisfied by γ. Therefore by setting x true if γ satisfies c'' and false otherwise, we can extend γ to an assignment on F that satisfies both of c', c''. On

the other hand, if c^* is not satisfied by γ then neither c' nor c'' is satisfied by γ, and any extension of γ will satisfy exactly one of c', c''. Therefore in either case $sat(F) \geq sat(F') + 1$. We conclude that $\text{sat}(F) = \text{sat}(F') + 1$, as required. □

Our next reduction rule is based on the following two lemmas. The first is proved in Fleischner *et al.* [29, Lemma 10], Kullmann [48, Lemma 7.7] and Szeider [67, Lemma 9].

Lemma 99. *Let F be a CNF formula. Given a maximum matching in B_F, in time $O(|F|)$ we can find an autarky $\pi : U \to \{\text{TRUE}, \text{FALSE}\}$ such that $F \setminus F_U$ is 1-expanding.*

Lemma 100 ([17])**.** *Let $\pi : U \to \{\text{TRUE}, \text{FALSE}\}$ be an autarky for a CNF formula F and let γ be any truth assignment on $V(F) \setminus U$. Then for the combined assignment $\tau := \pi \cup \gamma$, it holds that $\text{sat}_\tau(F) = |F_U| + \text{sat}_\gamma(F \setminus F_U)$. Clearly, τ can be constructed in polynomial time given π and γ.*

Rule 7. *Find an autarky $\pi : U \to \{\text{TRUE}, \text{FALSE}\}$ such that $F \setminus F_U$ is 1-expanding. Set $F' = F \setminus F_U$ and reduce α by $|F_U|$.*

The next lemma follows from Lemma 100.

Lemma 101. *For F and F' in Reduction Rule 7, $\text{sat}(F) = \text{sat}(F') + |F_U|$ and so Rule 7 is valid.*

After exhaustive application of Rule 7, we may assume that the resulting formula is 1-expanding. For the next reduction rule, we need the following results.

Theorem 102 (Szeider [67])**.** *Given a variable-matched formula F, with $|F| = |V(F)| + 1$, we can decide whether F is satisfiable in time $O(|V(F)|^3)$.*

Consider a bipartite graph $G = (A, B; E)$. Recall that a formula F is q-expanding if and only if B_F is q-expanding. From a bipartite graph $G = (A, B; E)$, $x \in A$ and $q \geq 1$, we obtain a bipartite graph G_{qx}, by adding new vertices $x_1, \ldots, x_q$ to A and adding edges such that new vertices have exactly the same neighborhood as x, that is, $G_{qx} = (A \cup \{x_1, \ldots, x_q\}, B; E \cup \{(x_i, y) : (x, y) \in E\})$. The following result is well known.

Lemma 103. [52, Theorem 1.3.6] *Let $G = (A, B; E)$ be a 0-expanding bipartite graph. Then G is q-expanding if and only if G_{qx} is 0-expanding for all $x \in A$.*

Lemma 104. *Let $G = (A, B; E)$ be a 1-expanding bipartite graph. In polynomial time, we can check whether G is 2-expanding, and if it is not, find a set $S \subseteq A$ such that $|N_G(S)| = |S| + 1$.*

Proof. Let $x \in A$. By Hall's Matching Theorem, G_{2x} is 0-expanding if and only if $\nu(G_{2x}) = |A| + 2$. Since we can check the last condition in polynomial time, by Lemma 103 we can decide whether G is 2-expanding in polynomial time. So, assume that G is not 2-expanding and we know this because G_{2y} is not 0-expanding for some $y \in A$. By Lemma 3(4) in [67], in polynomial time, we can find a set $T \subseteq A \cup \{y_1, y_2\}$ such that $|N_{G_{2y}}(T)| < |T|$. Since G is 1-expanding, $y_1, y_2 \in T$ and $|N_{G_{2y}}(T)| = |T| - 1$. Hence, $|S| + 1 = |N_G(S)|$, where $S = T \setminus \{y_1, y_2\}$. □

For a formula F and a set $S \subseteq V(F)$, $F[S]$ denotes the formula obtained from F_S by deleting all variables not in S.

Rule 8. *Let F be a 1-expanding formula and let $B = B_F$. Using Lemma 104, check whether F is 2-expanding. If it is then do not change F, otherwise find a set $S \subseteq V(F)$ with $|N_B(S)| = |S| + 1$. Let M be a matching that saturates S in $B[S \cup N_B(S)]$ (that exists as $B[S \cup N_B(S)]$ is 1-expanding). Use Theorem 102 to decide whether $F[S]$ is satisfiable, and proceed as follows.*

$F[S]$ **is satisfiable:** *Obtain a new formula F' by removing all clauses in $N_B(S)$ from F. Reduce α by $|N_B(S)|$.*

$F[S]$ **is not satisfiable:** *Let c' be the clause obtained by deleting all variables in S from $\cup_{c'' \in N_B(S)} c''$. That is, a literal l belongs to c' if and only if it belongs to some clause in $N_B(S)$ and the variable corresponding to l is not in S. Obtain a new formula F' by removing all clauses in $N_B(S)$ from F and adding c'. Reduce α by $|S|$.*

Lemma 105. *For F, F' and S introduced in Rule 8, if $F[S]$ is satisfiable* $\mathrm{sat}(F) = \mathrm{sat}(F') + |N_B(S)|$, *otherwise* $\mathrm{sat}(F) = \mathrm{sat}(F') + |S|$ *and thus Rule 8 is valid.*

Proof. We consider two cases.

Case 1: $F[S]$ **is satisfiable.** Observe that there is an autarky on S and thus by Lemma 100, $\mathrm{sat}(F) = \mathrm{sat}(F') + |N_B(S)|$.

Case 2: $F[S]$ **is not satisfiable.** Let $F'' = F' \setminus c'$. As any optimal truth assignment to F will satisfy at least $\mathrm{sat}(F) - |N_B(S)|$ clauses of F'', it follows that $\mathrm{sat}(F) \leq \mathrm{sat}(F'') + |N_B(S)| \leq \mathrm{sat}(F') + |N_B(S)|$.

Let y denote the clause in $N_B(S)$ that is not matched to a variable in S by M. Let S' be the set of variables, and Z the set of clauses, that can be reached from y with an M-alternating path in $B[S \cup N_B(S)]$. We argue now that $Z = N_B(S)$. Since Z is made up of clauses that are reachable in $B[S \cup N_B(S)]$ by an M-alternating path from the single unmatched clause y, $|Z| = |S'| + 1$. It follows that $|N_B(S) \setminus Z| = |S \setminus S'|$,

and M matches every clause in $N_B(S)\backslash Z$ with a variable in $S\backslash S'$. Furthermore, $N_B(S\backslash S') \cap Z = \emptyset$ as otherwise the matching partners of some elements of $S\backslash S'$ would have been reachable by an M-alternating path from y, contradicting the definition of $N_B(S)$ and S'. Thus $S \setminus S'$ has an autarky such that $F \setminus F_{S\backslash S'}$ is 1-expanding which would have been detected by Rule 7, hence $S \setminus S' = \emptyset$ and so $S = S'$. That is, all clauses in $N_B(S)$ are reachable from the unmatched clause y by an M-alternating path. We have now shown that $Z = N_B(S)$, as desired.

Suppose that there exists an assignment γ to F', that satisfies $\text{sat}(F')$ clauses of F' that also satisfies c'. Then there exists a clause $c'' \in N_B(S)$ that is satisfied by γ. As c'' is reachable from y by an M-alternating path, we can modify M to include y and exclude c'', by taking the symmetric difference of the matching and the M-alternating path from y to c''. This will give a matching saturating S and $N_B(S)\backslash c''$, and we use this matching to extend the assignment γ to one which satisfies all of $N_B(S)\backslash c''$. We therefore have satisfied all the clauses of $N_B(S)$. Therefore since c' is satisfied in F' but does not appear in F, we have satisfied extra $|N_B(S)| - 1 = |S|$ clauses. Suppose on the other hand that every assignment γ for F' that satisfies $\text{sat}(F')$ clauses does not satisfy c'. We can use the matching on $B[S \cup N_B(S)]$ to satisfy $|N_B(S)| - 1$ clauses in $N_B(S)$, which would give us an additional $|S|$ clauses in $N_B(S)$. Thus $\text{sat}(F) \geq \text{sat}(F') + |S|$.

As $|N_B(S)| = |S|+1$, it suffices to show that $\text{sat}(F) < \text{sat}(F')+|N_B(S)|$. Suppose that there exists an assignment γ to F that satisfies $\text{sat}(F') + |N_B(S)|$ clauses, then it must satisfy all the clauses of $N_B(S)$ and $\text{sat}(F')$ clauses of F''. As $F[S]$ is not satisfiable, variables in S alone can not satisfy all of $N_B(S)$. Hence there exists a clause $c'' \in N_B(S)$ such that there is a variable $v \in V(c'') \setminus S$ that satisfies c''. But then $v \in V(c')$ and hence c' would be satisfiable by γ, a contradiction as γ satisfies $\text{sat}(F')$ clauses of F''. □

Our algorithm first applies Reduction Rules 5, 6, 7 and 8 exhaustively on (F, α). Then it applies two branching rules we describe below, in the following order.

Branching on a variable x means that the algorithm constructs two instances of the problem, one by substituting $x = \text{TRUE}$ and simplifying the instance and the other by substituting $x = \text{FALSE}$ and simplifying the instance. Branching on x or y being false means that the algorithm constructs two instances of the problem, one by substituting $x = \text{FALSE}$ and simplifying the instance and the other by substituting $y = \text{FALSE}$ and simplifying the instance. Simplifying an instance is done as follows. For any clause c, if c contains a literal z with $z = \text{TRUE}$, remove c and reduce α by 1. If c contains a literal z with $z = \text{FALSE}$ and c contains other literals, remove z

from c. If c consists of the single literal $z = \text{FALSE}$, remove c.

A branching rule is correct if the instance on which it is applied is a YES-instance if and only if the simplified instance of (at least) one of the branches is a YES-instance.

Branching Rule 1. *If $n(x) \geq 2$ and $n(\bar{x}) \geq 2$ then we branch on x.*

Before attempting to apply Branching Rule 2, we apply the following rearranging step: For all variables x such that $n(\bar{x}) = 1$, swap literals x and $\bar{x}$ in all clauses. Clearly, this will not change $\text{sat}(F)$. Observe that now for every variable $n(x) = 1$ and $n(\bar{x}) \geq 2$.

Branching Rule 2. *If there is a clause c such that positive literals $x, y \in c$ then we branch on x being false or y being false.*

Branching Rule 1 is exhaustive and thus its correctness also follows. When we reach Branching Rule 2 for every variable $n(x) = 1$ and $n(\bar{x}) \geq 2$. As $n(x) = 1$ and $n(y) = 1$ we note that c is the only clause containing these literals. Therefore there exists an optimal solution with x or y being false (if they are both true just change one of them to false). Thus, we have the following:

Lemma 106. *Branching Rules 1 and 2 are correct.*

Let (F, α) be the given instance on which Reduction Rules 5, 6, 7 and 8, and Branching Rules 1 and 2 do not apply. Observe that for such an instance F the following holds:

1. For every variable x, $n(x) = 1$ and $n(\bar{x}) \geq 2$.
2. Every clause contains at most one positive literal.

We call a formula F satisfying the above properties *special*. In what follows we describe an algorithm for our problem on special instances. Let $c(x)$ denote the *unique* clause containing positive literal x. We can obtain a matching saturating $V(F)$ in B_F by taking the edge connecting the variable x and the clause $c(x)$. We denote the resulting matching by M_u.

We first describe a transformation that will be helpful in reducing our problem to $(m-k)$-HITTING SET. Given a formula F we obtain a new formula F' by changing the clauses of F as follows. If there exists some $c(x)$ such that $|c(x)| \geq 2$, do the following. Let $c' = c(x) - x$ (that is, c' contain the same literals as $c(x)$ except for x) and add c' to all clauses containing the literal $\bar{x}$. Furthermore remove c' from $c(x)$ (which results in $c(x) = (x)$ and therefore $|c(x)| = 1$).

Next we prove the validity of the above transformation.

Lemma 107. *Let F' be the formula obtained by applying the transformation described above on F. Then* $\mathrm{sat}(F') = \mathrm{sat}(F)$ *and* $\nu(B_F) = \nu(B_{F'})$.

Proof. We note that the matching M_u remains a matching in $B_{F'}$ and thus $\nu(B_F) = \nu(B_{F'})$. Let γ be any truth assignment to the variables in F (and F') and note that if c' is false under γ then F and F' satisfy exactly the same clauses under γ (as we add and subtract something false to the clauses). So assume that c' is true under γ.

If γ maximizes the number of satisfied clauses in F then clearly we may assume that x is false (as $c(x)$ is true due to c'). Now let γ' be equal to γ except the value of x has been flipped to true. Note that exactly the same clauses are satisfied in F and F' by γ and γ', respectively. Analogously, if an assignment maximizes the number of satisfied clauses in F' we may assume that x is true and by changing it to false we satisfy equally many clauses in F. Hence, $\mathrm{sat}(F') = \mathrm{sat}(F)$. □

Given a special instance (F, α) we apply the above transformation repeatedly until no longer possible and obtain an instance (F', α) such that $\mathrm{sat}(F') = \mathrm{sat}(F)$, $\nu(B_F) = \nu(B_{F'})$ and $|c(x)| = 1$ for all $x \in V(F')$. We call such an instance (F', α) *transformed special*. Observe that, it takes polynomial time, to obtain the transformed special instance from a given special instance.

For simplicity of presentation we denote the transformed special instance by (F, α). Let C^* denote all clauses that are not matched by M_u (and therefore only contain negated literals). We associate a hypergraph H^* with the transformed special instance. Let H^* be the hypergraph with vertex set $V(F)$ and edge set $E^* = \{V(c) \mid c \in C^*\}$.

We now show the following equivalence between $(\nu(F)+k)$-SAT on transformed special instances and $(m-k)$-Hitting Set.

Lemma 108. *Let (F, α) be the transformed special instance and H^* be the hypergraph associated with it. Then* $\mathrm{sat}(F) \geq \alpha$ *if and only if there is a hitting set in H^* of size at most $|E(H^*)| - k$, where $k = \alpha - \nu(F)$.*

Proof. We start with a simple observation about an assignment satisfying the maximum number of clauses of F. There exists an optimal truth assignment to F, such that all clauses in C^* are true. Assume that this is not the case and let γ be an optimal truth assignment satisfying as many clauses from C^* as possible and assume that $c \in C^*$ is not satisfied. Let $\bar{x} \in c$ be an arbitrary literal and note that $\gamma(x) = \textsc{true}$. However, changing x to false does not decrease the number of satisfied clauses in F and increases the number of satisfied clauses in C^*.

Now we show that $\text{sat}(F) \geq \alpha$ if and only if there is a hitting set in H^* of size at most $|E(H^*)| - k$. Assume that γ is an optimal truth assignment to F, such that all clauses in C^* are true. Let $U \subseteq V(F)$ be all variables that are false in γ and note that U is a hitting set in H^*. Analogously if U' is a hitting set in H^* then by letting all variables in U' be false and all other variables in $V(F)$ be true we get a truth assignment that satisfies $|F| - |U'|$ clauses in F. Therefore if $\tau(H^*)$ is the size of a minimum hitting set in H^* we have $\text{sat}(F) = |F| - \tau(H^*)$. Hence, $\text{sat}(F) = |F| - \tau(H^*) = |V(F)| + |C^*| - \tau(H^*)$ and thus $\text{sat}(F) \geq \alpha$ if and only if $|C^*| - \tau(H^*) \geq k$, which is equivalent to $\tau(H^*) \leq |E(H^*)| - k$. □

Lemma 108 and Theorem 85 together give us the following result:

Lemma 109. *There exists an algorithm solving a transformed special instance of* $(\nu(F) + k)$-SAT *in time* $O((2e)^{2k+O(\log^2 k)}(m+n)^{O(1)})$.

Lemma 108 and Theorem 90 together give us the following result:

Lemma 110. *There exists a randomized algorithm solving a transformed special instance of* $(\nu(F) + k)$-SAT *in expected time* $O(8^{k+O(\sqrt{k})}(m+n)^{O(1)})$.

We are now ready to describe the complete algorithm for an instance (F, α) of $(\nu(F) + k)$-SAT:

Find a maximum matching M on B_F and let $k = \alpha - |M|$. If $k \leq 0$, return Yes. Otherwise, apply Reduction Rules 5 to 8, whichever is applicable, in that order and then run the algorithm on the reduced instance and return the answer. If none of the Reduction Rules apply, then apply Branching Rule 1 if possible, to get two instances (F', α') and (F'', α''). Run the algorithm on both instances; if one of them returns Yes, return Yes, otherwise return No. If Branching Rule 1 does not apply then we rearrange the formula and attempt to apply Branching Rule 2 in the same way. Finally if $k > 0$ and none of the reduction or branching rules apply, then we have for all variables x, $n(x) = 1$ and every clause contains at most one positive literal, i.e. (F, α) is a special instance. Then solve the problem by first obtaining the transformed special instance as described above, then solving it in time $O((2e)^{2k+O(\log^2 k)}(m+n)^{O(1)})$ as guaranteed by Lemma 109

Correctness of all the preprocessing rules and the branching rules follows from Lemmas 97, 98, 101, 105 and 106.

Analysis of the algorithm. Let (F, α) be the input instance. Let $\mu(F) = \mu = \alpha - \nu(F)$ be the measure. We will first show that our preprocessing rules do not

increase this measure. Following this, we will prove a lower bound on the decrease in the measure occurring as a result of the branching, thus allowing us to bound the running time of the algorithm in terms of the measure μ. For each case, we let (F', α') be the instance resulting by the application of the rule or branch. Also let M' be a maximum matching of $B_{F'}$.

Reduction Rule 5: We consider the case when $n(x) = 0$; the other case when $n(\bar{x}) = 0$ is analogous. We know that $\alpha' = \alpha - n(\bar{x})$ and $\nu(F') \geq \nu(F) - n(\bar{x})$ as removing $n(\bar{x})$ clauses can only decrease the matching size by $n(\bar{x})$. This implies that $\mu(F) - \mu(F') = \alpha - \nu(F) - \alpha' + \nu(F') = (\alpha - \alpha') + (\nu(F') - \nu(F)) \geq n(\bar{x}) - n(\bar{x})$. Thus, $\mu(F') \leq \mu(F)$.

Reduction Rule 6: We know that $\alpha' = \alpha - 1$. We show that $\nu(F') \geq \nu(F) - 1$. In this case we remove the clauses c' and c'' and add $c^* = (c' - x) \cup (c'' - \bar{x})$. We can obtain a matching of size $\nu(F) - 1$ in $B_{F'}$ as follows. If at most one of the c' and c'' is the end-point of some matching edge in M then removing that edge gives a matching of size $\nu(F) - 1$ for $B_{F'}$. So let us assume that some edges (a, c') and (b, c'') are in M. Clearly, either $a \neq x$ or $b \neq x$. Assume $a \neq x$. Then $M \setminus \{(a, c'), (b, c'')\} \cup \{(a, c^*)\}$ is a matching of size $\nu(F) - 1$ in $B_{F'}$. Thus, we conclude that $\mu(F') \leq \mu(F)$.

Reduction Rule 7: The proof is the same as in the case of Reduction Rule 5.

Reduction Rule 8: The proof that $\mu(F') \leq \mu(F)$ in the case when $F[S]$ is satisfiable is the same as in the case of Reduction Rule 5 and in the case when $F[S]$ is not satisfiable is the same as in the case of Reduction Rule 6.

Branching Rule 1: Consider the case when we set $x = \text{TRUE}$. In this case, $\alpha' = \alpha - n(x)$. Also, since no reduction rules are applicable we have that F is 2-expanding. Hence, $\nu(F) = |V(F)|$. We will show that in (F', α') the matching size will remain at least $\nu(F) - n(x) + 1$ ($= |V(F)| - n(x) + 1 = |V(F')| - n(x) + 2$.) This will imply that $\mu(F') \leq \mu(F) - 1$. By Lemma 96 and the fact that $n(x) - 2 \geq 0$, it suffices to show that in $B' = B_{F'}$, every subset $S \subseteq V(F')$, $|N_{B'}(S)| \geq |S| - (n(x) - 2)$. The only clauses that have been removed by the simplification process after setting $x = \text{TRUE}$ are those where x appears positively and the singleton clauses $(\bar{x})$. Hence, the only edges of $G[S \cup N_B[S]]$ that are missing in $N_{B'}(S)$ from $N_B(S)$ are those corresponding to clauses that contain x as a pure literal and some variable in S. Thus, $|N_{B'}(S)| \geq |S| + 2 - n(x) = |S| - (n(x) - 2)$ (as F is 2-expanding).

The case when we set $x = \text{FALSE}$ is similar to the case when we set $x = \text{TRUE}$. Here, also we can show that $\mu(F') \leq \mu(F) - 1$. Thus, we get two instances, with each instance (F', α') having $\mu(F') \leq \mu(F) - 1$.

Branching Rule 2: The analysis here is the same as for Branching Rule 1 and again we get two instances with $\mu(F') \leq \mu(F) - 1$.

We therefore have a depth-bounded search tree of size of depth at most $\mu = \alpha - \nu(F) = k$, in which any branching splits an instance into two instances. Thus, the search tree has at most 2^k instances. As each reduction and branching rule takes polynomial time, every rule decreases the number of variables, the number of clauses, or the value of μ, and an instance to which none of the rules apply can be solved in time $O((2e)^{2\mu}\mu^{O(\log \mu)}(m+n)^{O(1)})$ (by Lemma 109), we have by induction that any instance can be solved in time

$$O(2 \cdot (2e)^{2(\mu-1)}(\mu-1)^{O(\log(\mu-1))}(m+n)^{O(1)}) = O((2e)^{2\mu}\mu^{O(\log \mu)}(m+n)^{O(1)}).$$

Thus the total running time of the algorithm is at most $O((2e)^{2k+O(\log^2 k)}(n+m)^{O(1)})$. Applying Lemma 110 instead of Lemma 109, we conclude that $(\nu(F)+k)$-SAT can be solved in expected time $O(8^{k+O(\sqrt{k})}(n+m)^{O(1)})$. Summarizing, we have the following:

Theorem 111. *There are algorithms solving* $(\nu(F)+k)$-SAT *in time* $O((2e)^{2k+O(\log^2 k)}(n+m)^{O(1)})$ *or expected time* $O(8^{k+O(\sqrt{k})}(n+m)^{O(1)})$.

6.1.2 No polynomial kernel for $\nu(F)+k$ SAT

In this section, we show that $(\nu(F)+k)$-SAT does not have a polynomial-size kernel, unless coNP $\subseteq$ NP/poly.

The proof of the next theorem is similar to the proof of Lemma 108.

Theorem 112. $(\nu(F)+k)$-SAT *has no polynomial-size kernel, unless coNP* $\subseteq$ *NP/poly.*

Proof. By Theorem 92, there is no polynomial-size kernel for the problem of deciding whether a hypergraph H has a hitting set of size $|E(H)|-k$, where k is the parameter unless coNP $\subseteq$ NP/poly. We prove the theorem by a polynomial parameter reduction from this problem. Then the theorem follows from Lemma 5, as $(\nu(F)+k)$-SAT is NP-complete.

Given a hypergraph H on n vertices, construct a CNF formula F as follows. Let the variables of F be the vertices of H. For each variable x, let the unit clause (x) be a clause in F. For every edge e in H, let c_e be the clause containing the literal $\bar{x}$ for every $x \in E$. Observe that F is matched, and that H has a hitting set of size $|E(H)| - k$ if and only if $\text{sat}(F) \geq n + k$.

□

6.2 Directed Nonblocker

In the problem NONBLOCKER, we are given a graph $G = (V, E)$ and an integer k, and asked whether there is a set $X \subseteq V$ of size at most $|V| - k$ such that each vertex $v \in V \setminus X$ is adjacent to a vertex in X. Here k is the parameter. Note that NONBLOCKER is a below-tight-upper-bound parameterization of the DOMINATING SET problem. It is well-known that NONBLOCKER can be reduced to $(n-k)$HITSET (see, e.g., [30], p. 18) and, thus, our $W[1]$-hardness result for that problem is in a sharp contrast to a linear-order-kernel result of Dehne et al. [22] for NONBLOCKER.

In a directed graph $G = (V, A)$, a *dominating set* is a set $V' \subseteq V$ such that for every vertex $u \in V \setminus V'$, there is a vertex $v \in V'$ such that there is an arc from v to u. Recall that in DIRECTED NONBLOCKER, we are given a directed graph G with n vertices and an integer k, and asked whether G has a dominating set with at most $n - k$ vertices.

Corollary 113. DIRECTED NONBLOCKER *has a kernel with at most* $k^2 + k - 1$ *vertices.*

Proof. Let $(G = (V, A), k)$ be an instance of DIRECTED NONBLOCKER with $|V| = n$. If G has a vertex v of out-degree at least k, then $V \setminus \{w \in V : \ vw \in A\}$ is a dominating set of size at most $n - k$. Thus, we may assume that the maximum out-degree of G is at most $k - 1$.

We construct an instance of $(n - k)$HITSET as follows. Let $H = (V, \mathcal{F})$, where $\mathcal{F} = \{N^-[v] : v \in V\}$, $N^-[v] = \{v\} \cup \{u \in V : \ uv \in A\}$. Observe that $N^-[v]$ is hit by a set $S \subseteq V$ if and only if $v \in S$ or v is dominated by a vertex in S. Therefore, H has a hitting set of size $|\mathcal{F}| - k = |V| - k$ if and only if G has a dominating set of size $|V| - k$.

Since the maximum out-degree of G is at most $k - 1$, the maximum degree of a vertex in H is at most k. Thus, the degeneracy d of H is at most k and the result follows from Theorem 95. □

6.2.1 Linear kernel for Directed Nonblocker

In this section, we improve the bound of Corollary 113 for $k > 2$.

It is well-known that every hypergraph $H = (V, \mathcal{F})$ in which each edge has at least two vertices, has a hitting set of cardinality at most $(|V| + |\mathcal{F}|)/3$, cf. [68]. We start from a minor extension of this result.

Lemma 114. *Let* $H = (V, \mathcal{F})$ *be a hypergraph such that every edge has at least two vertices apart from, possibly, one edge that has just one vertex. If* H *has a one-vertex*

edge $e = \{v\}$, let there be another edge f of H containing v. Then H has a hitting set of cardinality at most $(|V| + |\mathcal{F}|)/3$.

Proof. Let $n = |V|$ and $m = |\mathcal{F}|$. The proof is by induction on $n \geq 2$. If $n = 2$, then H has a hitting set of cardinality 1 and $n+m \geq 3$. Now assume that $n \geq 3$ and let $t(H)$ be the minimum cardinality of a hitting set in H. If H has a one-vertex edge $e = \{v\}$, then set $u = v$. Otherwise, let u be a vertex of H of maximum degree. Remove u from H together with all edges containing u and all vertices contained only in the removed edges. Denote the resulting hypergraph by H' and let n' and m' be the number of vertices and edges, respectively, in H'. Then $3t(H) \leq 3+3t(H') \leq 3+n'+m' \leq n+m$. The second inequality in this chain of inequalities is by the induction hypothesis and the third inequality is due to the fact that either we remove at least two edges and one vertex or at least two vertices and one edge. □

For a digraph D, let $\gamma(D)$ denote the minimum size of a dominating set in D. Using Lemma 114, it is easy to prove the following key lemma of this section.

Lemma 115. *Let D a digraph on n vertices, none of which are isolated, and let D have at most one vertex of in-degree zero. Then $\gamma(D) \leq 2n/3$.*

Proof. We construct an instance $H = (V, \mathcal{F})$ of $(n-k)$HitSet as in the proof of Corollary 113. The lemma follows from that facts that H satisfies the conditions of Lemma 114, $|V| = |\mathcal{F}|$, and the minimum cardinalities of a hitting set in H and a dominating set in D coincide. □

Theorem 116. Directed Nonblocker *has a kernel with at most $3k-1$ vertices.*

Proof. Let D be a digraph with n vertices. If D has isolated vertices, then delete them without changing the answer to Directed Nonblocker as all of them must be in any dominating set of D. Thus, we may assume that D has no isolated vertices. Let S be the set of all vertices of D of in-degree zero. Assume that $|S| > 1$. Then contract all vertices of S into one vertex s which dominates all vertices dominated by S. Let D' be the resulting digraph. Since all vertices of S must be in any dominating set of D, the answers to Directed Nonblocker on D and on D' are the same. Thus, we may assume that $|S| \leq 1$. Then, by Lemma 115, $\gamma(D) \leq 2n/3$ and, thus, if $n - k \geq 2n/3$, the answer to Directed Nonblocker is Yes. Otherwise, $n - k < 2n/3$ and $n \leq 3k - 1$. □

To obtain a smaller kernel for Directed Nonblocker, it might be helpful to use further results on hitting sets of hypergraphs with a lower bound on the minimum

size of an edge. Chvátal and McDiarmid [13] and Tuza [69] proved independently that a hypergraph $H = (V, \mathcal{F})$ with minimum edge size equal three, has a hitting set of size at most $(|V| + |\mathcal{F}|)/4$. Thomassé and Yeo [68] showed that if the minimum edge in a hypergraph $H = (V, \mathcal{F})$ is four and the minimum size of a hitting set of H is t, then $21t \leq 5|V| + 4|\mathcal{F}|$.

Chapter 7

Test Cover

Let $H = (V, \mathcal{E})$ be a hypergraph with n vertices and m edges. Given a set of vertices X, we say an edge e *cuts* X if $X \cap e \neq \emptyset$ and $X \setminus e \neq \emptyset$. We say an edge $e \in \mathcal{E}$ *separates* vertices x and y if $|e \cap \{x, y\}| = 1$, i.e. e cuts $\{x, y\}$. We say a set $T \subseteq \mathcal{E}$ is a *test cover* of H if every pair of vertices in V is separated by an edge in T. Thus, if T is a test cover of H then each vertex H is uniquely identified by the list of edges in T that contain it.

Observe that if H has a test cover then $\mathcal{E}$ itself is a test cover of H. If the hypergraph H has no test cover, then we can transform it into one that does by identifying any vertices that are in exactly the same edges. In what follows we will assume that $\mathcal{E}$ is a test cover for H.

In this chapter and the next, we define $\beta(H)$ as follows.

Definition 117. *Let H be a hypergraph. Then*

$$\beta(H) = \min\{|T| : T \text{ is a test cover of } H\}$$

The Test Cover problem is as follows:

Test Cover
Instance: A hypergraph $H = (V, \mathcal{E})$ such that $\mathcal{E}$ is test cover of H, an integer p.
Question: Is $\beta(G) \leq p$?

Test Cover arises naturally in the following general setting of identification problems: Given a set of items and a set of binary attributes that may or may not occur in each item, the aim is to find the minimum size subset of attributes (corresponding to a minimum test cover) such that each item can be uniquely identified

from the information on which of this subset of attributes it contains. TEST COVER arises in fault analysis, medical diagnostics, pattern recognition, and biological identification (see, e.g., [39, 40, 56]).

The TEST COVER problem has been also studied extensively from an algorithmic view point. The problem is NP-hard, as was shown by Garey and Johnson (under the name MINIMUM TEST SET) [32]. Moreover, TEST COVER is APX-hard [39]. There is an $O(\log n)$-approximation algorithm for the problem [56] and there is no $o(\log n)$-approximation algorithm unless P=NP [39]. These approximation results are obtained using reductions from TEST COVER to the well-studied SET COVER problem, where given a collection $\mathcal{S}$ of subsets of $[n]$ *covering* $[n]$ (i.e., $\cup_{X\in\mathcal{S}} X = [n]$) and integer t, we are to decide whether there is a subcollection of $\mathcal{S}$ of size t covering $[n]$.

As for HITSET, given a function $\Phi : \mathcal{H} \times \mathbb{N} \to \mathbb{N}$, define the parameterized problem $\Phi(H,k)$-TEST COVER as follows:

$\Phi(H,k)$-TEST COVER
Instance: A hypergraph H with n vertices, m edges, an integer k.
Parameter: k.
Question: Is $\beta(G) \leq \Phi(H,k)$?

TEST COVER arises naturally in the following general setting of identification problems: Given a set of items and a set of binary attributes that may or may not occur in each item, the aim is to find the minimum size subset of attributes (corresponding to a minimum test cover) such that each item can be uniquely identified from the information on which of this subset of attributes it contains. TEST COVER arises in fault analysis, medical diagnostics, pattern recognition, and biological identification (see, e.g., [39, 40, 56]).

The TEST COVER problem has been also studied extensively from an algorithmic view point. The problem is NP-hard, as was shown by Garey and Johnson [32]. Moreover, TEST COVER is APX-hard [39]. There is an $O(\log n)$-approximation algorithm for the problem [56] and there is no $o(\log n)$-approximation algorithm unless P=NP [39]. These approximation results are obtained using reductions from TEST COVER to the well-studied SET COVER problem, where given a hypergraph $H = (V, \mathcal{E})$ such that $\mathcal{E}$ *covers* V (i,e. $\bigcup \mathcal{E} = V$) and integer t, we are to decide whether there is a subset of $\mathcal{E}$ of size t covering V.

In this chapter we consider the problems $(n-k)$TEST COVER and $(m-k)$TEST COVER.

7.1 k-MINI TEST COVER

To solve $(n-k)$Test Cover, we introduce the k-Mini Test Cover problem. Similar to k-miniHitSet, the idea is to find a subset of $\mathcal{E}$ that does the job of a test cover on a smaller scale. To facilitate this idea, we introduce the idea of *induced classes*.

Definition 118. *Given a hypergraph $H = (V, \mathcal{E})$ and a set $T \subseteq \mathcal{E}$, let $\equiv_T$ be a relation on V such that $x \equiv_T y$ if an only if x and y are not separated by any edge in T. A* class induced by T *is a non-empty set of vertices C such that C is an equivalence class under the relation $\equiv_T$.*

Thus, a set of vertices is a class induced by T if it is a maximal set of vertices not cut by an edge in T. Observe that T is a test cover if and only if T induces exactly n classes in H.

Definition 119. *Let $H = (V, \mathcal{E})$ be a hypergraph and k an integer. A set $T \subset \mathcal{E}$ is a* k-mini test cover *for H if $|T| \leq 2k$ and T induces at least $|T| + k$ classes in H.*

In the remainder of this section we will show that a hypergraph has a test cover of size $(n-k)$ if and only if it has a k-mini test cover. Note that unlike the definition of a k-mini hitting set, which was required to have size at most k, here we only require that $|T| \leq 2k$. It turns out that $2k$ is the smallest size we can demand to ensure the claim is true.

Lemma 120. *Suppose that $\mathcal{E}$ is a test cover for H, $T \subseteq \mathcal{E}$ and T induces at least $|T| + k$ classes. Then T can be extended to a test cover of size $n - k$. Moreover, if $\mathcal{E}$ contains all singletons, this is possible by adding only singletons.*

Proof. Add edges from $\mathcal{E}$ to T one by one such that each edge increases the number of classes induced by T, until the number of classes is n. This can be done, since if we have less than n classes, there is a class C containing at least two vertices. For $x, y \in C$ there exists an edge $e \in \mathcal{E} \setminus T$ that separates x, y which may be added to T. If we are only permitted to add singletons, then pick $e = \{x\}$. Let T' be the subset produced from T in this way. Observe that T' is a test cover. Since T induces at least $|T| + k$ classes, we need to add at most $n - (|T| + k)$ tests to produce T'. Thus $|T'| \leq n - k$, as required. □

We now define the notion of a *C-test* as follows.

Definition 121. *Let $C \subseteq [n]$. An edge $e \in \mathcal{E}$ is a C-test if $C \setminus e \neq \emptyset$ and $e \cap C \neq \emptyset$ (i.e. e cuts C).*

Theorem 122. *Suppose that $\mathcal{E}$ is a test cover for $H = (V, \mathcal{E})$. If T is a k-mini test cover for H, then there exists a test cover T' for H with $|T'| \leq n - k$ such that $T \subseteq T'$. Conversely, if T' induces at least $|T'| + k$ classes (in particular if T' is a test cover for H with $|T'| \leq n - k$), then there exists a k-mini test cover T for H with $T \subseteq T'$.*

Proof. First suppose that H has a k-mini test cover T. Then by Lemma 120, T can be extended to a test cover of size at most $n - k$.

Conversely, suppose a set T' induces at least $|T'| + k$ classes. We will show that there exists a k-mini test cover which is a subset of T'. To obtain this subcollection we greedily construct a $T \subseteq T'$ as follows.

> Start with $T = \emptyset$. Add two edges e_i, e_j from T' to T if this will increase the number of classes induced by T by at least 3. Add an edge e_i from T' to T if this will increase the number of classes induced by T by at least 2. Stop the construction if we reach $|T| = 2k - 1$ or $|T| = 2k$.

We now show that T induces at least $|T| + k$ classes.. Observe that T induces at least $\lceil \frac{3}{2}|T| \rceil$ classes and $|T| \leq 2k$, so T is a k-mini test cover unless $|T| < 2k - 1$ and T induces less than $|T| + k$ classes. So consider this case. The construction must have stopped because no edge or pair of edges from T' would increase the number of classes by the required amount. Hence, the following two conditions hold:

1. For every edge $e_i \in T' \backslash T$, e_i does not cut more than one class induced by T.
2. For every class C induced by T, and for every pair e_i, e_j of C-tests in $T' \backslash T$, at least one of $(e_i \cap e_j) \cap C$, $(e_i \backslash e_j) \cap C$, $(e_j \backslash e_i) \cap C$ and $C \backslash (e_i \cup e_j)$ is empty.

It can be seen that these properties hold even if we add one extra edge from $T' \backslash T$ to T. Therefore if we add t edges from $T' \backslash T$, one at a time, this will subdivide a class C into at most $t + 1$ classes. Furthermore, since each edge cuts at most one class, adding t edges from $T' \backslash T$ to T will increase the number of classes induced by T by at most t. It follows that T' induces less than $T| + k + |T' \backslash T| = |T'| + k$ classes. But this is a contradiction. □

Define the problem k-Mini Test Cover as follows:

k-MINI TEST COVER
Instance: A hypergraph $H = (V, \mathcal{E})$ such that $\mathcal{E}$ is test cover of H, an integer k.
Parameter: k.
Question: Does H have a k-mini test cover?

By Lemma 122 we get the following result, which allows us to concentrate on k-MINI TEST COVER in the next subsection.

Corollary 123. *The problem* $(n-k)$TEST COVER *is FPT if and only if* k-MINI TEST COVER *is FPT.*

7.2 Base classes

Before solving the k-MINI TEST COVER problem, we will first attempt a greedy algorithm to find a k-mini test cover for H. If this does not work, the greedy algorithm will tell us something about the structure of the H. We will be able to split the vertices of H into a set of classes such that each edge cuts at most one of the classes, and within each class the graph has a very simple structure. Describing this structure is the purpose of this subsection. The results of this subsection will also be used in the next chapter, when we consider $(n-k)$TEST COVER with bounded edge sizes.

We start with the following easy observation.

Lemma 124. *Assume $\mathcal{E}$ is a test cover for $H = (V, \mathcal{E})$. Let H^* be the hypergraph formed by adding every singleton not already in $\mathcal{E}$ to $\mathcal{E}$. Then H^* has a k-mini test cover if and only if H also has a k-mini test cover.*

Proof. Assume H^* has a k-mini test cover T. Form T' from T by removing all singletons. For each singleton removed the number of induced classes decreases by at most one (since after removing the singleton $\{x\}$, there is exactly one class containing x, and this is the only class that can be cut by $\{x\}$). Hence, as T induces at least $|T| + k$ classes, T' induces at least $|T'| + k$ classes, and $|T'| \leq |T| \leq 2k$. Thus, T' is a k-mini test cover for H. The other direction is immediate since $\mathcal{E}(H) \subseteq \mathcal{E}(H^*)$. □

Due to Lemma 124, hereafter we assume that every singleton belongs to $\mathcal{E}$.

Lemma 125. *Given a hypergraph $H = (V, \mathcal{E})$ In polynomial time, we may either find a k-mini test cover for H, or find a partition of V into classes $C_1, \ldots, C_l$ such that:*

1. $l < 3k$

2. *Each edge in* $\mathcal{E}$ *cuts at most one class* C_i*.*

3. *For any* $e, e' \in \mathcal{E}$ *and any class* C_i*, at least one of* $(e \cap e') \cap C_i$*,* $(e \backslash e') \cap C_i$*,* $(e \backslash e') \cap C_i$ *and* $C_i \backslash (e \cup e')$ *is empty.*

We shall call $C_1, \ldots, C_l$ *the* base classes *of* H*.*

Proof. We start by constructing a set T in a similar way to the proof of Theorem 122. Start with $T = \emptyset$. Add two edges e_i, e_j from $\mathcal{E}$ to T if this will increase the number of classes induced by T by at least 3. Add an edge e_i from $\mathcal{E}$ to T if this will increase the number of classes induced by T by at least 2.

Observe that if we ever reach a point where $|T| \geq 2k$ then $|T|$ induces at least $|T| + k$ classes. In this case, by Lemma 120 we can find a test cover for H of size $n - k$. (Which is really enough for us to be done, but to strictly satisfy the claim, we can then apply Lemma 122 to find a k-mini test cover. In fact, it can be observed that at some point in the construction of T, T itself will be a k-mini test cover.)

So now assume that $|T| < 2k$, and furthermore that T induces less than $|T| + k$ classes. Then the construction stopped because no edge in $\mathcal{E}$ increases the number of classes induced by T by more than 1, and no pair of edges increases the number of classes induced by T by more than 2. Let $C_1, \ldots C_l$ be the classes induced by $\mathcal{T}$. Then observe that $l \leq 3k$, as required. We now show that the other properties of the claim are satisfied.

Suppose an edge e exists which cuts C_i and C_j, for $i \neq j$. Then adding e to to T would increase the number of classes induced by T by at least 2, a contradiction. Thus each edge in $\mathcal{E}$ cuts at most one class C_i.

Finally, consider two edges $e, e' \in \mathcal{E}$, and assume each of $(e \cap e') \cap C_i$, $(e \backslash e') \cap C_i$, $(e \backslash e') \cap C_i$ and $C_i \backslash (e \cup e')$ is non-empty. Then adding e, e' to T would increase the number of classes induced by T by at least 3, a contradiction. □

7.3 Fixed-parameter tractability of $(n - k)$ Test Cover

We may now assume that we are given a partition of the vertices of H into base classes $C_1, \ldots, C_l$, as described in the previous section.

Definition 126. *Let* C_i *be a base class. If an edge* e *is a* C_i*-test, we define the* local *portion of* e *as* $L(e) = C \cap e$ *and the* global *portion* $G(e) = e \setminus C_i$*. If an edge* e *is not a* C_i*-test for any base class* C_i*, then we define* $L(e) = \emptyset$ *and* $G(e) = e$*.*

Before continuing we need to give a small reduction rule.

Rule 9. *If e is a C_i-test for some base class C_i, and $|L(e)| > |C_i|/2$, then delete e from $\mathcal{E}$ and replace it with the edge $e' = V \setminus e$.*

Lemma 127. *Rule 9 is valid.*

Proof. Let $H' = (V, \mathcal{E}')$ be an instance of k-Mini Test Cover derived from $H = (V, \mathcal{E})$ by an application of Rule 9. Given a set of edges $T \subseteq \mathcal{E}$, let $T' = (T \setminus \{e\}) \cup \{e'\}$ if $e \in T$, and let $T' = T$ otherwise. Observe that two vertices in H' are separated by e' if and only if they are separated by e in H. It follows that the classes induced by $\mathcal{T}$ are exactly the same as the classes induced by $\mathcal{T}'$. Therefore H contains a k-mini test cover if and only if H' contains a k-mini test cover. □

Note that Lemma 125 still holds after applying Rule 9, since for all $j \neq i$ either $e' \cap C_j = \emptyset$ or $C_j \subseteq e'$. By Rule 9, we may now assume that for every C_i-test e, we have $|L(e)| \leq |C_i|/2$. By property 3 of Lemma 125, we now have that for any C_i-tests e, e', either $L(e)$ and $L(e')$ are disjoint or one is contained within the other.

Let S' be a set of vertices such that S' is a strict subset of some base class C_i. We define the *signature* of S' as follows.

$$Sig(S') = \{G(e) : \ e \in \mathcal{E} \text{ and } L(e) = S'\}$$

Lemma 128. *We have $|\{Sig(S') : \ S' \subset C_i\}| \leq 2^{2^{3k-1}}$.*

Proof. Let $\mathcal{S}_i$ denote all sets, S, with $C_j \cap S = \emptyset$ or $C_j \subseteq S$ for all j and furthermore $C_i \cap S = \emptyset$. Note that $|\mathcal{S}_i| \leq 2^{l-1} \leq 2^{3k-1}$, since $|\{C_1, \ldots, C_l\} \setminus \{C_i\}| \leq 3k-1$. Note that any two non-equal edges e and e' with $L(e) = S' = L(e')$ have $G(e) \neq G(e')$, as $e \neq e'$. Observe that all $G(S)$ in $Sig(S')$ belong to $\mathcal{S}_i$ implying that there is at most $2^{|\mathcal{S}_i|} = 2^{2^{3k-1}}$ different choices for a signature. □

An *out-tree* O is an orientation of a tree which has only one vertex of in-degree zero (called the *root*); a vertex of O of out-degree zero is a *leaf*.

For each $i \in [l]$, we now build an out-tree O_i as follows. The root of the tree, $r \in V(O_i)$ corresponds to the base class C_i. Each vertex $v \in V(O_i) \setminus r$ corresponds to a subset, $S_v \subseteq C_i$ such that there exists a C_i-test $e \in \mathcal{E}$ with $L(e) = S_v$. Note that for a pair of vertices $u, v \in V(O_i)$ if $u \neq v$, then $S_u \neq S_v$. Add an arc from v to w in O_i if $S_w \subset S_v$ and there is no u in O_i with $S_w \subset S_u \subset S_v$. Since the local parts of two C_i-tests are either disjoint or one is a subset of another, we note that O_i is indeed an out-tree.

Lemma 129. *Every non-leaf in O_i has out-degree at least two.*

Proof. Let v be a non-leaf in O_i, and note that $|S_v| \geq 2$. Let w be any child of v in O_i. By definition there exists a vertex in $S_v \setminus S_w$ (as $|S_v| > |S_w|$), say w'. As there is a singleton $\{w'\} \in \mathcal{E}$ there is a path from v to w' in O_i and as $w' \notin S_w$ the path does not use w. Therefore v has at least one other out-neighbour. □

Lemma 130. *There exists a function $f_1(k)$ such that either the depth of the tree O_i (i.e. the number of arcs in a longest path out of the root) is at most $f_1(k)$, or in polynomial time, we can find a vertex v in O_i such that if there is a solution to our instance of $(n-k)$-Test Cover then there is also a solution that does not use any edge e with $L(e) = S_v$.*

Proof. In what follows, let $F \subseteq \mathcal{E}$ be the set of at most $2k$ edges that induce the base classes $C_1, \ldots, C_l$. (Observe that F exists, and we can find it, by the construction of $C_1, \ldots, C_l$).

Let $f_1(k) = (32k-1)2^{2^{3k-1}}$. Assume that the depth of the tree O_i is more than $f_1(k)$ and let $p_0p_1p_2 \ldots p_a$ be a longest path in O_i (so $a > f_1(k)$). Let $p_{j_1}, p_{j_2}, \ldots, p_{j_q}$ be the longest subsequence of $p_0p_1p_2 \ldots p_a$ such that the sets $S_{p_{j_1}}, S_{p_{j_2}}, \ldots, S_{p_{j_q}}$ all have the same signature. By Lemma 128 and by the choice of $f_1(k)$, we may assume that $q \geq 32k$.

Let S^* be the set corresponding to $p_{j_{16k}}$. We will show that if there is a solution to our instance of $(n-k)$-Test Cover then there is also a solution that does not use any edge e with $L(e) = S^*$.

Assume that there is a solution to our instance of $(n-k)$-Test Cover and assume that we pick a solution T with as few edges, e, as possible with $L(e) = S^*$. For the sake of contradiction assume that there is at least one edge e' in our solution with $L(e') = S^*$. By Theorem 122 there is a k-mini test cover, T', taken from T. Initially let $T'' = T'$. While there exists a vertex $r \in C_q$ and $r' \in C_p$ $(q \neq p)$ which are not separated by T'' then add any edge from F which separates r and r' to T''. Note that this increases the size of T'' by 1 but also increases the number of classes induced by T'' by at least 1. We continue this process for as long as possible. As $T'' \subseteq F \cup T'$ we note that $|T''| \leq 2k + 2k = 4k$. Furthermore, by construction, vertices in different C_j's are separated by edge in T''. Also note that the number of classes induced by T'' is at least $|T''| + k$ (as the number of classes induced by T' is at least $|T'| + k$).

For every edge, e, in T'' color the vertex in O_i corresponding to $L(e)$ blue. For every vertex, $v \in V(O_i)$, color v red if all paths from v to a leaf in O_i use at least one

blue vertex and v is not already colored blue. Finally for every vertex, $w \in V(O_i)$, color w orange if all siblings of w (i.e. vertices with the same in-neighbour as w) are colored blue or red and w is not colored blue or red. We now need the following:

Claim A: The number of colored vertices in O_i is at most $16k - 2$.
Proof of Claim A: As $|T''| \leq 4k$ we note that the number of blue vertices is at most $4k$. We will now show that the number of red vertices is at most $4k - 1$. Consider the forest obtained from O_i by only keeping arcs out of red vertices. Note that any tree in this forest has all its leaves colored blue and all its internal vertices colored red. Furthermore, by Lemma 129 the out-degree of any internal vertex is at least 2. This implies that the number of red vertices in such a tree is less than the number of blue vertices. As this is true for every tree in the forest we conclude that the number of red vertices in O_i is less than the number of blue vertices in O_i and is therefore bounded by $4k - 1$.

We will now bound the number of orange vertices. Since every orange vertex in O_i has at least one sibling colored blue or red (by Lemma 129). and any blue or red vertex can have at most one orange sibling we note that the number of orange vertices cannot be more than the number of vertices colored blue or red. This implies that the number of orange vertices is at most $8k - 1$.

By Lemma 120, we note that some edge, e^x, in T'' has $L(e^x) = S^*$ (as otherwise extend T'' by singletons to a test cover where no edge, e, in the solution has $L(e) = S^*$, a contradiction to our assumption). Now create T^x as follows. Initially let T^x be obtained from T'' by removing the edge e^x. Let $p_{j_{i'}}$ be an uncolored vertex in $\{p_{j_1}, p_{j_2}, \ldots, p_{j_{16k-1}}\}$ and let $p_{j_{i''}}$ be an uncolored vertex in $\{p_{j_{16k+1}}, p_{j_{16k+2}}, \ldots, p_{j_{32k-1}}\}$ (note that we do not pick $p_{j_{32k}}$). Let e_1^x be an edge in $\mathcal{E}$ with $G(e_1^x) = G(e^x)$ and $L(e_1^x)$ corresponding to the vertex $p_{j_{i'}}$ and let e_2^x be an edge in $\mathcal{E}$ with $G(e_2^x) = G(e^x)$ and $L(e_2^x)$ corresponding to $p_{j_{i''}}$. These tests exist as the signature of all sets corresponding to vertices in $p_{j_1}, p_{j_2}, \ldots, p_{j_{32k}}$ are the same. Now add e_1^x and e_2^x to T^x. The following now holds.

Claim B: The number of classes induced by T^x is at least $|T^x| + k$.
Proof of Claim B: Let $u, v \in V(H)$ be arbitrary. If $u, v \notin C_i$ and they are separated by T'', then they are also separated by T^x, as if they were separated by e^x then they will now be separated by e_1^x (and e_2^x). Now assume that $u \in C_i$ and $v \notin C_i$. If $u \in L(e^x)$ and u and v were separated by e^x then they are also separated by e_1^x. If $u \notin L(e^x)$ and u and v were separated by e^x then they are also separated by e_2^x. So

as u and v were separated by T'' we note that they are also separated by T^x. We will now show that the number of classes completely within C_i induced by T^x is at least one larger than when using T''.

By the structure of the base classes, we note that deleting e^x from T'' can decrease the number of induce classes within C_i by at most one (it may decrease the number of induced classes overall by more than one). We first show that adding the test e_1^x to $T'' \setminus \{e^x\}$ increases the number of classes within C_i by at least one. As $p_{j_{i'}}$ is not colored there is a path from $p_{j_{i'}}$ to a leaf, say u_1, without any blue vertices. Furthermore as $p_{j_{i'}}$ is not orange we note that it has a sibling, say s', that is not colored and therefore has a path to a leaf, say u_2, without blue vertices. We now note that u_1 and u_2 are not separated in T'' (and therefore in $T'' \setminus \{e^x\}$). However adding the test e_1^x to $T'' \setminus \{e^x\}$ does separate u_1 and u_2 (as $u_1 \in S_1^x$ but $u_2 \notin e_1^x$). Therefore the classes within C_i has increased by at least one by adding e_1^x to $T'' \setminus \{e^x\}$.

Analogously we show that adding the test e_2^x to $T'' \cup \{e_1^x\} \setminus \{e^x\}$ increases the number of classes within C_i by at least one. As $p_{j_{i''}}$ is not colored there is a path from $p_{j_{i''}}$ to a leaf, say v_1, without blue vertices. Furthermore as $p_{j_{i''}}$ is not orange we note that it has a sibling, say s'', that is not colored and therefore has a path to a leaf, say v_2, without blue vertices. We now note that v_1 and v_2 are not separated by T'' (and therefore in $T'' \cup \{e_1^x\} \setminus \{e^x\}$, as $p_{j_{i'}}$ lies higher in the tree O_i and therefore the edge e_1^x does not separate u and v). However adding the edge e_2^x to $T'' \cup \{e_1^x\} \setminus \{e^x\}$ does separate v_1 and v_2 (as $v_1 \in e_2^x$ but $v_2 \notin e_2^x$). Therefore the classes within C_i has increased by at least one by adding e_2^x to $T'' \cup \{e_1^x\} \setminus \{e^x\}$. So we conclude that the number of classes within C_i has increased by at least one and as any vertex not in C_i is still separated from exactly the same vertices in T^x as it was in T'' we have proved Claim B.

By Lemma 120 and Claim B we get a solution with fewer edges, e, with $L(e) = S^*$, a contradiction. □

Suppose the depth of O_i is greater than $f_1(k)$, and let S^* be the set found by the above lemma. Then we can delete all edges, e, with $L(e) = S^*$ from $\mathcal{E}$ without changing the problem, as if there is a solution for the instance then there is one that does not contain any edge e with $L(e) = S^*$. Therefore we may assume that the depth of O_i is at most $f_1(k)$.

Lemma 131. *There exist functions $f_2(d,k)$ and $f_3(d,k)$, such that in polynomial time we can reduce (H,k) to an instance such that the following holds for all vertices*

$v \in O_i$, where d is the length (i.e. number of arcs) of a longest path out of v in O_i: (1) $N^+(v) \leq f_2(d,k)$ and (2) $|S_v| \leq f_3(d,k)$.

Proof. Let v be a vertex in O_i and let d be the length of a longest path out of v in O_i. We will prove the lemma by induction on d. If $d = 0$ then v is a leaf in O_i and $N^+(v) = 0$ and $|S_v| = 1$ (as all singletons exist in $\mathcal{E}$). So now assume that $d \geq 1$ and the lemma holds for all smaller values of d. We note that the way we construct $f_3(d,k)$ below implies that it is increasing in d.

We will first prove part (1). Let $N^+(v) = \{w_1, w_2, w_3, \ldots, w_b\}$ and note that $|S_{w_j}| \leq f_3(d-1,k)$ for all $j = 1, 2, \ldots, b$ (by induction and the fact that $f_3(d,k)$ is increasing in d). Let Q_j be the subtree of O_i that is rooted at w_j for all $j = 1, 2, \ldots, b$. As part (1) holds for all vertices in Q_j we note that there are at most $g(d,k)$ non-isomorphic trees in $\{Q_1, Q_2, \ldots, Q_b\}$, where $g(d,k) < 2^{f_2(d,k)^{d+1}}$ (note that each tree in $\{Q_1, Q_2, \ldots, Q_b\}$ is isomorphic to a subgraph of the $f_2(d,k)$-regular out-tree with depth d, which has less than $f_2(d,k)^{d+1}$ edges). Furthermore the number of vertices in each Q_j is bounded by $2f_3(d-1,k) - 1$ by Lemma 129 and induction (using part (2) and the fact that every leaf in Q_j corresponds to a singleton in C_i and the number of leaves are therefore bounded by $f_3(d-1,k)$). By Lemma 128 the number of distinct signatures is bounded by $2^{2^{3k-1}}$. Let $f_2(d,k)$ be defined as follows.

$$f_2(d,k) = 2k \cdot g(d,k) \left[2^{2^{3k-1}}\right]^{2f_3(d-1,k)-1}$$

So if $b > f_2(d,k)$ there exists at least $2k+1$ trees in $\{Q_1, Q_2, \ldots, Q_b\}$ which are strongly isomorphic, in the sense that a one-to-one mapping from one to the other maintains arcs as well as signatures (a vertex with a given signature is mapped into a vertex with the same signature). Without loss of generality assume that Q_1 is one of these at least $2k+1$ trees. We now remove all vertices in Q_1 as well as all edges e with $L(e)$ corresponding to a vertex in Q_1. Delete all the items in S_{w_1}. Let the resulting hypergraph be denoted H'. We show this reduction is valid in the following claim.

Claim: This reduction is valid (i.e. (H,k) and (H',k) are equivalent).
Proof of Claim: Observe that any k-mini test cover in H' is a k-mini test cover in H, and so (H,k) is a YES-instance if (H',k) is a YES-instance.

For the converse, assume H contains a k-mini test cover T', and for each edge e in T', color the vertex in O_i corresponding to $L(e)$ blue. We first show we may assume Q_1 is uncolored. For suppose not, then since $|T'| \leq 2k$, then some other tree Q_j that is strongly isomorphic to Q_1 is uncolored. In this case, we may replace the

edges e in T' with $L(e)$ corresponding to a vertex in Q_1, by the equivalent tests e' with $L(e')$ corresponding to a vertex in Q_j.

So assume Q_1 is uncolored. Then T' is still a subset of $\mathcal{E}(H')$. It remains to show that T' still induces at least $|T'| + k$ classes over $[n']$. Observe that this holds unless there is some class C induced by $|T'|$ that only contains items from S_{w_1}. But this can only happen if some item in S_{w_1} is separated from S_{w_j} by an edge in T', for all $j \in \{2, \dots b\}$. But since $b > |F| + 1$, there exists $j \neq 1$ such that Q_j is not coloured. Then since w_1, w_j are siblings, no test in T' can separate S_{w_1} from S_{w_j}. Thus T' induces at least $|T'| + k$ classes in H', and so T' is still a k-mini test cover in the new instance. Thus, $H', k)$ is a YES-instance if and only if (H, k) is a YES-instance.

By the above claim we may assume that $b \leq f_2(d, k)$, which proves part (1). We will now prove part (2). As we have just proved that $b \leq f_2(d, k)$ and $|S_{w_j}| \leq f_3(d-1, k)$ for all $j = 1, 2, \dots, b$, we note that (2) holds with $f_3(d, k) = f_3(d-1, k) \times f_2(d, k)$. □

Theorem 132. *The* $(n - k)$TEST COVER *problem is fixed-parameter tractable.*

Proof. By Lemma 125, we can either find a k-mini test cover, and thus have a YES-instance, or find a partition of $V(H)$ into base classes $C_1, \dots, C_l$.

By Lemma 130, we may assume that O_i has depth at most $d = f_1(k)$, and by Lemma 131 part (2) we may assume that $|C_i| \leq f_3(d, k)$, for each base class C_i. Thus $|C_i| \leq f_3(f_1(k), k)$.

Hence there are at most $3k$ base classes, the size of each bounded by a function of k, so the number of items in the problem is bounded by a function of k. Thus, the problem can be solved by an algorithm of running time depending on k only.

□

w, and adding v as a child of w.

7.4 $W[1]$-hardness of $(m - k)$ TEST COVER

In this section we give the hardness result for $(m - k)$TEST COVER.

Theorem 133. $(m - k)$TEST COVER *is W[1]-complete.*

Proof. We will give a reduction from the W[1]-hard k-INDEPENDENT SET problem to $(m-k)$-TEST COVER. An input to k-INDEPENDENT SET consists of an undirected graph $G = (V, E)$ and a positive integer k (the parameter) and the objective is to

decide whether there exists an independent set of size at least k in G. A set $I \subseteq V$ is *independent* if no edge of G has both end-vertices in I.

Let G be an input graph to k-Independent Set with vertices $v_1, \ldots, v_p$ and edges $e_1, \ldots, e_q$. We construct an instance (H, k) of $(m-k)$-Test Cover as follows. The set of vertices of H is $\{e_i, e'_i : \ i \in [q]\}$ and the collection of edges is $\{T_j : j \in [p]\} \cup \{T'_i : i \in [q-1]\}$, where $T_j = \{e_i : \ v_j \in e_i\}$, the set of of edges of G incident to v_j, and $T'_i = \{e_i, e'_i\}$.

A set U of vertices of $G = (V, E)$ is a *vertex cover* if every edge of G has at least one end-vertex in U. It is well-known and easy to see that U is a vertex cover if and only if $V \setminus U$ is an independent set. Consider a minimum size vertex cover U of G, and a subset of $\mathcal{E}(H)$ $\{T_j : v_j \in U\} \cup \{T'_i : i \in [q-1]\}$. Observe that the latter is a test cover, since a pair e_i, e'_j $(i \neq j)$ is separated by $T'_{\min\{i,j\}}$, as are the pairs e_i, e_j and e'_i, e'_j, and a pair e_i, e'_i is separated by T_j for some $v_j \in U$ such that $v_j \in e_i$. Such a v_j exists since U is a vertex cover.

A test cover must use all T'_i as otherwise we cannot separate e'_i, e'_q for some $i \neq q$. A test cover must also use at least $|U|$ of T_j edges. Suppose not, and consider the corresponding set W of vertices of G, such that $|W| < |U|$. Then every e_i is separated from e'_i by T_j for some $v_j \in W$, and so W forms a vertex cover, contradicting the minimality of U. Hence G has a vertex cover of size t if and only if there is a test cover of size $q - 1 + t$.

The number of edges is $M = q - 1 + p$, and so there is a test cover of size $M - k = q - 1 + p - k$ if and only if G has an independent set with at least k vertices. Since k-Independent Set is W[1]-hard, $(m-k)$-Test Cover is W[1]-hard as well.

To prove that $(m-k)$-Test Cover is in W[1], we will use the following reduction of Test Cover to Hitting Set, based on a reduction by Moret and Shapiro [56][1].

Consider a hypergraph H with $V(H) = \{v_1, \ldots, v_n\}$ and $\mathcal{E}(H)\{e_1, \ldots, e_m\}$ of tests.

We now construct a hypergraph H^* with $V(H^*) = \{e_1, \ldots e_m\}$ and $\mathcal{E}(H^*) = \{E_{i,j} : 1 \leq i < j \leq n\}$, where $E_{i,j} = \{e_q : e_q \text{ separates } v_i, v_j\}$. Observe that H has a test cover of size $m-k$ if and only if H^* has a hitting set of size $m - k = |V(H^*)| - k$. Thus (H, k) is a Yes-instance of $(m-k)$Test Cover if and only if (H^*, k) is a Yes-instance of $(n-k)$HitSet. But by Theorem 93, $(n-k)$HitSet is in $W[1]$, and therefore so is $(m-k)$Test Cover.

□

[1]The original reduction of [56] is from Test Cover to Set Cover. However, any instance of Set Cover can be transformed into an instance of Hitting Set by switching the vertices and edges (and vice versa).

Chapter 8

Test Cover with bounded edge sizes

In the previous chapter, we weren't able to obtain any practical algorithms for our parameterizations of Test Cover. $(m-k)$Test Cover was shown to be $W[1]$-hard, and even our technically fixed-parameter tractable algorithm for $(n-k)$Test Cover was impractically large. Thus, in this chapter, we consider an important special case of Test Cover, namely the case in which every edge contains at most r vertices, for some constant r. De Bontridder *et al.* [21] claim that "this is the common restriction" for the problem and provide, as an example, protein identification. A question is whether the parameterizations of Test Cover become easier in this case. Already [34] indicated that this can be true by proving that TestCover(k) does admit a polynomial-size kernel if r is a constant.

We use the same definitions as the previous chapter. In particular, a *test cover* for a hypergraph H is still defined as a set of edges in H that separate every pair of vertices in H, and $\beta(H)$ is still defined as the minimum size of a test cover for H.

For any constant r, the problem Test-r-Cover is as follows:

Test-r-Cover
Instance: A hypergraph $H = (V, \mathcal{E})$ such that $\mathcal{E}$ is test cover of H and $\|e\| \leq r$ for every $e \in \mathcal{E}$, an integer p.
Question: Is $\beta(G) \leq p$?

Given a function $\Phi : \mathcal{H} \times \mathbb{N} \to \mathbb{N}$, define the parameterized problem $\Phi(H,k)$-Test-r-Cover as follows:

$\Phi(H,k)$-TEST-r-OVER

Instance: A hypergraph H , with n vertices, m edges such that every edge contains at most r vertices, an integer k.

Parameter: k.

Question: Is

$$\beta(G) \leq \Phi(H,k)?$$

8.1 Polynomial kernel for $(n-k)$TEST-r-COVER

In this section, we show that $(n-k)$ TEST-r-COVER admits a kernel which is polynomial in the number of vertices.

Consider a hypergraph $H = (V, \mathcal{E})$ and integer k.

Recall the definition of a k-mini test cover from the previous chapter, and recall that H contains a test cover of size $(n-k)$ if and only if if contains a k-test mini cover. Also recall that by Lemma 125, we may assume that we have a partition of V into base classes $C_1, \ldots, C_l$, such that

1. $l < 3k$
2. Each edge in $\mathcal{E}$ cuts at most one class C_i.
3. For any $e, e' \in \mathcal{E}$ and any class C_i, at least one of $(e \cap e') \cap C_i$, $(e \backslash e') \cap C_i$, $(e \backslash e') \cap C_i$ and $C_i \backslash (e \cup e')$ is empty.

(Lemma 125 implies that if this is not the case, then we can find a k-mini test cover of H and we are done.)

Also recall that $C_1, \ldots, C_l$ are the classes induced by some set of at most $2k$ edges in H, and that therefore there is at most one base class C_i that is not fully included in an edge from H. It will be useful in what follows to treat this class differently from the others. Therefore we will assume in what follows that the base classes are $G, C_1, \ldots, C_l$, where $l < 3k$, and furthermore all of $C_1, \ldots C_l$ are subsets of edges in H. Let $\mathcal{C}$ be the set of classes $C_1, \ldots, C_l$, and let C be the set of vertices contained in such classes. Observe that by construction (in particular the fact that every vertex in C is contained in at least one of a set of less than $2k$ edges), $|C| \leq (2k-1)r$. Therefore, in order to prove a kernel with a polynomial number of vertices, it will be enough to bound $|G|$.

Theorem 134. *Given an instance $(V, \mathcal{E}, k)$, it is possible to reduce it in polynomial time to an equivalent instance with at most $18k^3 r$ vertices and $(18k^3 r)^r$ edges.*

Proof. We may assume that $|V| > (7k+2)r$ as otherwise our instance is already small enough (every edge has at most r vertices and so the number of edges is at most $((7k+2)r)^r$). Observe that if there exists an edge e such that $|e \cap G| \geq |G|/2$ then $|G| \leq 2|e| \leq 2r$, and we conclude that $|V| \leq |C| + 2r \leq (2k+1)r$, a contradiction. Therefore, $|e \cap G| < |G|/2$, and so by Part 3 of Lemma 125 if X, Y are different G-portions then either $X \subset Y$, $Y \subset X$ or $X \cap Y = \emptyset$.

Apply the following algorithm:

Step 1: For each pair (C_i, C_j) $(i \neq j)$ in turn, mark $2k$ unmarked components of G which contain the G-portion of an edge containing C_i and having empty intersection with C_j (also mark these edges). If there are less than $2k$ such components, mark them all. Let $E_{i,j}$ denote the set of marked edges.

For each C_i in turn, mark $2k+1$ unmarked components of G which contain the G-portion of an edge containing C_i (also mark these edges). If there are less than $2k+1$ such components, mark them all. Let E_i denote the set of marked edges.

Step 2: Delete every edge in $\mathcal{G}$ whose G-portion is not contained in a marked component of G. Delete every vertex which is not contained in any edge anymore, except one vertex y (if it exists).

Let $\mathcal{E}'$ be the set of edges which have not been deleted by this algorithm. Notice that the number of marked components in G is at most $(3k-1)(3k-2)(2k) + (3k-1)(2k+1) < (3k-1)^2(2k+1) = 18k^3 - 3k^2 - 4k + 1 < 18k^3 - 2k$ (here we use the assumption that $k \geq 1$). Let G' be the set of vertices of G which have not been deleted by the algorithm. Notice that $|G'| \leq (18k^3 - 2k)r$.

In the instance which is produced, $|V'| = |C| + |G'| \leq 18k^3 r$, and since each edge contains at most r vertices, $|\mathcal{E}'| \leq (18k^3 r)^r$. Hence it is sufficient to show that $(V', \mathcal{E}', k)$ admits a k-mini test cover if and only if $(V, \mathcal{E}, k)$ admits one.

Obviously, if $(V', \mathcal{E}', k)$ admits a k-mini test cover, this is a k-mini test cover for $(V, \mathcal{E}, k)$ too. For the other direction, suppose $\mathcal{T}$ is a k-mini test cover for $(V, \mathcal{E}, k)$ such that $\mathcal{T} \setminus \mathcal{E}'$ is as small as possible. For the sake of contradiction, suppose that $\mathcal{T}$ contains at least one edge e in $\mathcal{T} \setminus \mathcal{E}'$. We claim that it is possible to construct a set $\mathcal{T}'''$ which induces at least $|\mathcal{T}'''| + k$ classes, such that $\mathcal{T}''' \setminus \mathcal{E}' = (\mathcal{T} \setminus \mathcal{E}') \setminus \{e\}$. By applying Theorem 122 to the hypergraph H' with edge set $\mathcal{T}'''$, and vertex set formed by identifying the vertices in each class induced by $\mathcal{T}'''$, observe that there exists a k-mini test cover in H'. Observe that the edges of this k-mini test cover in H' also form a k-mini test cover in the original instance, and this k-mini test cover

is a subcollection of $\mathcal{T}'''$. Since this k-mini test cover contains fewer edges from $\mathcal{E} \setminus \mathcal{E}'$ than $\mathcal{T}$ does, we have a contradiction.

Start with $\mathcal{T}' = \mathcal{T} \setminus \{e\}$. Since e is not in $\mathcal{E}'$, e must be in $\mathcal{G}$, and the G-portion of e must not be contained in any marked component. Furthermore, for each $C_i, C_j \in \mathcal{C}$ with $C_i \subseteq e$ and $e \cap C_j = \emptyset$ we note that $E_{i,j}$ must contain $2k$ edges, as otherwise e would be in $\mathcal{E}'$. Similarly, for each C_i contained in e we note that E_i must contain $2k+1$ edges. For any i, j such that $|E_{i,j}| = 2k$, let $e_{i,j}$ be an edge in $E_{i,j}$ whose G-portion is disjoint from any edge in $\mathcal{T}'$. This must exist as $|\mathcal{T}'| \leq 2k-1$.

For any i such that $|E_i| = 2k+1$ let e_i, e_i' be edges in E_i whose G-positions are disjoint from any edge in $\mathcal{T}'$. These edges must exist as $|\mathcal{T}'| \leq 2k-1$.

Let C_0^* be the class induced by $\mathcal{T}'$ that consists of all vertices not in any edge in $\mathcal{T}'$ (which exists by Claim C below). We will need the following claims.

Claim A: There is at most one class C_G^* induced by $\mathcal{T}'$, such that $G \cap (C_G^* \cap e) \neq \emptyset$ and $G \cap (C_G^* \setminus e) \neq \emptyset$.

Proof of Claim A: For the sake of contradiction assume that there are two such classes C_G' and C_G''. This implies that there exist vertices $x' \in G \cap (C_G' \cap e)$, $y' \in G \cap (C_G' \setminus e)$, $x'' \in G \cap (C_G'' \cap e)$ and $y'' \in G \cap (C_G'' \setminus e)$. Some edge $e' \in \mathcal{T}'$ separates C_G' and C_G''. Note that adding e' and e to $\mathcal{F}$ separates x', y', x'' and y'' into different classes, contradicting Part 3 of Lemma 125 This contradiction complete the proof of Claim A.

Claim B: For each edge e' that cuts G and every C_i we have $C_i \subseteq e'$ or $C_i \cap e' = \emptyset$. In particular, $C_i \subseteq e$ or $C_i \cap e = \emptyset$.

Proof of Claim B: If Claim B is false then there exist $x \in C_i \cap e'$ and $y \in C_i \setminus e'$. So adding e' to $\mathcal{F}$ cuts G (as G is not a subset of any edge and e' contains vertices from G) and C_i, a contradiction to Part 2 of Lemma 125 .

Claim C: C_0^* exists and $|G \cap C_0^*| \geq (3k+2)r$.

Proof of Claim C: If C_0^* does not exist then every vertex of V belongs to some edge in $\mathcal{T}'$, which implies that $|V| \leq 2kr$, so C_0^* does exist. If $|G \cap C_0^*| < (3k+2)r$, then the following holds and we have a contradiction to the assumption on $|V|$ in the beginning of the proof:

$$|V| \leq 2kr + |C_0^*| \leq 2kr + (|C| + |G \cap C_0^*|) < 2kr + 2kr + (3k+2)r = (7k+2)r.$$

By Claim A there exists at most one class, say C_G^*, induced by $\mathcal{T}'$ that is cut by e and only contains vertices from G. Let $C_1^*, \ldots, C_t^*$ be all classes induced by $\mathcal{T}'$,

different from C^*_G and C^*_0, that are cut by e. Note that $t \leq 3k$, as $\mathcal{T}$ is a k-mini test cover. Each C^*_s, $(1 \leq s \leq t)$, must be contained in an edge, say e^*_s, in $\mathcal{T}'$ and contain vertices from C, by the definitions on C^*_G and C^*_0. We are going to create a collection of edges $\mathcal{T}''$ such that each C^*_s is cut by an edge in $\mathcal{T}''$ and also $\mathcal{T}''$ induces $|\mathcal{T}''|$ extra classes in C^*_0. Initially let $\mathcal{T}'' = \emptyset$. For each $s \in [t]$ in turn, consider the following two cases.

Case 1: For some $i \neq j$, e contains C_i but not C_j and $C_i \cap C^*_s \neq \emptyset, C_j \cap C^*_s \neq \emptyset$.

In this case observe that $|E_{i,j}| = 2k$, as otherwise e would be marked. Then add the edge $e_{i,j}$ to $\mathcal{T}''$, if $e_{i,j}$ is not in $\mathcal{T}''$ already. Note that $e_{i,j}$ separates C_i from C_j and therefore cuts C^*_s, and also creates an extra class in C^*_0, as desired.

Case 2: Case 1 does not hold. That is, $C \cap C^*_s \subseteq e$ or $(C \cap C^*_s) \cap e = \emptyset$.

Recall that there exists a C_i such that C^*_s contains vertices from C_i and, since e cuts C^*_s, we have $C^*_s \cap G \neq \emptyset$. Suppose e does not contain C_i. Then $(C \cap C^*_s) \cap e = \emptyset$ and, since e cuts C^*_s, it must contain vertices from $C^*_s \cap G \subseteq e^*_s \cap G$. Then e^*_s cuts G and the G-portion of e^*_s is in the same component as the G-portion of e, and therefore e^*_s is an unmarked edge. Furthermore since e^*_s cuts G it does not cut C_i, and therefore $C_i \subseteq e^*_s$. Thus, we have that either e or e^*_s is an unmarked edge containing C_i, and therefore $|E_i| = 2k + 1$. Then add e_i to $\mathcal{T}''$, if e_i is not already in $\mathcal{T}''$. Observe that e_i cuts C^*_s as it contains vertices in $C_i \cap C^*_s$ but no vertex from $C^*_s \cap G$, and e_i creates an extra class in C^*_0, as required.

This completes Case 1 and Case 2. Note that the edges in $\mathcal{T}''$ all have vertex disjoint G-portions, as they are in distinct $E_{i,j}$'s and E_i's. We now consider Case (i) and Case (ii) below, which will complete the proof.

Case (i): C^*_G does not exist or is equal to C^*_0 or e does not cut C^*_0 or does not cut C^*_G.

In this case e cuts at most $t + 1$ classes induced by $\mathcal{T}'$. Note that if we add the edges from $\mathcal{T}''$ to $\mathcal{T}'$, each edge in $\mathcal{T}'$ increase the number of classes in C^*_0 by at least one. Also note that for every $s \in [t]$ some edge in $\mathcal{T}''$ cuts C^*_s. So let $\mathcal{T}''' = \mathcal{T}' \cup \mathcal{T}'' = (\mathcal{T} \setminus e) \cup \mathcal{T}''$. Removing e from $\mathcal{T}$ decreases the number of classes by at most $t + 1$ and adding $\mathcal{T}''$ increases the number of classes by at least $t + |\mathcal{T}''|$. So by increasing the number of edges by $|\mathcal{T}''| - 1$ we have increased the number of classes by at least $|\mathcal{T}''| - 1$ and therefore we still have at least k more classes than edges.

Case (ii): Case (i) does not hold. That is, C^*_G exists and is distinct from C^*_0 and

e cuts both C^*_G and C^*_0.

By Claim A we note that e either contains all of $C^*_0 \cap G$ or none of $C^*_0 \cap G$. By Claim C e must contain none of $C^*_0 \cap G$. As e cuts C^*_0 we must have $C \cap e \cap C^*_0 \neq \emptyset$. Therefore there exists C_i such that e contains vertices from $C_i \cap C^*_0$, and so $|E_i| = 2k+1$. Add e_i and e'_i to $\mathcal{T}''$ (unless e_i is already in $\mathcal{T}'$, in which case just add e'_i to $\mathcal{T}''$). Observe that the G-portions of e_i and e'_i are vertex disjoint by construction, and so the G-portions of all edges in $\mathcal{T}''$ are still vertex disjoint.

Note that adding e_i and e'_i to $\mathcal{T}'$ creates three new classes in C^*_0 (C^*_0 now being split into the class $C^*_0 \cap e \cap e'$ which contains vertices from C_i, the G-portion of e_i, the G-portion of e'_i and the class of vertices not in any edge). Adding each other edge from $\mathcal{T}''$ to $\mathcal{T}'$ increases the number of classes in C^*_0 by one (as by Claim C we note that some vertex in $G \cap C^*_0$ is not contained in any edge in $\mathcal{T}''$). Also note that for every $s \in [t]$ some edge in $\mathcal{T}''$ cuts C^*_s.

So let $\mathcal{T}''' = \mathcal{T}' \cup \mathcal{T}'' = (\mathcal{T} \setminus e) \cup \mathcal{T}''$. Removing e from $\mathcal{T}$ decreases the number of classes by $t+2$ and adding $\mathcal{T}''$ increases the number of classes by at least $t+|\mathcal{T}''|+1$. So by increasing the number of edges by $|\mathcal{T}''|-1$ we have increased the number of classes by at least $|\mathcal{T}''|-1$ and therefore we still have at least k more classes than edges. □

8.2 Fixed-parameter tractability of $(m-k)$ Test-r-Cover

In this section, we show that $(m-k)$ Test-r-Cover is fixed-parameter tractable. The proof is short but does not lead to a polynomial kernel; this requires more work, and is the subject of the next section.

Lemma 135. *If $\mathcal{E}$ induces $t \geq 2$ classes in a hypergraph $H = (V, \mathcal{E})$ and $i \in [t-1]$ then there is a subset $\mathcal{F}$ of $\mathcal{E}$ with i edges that induces at least $i+1$ classes.*

Proof. By induction on $i \in [t-1]$. To see that the lemma holds for $i = 1$ set $\mathcal{F} = \{e\}$, where e is any edge of $\mathcal{E}$ with less than $|V|$ vertices. Let $\mathcal{F}$ be a subset of $\mathcal{E}$ with $i-1$ edges that induces at least i classes, let x, y be vertices separated by $\mathcal{E}$ not separated by $\mathcal{F}$, and let e be an edge separating x and y. It remains to observe that $\mathcal{F} \cup \{e\}$ induces at least $i+1$ classes. □

Corollary 136. *In a hypergraph $H = (V, \mathcal{E})$, let $X, Y \subseteq V$ be such that $X \cap Y = \emptyset$ and $\mathcal{E}$ separates X and Y. Then there is a subset $\mathcal{F}$ of $\mathcal{E}$ that separates X and Y and has at most $t_X + t_Y - 1$ edges, where t_X (t_Y) is the number of classes induced by $\mathcal{E}$ intersecting X (Y).*

Proof. Apply Lemma 135 to $H[X \cup Y]$ and then extend the obtained edges of $H[X \cup Y]$ beyond $X \cup Y$ such that they correspond to edges of H. □

Theorem 137. *There is an algorithm for* $(m-k)$Test-r-Cover *that runs in time* $2(r^2+1)^k(n+m)^{O(1)}$.

Proof. We first need to guess whether there will be a vertex not contained in any edge in the solution, and if so, which vertex it will be. If there already exists a vertex y_0 not in any edge in $\mathcal{E}$, then we are done. Otherwise, either pick a vertex x or guess that every vertex in V will be covered by the solution. If a vertex x is picked, delete all the edges containing x, and reduce k by the number of deleted edges. If it is guessed that every vertex in V will be covered by the solution, add a new vertex y_0 which is not in any edge. Observe that this does not change the solution to the problem. By doing this we have split the problem into $n+1$ separate instances, with each instance containing an isolated vertex. Thus we may now assume that there exists a vertex y_0 which is not contained in any edge in $\mathcal{E}$.

Consider an edge $e \in \mathcal{E}$, and suppose that $\mathcal{E} \setminus \{e\}$ is a test cover. Let $\mathcal{B}_0$ be a minimal set of edges in $\mathcal{E} \setminus \{e\}$ which covers e. Note that such a set must exist, as otherwise x is not separated from y_0 in $\mathcal{E} \setminus \{e\}$ for some $x \in e$, and so $\mathcal{E} \setminus \{e\}$ is not a test cover. Furthermore, we may assume $|\mathcal{B}_0| \leq |e| \leq r$. Now for each $b \in \mathcal{B}_0$, let $\mathcal{B}_b$ be a minimal set of edges in $\mathcal{E} \setminus \{e\}$ separating every vertex in $b \setminus e$ from every vertex in $b \cap e$. By Corollary 136, we may assume that $|\mathcal{B}_b| \leq r-1$.

Now let $\mathcal{B} = \mathcal{B}_0 \cup (\bigcup_{b \in \mathcal{B}_0} \mathcal{B}_b)$, and observe that $\mathcal{B}$ isolates the vertex set of e. Thus, in any solution with minimum number of edges, at least one edge from $\mathcal{B} \cup \{e\}$ will be missing. Note that $|\mathcal{B}| \leq r + r(r-1) = r^2$.

We now describe a depth-bounded search tree algorithm for $(m-k)$Test-r-Cover. If $\mathcal{E}$ is not a test cover, return No. Otherwise if $k = 0$ return Yes. Otherwise, for each edge $e \in \mathcal{E}$ check whether $\mathcal{E} \setminus \{e\}$ is a test cover. If for all $e \in \mathcal{E}$, $\mathcal{E} \setminus \{e\}$ is not a test cover, then a test cover must contain all m edges and so we return No. Otherwise, let e be an edge such that $\mathcal{E} \setminus \{e\}$ is a test cover, and construct the set $\mathcal{B}$ as described above. Then we may assume one of $\mathcal{B} \cup \{e\}$ is not in the solution. Thus we may pick one edge from $\mathcal{B} \cup \{e\}$, delete it, and reduce k by 1. So we split into r^2+1 instances with reduced parameter.

We therefore have a search tree with at most $(r^2+1)^k$ leaves. As every internal node has at least 2 children, the total number of nodes is at most $2(r^2+1)^k - 1$. Note also that guessing the isolated vertex at the start split the problem into $n+1$ instances, so there are at most $(n+1)(2(r^2+1)^k - 1)$ nodes to compute in total.

As each node in the tree takes polynomial time to compute, we have an algorithm with total running time $2(r^2+1)^k(n+m)^{O(1)}$. □

8.3 Polynomial Kernel for $(m-k)$Test-r-Cover

In this section, we show that $(m-k)$Test-r-Cover admits a polynomial kernel. It will be useful to consider a slight generalization of $(m-k)$Test-r-Cover, which we call $(m-k)$Subset-Test-r-Cover. In this problem, we are given a special subset $\mathcal{B} \subseteq \mathcal{E}$ of edges which are required to be in the solution. For convenience we will say these edges are *colored black*.

Given $\mathcal{F} \subseteq \mathcal{E}$, define the neighborhood $N_1(\mathcal{F}) = \{e \in \mathcal{E} \setminus \mathcal{F} \ : \ \exists f \in \mathcal{F}, \ \ f \cap e \neq \emptyset\}$, $N_1[\mathcal{F}] = N_1(\mathcal{F}) \cup \mathcal{F}$ and $N_j[\mathcal{F}] = N_1[N_{j-1}[\mathcal{F}]]$.

We begin with the following reduction rules, which must be applied whenever possible:

Rule 10. *Given a vertex x of degree 1 and a black edge b which contains only x, delete b from $\mathcal{B}$ and $\mathcal{E}$, delete x and leave k the same.*

Rule 11. *Given a black edge b, if there exists any other edge e such that $b \subset e$, then replace e with $e \setminus b$. If there exists a black edge b' such that b cuts b' and b' cuts b, delete b and b' and add black edges $b \setminus b'$, $b' \setminus b$ and $b \cap b'$. Leave k the same.*

Lemma 138. *Let $(V, \mathcal{E}', \mathcal{B}', k)$ be an instance of $(m-k)$Subset-Test-r-Cover derived from $(V, \mathcal{E}, \mathcal{B}, k)$ by an application of Rule 10 or 11. Then $(V, \mathcal{E}', \mathcal{B}', k)$ is a* Yes*-instance if and only if $(V, \mathcal{E}, \mathcal{B}, k)$ is a* Yes*-instance.*

Proof. We will show for each rule that for any t, $(V, \mathcal{E}, \mathcal{B}, k)$ has a solution of size $|\mathcal{E}| - k$ if and only if $(V, \mathcal{E}', \mathcal{B}', k)$ has a solution of size $|\mathcal{E}'| - k$.

Rule 10: Suppose $(V, \mathcal{E}', \mathcal{B}', k)$ is a Yes-instance, with solution $\mathcal{T}'$. Then observe that $\mathcal{T} = \mathcal{T}' \cup \{b\}$ is a solution for $(V, \mathcal{E}, \mathcal{B}, k)$. Conversely, if $\mathcal{T}$ is a solution for $(V, \mathcal{E}, \mathcal{B}, k)$ then $\mathcal{T}$ must contain b, and $\mathcal{T} \setminus \{b\}$ is a solution for $(V, \mathcal{E}', \mathcal{B}', k)$.

Rule 11: First consider the case when $b \subset e$ for some other edge e. It is sufficient to show that for any $\mathcal{T} \subseteq \mathcal{E}$ containing e and b, $\mathcal{T}$ is a test cover if and only if $(\mathcal{T} \setminus \{e\}) \cup \{e \setminus b\}$ is a test cover. To see this, observe that for any $x \in e, y \notin e$, x and y are separated either by b or $e \setminus b$, and for any $x \in e \setminus b$, $y \notin e \setminus b$, x and y are separated either by b or e.

Now consider the case when b, b' are intersecting black edges. Similar to the previous case, if x and y are separated by one of b, b' then they are also separated by at least one of $b \setminus b'$, $b' \setminus b$, $b \cap b'$, and if they are separated by one of $b \setminus b'$, $b' \setminus b$, $b \cap b'$ then they are also separated by at least one of b, b'. □

Lemma 139. *Let $(V, \mathcal{E}, \mathcal{B}, k)$ be an instance irreducible by Rules 10 and 11. The instance can be reduced, in polynomial time, to an equivalent instance such that every vertex has degree at most kr^2.*

Proof. Assume that there exists a vertex x with degree in $\mathcal{E}$ greater than kr^2. We will be able to produce an equivalent instance in which either k or the degree of x is reduced. Clearly this reduction can only take place a polynomial number of times, so in polynomial time we will reduce to an instance in which every vertex has degree bounded by kr^2.

We produce a special set $\widetilde{X}$ such that $\widetilde{X}$ is still isolated when at most k edges are deleted according to the following algorithm.

Set $\widetilde{\mathcal{E}} = \mathcal{E}$, $i = 1$, $X = \{x\}$, $j = 1$;
while $i \leq k + 1$ **do**
  if $\widetilde{\mathcal{E}}$ *isolates* X **then**
    Let e_i be an edge containing X, and construct a set $E_i \subseteq \widetilde{\mathcal{E}}$ such that $E_i \cup \{e_i\}$ isolates X and $|E_i| \leq r - 1$;
    Set $\widetilde{\mathcal{E}} = \widetilde{\mathcal{E}} \setminus (E_i \cup \{e_i\})$;
    Set $i = i + 1$;
  else
    Let X' be the class induced by $\widetilde{\mathcal{E}}$ containing X;
    Set $X = X', i = 1, j = j + 1$;
  end
end
Set $\widetilde{X} = X$.

Observe that throughout the algorithm, by construction any edge in $\widetilde{\mathcal{E}}$ which contains x also contains X as a subset, $|X| \geq j$ and $\widetilde{\mathcal{E}} \subseteq \mathcal{E}$. Notice if the algorithm ever sets $j = r + 1$, then at that point at most kr^2 edges have been deleted from $\widetilde{\mathcal{E}}$, and as $|X| \geq r + 1$, no remaining edges in $\widetilde{\mathcal{E}}$ contain x. But this is a contradiction as the degree of x is greater than kr^2. Therefore we may assume the algorithm never reaches $j = r + 1$. Hence the algorithm must terminate for some $j \leq r$.

We now show that we can always find e_i and E_i for $i \leq k + 1$. Since x has degree greater than kr^2 and at most $kr(r - 1)$ edges are removed from $\widetilde{\mathcal{E}}$ earlier in the algorithm, we can always find an edge e_i containing x and therefore containing X. To see that E_i can be constructed using at most $r - 1$ edges, apply Corollary 136 to X and $e_i \setminus X$.

Now consider the set $\widetilde{X}$ formed by the algorithm. Since when the algorithm terminated, e_i and E_i were found for all $i \leq k + 1$ if we remove k arbitrary edges

from $\mathcal{E}$, it is still possible to find i such that no edges in $E_i \cup \{e_i\}$ have been deleted. This means that as long as we delete at most k edges, $\widetilde{X}$ is still isolated. Therefore if $\widetilde{X}$ is an edge in $\mathcal{E}$, delete it and reduce k by 1. If $\widetilde{X}$ is not an edge in $\mathcal{E}$, add a new black edge $\widetilde{X}$ to $\mathcal{E}$ and $\mathcal{B}$, keeping k the same, and apply Rule 10 and 11. Observe that since $\widetilde{X}$ is properly contained in at least two edges, this will decrease the degree of every vertex in $\widetilde{X}$. □

Now assume that $(V, \mathcal{E}, \mathcal{B}, k)$ is reduced by Rules 10 and 11 and that every vertex has degree at most kr^2. We will color the uncolored edges in $\mathcal{E}$ as follows. For every edge e which is not black, if $\mathcal{E} \setminus \{e\}$ is not a test cover, color e black, adding it to $\mathcal{B}$ (and apply Rules 10 and 11). If $\mathcal{E} \setminus \{e\}$ is a test cover and e contains a degree one vertex, color e orange. Otherwise, color e green.

Remark 140. *Notice that an edge is colored orange only if there is no isolated vertex.*

Lemma 141. *If G is a set of green edges such that, for every pair $g_1, g_2 \in G$, $N_1[g_1] \cap N_1[g_2]$ is empty, then $\mathcal{E} \setminus G$ is a test cover.*

Proof. We proceed by induction on $|G|$. If $|G| = 1$ this is obviously true. If $|G| = j+1$, delete the first j edges and consider the last one, denoted g. The only problem that could occur removing g is that a vertex $x \in g$ may no longer be separated from another vertex y. If y is not in one of the edges in $G \setminus \{g\}$, then x and y are not separated even by $\mathcal{E} \setminus \{g\}$, which is a contradiction since g is green. Therefore, denote by g' the edge in G which contains y. The degree of y is at least 2, hence there exists an edge different from g' which contains y; this edge cannot contain x too, or $N_1[g] \cap N_1[g']$ would not be empty. This ensures that x and y are separated by $\mathcal{E} \setminus G$. □

Rule 12. *Given an orange edge o, if $N_2[o]$ contains no green edges, delete o and decrease k by 1. (Notice that this creates an isolated vertex, which means that every other orange edge will become black.)*

Lemma 142. *Let $(V, \mathcal{E}', \mathcal{B}', k-1)$ be an instance of $(m-k)$*Subset-Test-r-Cover *derived from $(V, \mathcal{E}, \mathcal{B}, k)$ by an application of Rule 12. Then $(V, \mathcal{E}', \mathcal{B}', k-1)$ is a* Yes*-instance if and only if $(V, \mathcal{E}, \mathcal{B}, k)$ is a* Yes*-instance.*

Proof. Let $(V, \mathcal{E}, \mathcal{B}, k)$ be a Yes-instance, and suppose there is an orange edge o such that $N_2[o]$ contains no green edges. It is sufficient to prove that there exists a solution that does not include o.

Suppose $\mathcal{T} \subseteq \mathcal{E}$ is a solution and suppose it contains o. If there is a vertex x which is not contained in any edge of $\mathcal{T}$, consider an edge $e \in \mathcal{E}$ which contains it (which exists by Remark 140). If the isolated vertex x does not exist, take any edge e from $\mathcal{E} \setminus \mathcal{T}$.

Consider $\mathcal{T}' = (\mathcal{T} \setminus \{o\}) \cup \{e\}$. We claim that $\mathcal{T}'$ is still a test cover.

First of all, note that if there is an orange edge $o' \in \mathcal{E} \setminus \mathcal{T}$, this edge must be e: removing o' creates an isolated vertex and o' is the only edge containing that vertex, which means that o' is the edge that we add to make $\mathcal{T}'$. Therefore, we may assume $(N_2[o] \setminus \{o\}) \subseteq \mathcal{T}'$.

Now, the only problem that can occur removing o is that a vertex $x \in o$ is no longer separated from a vertex $y \in V \setminus o$. Vertices x and y must be separated by some other edge in $\mathcal{E} \setminus \mathcal{T}'$ as o is not black; furthermore this edge must contain y and not x, as any other edge containing x is in $(N_2[o] \setminus \{o\}) \subseteq \mathcal{T}'$. Let this edge be $\widetilde{e}$.

If x is the degree 1 vertex, it is now the only isolated vertex and therefore it is separated from any other vertex. If x is a vertex of degree at least 2, then there is an edge (different from o) containing it; moreover, this edge cannot contain any vertices of $\widetilde{e}$, because in this case $\widetilde{e} \in N_2[o]$. Hence, even deleting o, every vertex $x \in o$ is still separated from any other vertex, which ensures that $\mathcal{T}'$ is a test cover. □

Lemma 143. *Let $(V, \mathcal{E}, \mathcal{B}, k)$ be an instance obtained after applying Rules 10, 11 and 12, and let x be a non-isolated vertex. Then $x \in V(N_3[g])$ for some green edge g.*

Proof. If x is contained in a green edge, we are done. If x is contained in an orange edge o, Lemma 142 implies that there exists a green edge g in $N_2[o]$, which means that $x \in V(N_2[g]) \subseteq V(N_3[g])$. If, finally, x is contained in a black edge b, this edge must intersect one other edge, that can be either green or orange (due to Rules 10 and 11). In both cases, $x \in V(N_3[g])$ for some green edge g. □

Theorem 144. *There is a kernel for $(m-k)$*Subset-Test-r-Cover *with $|V| \leq (k-1)k^5r^{16}+1$ and $|\mathcal{E}| \leq (k-1)k^5r^{16}+k$. This gives a kernel for $(m-k)$* Test-r-Cover *with $|V| \leq 5(k-1)k^5r^{16}+4k+1$ and $|\mathcal{E}| \leq 3(k-1)k^5r^{16}+3k$.*

Proof. Let $(V, \mathcal{E}, \mathcal{B}, k)$ be an instance irreducible by Rules 10, 11 and 12. Construct greedily a set G of green edges which satisfy the hypothesis of Lemma 141. If $|G| \geq k$, we answer Yes using Lemma 141. Otherwise, $|G| \leq k-1$ and every green edge which is not in G must be in $N_2[G]$. By Lemma 143, this means that every vertex (except the one of degree zero, if it exists) must be in $V(N_3[N_2[G]]) = V(N_5[G])$.

However, $|V(N_5[G])| \leq r|N_5[G]|$ and, given $\mathcal{F} \subseteq \mathcal{E}$, $|N_1[\mathcal{F}]| \leq |\mathcal{F}|(r)(kr^2)$ (by Lemma 139), which means that $|N_5[G]| \leq |G|(kr^3)^5$. To sum up, $|V(N_5[G])| \leq (k-1)k^5r^{16}$, which gives us the required bound on the number of vertices.

To bound the number of edges, we show that there is a solution of size at most $|V|$. First let $\mathcal{T}$ be the set of black edges. By Rule 11, the black edges are disjoint and therefore $|\mathcal{T}| \leq |V|$ and $\mathcal{T}$ induces at least $|\mathcal{T}|$ classes. Now if $\mathcal{T}$ is not a test cover, add an edge to $\mathcal{T}$ that increases the number of induced classes. Then eventually we have that $|\mathcal{T}| \leq |V|$ and $\mathcal{T}$ induces $|V|$ classes as required. Therefore if $|\mathcal{E}| - k \geq |V|$, the answer is YES. Hence $|\mathcal{E}| \leq |V| + k - 1 \leq (k-1)k^5r^{16} + k$, proving the kernel for $(m-k)$ SUBSET-TEST-r-COVER.

We now prove the kernel for $(m-k)$ TEST-r-COVER. First transform an instance $(V, \mathcal{E}, k)$ into an equivalent instance $(V, \mathcal{E}, \mathcal{B}, k)$ of $(m-k)$ SUBSET-TEST-r-COVER, by letting $\mathcal{B} = \emptyset$. Then reduce this instance to a kernel $(V', \mathcal{E}', \mathcal{B}', k')$. Now we reduce $(V', \mathcal{E}', \mathcal{B}', k')$ to an instance $(V'', \mathcal{E}'', k'')$ of $(m-k)$TEST-r-COVER, completing the proof. If an edge b is colored black, and $\mathcal{E} \setminus \{b\}$ is not a test cover, then uncolor b. Otherwise, observe that by construction, b must have been created during the algorithm of Lemma 139, and in such a case b was contained within $k''+1$ other edges. Therefore b contains at most $r-1$ vertices. We will replace b with a small gadget such that the component b is still generated in any Test Cover solution.

To make the gadget, add vertices x_1, x_2, x_3, x_4 to the instance, replace b with $b \cup \{x_1\}$ and add edges $e' = \{x_1, x_2, x_3\}$ and $e'' = \{x_3, x_4\}$. Observe edge e' is necessary to separate x_3 from x_4, e'' is necessary to separate x_2 from x_3 and $b \cup \{x_1\}$ is necessary to separate x_1 from x_2. Hence all three edges must be in a test cover, generating the region previously generated by the black edge, as required.

Finally observe that for each original black edge we added four new vertices and two new edges. Hence $|V''| \leq |V'| + 4|\mathcal{E}'| \leq 5(k-1)k^5r^{16} + 4k + 1$ and $|\mathcal{E}''| \leq |\mathcal{E}'| + 2|\mathcal{E}'| \leq 3(k-1)k^5r^{16} + 3k$. □

8.4 TEST-r-COVER$(\frac{2(n-1)}{r+1} + k)$

In this final section, we prove a new *lower bound* for TEST-r-COVER, and consider TEST-r-COVER parameterized *above* this bound.

Proposition 145. *If the size of every edge is at most r, then every test cover has at least $\lceil \frac{2(n-1)}{r+1} \rceil$ edges. This lower bound on the size of a test cover is tight.*

Proof. We first prove that $\lceil \frac{2(n-1)}{r+1} \rceil$ is indeed a lower bound. Given a test cover of size m, observe that at most one vertex may be contained in no edges. For each

of the m edges, at most one vertex is contained in only that edge and every other vertex is contained in at least two edges. Hence $n \leq 1 + m + \frac{m(r-1)}{2}$, which implies $m \geq \frac{2(n-1)}{r+1}$. Observe that no vertex being contained in three edges is a necessary condition for the bound to be tight.

To see that this bound is tight, consider a set $V = \{x_{i,j} : i, j \in [r]\}$ of vertices, and a set $\mathcal{E} = \{e_q : q \in [r-1]\} \cup \{e'_s : s \in [r-1]\}$ of edges, where $e_q = \{x_{q,j} : j \in [r]\}$, $e'_s = \{x_{i,s} : i \in [r]\}$ (see Figure 8.1). Since $n = r^2$, we have $|\mathcal{E}| = 2(r-1) = \frac{2(n-1)}{r+1}$. Consider two vertices $x_{i,j}$ and $x_{i',j'}$. If $i \neq i'$, then $x_{i,j}$ and $x_{i',j'}$ are separated by $e_{\min\{i,i'\}}$ and if $j \neq j'$, then $x_{i,j}$ and $x_{i',j'}$ are separated by $e'_{\min\{j,j'\}}$. Thus, $\mathcal{E}$ is a test cover of minimum possible size. By using multiple copies of this construction (except for $x_{r,r}$), one can see that the bound is tight for arbitrarily large n. □

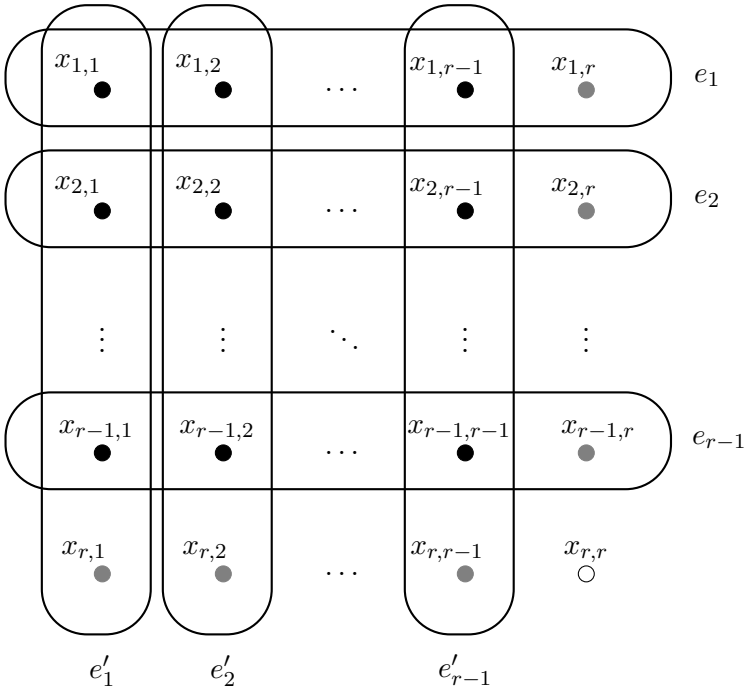

Figure 8.1: Illustration of the hypergraph $(V, \mathcal{E})$ in Proposition 145. Vertices of degree two are labelled in black, vertices of degree one in gray and the vertex of degree zero in white.

From this lower bound, a kernel for Test-r-Cover(k), with less vertices than in [34], immediately follows:

Corollary 146. (k) Test-r-Cover *admits a kernel with at most* $k(r+1)/2 + 1$ *vertices.*

It follows from the proof of Proposition 145 that for a test cover of size $\frac{2(n-1)}{r+1}$, there must be exactly one vertex contained in no edges, each edge must contain one vertex of degree 1, and every other vertex must be of degree 2.

To prove the next lemma we give a reduction from the r-DIMENSIONAL MATCHING problem, which is one of Karp's 21 NP-complete problems [45], for each $r \geq 3$. To state the problem, we will give the following definitions. A collection $\mathcal{E}$ of edges of a hypergraph H is called a *matching* if no two edges of $\mathcal{E}$ have a common vertex. A matching is *perfect* if every vertex of H belongs to an edge of the matching. A hypergraph H is *r-partite r-uniform* if the vertex set $V(H)$ of H can be partitioned into r sets $X_1, \ldots, X_r$ such that each edge of H has exactly one vertex from each X_i. In r-DIMENSIONAL MATCHING, given an r-partite r-uniform hypergraph H, the aim is to decide whether H has a perfect matching.

Lemma 147. ($\frac{2(n-1)}{r+1} + k$) TEST-r-COVER *is NP-hard for* $k = 0$ *and* $r \geq 3$.

Proof. We give a reduction from r-DIMENSIONAL MATCHING to ($\frac{2(n-1)}{r+1} + k$)TEST-r-COVER for $k = 0$. We assume we have an instance of r-DIMENSIONAL MATCHING given by an r-partite r-uniform hypergraph G with vertex set $V(G) = \{x_{i,j} : \ i \in [r], j \in [n']\}$, where $X_i = \{x_{i,j} : j \in [n']\}$ form the partition of $V(G)$, i.e. each edge has exactly one vertex in each X_i. (We may assume all X_i have the same size, as otherwise there is no perfect matching.) The edge set of G will be denoted by $\mathcal{E}(G)$. Additionally, we may assume that n' is divisible by $r - 1$.

We now give the construction of $(V, \mathcal{E})$, the TEST-r-COVER instance. Let the vertex set $V = V(G) \cup Y$, and the edge set $\mathcal{E} = \mathcal{E}(G) \cup \mathcal{E}'$, where Y and $\mathcal{E}'$ are defined as follows:

$$Y = \{y_{i,j} : \ i \in [r-1], j \in \{(r-1), 2(r-1), ..., n'\}\} \cup \{y_0\}$$

$$\mathcal{E}' = \{e_{i,p} : \ i \in [r-1], p \in \{(r-1), 2(r-1), ..., n'\}\}$$

where $e_{i,p} = \{x_{i,p-r+2}, x_{i,p-r+3}, \ldots, x_{i,p}, y_{i,p}\}$ (See Figure 8.2) Note that $n = |V| = rn' + n' + 1$.

We first show that if $\mathcal{E}'' \subseteq \mathcal{E}(G)$ is a perfect matching in G, then $\mathcal{E}' \cup \mathcal{E}''$ is a test cover of size $\frac{2(n-1)}{r+1}$. For any pair $x_{i,j}, x_{i',j'} \in V(G)$, if $i = i'$ then $x_{i,j}, x_{i',j'}$ appear in different edges in $\mathcal{E}''$ and are separated, if $i < i'$ they are separated by $e_{i,(r-1)\lceil \frac{j}{r-1} \rceil}$, and if $i > i'$ they are separated by $e_{i',(r-1)\lceil \frac{j'}{r-1} \rceil}$. Any pair $y_{i,j}, y_{i',j'} \in Y$ is separated by $e_{i,j}$. As $\mathcal{E}'$ covers $V(G)$, every $x_{i,j} \in V(G)$ is separated from every $y_{i',j'} \in Y$. Finally, y_0 is separated from every other vertex as it is the only vertex not covered by $\mathcal{E}' \cup \mathcal{E}''$. Thus, $\mathcal{E}' \cup \mathcal{E}''$ is a test cover. Since $|\mathcal{E}' \cup \mathcal{E}''| = 2n'$ and $n = rn' + n' + 1$, we have that $\mathcal{E}' \cup \mathcal{E}''$ is a test cover of size $\frac{2(n-1)}{r+1}$.

It remains to show that if $(V, \mathcal{E})$ has a test cover of size $\frac{2(n-1)}{r+1}$, then G has a perfect matching. Firstly, observe that to separate y_0 from $y_{i,p}$, every edge $e_{i,p}$ must be in the test cover. Hence $\mathcal{E}'$ is contained in any test cover for $(V, \mathcal{E})$. So a test cover is a set $\mathcal{E}' \cup \mathcal{E}''$, where $\mathcal{E}'' \subseteq \mathcal{E}(G)$. Next, observe that for every vertex $x_{i,j}$, there must exist an edge in $\mathcal{E}''$ containing $x_{i,j}$, as otherwise there would be no edge separating $x_{i,j}$ from $y_{i,(r-1)\lceil \frac{j}{r-1} \rceil}$.

We may also observe that no two edges in $\mathcal{E}''$ intersect. Suppose two edges intersected, but not in partite set X_r. Then there is another edge in $\mathcal{E}'$ also intersecting these edges at the same vertex. But we know from Proposition 145 that in a test cover of size $\frac{2(n-1)}{r+1}$, at most two edges intersect at any vertex, so this can not happen. Suppose instead, that edges e and e' intersect in partite set X_r. Then e also intersects an edge in $\mathcal{E}'$ in each of the other partitions, so every vertex in e is of degree 2. But we know that in a test cover of size $\frac{2(n-1)}{r+1}$, one vertex in each edge has degree 1, so this case is also not possible. Hence, a test cover with $\frac{2(n-1)}{r+1}$ edges for $(V, \mathcal{E})$ would give a perfect matching in G. □

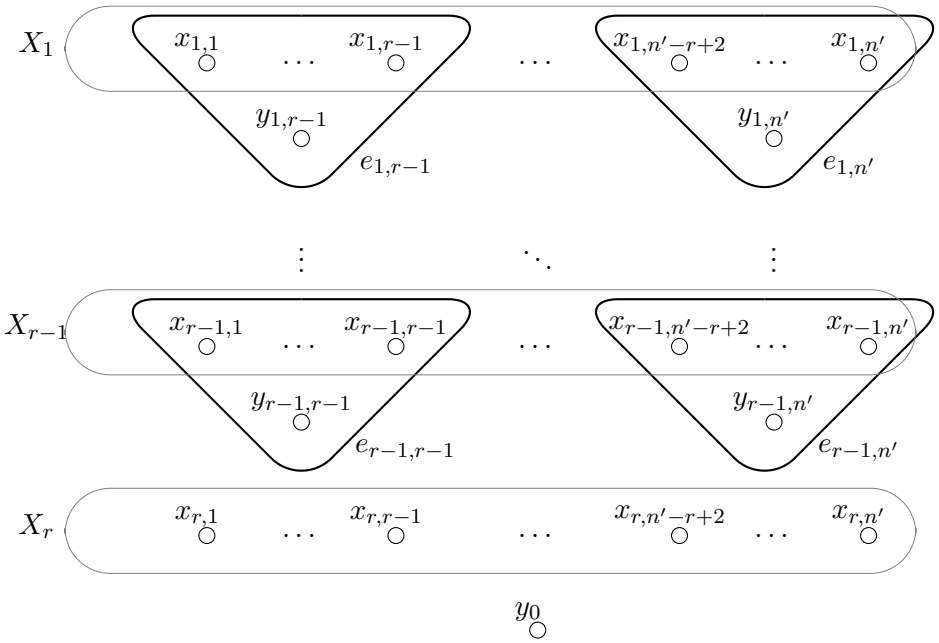

Figure 8.2: Illustration of the edge set $\mathcal{E}'$ in Lemma 147. Edges of $\mathcal{E}'$ are in black. The sets $X_1, \ldots X_r$ are in gray. Edges of $\mathcal{E}(G)$ (not depicted) contain one vertex from each of $X_1, \ldots X_r$.

Lemma 148. $(2(n-1)/3 + k)$Test-2-Cover *is NP-complete for* $k = 0$.

Proof. We will show NP-completeness of our problem by reduction from the P_3-packing problem, which asks, given a graph G with n vertices, n divisible by 3, whether G has $n/3$ vertex-disjoint copies of P_3. The problem is NP-complete [32]. Consider the graph H obtained from G by adding to it an isolated vertex x. We will view H as an instance of Test-2-Cover and we will prove that G is a Yes-instance of the P_3-packing problem if and only if H has a test cover of size $2n/3$. If G contains a set of edges F forming $n/3$ disjoint copies of P_3, then observe that F forms a test cover in H of size $2n/3$. Now assume that H has a test cover F of size $2n/3$. We will prove that G is a Yes-instance of the P_3-packing problem. Observe that every vertex of G has positive degree in $G[F]$. Let n' be the number of vertices of degree 1 in $G[F]$. We know that no vertex can have degree larger than 2, as otherwise F must have more than $2n/3$ edges. Since $|F| = 2n/3$, we have $n' + 2(n - n') = 2|F| = 4n/3$ and so $n' = 2n/3$. Finally, F cannot have any isolated edges as it is a test cover. Thus, F is a collection of $n/3$ vertex-disjoint copies of P_3. □

The next theorem follows from the straightforward fact that $(\frac{2(n-1)}{r+1} + k)$Test-r-Cover is in para-NP and the previous two lemmas.

Theorem 149. $(\frac{2(n-1)}{r+1} + k)$ Test-r-Cover *is para-NP-complete for each fixed* $r \geq 2$.

Bibliography

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

[2] Ron Aharoni and Nathan Linial. Minimal non-two-colorable hypergraphs and minimal unsatisfiable formulas. *Journal of Combinatorial Theory, Series A*, 43(2):196–204, 1986.

[3] Noga Alon, Gregory Gutin, Eun Kim, Stefan Szeider, and Anders Yeo. Solving Max-r-Sat above a tight lower bound. *Algorithmica*, 61:638–655, 2011.

[4] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995.

[5] Jørgen Bang-Jensen and Gregory Gutin. *Digraphs: theory, algorithms and applications*. Springer, 2009.

[6] Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. System Sci.*, 75(8):423–434, 2009.

[7] Hans L Bodlaender, Bart MP Jansen, and Stefan Kratsch. Cross-composition: A new technique for kernelization lower bounds. *arXiv preprint arXiv:1011.4224*, 2010.

[8] Hans L Bodlaender, Stéphan Thomassé, and Anders Yeo. Kernel bounds for disjoint cycles and disjoint paths. *Theoretical Computer Science*, 412(35):4570–4578, 2011.

[9] Béla Bollobás. *Modern graph theory*, volume 184. Springer Verlag, 1998.

[10] Hans Kleine Büning. On subclasses of minimal unsatisfiable formulas. *Discrete Applied Mathematics*, 107(1):83–98, 2000.

[11] Jianer Chen, Yang Liu, Songjian Lu, Barry O'sullivan, and Igor Razgon. A fixed-parameter algorithm for the directed feedback vertex set problem. *Journal of the ACM (JACM)*, 55(5):21, 2008.

[12] Charles Chiang, Andrew B Kahng, Subarnarekha Sinha, Xu Xu, and Alexander Z Zelikovsky. Fast and efficient bright-field aapsm conflict detection and correction. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 26(1):115–126, 2007.

[13] Vašek Chvátal and Colin McDiarmid. Small transversals in hypergraphs. *Combinatorica*, 12(1):19–26, 1992.

[14] Robert Crowston, Michael Fellows, Gregory Gutin, Mark Jones, Frances Rosamond, Stéphan Thomassé, and Anders Yeo. Simultaneously satisfying linear equations over $\mathbb{F}_2$: Maxlin2 and max-r-lin2 parameterized above average. *arXiv preprint arXiv:1104.1135*, 2011.

[15] Robert Crowston, Gregory Gutin, Mark Jones, Eun Jung Kim, and Imre Z Ruzsa. Systems of linear equations over $\mathbb{F}_2$ and problems parameterized above average. In *Algorithm Theory-SWAT 2010*, pages 164–175. Springer, 2010.

[16] Robert Crowston, Gregory Gutin, Mark Jones, Venkatesh Raman, and Saket Saurabh. Parameterized complexity of maxSat above average. *Theoretical Computer Science*, 2013.

[17] Robert Crowston, Gregory Gutin, Mark Jones, and Anders Yeo. A new lower bound on the maximum number of satisfied clauses in max-sat and its algorithmic applications. *Algorithmica*, 64(1):56–68, 2012.

[18] Robert Crowston, Mark Jones, and Matthias Mnich. Max-cut parameterized above the edwards-erdős bound. In *Automata, Languages, and Programming*, pages 242–253. Springer, 2012.

[19] Marek Cygan, Marcin Pilipczuk, Michał Pilipczuk, and Jakub Onufry Wojtaszczyk. On multiway cut parameterized above lower bounds. In *Parameterized and Exact Computation*, pages 1–12. Springer, 2012.

[20] Bhaskar DasGupta, German Andres Enciso, Eduardo Sontag, and Yi Zhang. Algorithmic and complexity results for decompositions of biological networks into monotone subsystems. *Biosystems*, 90(1):161–178, 2007.

[21] Koen MJ De Bontridder, Bjarne V Halldórsson, Magnús M Halldórsson, Cor AJ Hurkens, Jan K Lenstra, R Ravi, and Leen Stougie. Approximation algorithms for the test cover problem. *Mathematical Programming*, 98(1-3):477–491, 2003.

[22] Frank Dehne, Mike Fellows, Frances Rosamond, and Peter Shaw. Greedy localization, iterative compression, and modeled crown reductions: New fpt techniques, an improved algorithm for set splitting, and a novel 2k kernelization for vertex cover. In *Parameterized and Exact Computation*, pages 271–280. Springer, 2004.

[23] Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Incompressibility through colors and ids. In *Automata, Languages and Programming*, pages 378–389. Springer, 2009.

[24] Rod Downey. A basic parameterized complexity primer. In *The Multivariate Algorithmic Revolution and Beyond*, pages 91–128. Springer, 2012.

[25] Rod G Downey and Michael R Fellows. *Fixed-parameter tractability and completeness*. Cornell University, Mathematical Sciences Institute, 1992.

[26] Rod G Downey and Michael Ralph Fellows. *Parameterized complexity*, volume 127. springer Heidelberg, 1999.

[27] CS Edwards. Some extremal properties of bipartite subgraphs. *Canad. J. Math*, 25(3):475–483, 1973.

[28] CS Edwards. An improved lower bound for the number of edges in a largest bipartite subgraph. In *Proc. 2nd Czechoslovak Symposium on Graph Theory, Prague*, pages 167–181, 1975.

[29] Herbert Fleischner, Oliver Kullmann, and Stefan Szeider. Polynomial-time recognition of minimal unsatisfiable formulas with fixed clause-variable difference. *Theoretical Computer Science*, 289(1):503–516, 2002.

[30] Jörg Flum and Martin Grohe. *Parameterized complexity theory*, volume 3. Springer Heidelberg, 2006.

[31] Fedor V Fomin, Daniel Lokshtanov, Venkatesh Raman, Saket Saurabh, and BV Rao. Faster algorithms for finding and counting subgraphs. *Journal of Computer and System Sciences*, 78(3):698–706, 2012.

[32] Michael R Garey and David S Johnson. *Computers and intractability*, volume 174. freeman New York, 1979.

[33] Nalân Gülpinar, Gregory Gutin, Gautam Mitra, and Alexey Zverovitch. Extracting pure network submatrices in linear programs using signed graphs. *Discrete Applied Mathematics*, 137(3):359–372, 2004.

[34] G Gutin, G Muciaccia, and A Yeo. (non-) existence of polynomial kernels for the test cover problem. *Information Processing Letters*, 2012.

[35] Gregory Gutin, Eun Jung Kim, Stefan Szeider, and Anders Yeo. A probabilistic approach to problems parameterized above or below tight bounds. *Journal of Computer and System Sciences*, 77(2):422–429, 2011.

[36] Gregory Gutin, Leo Van Iersel, Matthias Mnich, and Anders Yeo. Every ternary permutation constraint satisfaction problem parameterized above average has a kernel with a quadratic number of variables. *Journal of Computer and System Sciences*, 78(1):151–163, 2012.

[37] Gregory Gutin and Anders Yeo. Some parameterized problems on digraphs. *The Computer Journal*, 51(3):363–371, 2008.

[38] Gregory Gutin and Alexei Zverovitch. Extracting pure network sub-matrices in linear programs using signed graphs, part ii. *Communications in Dependability and Quality Management*, 6(1):58–65, 2003.

[39] Bjarni V Halldórsson, Magnús M Halldórsson, and R Ravi. On the approximability of the minimum test collection problem. In *AlgorithmsESA 2001*, pages 158–169. Springer, 2001.

[40] BV Halldórsson, JS Minden, and R Ravi. Pier: Protein identification by epitope recognition. *Currents in Computational Molecular Biology*, 2001:109–110, 2001.

[41] Frank Harary. On the notion of balance of a signed graph. *The Michigan Mathematical Journal*, 2(2):143–146, 1953.

[42] Godfrey H Hardy and Srinivasa Ramanujan. Asymptotic formulaæ in combinatory analysis. *Proceedings of the London Mathematical Society*, 2(1):75–115, 1918.

[43] Falk Hüffner, Nadja Betzler, and Rolf Niedermeier. Optimal edge deletions for signed graph balancing. In *Experimental Algorithms*, pages 297–310. Springer, 2007.

[44] James A. Jones and Mary Jean Harrold. Test-suite reduction and prioritization for modified condition/decision coverage. *Software Engineering, IEEE Transactions on*, 29(3):195–209, 2003.

[45] Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.

[46] Fabian Kuhn, Pascal Von Rickenbach, Roger Wattenhofer, Emo Welzl, and Aaron Zollinger. Interference in cellular networks: The minimum membership set cover problem. In *Computing and Combinatorics*, pages 188–198. Springer, 2005.

[47] Oliver Kullmann. An application of matroid theory to the sat problem. In *Computational Complexity, 2000. Proceedings. 15th Annual IEEE Conference on*, pages 116–124. IEEE, 2000.

[48] Oliver Kullmann. Lean clause-sets: Generalizations of minimally unsatisfiable clause-sets. *Discrete Applied Mathematics*, 130(2):209–249, 2003.

[49] Michael A Langston. Fixed-parameter tractability, a prehistory. In *The Multivariate Algorithmic Revolution and Beyond*, pages 3–16. Springer, 2012.

[50] Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Kernelization–preprocessing with a guarantee. In *The Multivariate Algorithmic Revolution and Beyond*, pages 129–161. Springer, 2012.

[51] Daniel Loksthanov, NS Narayanaswamy, Venkatesh Raman, MS Ramanujan, and Saket Saurabh. Faster parameterized algorithms using linear programming. *arXiv preprint arXiv:1203.0833*, 2012.

[52] L Lovász and MD Plummer. *Matching theory*, volume 121. North Holland, 1986.

[53] Meena Mahajan and Venkatesh Raman. Parameterizing above guaranteed values: MaxSat and maxCut. *Journal of Algorithms*, 31(2):335–354, 1999.

[54] Meena Mahajan, Venkatesh Raman, and Somnath Sikdar. Parameterizing above or below guaranteed values. *Journal of Computer and System Sciences*, 75(2):137–153, 2009.

[55] Matthias Mnich, Geevarghese Philip, Saket Saurabh, and Ondrej Suchy. Beyond Max-Cut: lambda-Extendible Properties Parameterized Above the Poljak-Turzik Bound. In Deepak D'Souza, Telikepalli Kavitha, and Jaikumar Radhakrishnan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2012)*, volume 18 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 412–423, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[56] Bernard ME Moret and Henry D Shapiro. On minimizing a set of tests. *SIAM Journal on Scientific and Statistical Computing*, 6(4):983–1003, 1985.

[57] NS Narayanaswamy, Venkatesh Raman, MS Ramanujan, Saket Saurabh, et al. LP can be a cure for parameterized problems. In *Symposium on Theoretical Aspects of Computer Science*, volume 14, pages 338–349, 2012.

[58] Rolf Niedermeier. Invitation to fixed-parameter algorithms. *Habilitationschrift, University of Tübingen*, 2002.

[59] Albert Nijenhuis and Herbert S Wilf. *Combinatorial Algorithms for Computers and Calculators: 2d Ed.* Academic Press, 1978.

[60] Christos H Papadimitriou and David Wolfe. The complexity of facets resolved. *Journal of Computer and System Sciences*, 37(1):2–13, 1988.

[61] Svatopluk Poljak and Daniel Turzik. A polynomial algorithm for constructing a large bipartite subgraph, with an application to a satisfiability problem. *Canadian Journal of Mathematics*, 34(3):519–524, 1982.

[62] Svatopluk Poljak and Daniel Turzík. A polynomial time heuristic for certain subgraph optimization problems with guaranteed worst case bound. *Discrete Mathematics*, 58(1):99–104, 1986.

[63] Venkatesh Raman and Saket Saurabh. Parameterized algorithms for feedback set problems and their duals in tournaments. *Theoretical Computer Science*, 351(3):446–458, 2006.

[64] DP Ruchkys and SW Song. A parallel approximation hitting set algorithm for gene expression analysis. In *Computer Architecture and High Performance Computing, 2002. Proceedings. 14th Symposium on*, pages 75–81. IEEE, 2002.

[65] Joel Spencer. Optimal ranking of tournaments. *Networks*, 1(2):135–138, 1971.

[66] Aravind Srinivasan. Improved approximations of packing and covering problems. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 268–276. ACM, 1995.

[67] Stefan Szeider. Minimal unsatisfiable formulas with bounded clause-variable difference are fixed-parameter tractable. In *Computing and Combinatorics*, pages 548–558. Springer, 2003.

[68] Stéphan Thomassé and Anders Yeo. Total domination of graphs and small transversals of hypergraphs. *Combinatorica*, 27(4):473–487, 2007.

[69] Zsolt Tuza. Covering all cliques of a graph. *Discrete Mathematics*, 86(1):117–126, 1990.

[70] Thomas Zaslavsky. A mathematical bibliography of signed and gain graphs and allied areas. *The Electronic Journal of Combinatorics*, 1000:DS8–Sep, 2012.