

# A Survey on Approximation in Parameterized Complexity: Hardness and Algorithms

Andreas Emil Feldmann\*  
Charles University  
feldmann.a.e@gmail.com

Karthik C. S.†  
Tel Aviv University  
karthik0112358@gmail.com

Euiwoong Lee  
New York University  
euiwoong@cims.nyu.edu

Pasin Manurangsi  
Google Research  
pasin@google.com

5<sup>th</sup> June 2020

## Abstract

Parameterization and approximation are two popular ways of coping with NP-hard problems. More recently, the two have also been combined to derive many interesting results. We survey developments in the area both from the algorithmic and hardness perspectives, with emphasis on new techniques and potential future research directions.

---

\*Supported by the Czech Science Foundation GAČR (grant #19-27871X), and by the Center for Foundations of Modern Computer Science (Charles Univ. project UNCE/SCI/004).

†Supported by ERC-CoG grant 772839, the Israel Science Foundation (grant number 552/16), and from the Len Blavatnik and the Blavatnik Family foundation.

# 1 Introduction

In their seminal papers of the mid 1960s, Cobham [Cob64] and Edmonds [Edm65] independently phrased what is now known as the Cobham-Edmonds thesis. It states that an optimization problem is *feasibly solvable* if it admits an algorithm with the following two properties:

1. **Accuracy:** the algorithm should always compute the best possible (optimum) solution.
2. **Efficiency:** the runtime of the algorithm should be polynomial in the input size  $n$ .

Shortly after the Cobham-Edmonds thesis was formulated, the development of the theory of NP-hardness and reducibility identified a whole plethora of problems that are seemingly intractable, i.e., for which algorithms with the above two properties do not seem to exist. Even though the reasons for this phenomenon remain elusive up to this day, this has not hindered the development of algorithms for such problems. To obtain an algorithm for an NP-hard problem, at least one of the two properties demanded by the Cobham-Edmonds thesis needs to be relaxed. Ideally, the properties are relaxed as little as possible, in order to stay close to the notion of feasible solvability suggested by the thesis.

A very common approach is to relax the accuracy condition, which means aiming for *approximation algorithms* [Vaz01; WS11]. The idea here is to use only polynomial time to compute an  $\alpha$ -approximation, i.e., a solution that is at most a factor  $\alpha$  times worse than the optimum solution obtainable for the given input instance. Such an algorithm may also be randomized, i.e., there is either a high probability that the output is an  $\alpha$ -approximation, or the runtime is polynomial in expectation.

In a different direction, several relaxations of the efficiency condition have also been proposed. Popular among these is the notion of *parameterized algorithms* [Cyg+15; DF13]. Here the input comes together with some parameter  $k \in \mathbb{N}$ , which describes some property of the input and can be expected to be small in typical applications. The idea is to isolate the seemingly necessary exponential runtime of NP-hard problems to the parameter, while the runtime dependence on the input size  $n$  remains polynomial. In particular, the algorithm should compute the optimum solution in  $f(k)n^{O(1)}$  time, for some computable function  $f : \mathbb{N} \rightarrow \mathbb{N}$  independent of the input size  $n$ . If such an algorithm exists for a problem it is *fixed-parameter tractable (FPT)*, and the algorithm is correspondingly referred to as an *FPT algorithm*.

Approximation and FPT algorithms have been studied extensively for the past few decades, and this has led to a rich literature on algorithmic techniques and deep links to other research fields within mathematics. However, in this process the limitations of these approaches have also become apparent. Some NP-hard problems can fairly be considered to be feasibly solvable in the respective regimes, as they admit polynomial-time algorithms with rather small approximation factors, or can be shown to be solvable optimally with only a fairly small exponential runtime overhead due to the parameter. But many problems can also be shown not to admit any reasonable algorithms in either of these regimes, assuming some standard complexity assumptions. Thus considering only approximation and FPT algorithms, as has been mostly done in the past, we are seemingly stuck in a swamp of problems for which we have substantial evidence that they cannot be feasibly solved.

To find a way out of this dilemma, an obvious possibility is to lift both the accuracy and the efficiency requirements of the Cobham-Edmonds thesis. In this way we obtain a *parameterized  $\alpha$ -approximation algorithm*, which computes an  $\alpha$ -approximation in  $f(k)n^{O(1)}$  time for some computable function  $f$ , given an input of size  $n$  with parameter  $k$ . The study of such algorithms had been suggested dating back to the early days of parameterized complexity (cf. [CC97; DF13;

FG06; Mar08]), and we refer the readers to an excellent survey of Marx [Mar08] for discussions on the earlier results in the area.

Recently this approach has received some increased interest, with many new results obtained in the past few year both in terms of algorithms and hardness of approximation. The aim of this survey is to give an overview on some of these newer results. We would like to caution the readers that the goal of this survey is not to compile all known results in the field but rather to give examples that demonstrate the flavor of questions studied, techniques used to obtain them, and some potential future research directions. Finally, we remark that on the broader theme of approximation in P, there was an excellent survey recently made available by Rubinfeld and Williams [RW19] focusing on the approximability of popular problems in P which admit simple quadratic/cubic algorithms.

## Organization of the Survey

The main body of the survey is organized into two sections: one on FPT hardness of approximation (Section 3) and the other on FPT approximation algorithms (Section 4). Before these two main sections, we list some notations and preliminaries in Section 2. Finally, in Section 5, we highlight some open questions and future directions (although a large list of open problems have also been detailed throughout Sections 3 and 4).

## 2 Preliminaries

In this section, we review several notions that will appear regularly throughout the survey. However, we skip including definitions of basic concepts such as W-hardness, para-NP-hardness, APX-hardness, and so forth; the interested reader may refer to [Cyg+15; DF13; Vaz01; WS11] for these definitions.

**Parameterized approximation algorithms.** We briefly specify the different types of algorithms we will consider. As already defined in the introduction, an FPT algorithm computes the optimum solution in  $f(k)n^{O(1)}$  time for some parameter  $k$  and computable function  $f : \mathbb{N} \rightarrow \mathbb{N}$  on inputs of size  $n$ . The common choices of parameters are the standard parameters based on solution size, structural parameters, guarantee parameters, and dual parameters.

An algorithm that computes the optimum solution in  $f(k)n^{g(k)}$  time for some parameter  $k$  and computable functions  $f, g : \mathbb{N} \rightarrow \mathbb{N}$ , is called a *slice-wise polynomial (XP) algorithm*. If the parameter is the approximation factor, i.e., the algorithm computes a  $(1 + \epsilon)$ -approximation in  $f(\epsilon)n^{g(\epsilon)}$  time, then it is called a *polynomial-time approximation scheme (PTAS)*. The latter type of algorithm has been studied *avant la lettre* for quite a while. This is also true for the corresponding FPT algorithm, which computes a  $(1 + \epsilon)$ -approximation in  $f(\epsilon)n^{O(1)}$  time, and is referred to as an *efficient polynomial-time approximation scheme (EPTAS)*. Note that if the standard parameterization of an optimization problem is W[1]-hard, then the optimization problem does not have an EPTAS (unless  $\text{FPT} = \text{W}[1]$ ) [CT97].

Some interesting links between these algorithms, traditionally studied from the perspective of polynomial-time approximation algorithms, and parameterized complexity have been uncovered more recently [CC93; Che+07; CT97; Cyg+16; GKK11; Kra12; Mar08].

As also mentioned in the introduction, a *parameterized  $\alpha$ -approximation algorithm* computes

an  $\alpha$ -approximation in  $f(k)n^{O(1)}$  time for some parameter  $k$  on inputs of size  $n$ . If  $\alpha$  can be set to  $1 + \varepsilon$  for any  $\varepsilon > 0$  and the runtime is  $f(k, \varepsilon)n^{g(\varepsilon)}$ , then we obtain a *parameterized approximation scheme (PAS)* for parameter  $k$ . Note that this runtime is only truly FPT if we assume that  $\varepsilon$  is constant. If we forbid this and consider  $\varepsilon$  as a parameter as well, i.e., the runtime should be of the form  $f(k, \varepsilon)n^{O(1)}$ , then we obtain an *efficient parameterized approximation scheme (EPAS)*.

**Kernelization.** A further topic closely related to FPT algorithms is kernelization. Here the idea is that an instance is efficiently pre-processed by removing the “easy parts” so that only the NP-hard core of the instance remains. More concretely, a *kernelization algorithm* takes an instance  $I$  and a parameter  $k$  of some problem and computes a new instance  $I'$  with parameter  $k'$  of the same problem. The runtime of this algorithm is polynomial in the size of the input instance  $I$  and  $k$ , while the size of the output  $I'$  and  $k'$  is bounded as a function of the input parameter  $k$ . For optimization problems it should also be the case that any optimum solution to  $I'$  can be converted to an optimum solution of  $I$  in polynomial time. The new instance  $I'$  is called the *kernel* of  $I$  (for parameter  $k$ ). A fundamental result in fixed-parameter tractability is that an (optimization) problem parameterized by  $k$  is FPT if and only if it admits a kernelization algorithm for the same parameter [Cai+97]. However the size of the guaranteed kernel will in general be exponential (or worse) in the input parameter. Therefore an interesting question is whether an NP-hard problem admits small kernels of polynomial size. This can be interpreted as meaning that the problem has a very efficient pre-processing algorithm, which can be used prior to solving the kernel. It also gives an additional dimension to the parameterized complexity landscape.

Kernelization has played a fundamental role in the development of FPT algorithms, where often a pre-processing step is used to simplify the structure of the input instance. It is therefore only natural to consider such pre-processing algorithms for parameterized approximation algorithms as well. Lokshantov et al. [Lok+17] define an  *$\alpha$ -approximate kernelization algorithm*, which computes a kernel  $I'$  such that any  $\beta$ -approximation in  $I'$  can be converted into an  $\alpha\beta$ -approximation to the input instance  $I$  in polynomial time. Again the size of  $I'$  and  $k'$  need to be bounded as a function of the input parameter  $k$ , and the algorithm needs to run in polynomial time. The instance  $I'$  is now called an  *$\alpha$ -approximate kernel*. Analogous to exact kernels, any problem has a parameterized  $\alpha$ -approximation algorithm if and only if it admits an  $\alpha$ -approximate kernel for the same parameter [Lok+17], which however might be of exponential size in the parameter.

An  $\alpha$ -approximate kernelization algorithm that computes a polynomial-sized kernel, and for which we may set  $\alpha$  to  $1 + \varepsilon$  for any  $\varepsilon > 0$ , is called a *polynomial-sized approximate kernelization scheme (PSAKS)*. In this case  $\varepsilon$  is necessarily considered to be a constant, since any kernelization algorithm needs to run in polynomial time.

We remark here that apart from  $\alpha$ -approximate kernels, there is another common work-around for problems with no polynomial kernels, captured using the notion of Turing kernels. There is also a lower bound framework for Turing kernels [Her+15], and the question of approximate kernels for problems that do not even admit Turing kernels is fairly natural to ask. However we skip this discussion for the sake of brevity.

Finally, note that in literature there is another notion of approximate kernels called  $\alpha$ -fidelity kernelization [Fel+12] which is different from the one mentioned above. Essentially, an  $\alpha$ -fidelity kernel is a polynomial time preprocessing procedure such that an optimal solution to the reduced instance translates to an  $\alpha$ -approximate solution to the original. This definition

allows a loss of precision in the preprocessing step, but demands that the reduced instance has to be solved to optimality. See [Lok+17] for a detailed discussion on the differences between the two approximate kernel notions.

**Complexity-Theoretic Hypotheses.** We assume that the readers have basic knowledge of (classic) parameterized complexity theory, including the W-hierarchy, the exponential time hypothesis (ETH), and the strong exponential time hypothesis (SETH). The reader may choose to recapitulate these definitions by referring to [Cyg+15, Sections 13 and 14].

We will additionally discuss two hypotheses that may not be standard to the community. First is the *Gap Exponential Time Hypothesis (Gap-ETH)*, which is a strengthening of ETH. Roughly speaking, it states that even the approximate version of 3SAT cannot be solved in subexponential time; a more formal statement of Gap-ETH can be found in [Hypothesis 3.14](#). Another hypothesis we will discuss is the *Parameterized Inapproximability Hypothesis (PIH)*, which states that the multicolored version of the DENSEST  $k$ -SUBGRAPH is hard to approximate in FPT time. Once again, we do not define PIH formally here; please refer to [Hypothesis 3.11](#) for a formal statement.

### 3 FPT Hardness of Approximation

In this section, we focus on showing barriers against obtaining good parameterized approximation algorithms. The analogous field of study in the non-parameterized (NP-hardness) regime is the theory of *hardness of approximation*. The celebrated PCP Theorem [Aro+98; AS92] and numerous subsequent works have developed a rich set of tools that allowed researchers to show tight inapproximability results for many fundamental problems. In the context of parameterized approximation, the field is still in the nascent stage. Nonetheless, there have been quite a few tools that have already been developed, which are discussed in the subsequent subsections.

We divide this section into two parts. In [Section 3.1](#), we discuss the results and techniques in the area of hardness of parameterized approximation under the standard assumption of  $W[1] \neq \text{FPT}$ . In [Section 3.2](#), we discuss results and techniques in hardness of parameterized approximation under less standard assumptions such as the Gap Exponential Time Hypothesis, where the gap is inherent in the assumption, and the challenge is to construct gap-preserving reductions.

#### 3.1 $W[1]$ -Hardness of Gap Problems

In this subsection, we discuss  $W[1]$ -hardness of approximation of a few fundamental problems. In particular, we discuss about the parameterized inapproximability (i.e.,  $W[1]$ -hard to even approximate) of the DOMINATING SET problem, the (ONE-SIDED) BICLIQUE problem, the EVEN SET problem, the SHORTEST VECTOR problem, and the STEINER ORIENTATION problem. We emphasize here that the main difficulty that is addressed in this subsection is *gap generation*, i.e., we focus on how to start from a hard problem (with no gap), say  $k$ -CLIQUE (which is the canonical  $W[1]$ -complete problem), and reduce it to one of the aforementioned problems, while generating a non-trivial gap in the process.

### 3.1.1 Parameterized Intractability of Biclique and Applications to Parameterized Inapproximability

In this subsection, we will discuss the parametrized inapproximability of the one-sided biclique problem, and show how both that result and its proof technique lead to more inapproximability results.

We begin our discussion by formally stating the  $k$ -BICLIQUE problem where we are given as input a graph  $G$  and an integer  $k$ , and the goal is to determine whether  $G$  contains a complete bipartite subgraph with  $k$  vertices on each side. The complexity of  $k$ -BICLIQUE was a long standing open problem and was resolved only recently by Lin [Lin18] where he showed that it is  $W[1]$ -hard. In fact, he showed a much stronger result and this shall be the focus of attention in this subsection.

**Theorem 3.1** ([Lin18]). *Given a bipartite graph  $G(L \dot{\cup} R, E)$  and  $k \in \mathbb{N}$  as input, it is  $W[1]$ -hard to distinguish between the following two cases:*

- **Completeness:** *There are  $k$  vertices in  $L$  with at least  $n^{\Theta(\frac{1}{k})}$  common neighbors in  $R$ ;*
- **Soundness:** *Any  $k$  vertices in  $L$  have at most  $(k + 6)!$  common neighbors in  $R$ .*

We shall refer to the gap problem in the above theorem as the ONE-SIDED  $k$ -BICLIQUE problem. To prove the above result, Lin introduced a technique which we shall refer to as *Gadget Composition*. The gadget composition technique has found more applications since [Lin18]. We provide below a failed approach (given in [Lin18]) to prove the above theorem; nonetheless it gives us good insight into how the gadget composition technique works.

Suppose we can construct a set family  $\mathcal{T} = \{S_1, S_2, \dots, S_n\}$  of subsets of  $[n]$  for some integers  $k, n$  and  $h > \ell$  (for example,  $h = n^{1/k}$  and  $\ell = (k + 1)!$ ) such that:

**Property 1:** Any  $k + 1$  distinct subsets in  $\mathcal{T}$  have intersection size at most  $\ell$ ;

**Property 2:** Any  $k$  distinct subsets in  $\mathcal{T}$  have intersection size at least  $h$ .

Then we can combine  $\mathcal{T}$  with an instance of  $k$ -CLIQUE to obtain a gap instance of ONE-SIDED  $k$ -BICLIQUE as follows. Given a graph  $G$  and parameter  $k$  with  $V(G) \subseteq [n]$ , we construct our instance of ONE-SIDED  $k$ -BICLIQUE, say  $H(L \dot{\cup} R, E(H))$  by setting  $L := E(G)$  and  $R := [n]$ , where for any  $(v_i, v_j) \in L$  and  $v \in [n]$ , we have that  $((v_i, v_j), v) \in E(H)$  if and only if  $v \in S_i \cap S_j$ . Let  $s := k(k - 1)/2$ . It is easy to check that if  $G$  has a  $k$ -vertex clique, say  $\{v_1^*, \dots, v_k^*\}$  is a clique in  $G$ , then Property 2 implies that  $|\Delta := \bigcap_{i \in [k]} S_{v_i^*}| \geq h$ . It follows that the set of  $s$  vertices in  $L$  given by  $\{(v_i^*, v_j^*) : \text{for all } \{i, j\} \in \binom{[k]}{2}\}$  are neighbors of every vertex in  $\Delta \subseteq R$ . On the other hand, if  $G$  contains no  $k$ -vertex clique, then any  $s$  distinct vertices in  $L$  (i.e.,  $s$  edges in  $G$ ) must have at least  $k + 1$  vertices in  $G$  as their end points. Say  $V'$  was the set of all vertices contained the  $s$  edges. By Property 1, we know that  $|\Delta' := \bigcap_{v \in V'} S_v| \leq \ell$ , and thus any  $s$  distinct vertices in  $L$  have at most  $\ell$  common neighbors in  $R$ .

It is indeed very surprising that this technique can yield non-trivial inapproximability results, as the gap is essentially produced from the gadget and is oblivious to the input! This also stands in stark contrast to the PCP theorem and hardness of approximation results in NP, where all known results were obtained by global transformations on the input. The key difference between the parameterized and NP worlds is the notion of locality. For example, consider the  $k$ -CLIQUE problem, if a graph does not have a clique of size  $k$ , then given any  $k$  vertices, a random vertex pair in these  $k$  vertices does not have an edge with probability

at least  $1/k^2$ . It is philosophically possible to compose the input graph with a simple error correcting code to amplify this probability to a constant, as we are allowed to blowup the input size by any function of  $k$ . In contrast, when  $k$  is not fixed, like in the NP world,  $k$  is of the same magnitude as the input size, and thus we are only allowed to blow up the input size by  $\text{poly}(n)$  factor. Nonetheless, we have to point out that the gadgets typically needed to make the gadget composition technique work, must be extremely rich in combinatorial structure (and are typically constructed from random objects or algebraic objects), and were previously studied extensively in the area of extremal combinatorics.

Returning to the reduction above from  $k$ -CLIQUE to ONE-SIDED  $k$ -BICLIQUE, it turns out that we do not know how to construct the set system  $\mathcal{T}$ , and hence the reduction does not pan out. Nonetheless Lin constructed a variant of  $\mathcal{T}$ , where Property 2 was more refined and the reduction from  $k$ -CLIQUE to ONE-SIDED  $k$ -BICLIQUE, went through with slightly more effort.

Before we move on to discussing some applications of Theorem 3.1 and the gadget composition technique, we remark about known stronger time lower bound for ONE-SIDED  $k$ -BICLIQUE under stronger running time hypotheses. Lin [Lin18] showed a lower bound of  $n^{\Omega(\sqrt{k})}$  for ONE-SIDED  $k$ -BICLIQUE assuming ETH. We wonder if this can be further improved.

**Open Question 3.2** (Lower bound of ONE-SIDED  $k$ -BICLIQUE under ETH and SETH). *Can the running time lower bound on ONE-SIDED  $k$ -BICLIQUE be improved to  $n^{\Omega(k)}$  under ETH? Can it be improved to  $n^{k-o(1)}$  under SETH?*

We remark that a direction to address the above question was detailed in [KM19]. While on the topic of the  $k$ -BICLIQUE problem, it is worth noting that the lower bound of  $n^{\Omega(\sqrt{k})}$  for ONE-SIDED  $k$ -BICLIQUE assuming ETH yields a running time lower bound of  $n^{\Omega(\frac{\log k}{\log \log k})}$  for the  $k$ -BICLIQUE problem (due to the soundness parameters in Theorem 3.1). However, assuming *randomized* ETH, the running time lower bound for the  $k$ -BICLIQUE problem can be improved to  $n^{\Omega(\sqrt{k})}$  [Lin18]. Can this improved running time lower bound be obtained just under (deterministic) ETH? Finally, we remark that we shall discuss about the hardness of approximation of the  $k$ -BICLIQUE problem in Section 3.2.3.

**Inapproximability of  $k$ -DOMINATING SET via Gadget Composition.** We shall discuss about the inapproximability of  $k$ -DOMINATING SET in detail in the next subsection. We would like to simply highlight here how the above framework was used by Chen and Lin [CL19a] and Lin [Lin19] to obtain inapproximability results for  $k$ -DOMINATING SET.

In [CL19a], the authors starting from Theorem 3.1, obtain the W[1]-hardness of approximating  $k$ -DOMINATING SET to a factor of almost two. Then they amplify the gap to any constant by using a specialized graph product.

We now turn our attention to a recent result of Lin [Lin19] who provided strong inapproximability result for  $k$ -DOMINATING SET (we refer the reader to Section 3.1.2 to obtain the context for this result). Lin's proof of inapproximability of  $k$ -DOMINATING SET is a one-step reduction from an instance of  $k$ -SET COVER on a universe of size  $O(\log n)$  (where  $n$  is the number of subsets given in the collection) to an instance of  $k$ -SET COVER on a universe of size  $\text{poly}(n)$  with a gap of  $\left(\frac{\log n}{\log \log n}\right)^{1/k}$ . Lin then uses this gap producing self-reduction to provide running time lower bounds (under different time hypotheses) for approximating  $k$ -set cover to a factor of  $(1 - o(1)) \cdot \left(\frac{\log n}{\log \log n}\right)^{1/k}$ . Recall that  $k$ -DOMINATING SET is essentially<sup>1</sup> equivalent to  $k$ -SET

<sup>1</sup>Recall that there is a pair of polynomial-time  $L$ -reductions between the minimum dominating set problem and

COVER.

Elaborating, Lin designs a gadget by combining the hypercube partition gadget of Feige [Fei98] with a derandomizing combinatorial object called *universal set*, to obtain a gap gadget, and then combines the gap gadget with the input  $k$ -SET COVER instance (on small universe but with no gap) to obtain a gap  $k$ -SET COVER instance. This is another success story of the gadget composition technique.

Finally, we remark that Lai [Lai19] recently extended Lin's inapproximability results for dominating set (using the same proof framework) to rule out constant-depth circuits of size  $f(k)n^{o(\sqrt{k})}$  for any computable function  $f$ .

**Even Set.** A recent success story of Theorem 3.1 is its application to resolve a long standing open problem called  $k$ -MINIMUM DISTANCE PROBLEM (also referred to as  $k$ -EVEN SET), where we are given as input a generator matrix  $\mathbf{A} \in \mathbb{F}_2^{n \times m}$  of a binary linear code and an integer  $k$ , and the goal is to determine whether the code has distance at most  $k$ . Recall that the distance of a linear code is  $\min_{\vec{0} \neq \mathbf{x} \in \mathbb{F}_2^m} \|\mathbf{A}\mathbf{x}\|_0$  where  $\|\cdot\|_0$  denote the 0-norm (aka the Hamming norm).

In [Bha+19], the authors showed that  $k$ -EVEN SET is  $W[1]$ -hard under randomized reductions. The result was obtained by starting from the inapproximability result stated in Theorem 3.1 followed by a series of intricate reductions. In fact they proved the following stronger inapproximability result.

**Theorem 3.3 ([Bha+19]).** *For any  $\gamma \geq 1$ , given input  $(\mathbf{A}, k) \in \mathbb{F}^{n \times m} \times \mathbb{N}$ , it is  $W[1]$ -hard (under randomized reductions) to distinguish between*

- **Completeness:** *Distance of the code generated by  $\mathbf{A}$  is at most  $k$ , and,*
- **Soundness:** *Distance of the code generated by  $\mathbf{A}$  is more than  $\gamma \cdot k$ .*

We emphasize that even to obtain the  $W[1]$ -hardness of  $k$ -EVEN SET (with no gap), they needed to start from the gap problem given in Theorem 3.1.

The proof of the above theorem proceeds by first showing FPT hardness of approximation of the non-homogeneous variant of  $k$ -MINIMUM DISTANCE PROBLEM called the  $k$ -NEAREST CODEWORD PROBLEM. In  $k$ -NEAREST CODEWORD PROBLEM, we are given a target vector  $\mathbf{y}$  (in  $\mathbb{F}^n$ ) in addition to  $(\mathbf{A}, k)$ , and the goal is to find whether there is any  $\mathbf{x}$  (in  $\mathbb{F}^m$ ) such that the Hamming norm of  $\mathbf{A}\mathbf{x} - \mathbf{y}$  is at most  $k$ . As an intermediate step of the proof of Theorem 3.3, they showed that  $k$ -NEAREST CODEWORD PROBLEM is  $W[1]$ -hard to approximate to any constant factor.

An important intermediate problem which was studied by [Bha+19] to prove the inapproximability of  $k$ -NEAREST CODEWORD PROBLEM, was the  $k$ -LINEAR DEPENDENT SET problem where given a set  $\mathbf{A}$  of  $n$  vectors over a finite field  $\mathbb{F}_q$  and an integer  $k$ , the goal is to decide if there are  $k$  vectors in  $\mathbf{A}$  that are linearly dependent. They ruled out constant factor approximation algorithms for this problem running in FPT time. Summarizing, the high level proof overview of Theorem 3.3 follows by reducing ONE-SIDED  $k$ -BICLIQUE to  $k$ -LINEAR DEPENDENT SET, which is then reduced to  $k$ -NEAREST CODEWORD PROBLEM, followed by a final randomized reduction to  $k$ -MINIMUM DISTANCE PROBLEM.

Finally, we note that there is no reason to define  $k$ -MINIMUM DISTANCE PROBLEM only for binary code, but can instead be defined over larger fields as well. It turns out that [Bha+19]

---

the set cover problem [Kan92].



cannot rule out FPT algorithms for  $k$ -MINIMUM DISTANCE PROBLEM over  $\mathbb{F}_p$  with  $p > 2$ , when  $p$  is fixed and is not part on the input. Thus we have the open problem.

**Open Question 3.4.** *Is it W[1]-hard to decide  $k$ -MINIMUM DISTANCE PROBLEM over  $\mathbb{F}_p$  with  $p > 2$ , when  $p$  is fixed and is not part on the input?*

**Shortest Vector Problem.** [Theorem 3.1](#) (or more precisely the constant inapproximability of  $k$ -LINEAR DEPENDENT SET stated above) was also used to resolve the complexity of the parameterized  $k$ -SHORTEST VECTOR PROBLEM in lattices, where the input (in the  $\ell_p$  norm) is an integer  $k \in \mathbb{N}$  and a matrix  $\mathbf{A} \in \mathbb{Z}^{n \times m}$  representing the basis of a lattice, and we want to determine whether the shortest (non-zero) vector in the lattice has length at most  $k$ , i.e., whether  $\min_{\vec{0} \neq \mathbf{x} \in \mathbb{Z}^m} \|\mathbf{A}\mathbf{x}\|_p \leq k$ . Again,  $k$  is the parameter of the problem. It should also be noted here that (as in [\[Dow+99\]](#)), we require the basis of the lattice to be integer valued, which is sometimes not enforced in literature (e.g. [\[Ajt98; EB81\]](#)). This is because, if  $\mathbf{A}$  is allowed to be any matrix in  $\mathbb{R}^{n \times m}$ , then parameterization is meaningless because we can simply scale  $\mathbf{A}$  down by a large multiplicative factor.

In [\[Bha+19\]](#), the authors showed that  $k$ -SHORTEST VECTOR PROBLEM is W[1]-hard under randomized reductions. In fact they proved the following stronger inapproximability result.

**Theorem 3.5** ([\[Bha+19\]](#)). *For any  $p > 1$ , there exists a constant  $\gamma_p > 1$  such that given input  $(\mathbf{A}, k) \in \mathbb{Z}^{n \times m} \times \mathbb{N}$ , it is W[1]-hard (under randomized reductions) to distinguish between*

- **Completeness:** *The  $\ell_p$  norm of the shortest vector of the lattice generated by  $\mathbf{A}$  is  $\leq k$ , and,*
- **Soundness:** *The  $\ell_p$  norm of the shortest vector of the lattice generated by  $\mathbf{A}$  is  $> \gamma_p \cdot k$ .*

Notice that [Theorem 3.3](#) rules out FPT approximation algorithms with *any* constant approximation ratio for  $k$ -EVEN SET. In contrast, the above result only prove FPT inapproximability with *some* constant ratio for  $k$ -SHORTEST VECTOR PROBLEM in  $\ell_p$  norm for  $p > 1$ . Also, as with  $k$ -EVEN SET, even to prove the W[1]-hardness of  $k$ -SHORTEST VECTOR PROBLEM (with no gap), they needed to start from the gap problem given in [Theorem 3.1](#).

The proof of the above theorem proceeds by first showing FPT hardness of approximation of the non-homogeneous variant of  $k$ -SHORTEST VECTOR PROBLEM called the  $k$ -NEAREST VECTOR PROBLEM. In  $k$ -NEAREST VECTOR PROBLEM, we are given a target vector  $\mathbf{y}$  (in  $\mathbb{Z}^n$ ) in addition to  $(\mathbf{A}, k)$ , and the goal is to find whether there is any  $\mathbf{x}$  (in  $\mathbb{Z}^m$ ) such that the  $\ell_p$  norm of  $\mathbf{A}\mathbf{x} - \mathbf{y}$  is at most  $k$ . As an intermediate step of the proof of [Theorem 3.3](#), they showed that  $k$ -NEAREST VECTOR PROBLEM is W[1]-hard to approximate to any constant factor. Summarizing, the high level proof overview of [Theorem 3.5](#) follows by reducing ONE-SIDED  $k$ -BICLIQUE to  $k$ -LINEAR DEPENDENT SET, which is then reduced to  $k$ -NEAREST VECTOR PROBLEM, followed by a final randomized reduction to  $k$ -SHORTEST VECTOR PROBLEM.

An immediate open question left open from their work is whether [Theorem 3.5](#) can be extended to  $k$ -SHORTEST VECTOR PROBLEM in the  $\ell_1$  norm. In other words,

**Open Question 3.6** (Approximation of  $k$ -SHORTEST VECTOR PROBLEM in  $\ell_1$  norm). *Is  $k$ -SHORTEST VECTOR PROBLEM in the  $\ell_1$  norm in FPT?*

### 3.1.2 Parameterized Inapproximability of Dominating Set

In the  $k$ -DOMINATING SET problem we are given an integer  $k$  and a graph  $G$  on  $n$  vertices as input, and the goal is to determine if there is a dominating set of size at most  $k$ . It was a long

standing open question to design an algorithm which runs in time  $T(k) \cdot \text{poly}(n)$  (i.e., FPT-time), that would find a dominating set of size at most  $F(k) \cdot k$  whenever the graph  $G$  has a dominating set of size  $k$ , for any computable functions  $T$  and  $F$ .

The first non-trivial progress on this problem was by Chen and Lin [CL19a] who ruled out the existence of such algorithms (under  $W[1] \neq \text{FPT}$ ) for all constant functions  $F$  (i.e.,  $F(n) = c$ , where  $c$  is any universal constant). We discussed their proof technique in the previous subsection. A couple of years later, Karthik C. S. et al. [KLM19] completely settled the question, by ruling out the existence of such an algorithm (under  $W[1] \neq \text{FPT}$ ) for any computable function  $F$ . Thus,  $k$ -DOMINATING SET was shown to be *totally inapproximable*. We elaborate on their proof below.

**Theorem 3.7 ([KLM19]).** *Let  $F : \mathbb{N} \rightarrow \mathbb{N}$  be any computable function. Given an instance  $(G, k)$  of  $k$ -DOMINATING SET as input, it is  $W[1]$ -hard to distinguish between the following two cases:*

- **Completeness:**  $G$  has a dominating set of size  $k$ .
- **Soundness:** Every dominating set of  $G$  is of size at least  $F(k) \cdot k$ .

The overall proof follows by reducing  $k$ -MULTICOLOR CLIQUE to the gap  $k$ -DOMINATING SET with parameters as given in the theorem statement. In the  $k$ -MULTICOLOR CLIQUE problem, we are given an integer  $k$  and a graph  $G$  on vertex set  $V := V_1 \dot{\cup} V_2 \dot{\cup} \dots \dot{\cup} V_k$  as input, where each  $V_i$  is an independent set of cardinality  $n$ , and the goal is to determine if there is a clique of size  $k$  in  $G$ . Following a straightforward reduction from the  $k$ -CLIQUE problem, it is fairly easy to see that  $k$ -MULTICOLOR CLIQUE is  $W[1]$ -hard.

The reduction from  $k$ -MULTICOLOR CLIQUE to the gap  $k$ -DOMINATING SET proceeds in two steps. In the first step we reduce  $k$ -MULTICOLOR CLIQUE to  $k$ -GAP CSP. This is the step where we generate the gap. In the second step, we reduce  $k$ -GAP CSP to gap  $k$ -DOMINATING SET. This step is fairly standard and mimics ideas from Feige’s proof of the NP-hardness of approximating the MAX COVERAGE problem [Fei98].

Before we proceed with the details of the above two steps, let us introduce a small technical tool from coding theory that we would need. We need codes known in literature as *good codes*, these are binary error correcting codes whose rate and relative distances are both constants bounded away from 0 (see [Gol08, Appendix E.1.2.5] for definitions). The reader may think of them as follows: for every  $\ell \in \mathbb{N}$ , we say that  $C_\ell \subseteq \{0, 1\}^\ell$  is a good code if (i)  $|C_\ell| = 2^{\rho\ell}$ , for some universal constant  $\rho > 0$ , (ii) for any distinct  $c, c' \in C_\ell$  we have that  $c$  and  $c'$  have different values on at least  $\delta\ell$  fraction of coordinates, for some universal constant  $\delta > 0$ . An encoding of  $C_\ell$  is an *injective* function  $\mathcal{E}_{C_\ell} : \{0, 1\}^{\rho\ell} \rightarrow C_\ell$ . The encoding is said to be efficient if  $\mathcal{E}_{C_\ell}(x)$  can be computed in  $\text{poly}(\ell)$  time for any  $x \in \{0, 1\}^{\rho\ell}$ .

Let us fix  $k \in \mathbb{N}$  and  $F : \mathbb{N} \rightarrow \mathbb{N}$  as in the theorem statement. We further define

$$\alpha := 1 - \frac{1}{\binom{k}{2} \cdot F\left(\binom{k}{2}\right)^{\binom{k}{2}}}.$$

**From  $k$ -MULTICOLOR CLIQUE to  $k$ -GAP CSP.** Starting from an instance of  $k$ -MULTICOLOR CLIQUE, say  $G$  on vertex set  $V := V_1 \dot{\cup} V_2 \dot{\cup} \dots \dot{\cup} V_k$ , we write down a set of constraints  $\mathcal{P}$  on a variable set  $X := \{x_{i,j} \mid i, j \in [k], i \neq j\}$  as follows. For every  $i, j \in [k]$ , such that  $i \neq j$ , define  $E_{i,j}$  to be the set of all edges in  $G$  whose end points are in  $V_i$  and  $V_j$ . An assignment to variable  $x_{i,j}$  is an element of  $E_{i,j}$ , i.e., a pair of vertices, one from  $V_i$  and the other from  $V_j$ . Suppose that  $x_{i,j}$

was assigned the edge  $\{v_i, v_j\}$ , where  $v_i \in V_i$  and  $v_j \in V_j$ . Then we define the assignment of  $x_{i,j}^i$  to be  $v_i$  and the assignment of  $x_{i,j}^j$  to be  $v_j$ . We define  $\mathcal{P} := \{P_1, \dots, P_k\}$ , where the constraint  $P_i$  is defined to be satisfied if the assignment to all of  $x_{1,i}^i, x_{2,i}^i, \dots, x_{i-1,i}^i, x_{i+1,i}^i, \dots, x_{k,i}^i$  are the same. We refer to the problem of determining if there is an assignment to the variables in  $X$  such that all the constraints are satisfied as the  $k$ -CSP problem. Notice that while this is a natural way to write  $k$ -MULTICOLOR CLIQUE as a CSP, where we have tried to check if all variables having a vertex in common, agree on its assignment, there is no gap yet in the  $k$ -CSP problem. In particular, if there was a clique of size  $k$  in  $G$  then there is an assignment to the variables of  $X$  (by assigning the edges of the clique in  $G$  to the corresponding variable in  $X$ ) such that all the constraints in  $\mathcal{P}$  are satisfied; however if every clique in  $G$  is of size less than  $k$  then there every assignment to the variables of  $X$  may violate only one constraint in  $\mathcal{P}$  (and not more).

In order to amplify the gap, we rewrite the set of constraints  $\mathcal{P}$  in a different way to obtain the set of constraints  $\mathcal{P}'$ , on the same variable set  $X$ , as follows. Suppose that  $x_{i,j}$  was assigned the edge  $\{v_i, v_j\}$ , where  $v_i \in V_i$  and  $v_j \in V_j$ , then for  $\beta \in [\log n]$ , we define the assignment of  $x_{i,j}^{i,\beta}$  to be the  $\beta^{\text{th}}$  coordinate of  $v_i$ ; recall that  $|V_i| = n$  and therefore we can label all vertices in  $V_i$  by vectors in  $\{0, 1\}^{\log n}$ . We define  $\mathcal{P}' := \{P'_1, \dots, P'_{\log n}\}$ , where the constraint  $P'_\beta$  is defined to be satisfied if and only if the following holds for all  $i \in [k]$ : the assignment to all of  $x_{1,i}^{i,\beta}, x_{2,i}^{i,\beta}, \dots, x_{i-1,i}^{i,\beta}, x_{i+1,i}^{i,\beta}, \dots, x_{k,i}^{i,\beta}$  are the same. Again notice that there is an assignment to the variables of  $X$  such that all the constraints in  $\mathcal{P}$  are satisfied if and only if the same assignment also satisfies all the constraints in  $\mathcal{P}'$ .

However, rewriting  $\mathcal{P}$  as  $\mathcal{P}'$  allows us to simply apply the error correcting code  $C_\ell$  (with parameters  $\rho$  and  $\delta$ , and encoding function  $\mathcal{E}_{C_\ell}$ ) to the constraints in  $\mathcal{P}'$ , to obtain a gap! In particular, we choose  $\ell$  to be such that  $\rho\ell = \log n$ . Consider a new set of constraints  $\mathcal{P}''$ , on the same variable set  $X$ , as follows. For any  $z \in \{0, 1\}^{\log n}$  and  $\beta \in [\ell]$ , we denote by  $\mathcal{E}_{C_\ell}(z)_\beta$ , the  $\beta^{\text{th}}$  coordinate of  $\mathcal{E}_{C_\ell}(z)$ . We define  $\mathcal{P}'' := \{P''_1, \dots, P''_\ell\}$ , where the constraint  $P''_\beta$  is defined to be satisfied if and only if the following holds for all  $i \in [k]$ : the assignment to all of  $\mathcal{E}_{C_\ell}(x_{1,i}^i)_\beta, \mathcal{E}_{C_\ell}(x_{2,i}^i)_\beta, \dots, \mathcal{E}_{C_\ell}(x_{i-1,i}^i)_\beta, \mathcal{E}_{C_\ell}(x_{i+1,i}^i)_\beta, \dots, \mathcal{E}_{C_\ell}(x_{k,i}^i)_\beta$  are the same.

Notice as before that there is an assignment to the variables of  $X$  such that all the constraints in  $\mathcal{P}'$  are satisfied if and only if the same assignment also satisfies all the constraints in  $\mathcal{P}''$ . However, for every assignment to  $X$  that violates at least one constraint in  $\mathcal{P}'$ , we have that the same assignment violates at least  $\delta$  fraction of the constraints in  $\mathcal{P}''$ . To see this consider an assignment that violates the constraint  $P_1$  in  $\mathcal{P}'$ . This implies that there is some  $i \in [k]$  such that the assignment to all of  $x_{1,i}^{i,1}, x_{2,i}^{i,1}, \dots, x_{i-1,i}^{i,1}, x_{i+1,i}^{i,1}, \dots, x_{k,i}^{i,1}$  are *not* the same. Let us suppose, without loss of generality, that the assignment to  $x_{1,i}^{i,1}$  and  $x_{2,i}^{i,1}$  are different. In other words, we have that  $x_{1,i}^i \neq x_{2,i}^i$ , where we think of  $x_{1,i}^i, x_{2,i}^i$  as  $\log n$  bit vectors. Let  $\Delta \subseteq [\ell]$  such that  $\beta \in \Delta$  if and only if  $\mathcal{E}_{C_\ell}(x_{1,i}^i)_\beta \neq \mathcal{E}_{C_\ell}(x_{2,i}^i)_\beta$ . By the distance of the code  $C_\ell$  we have that  $|\Delta| \geq \delta\ell$ . Finally, notice that for all  $\beta \in \Delta$ , we have that the assignment does not satisfy constraint  $P_\beta$  in  $\mathcal{P}''$ . We refer to the problem of distinguishing if there is an assignment to  $X$  such that all the constraints are satisfied or if every assignment to  $X$  does not satisfy a constant fraction of the constraints, as the  $k$ -GAP CSP problem.

In order to rule out  $F(k)$  approximation FPT algorithms for  $k$ -DOMINATING SET, we will need that for every assignment to  $X$  that violates at least one constraint in  $\mathcal{P}'$ , we have that the same assignment violates at least  $\alpha$  fraction of the constraints in  $\mathcal{P}''$  (instead of just  $\delta$ ; note that  $\alpha$  is very close to 1, whereas  $\delta$  can be at most half). To boost the gap<sup>2</sup>, we apply a simple

<sup>2</sup>We could have skipped this boosting step, had we chosen a different good code with distance  $\alpha$  but over a

repetition/direct-product trick to our constraint system. Starting from  $\mathcal{P}''$ , we construct a new set of constraints  $\mathcal{P}^*$ , on the same variable set  $X$ , as follows.

$$\mathcal{P}^* := \{P_S \mid S \in [\ell]^t\},$$

where  $t = \frac{\log(1-\alpha)}{\log(1-\delta)}$ . For every  $S \in [\ell]^t$ , we define  $P_S$  to be satisfied if and only if for all  $\beta \in S$ , the constraint  $P_\beta$  is satisfied.

It is easy to see that  $\mathcal{P}''$  and  $\mathcal{P}^*$  have the same set of completely satisfying assignments. However, for every assignment to  $X$  that violates  $\delta$  fraction of constraints in  $\mathcal{P}''$ , we have that the same assignment violates at least  $\alpha$  fraction of the constraints in  $\mathcal{P}^*$ . To see this consider an assignment that violates  $\delta$  fraction of constraints in  $\mathcal{P}''$ , say it violates all constraints  $P_\beta \in \mathcal{P}''$ , for every  $\beta \in \Delta \subseteq [\ell]$ . This implies that the assignment satisfies constraint  $P_S$  if and only if  $S \in ([\ell] \setminus \Delta)^t$ . This implies that the fraction of constraints in  $\mathcal{P}^*$  that the assignment can satisfy is upper bounded by  $(1 - \delta)^t = 1 - \alpha$ .

**From  $k$ -GAP CSP to gap  $k$ -DOMINATING SET.** In the second part, starting from the aforementioned instance of  $k$ -GAP CSP (after boosting the gap), we construct an instance  $H$  of  $k$ -DOMINATING SET. The construction is due to Feige [Fei98]<sup>3</sup> and it proceeds as follows. Let  $\mathcal{F}$  be the set of all functions from  $\{0, 1\}^{tk}$  to  $\binom{k}{2}$ , i.e.,  $\mathcal{F} := \{f : \{0, 1\}^{tk} \rightarrow \binom{k}{2}\}$ . The graph  $H$  is on vertex set  $U = A \dot{\cup} B$ , where  $A = \mathcal{P}^* \times \mathcal{F}$  and  $B = E(G)$ , i.e.,  $B$  is simply the edge set of  $G$ . We introduce an edge between all pairs of vertices in  $B$ . We introduce an edge between  $a := (S := (s_1, \dots, s_t) \in [\ell]^t, f : \{0, 1\}^{tk} \rightarrow \binom{k}{2}) \in A$  and  $e := (v_i, v_j) \in E$  if and only if the following holds.

$$\begin{aligned} \exists \tau := (\tau_1, \dots, \tau_t) \in \{0, 1\}^{kt}, \text{ such that } f(\tau) = \{i, j\} \text{ and} \\ \forall r \in [t], \text{ we have } \mathcal{E}_{C_\ell}(v_i)_{s_r} = (\tau_r)_i \text{ and } \mathcal{E}_{C_\ell}(v_j)_{s_r} = (\tau_r)_j. \end{aligned}$$

Notice that the number of vertices in  $H$  is  $|A| + |B| \leq \left(\frac{\log n}{\rho}\right)^t \cdot \binom{k}{2}^{2tk} + n^2 < \eta(k) \cdot n^{2.01}$ , for some computable function  $\eta$ . It is not hard to check that the following hold:

- (Completeness) If there is an assignment to  $X$  that satisfies all constraints in  $\mathcal{P}^*$ , then the corresponding  $\binom{k}{2}$  vertices in  $B$  dominate all vertices in the graph  $H$ .
- (Soundness) If each assignment can only satisfy  $(1 - \alpha)$  fraction of constraints in  $\mathcal{P}^*$ , then any dominating set of  $H$  has size at least  $F \left(\binom{k}{2}\right) \cdot \binom{k}{2}$ .

We skip presenting details of this part of the proof here. The proofs have been derived many times in literature; if needed, the readers may refer to Appendix A of [KLM19]. This completes our sketch of the proof of [Theorem 3.7](#).

A few remarks are in order. First, the  $k$ -GAP CSP problem described in the proof above, is formalized as the  $k$ -MAXCOVER problem in [KLM19] (and was originally introduced in [Cha+17]). In particular, the formalism of  $k$ -MAXCOVER (which may be thought of as the parameterized label cover problem) is generic enough to be used as an intermediate gap problem to reduce to both  $k$ -DOMINATING SET (as in [KLM19]) and  $k$ -CLIQUE (as in [Cha+17]).

---

larger alphabet. For example, taking the Reed Solomon code over alphabet  $\frac{\log n}{1-\alpha}$  would have sufficed. We chose not to do so, to keep the proof as elementary as possible.

<sup>3</sup>This reduction (which employs the hypercube set system) is used in [Fei98] for proving hardness of approximating MAX  $k$ -COVERAGE; for SET COVER, Feige used a more efficient set system which is not needed in our context.

Moreover, it was robust enough to capture stronger running time lower bounds (under stronger hypotheses); this will be elaborated below. However, in order to keep the above proof succinct, we skipped introducing the  $k$ -MAXCOVER problem, and worked with  $k$ -GAP CSP, which was sufficient for the above proof.

Second, [KLM19] additionally showed that for every computable functions  $T, F : \mathbb{N} \rightarrow \mathbb{N}$  and every constant  $\varepsilon > 0$ :

- Assuming the Exponential Time Hypothesis (ETH), there is no  $F(k)$ -approximation algorithm for  $k$ -DOMINATING SET that runs in  $T(k) \cdot n^{o(k)}$  time.
- Assuming the Strong Exponential Time Hypothesis (SETH), for every integer  $k \geq 2$ , there is no  $F(k)$ -approximation algorithm for  $k$ -DOMINATING SET that runs in  $T(k) \cdot n^{k-\varepsilon}$  time.

In order to establish [Theorem 3.7](#) and the above two results, [KLM19] introduced a framework to prove parameterized hardness of approximation results. In this framework, the objective was to start from either the  $W[1] \neq \text{FPT}$  hypothesis, ETH, or SETH, and end up with the gap  $k$ -DOMINATING SET, i.e., they design reductions from instances of  $k$ -CLIQUE, 3-CNF-SAT, and  $\ell$ -CNF-SAT, to an instance of gap  $k$ -DOMINATING SET. A prototype reduction in this framework has two modular parts. In the first part, which is specific to the problem they start from, they generate a gap and obtain hardness of gap  $k$ -MAXCOVER. In the second part, they show a gap preserving reduction from gap  $k$ -MAXCOVER to gap  $k$ -DOMINATING SET, which is essentially the same as the reduction from  $k$ -GAP CSP to  $k$ -DOMINATING SET in the proof of [Theorem 3.7](#).

The first part of a prototype reduction from the computational problem underlying a hypothesis of interest to gap  $k$ -MAXCOVER follows by the design of an appropriate communication protocol. In particular, the computational problem is first reduced to a constraint satisfaction problem (CSP) over  $k$  (or some function of  $k$ ) variables over an alphabet of size  $n$ . The predicate of this CSP would depend on the computational problem underlying the hypothesis from which we started. Generalizing ideas from [ARW17], they then show how a protocol for computing this predicate in the multiparty (number of players is the number of variables of the CSP) communication model, can be combined with the CSP to obtain an instance of gap  $k$ -MAXCOVER. For example, for the  $W[1] \neq \text{FPT}$  hypothesis and ETH, the predicate is a variant of the equality function, and for SETH, the predicate is the well studied disjointness function. The completeness and soundness of the protocols computing these functions translate directly to the completeness and soundness of  $k$ -MAXCOVER.

Third, we recall that Lin [Lin19] recently provided alternate proofs of [Theorem 3.7](#) and the above mentioned stronger running time lower bounds. While we discussed about his proof technique in [Section 3.1.1](#), we would like to discuss about his result here. Following the right setting of parameters in the proof of [Theorem 3.7](#) (for example set  $\alpha = 1 - \frac{1}{(\log n)^{\Omega(1/k)}}$ ), we can obtain that approximating  $k$ -DOMINATING SET to a factor of  $(\log n)^{1/k^3}$  is  $W[1]$ -hard. Lin improved the exponent of  $1/k^3$  in the approximation factor to  $h(k)$  for any computable function  $h$ . Can this inapproximability be further improved? On the other hand, can we do better than the simple polynomial time greedy algorithm which provides a  $(1+\ln n)$  factor approximation? This leads us to the following question:

**Open Question 3.8** (Tight inapproximability of  $k$ -DOMINATING SET). *Is there a  $(\log n)^{1-o(1)}$  factor approximation algorithm for  $k$ -DOMINATING SET running in time  $n^{k-0.1}$ ?*

We conclude the discussion on  $k$ -DOMINATING SET with an open question on  $W[2]$ -hardness of approximation. As noted earlier,  $k$ -DOMINATING SET is a  $W[2]$ -complete problem, and [Theorem 3.7](#) shows that the problem is  $W[1]$ -hard to approximate to any  $F(k)$  factor. But is

there some computable function  $F$  for which approximating  $k$ -DOMINATING SET is in  $W[1]$ ? In other words we have:

**Open Question 3.9** ( $W[2]$ -completeness of approximating  $k$ -DOMINATING SET). *Can we base total inapproximability of  $k$ -DOMINATING SET on  $W[2] \neq FPT$ ?*

### 3.1.3 Parameterized Inapproximability of Steiner Orientation by Gap Amplification

Gap amplification is a widely used technique in the classic literatures on (NP-)hardness of approximation (e.g. [BS92; Din07; Raz98]). In fact, the arguably simplest proof of the PCP theorem, due to Dinur [Din07], is indeed via repeated gap amplification. The overall idea here is simple: we start with a hardness of approximation for a problem with small factor (e.g.  $1 + 1/n$ ). At each step, we perform an operation that transform an instance of our problem to another instance, in such a way that the gap becomes bigger; usually this new instance will also be bigger than our instance. By repeatedly applying this operation, one can finally arrive at a constant, or even super constant, factor hardness of approximation.

There are two main parameters that determine the success/failure of such an approach: how large the new instance is compared to the old instance (i.e. size blow-up) and how large the new gap is compared to the old gap, in each operation. To see how these two come into the picture, let us first consider a case study where a (straightforward) gap amplification procedure *does not* work:  $k$ -CLIQUE. The standard way to amplify the gap for  $k$ -CLIQUE is through graph product. Recall that the (tensor) graph product of a graph  $G = (V, E)$  with itself, denoted by  $G^{\otimes 2}$ , is a graph whose vertex set is  $V^2$  and there is an edge between  $(u_1, u_2)$  and  $(v_1, v_2)$  if and only if  $(u_1, v_1) \in E$  and  $(u_2, v_2) \in E$ . It is not hard to check that, if we can find a clique of size  $t$  in  $G^{\otimes 2}$ , then we can find one of size  $\sqrt{t}$  in  $G$  (and vice versa). This implies that, if we have an instance of clique that is hard to approximate to within a factor of  $(1 + \varepsilon)$ , then we may take the graph product with itself which yields an instance of CLIQUE that is hard to approximate to within a factor of  $(1 + \varepsilon)^2$ .

Now, let us imagine that we start with the hard instance of an exact version of  $k$ -CLIQUE. We may think of this as being hard to approximate to within a factor of  $(1 - 1/k)$ . Hence, we may apply the above gap amplification procedure  $\log k$  times, resulting in an instance of CLIQUE that is hard to approximate to within a factor of  $(1 - 1/k)^{2^{\log k}}$ , which is a constant bounded away from one (i.e.  $\approx 1/e$ ). The bad news here is that the number of the vertices of the final graph is  $n^{2^{\log k}} = n^k$ , where  $n$  is the number of vertices of the initial graph. This does not give any lower bound, because we can solve  $k$ -CLIQUE in the original graph in  $n^{O(k)}$  time trivially! In the next subsection, we will see a simple way to prove hardness of approximating  $k$ -CLIQUE, assuming stronger assumptions. However, it remains an interesting and important open question how to prove such hardness from a non-gap assumption:

**Open Question 3.10.** *Is it  $W[1]$ -hard or ETH-hard to approximate  $k$ -CLIQUE to within a constant factor in FPT time?*

Having seen a failed attempt, we may now move on to a success story. Remarkably, Wlodarczyk [Wlo19] recently managed to use gap amplification to prove hardness of approximation for connectivity problems, including the  $k$ -STEINER ORIENTATION problem. Here we are given a *mixed graph*  $G$ , whose edges are either directed or undirected, and a set of  $k$  terminal pairs  $\{(s_i, t_i)\}_{i \in [k]}$ . The goal is to orient all the undirected edges in such a way that maximizes the number of  $t_i$  that can be reached from  $s_i$ . The problem is known to be in XP [CKN13]

but is  $W[1]$ -hard even when all terminal pairs can be connected [PW18]. Starting from this  $W[1]$ -hardness, Włodarczyk [Wlo19] devises a gap amplification step that implies a hardness of approximation with factor  $(\log k)^{o(1)}$  for the problem. Due to the technicality of the gap amplification step, we will not go into the specifics in this survey. However, let us point out the differences between this gap amplification and the (failed) one for CLIQUE above. The key point here is that the new instance of Włodarczyk’s gap amplification has size of the form  $f(k) \cdot n$  instead of  $n^2$  as in the graph product. This means that, even if we are applying Włodarczyk’s gap amplification step  $\log(k)$  times or more generally  $g(k)$  times, it only results in an instance of size  $f(\underbrace{f(\dots(f(k)))}_{g(k)\text{ times}}) \cdot n$ , which is still FPT! Since the technique is still quite new, it is an exciting frontier to examine whether other parameterized problems allow such similar gap amplification steps.

### 3.2 Hardness from Gap Hypotheses

In the previous subsection, we have seen that several hardness of approximation results can be proved based on standard assumptions. However, as alluded to briefly, some basic problems, including  $k$ -CLIQUE, still evades attempts at proving such results. This motivates several researchers in the community to come up with new assumptions that allow more power and flexibility in proving inapproximability results. We will take a look at two of these hypotheses in this subsection; we note that there have also been other assumptions formulated, but we only focus on these two since they arguably have been used most often.

The first assumption, called the Parameterized Inapproximability Hypothesis or PIH for short, can be viewed as a gap analogue of the  $W[1] \neq \text{FPT}$  assumption. There are many (equivalent) ways to state PIH. We choose to state it in terms of an inapproximability of the colored version of DENSEST  $k$ -SUBGRAPH. In MULTICOLORED DENSEST  $k$ -SUBGRAPH, we are given a graph  $G = (V, E)$  where the vertex set  $V$  is partitioned into  $k$  parts  $V_1, \dots, V_k$ . The goal is to select  $k$  vertices  $v_1 \in V_1, v_2 \in V_2, \dots, v_k \in V_k$  such that  $\{v_1, \dots, v_k\}$  induces as many edges as possible.

It is easy to see that the exact version of this problem is  $W[1]$ -hard, via a straightforward reduction from  $k$ -CLIQUE. PIH postulates that even the approximate version of this problem is hard:

**Hypothesis 3.11** (Parameterized Inapproximability Hypothesis (PIH)<sup>4</sup> [Lok+20]). *For some constant  $\epsilon > 0$ , there is no  $(1 + \epsilon)$  factor FPT approximation algorithm for MULTICOLORED DENSEST  $k$ -SUBGRAPH.*

There are two important remarks about PIH. First, the factor  $(1 + \epsilon)$  is not important, and the conjecture remains equivalent even if we state it for a factor  $C$  for any arbitrarily large constant  $C$ ; this is due to gap amplification via parallel repetition [Raz98]. Second, PIH implies that  $k$ -CLIQUE is hard to approximate to within any constant factor:

**Lemma 3.12.** *Assuming PIH, there is no constant factor FPT approximation algorithm for  $k$ -CLIQUE.*

The above result can be shown via a classic reduction of Feige, Goldwasser, Lovász, Safra and Szegedy (henceforth FGLSS) [Fei+96], which was one of the first works connecting proof

<sup>4</sup>We remark that the original conjecture in [Lok+20] says that the problem is  $W[1]$ -hard to approximate. However, we choose to state the more relaxed form here.

systems and hardness of approximation. Specifically, the FGLSS reduction transforms  $G$  to another graph  $G'$  by viewing the edges of  $G$  as vertices of  $G'$ . Then, we connect  $\{u_1, v_1\}$  and  $\{u_2, v_2\}$  *except* when the union  $\{u_1, v_1\} \cup \{u_2, v_2\}$  contains two distinct vertices from the same partition. One can argue that the size of the largest clique in  $G'$  is exactly equal to the number of edges in the optimal solution of MULTICOLORED DENSEST  $k$ -SUBGRAPH on  $G$ . As a result, PIH implies hardness of approximation of the former. Interestingly, however, it is not known if the inverse is true and this remains an interesting open question:

**Open Question 3.13.** *Does PIH hold if we assume that  $k$ -CLIQUE is FPT inapproximable to within any constant factor?*

As demonstrated by the FGLSS reduction, once we have a gap, it is much easier to give a reduction to another hardness of approximation result, because we do not have to create the initial gap ourselves (as in the previous subsection) but only need to preserve or amplify the gap. Indeed, PIH turns out to be a pretty robust hypothesis that gives FPT inapproximability for many problems, including  $k$ -CLIQUE, DIRECTED ODD CYCLE TRAVERSAL [Lok+20] and STRONGLY CONNECTED STEINER SUBGRAPH [CFM18]. We remark that the current situation here is quite similar to that of the landscape of the classic theory of hardness of approximation before the PCP Theorem [Aro+98; AS92] was proved. There, Papadimitriou and Yannakakis introduced a complexity class MAX-SNP and show that many optimization problems are hard (or complete) for this class [PY91]. Later, the PCP Theorem confirms that these problems are NP-hard. In our case of FPT inapproximability, PIH seems to be a good analogy of MAX-SNP for problems in  $W[1]$  and, as mentioned before, PIH has been used as a starting point of many hardness of approximation results. However, there has not yet been many reverse reductions to PIH, and this is one of the motivation behind Question 3.13 above.

Despite the aforementioned applications of PIH, there are still quite a few questions that seem out of reach of PIH, such as whether there is an  $o(k)$  factor FPT approximation for  $k$ -CLIQUE or questions related to running time lower bounds of approximation algorithms. On this front, another stronger conjecture called the Gap Exponential Time Hypothesis (Gap-ETH) is often used instead:

**Hypothesis 3.14** (Gap Exponential Time Hypothesis (Gap-ETH) [Din16; MR17]). *For some constants  $\varepsilon, \delta > 0$ , there is no  $O(2^{\delta n})$ -time algorithm that can, given a 3CNF formula, distinguish between the following two cases:*

- (Completeness) the formula is satisfiable.
- (Soundness) any assignment violates more than  $\varepsilon$  fraction of the clauses.

Here  $n$  denotes the number of clauses<sup>5</sup>.

Clearly, Gap-ETH is a strengthening of ETH, which can be thought of in the above form but with  $\varepsilon = 1/n$ . Another interesting fact is that Gap-ETH is stronger than PIH. This can be shown via the standard reduction from 3SAT to  $k$ -CLIQUE that establishes  $N^{\Omega(k)}$  lower bound for the latter. The reduction, due to Chen et al. [Che+04; Che+06], proceed as follows. First, we partition the set of clauses  $C$  into  $C_1, \dots, C_k$  each of size  $n/k$ . For each  $C_i$ , we create a partition  $V_i$  in the new graph where each vertex corresponds to all partial assignments (to variables that appear in at least one clause of  $C_k$ ) that satisfy all the clauses in  $C_k$ . Two vertices are connected

<sup>5</sup>The version where  $n$  denotes the number of variables is equivalent to the current formulation, because we can always assume without loss of generality that  $m = O(n)$ . (See [Din16; MR17].)



if the corresponding partial assignments are consistent, i.e., they do not assign a variable to different values.

If there is an assignment that satisfies all the clauses, then clearly the restrictions of this assignment to each clause corresponds to  $k$  vertices from different partitions that form a clique. On the other hand, it is also not hard to argue that, in the soundness case, the number of edges induced by any  $k$  vertices from different partitions is at most  $1 - \Theta(\varepsilon)$ . Thus, Gap-ETH implies PIH as claimed.

Now that we have demonstrated that Gap-ETH is at least as strong as PIH, we may go further and ask how much more can we achieve from Gap-ETH, compared to PIH. The obvious consequences of Gap-ETH is that it can give explicit running time lower bounds for FPT hardness of approximation results. Perhaps more surprising, however, is that it can be used to improve the *inapproximability ratio* as well. The rest of this subsection is devoted to present some of these examples, together with brief overviews of how the proofs of these results work.

### 3.2.1 Strong Inapproximability of $k$ -Clique

Our first example is the  $k$ -CLIQUE problem. Obviously, we can approximate  $k$ -CLIQUE to within a factor of  $k$ , by just outputting any single vertex. It had long been asked whether an  $o(k)$ -approximation is achievable in FPT time. As we saw above, PIH implies that a constant factor FPT approximation does not exist, but does not yet resolve this question. Nonetheless, assuming Gap-ETH, this question can be resolved in the negative:

**Theorem 3.15** ([Cha+17]). *Assuming Gap-ETH, there is no  $o(k)$ -FPT-approximation for  $k$ -CLIQUE.*

The reduction used in [Cha+17] to prove the above inapproximability is just a simple modification of the above reduction [Che+04; Che+06] that we saw for  $k$ -CLIQUE. Suppose that we would like to rule out a  $\frac{k}{g}$ -approximation, where  $g = g(k)$  is a function such that  $\lim_{k \rightarrow \infty} g(k) = \infty$ . The only change in the reduction is that, instead of letting  $C_1, \dots, C_k$  be the partition of the set of clauses  $C$ , we let each  $C_i$  be a set of  $\frac{Dn}{g}$  clauses for some sufficiently large constant  $D > 0$ . The rest of the reduction works similar to before: for each  $C_i$ , we create a vertex corresponding to each partial assignment that satisfies all the clauses in  $C_i$ . Two vertices are joined by an edge if and only if they are consistent. This completes the description of the reduction.

To see that the reduction yields [Theorem 3.15](#), first note that, if there is an assignment that satisfies the CNF formula, then we can again pick the restrictions on this formula onto  $C_1, \dots, C_k$ ; these gives  $k$  vertices that induces a clique in the graph.

On the other hand, suppose that every assignment violates more than an  $\varepsilon$  fraction of clauses. We will argue that there is no clique of size  $g$  in the constructed graph. The only property we need from the subsets  $C_1, \dots, C_k$  is that the union of any  $g$  such subsets contain at least  $(1 - \varepsilon)$  fraction of the clauses. It is not hard to show that this is true with high probability, when we choose  $D$  to be sufficiently large. Now, suppose for the sake of contradiction that there exists a clique of size  $g$  in the graph. Since the vertices corresponding to the same subset  $C_i$  form an independent set, it must be that these  $g$  vertices are from different subsets. Let us call these subsets  $C_{i_1}, \dots, C_{i_g}$ . Because these vertices induce a clique, we can find a global assignment that is consistent with each vertex. This global assignment satisfies all the clauses in  $C_{i_1} \cup \dots \cup C_{i_g}$ . However,  $C_{i_1} \cup \dots \cup C_{i_g}$  contains at least  $1 - \varepsilon$  fraction of all clauses, which contradicts to our assumption that every assignment violates more than  $\varepsilon$  fraction of the clauses.

Now, if we can  $o(k)$ -approximate  $k$ -CLIQUE in  $T(k) \cdot N^{O(1)}$  time. Then, we may run this algorithm to distinguish the two cases in Gap-ETH in  $f(k) \cdot (2^{Dn/g})^{O(1)} = 2^{o(n)}$  time, which violates Gap-ETH. This concludes our proof sketch. We end by remarking that the reduction may also be viewed as an instantiation of the randomized graph product [BGS98; BS92; Zuc96], and it can also be derandomized. We omit the details of the latter here. Interested readers may refer to [Cha+17] for more detail.

### 3.2.2 Strong Inapproximability of Multicolored Densest $k$ -Subgraph and Label Cover

For our second example, we go back to the MULTICOLORED DENSEST  $k$ -SUBGRAPH once again. Recall that PIH asserts that this problem is hard to approximate to some constant factor, and we have seen above that Gap-ETH also implies this. On the approximation front, however, only the trivial  $k$ -approximation algorithm is known: just pick a vertex that has edges to as many partitions as possible. Then, output that vertex and one of its neighbors from each partition. It is hence a natural question to ask whether it is possible to beat this approximation ratio. This question has been, up to lower order terms, answered in the negative, assuming Gap-ETH:

**Theorem 3.16** ([DM18]). *Assuming Gap-ETH, there is no  $k^{1-o(1)}$ -approximation for MULTICOLORED DENSEST  $k$ -SUBGRAPH.*

An interesting aspect of the above result is that, even in the NP-hardness regime, no NP-hardness of factor  $k^\gamma$  for some constant  $\gamma > 0$  is known. In fact, the problem is closely related to (and is a special case of) a well-known conjecture in the hardness of approximation community called the Sliding Scale Conjecture (SSC) [Bel+93]<sup>6</sup>. (See [DM18] for more discussion on the relation between the two.) Thus, this is yet another instance where taking a parameterized complexity perspective helps us advance knowledge even in the classical settings.

To prove Theorem 3.16, arguably the most natural reduction here is the above reduction for Clique! Note that we now view the vertices corresponding to each subset  $C_i$  as forming a partition  $V_i$ . The argument in the YES case is exactly the same as before: if the formula is satisfiable, then there is a (multicolored)  $k$ -clique. However, as the readers might have noticed, the argument in the NO case does not go through anymore. In particular, even when the graph is quite dense (e.g. having half of the edges present), it may not contain any large clique at all and hence it is unclear how to recover back an assignment that satisfies a large fraction of constraints.

This obstacle was overcome in [DM18] by proving an *agreement testing theorem* (i.e. *direct product theorem*), which is of the following form. Given  $k$  local functions  $f_1, \dots, f_k$ , where  $f_i : S_i \rightarrow \{0, 1\}$  is a boolean function whose domain  $S_i$  is a subset of a universe  $\mathcal{U}$ . If some (small)  $\zeta$  fraction of the pairs agree<sup>7</sup> with each other, then we can find (i.e. “decode”) a global function  $h : [n] \rightarrow \{0, 1\}$  that “approximately agrees” with roughly  $\zeta$  fraction of the local functions. The theorem in [DM18] works when  $S_1, \dots, S_k$  are sets of size  $\Omega(n)$ .

Due to the technical nature of the definitions, we will not fully formalize the notions in the previous paragraph. Nonetheless, let us sketch how to apply the agreement testing theorem to prove the NO case for our reduction. Suppose for the sake of contradiction that the formula is not  $(1 - \delta)$ -satisfiable and that there exists a  $k$ -subgraph with density  $\zeta \geq \frac{1}{k^{1-o(1)}}$ . Recall that each selected vertex is simply a partial assignment onto the subset of clauses  $C_i$  for some  $i$ ; we

<sup>6</sup>See also the related Projection Game Conjecture (PGC) [Mos15].

<sup>7</sup>Naturally, we say that two functions  $f_i$  and  $f_j$  agree iff  $f_i(x) = f_j(x)$  for all  $x \in S_i \cap S_j$ .

may view this as a function  $f_i : S_i \rightarrow \{0, 1\}$  where  $S_i$  denote the set of variables that appear in  $C_i$ . Here the universe  $\mathcal{U}$  is the set of all variables. With this perspective, we can apply the agreement testing theorem to recover a global function  $h : \mathcal{U} \rightarrow \{0, 1\}$  that “approximately agrees” with roughly  $\zeta k$  of the local functions. Notice that, in this context,  $h$  is simply a global assignment for the CNF formula. Previously in the proof for inapproximability of Clique, we had a global assignment that (perfectly) agrees with  $g$  local functions, from which we can conclude that this assignment satisfies all but  $\delta$  fraction of the clauses. It turns out that relaxing “perfect agreement” to “approximate agreement” does not affect the proof too much, and the latter still implies that  $h$  satisfies all but  $\delta$  fraction of clauses as desired.

As for the proof of the agreement testing theorem itself, we will not delve too much into detail here. However, we note that the proof is based on looking at different “agreement levels” and the graph associated with them. It turns out that such a graph has a certain transitivity property, which allows one to “decode” back the global function  $h$ . This general approach of looking at different agreement levels and their transitivity properties is standard in the direct product/agreement testing literature [DN17; IKW12; RS97]. The main challenge in [DM18] is to make the proof works for  $\zeta$  as small as  $1/k$ , which requires a new notion of transitivity.

To end this subsection, we remark that the MULTICOLORED DENSEST  $k$ -SUBGRAPH is known as the 2-ary Constraint Satisfaction Problem (2-CSP) in the classical hardness of approximation community. The problem, and in particular its special case called *Label Cover*, serves as the starting point of almost all known NP-hardness of approximation (see e.g. [Aro+93; Cha16; Hås01]). The technique in [DM18] can also be used to show inapproximability for Label Cover with strong running time lower bound of the form  $f(k) \cdot N^{\Omega(k)}$  [Man]. Due to known reductions, this has numerous consequences. For example, it implies, assuming Gap-ETH, that approximating  $k$ -EVEN SET to within any factor less than two cannot be done in  $f(k) \cdot N^{o(k)}$  time, considerably improving the lower bound mentioned in the previous subsection.

### 3.2.3 Inapproximability of $k$ -Biclique and Densest $k$ -Subgraph

While PIH (or equivalently the MULTICOLORED DENSEST  $k$ -SUBGRAPH problem) can serve as a starting point for hardness of approximation of many problems, there are some problems for which not even a constant factor hardness is known under PIH, but strong inapproximability results can be obtained via Gap-ETH. We will see two examples of this here.

First is the  $k$ -BICLIQUE problem. Recall that in this problem, we are given a bipartite graph and we would like to determine whether there is a complete bipartite subgraph of size  $k$ . As stated earlier in the previous subsection, despite its close relationship to  $k$ -CLIQUE,  $k$ -BICLIQUE turned out to be a much more challenging problem to prove intractibility and even its W[1]-hardness was only shown recently [Lin18]. This difficulty is corroborated by its approximability status in the classical (non-parameterized) regime; while CLIQUE is long known to be NP-hard to approximate to within  $N^{1-o(1)}$  factor [Hås96], BICLIQUE is not even known to be NP-hard to approximate to within say 1.01 factor<sup>8</sup>. With this in mind, it is perhaps not a surprise that  $k$ -BICLIQUE is not known to be hard to approximate under PIH. Nonetheless, when we assume Gap-ETH, we can in fact prove a very strong hardness of approximation for the problem:

**Theorem 3.17** ([Cha+17]). *Assuming Gap-ETH, there is no  $o(k)$ -FPT-approximation for  $k$ -BICLIQUE.*

<sup>8</sup>We note, however, that strong inapproximability of BICLIQUE is known under stronger assumptions [Bha+17; Kho06; Man18b].

Note that, similar to  $k$ -CLIQUE, a  $k$ -approximation for BICLIQUE can be easily achieved by outputting a single edge. Hence, in terms of the inapproximability ratio, the above result is tight.

Due to its technicality, we only sketch an outline of the proof of [Theorem 3.17](#) here. Firstly, the reduction starts by constructing a graph that is similar (but not the same) to that of  $k$ -CLIQUE that we describe above. The main properties of this graph is that (i) in the YES case where the formula is satisfiable, the graph contains many copies of  $k$ -BICLIQUE, and (ii) in the NO case where the formula is not even  $(1 - \delta)$ -satisfiable, the graph contains few copies of  $g$ -BICLIQUE. The construction and these properties were in fact shown in [\[Man17\]](#). In [\[Cha+17\]](#), it was observed that, if we subsample the graph by keeping each vertex independently with probability  $p$  for an appropriate value of  $p$ , then (i) ensures that at least one of the  $k$ -BICLIQUE survives the subsampling, whereas (ii) ensures that no  $g$ -BICLIQUE survives. This indeed gives the claimed result in the above theorem.

We remark that, while [Theorem 3.17](#) seems to resolve the approximability of  $k$ -BICLIQUE, there is still one aspect that is not yet completely understood: the running time lower bound. To demonstrate this, recall that, for  $k$ -CLIQUE, the reduction that gives hardness of  $k$ -VS- $g$  CLIQUE has size  $2^{O(n/g)}$ ; this means that we have a running time lower bound of  $f(k) \cdot N^{\Omega(g)}$  on the problem. This is of course tight, because we can determine whether a graph has a  $g$ -clique in  $N^{O(g)}$  time. However, for  $k$ -BICLIQUE, the known reduction that gives hardness for  $k$ -VS- $g$  BICLIQUE has size  $2^{O(n/\sqrt{g})}$ . This results in a running time lower bound of only  $f(k) \cdot N^{\Omega(\sqrt{g})}$ . Specifically, for the most basic setting of constant factor approximation, [Theorem 3.17](#) only rules out algorithms with running time  $f(k) \cdot N^{o(\sqrt{k})}$ . Hence, an immediate question here is:

**Open Question 3.18.** *Is there an  $f(k) \cdot N^{o(k)}$ -time algorithm that approximates  $k$ -BICLIQUE to within a constant factor?*

To put things into perspective, we note that, even for *exact* algorithms for  $k$ -BICLIQUE, the best running time lower bound is still  $f(k) \cdot N^{\Omega(\sqrt{k})}$  [\[Lin18\]](#) (under any reasonable complexity assumption). This means that, to answer [Question 3.18](#), one has to first settle the best known running time lower bound for exact algorithms, which would already be a valuable contribution to the understanding of the problem.

Let us now point out an interesting consequence of [Theorem 3.17](#) for the DENSEST  $k$ -SUBGRAPH problem. This is the “uncolored” version of the MULTICOLORED DENSEST  $k$ -SUBGRAPH problem as defined above, where there are no partitions  $V_1, \dots, V_k$  and we can pick any  $k$  vertices in the input graph  $G$  with the objective of maximizing the number of induced edges. The approximability status of DENSEST  $k$ -SUBGRAPH very much mirrors that of  $k$ -BICLIQUE. Namely, in the parameterized setting, PIH is not known to imply hardness of approximation for DENSEST  $k$ -SUBGRAPH. Furthermore, in the classic (non-parameterized) setting, DENSEST  $k$ -SUBGRAPH is not known<sup>9</sup> to be NP-hard to approximate even to within a factor of say 1.01. Despite these, Gap-ETH does give a strong inapproximability for DENSEST  $k$ -SUBGRAPH, as stated below:

**Theorem 3.19** ([\[Cha+17\]](#)). *Assuming Gap-ETH, there is no  $k^{o(1)}$ -FPT-approximation for DENSEST  $k$ -SUBGRAPH.*

In fact, the above result is a simple consequence of [Theorem 3.17](#). To see this, recall the following classic result in extremal graph theory commonly referred to as the Kővári-Sós-Turán

<sup>9</sup>Again, similar to BICLIQUE, DENSEST  $k$ -SUBGRAPH is known to be hard to approximate under stronger assumptions [\[Alo+11; Kho06; Man17; RS10\]](#).

(KST) Theorem [KST54]: any  $k$ -vertex graph that does not contain a  $g$ -biclique as a subgraph has density at most  $O(k^{-1/g})$ . Now, the hardness for  $k$ -BICLIQUE from Theorem 3.17 tells us that there is no FPT time algorithm that can distinguish between the graph containing  $k$ -biclique from one that does not contain  $g$ -biclique for any  $g = \omega(1)$ . When the graph contains a  $k$ -biclique, we have a  $k$ -vertex subgraph with density (at least)  $1/2$ . On the other hand, when the graph does not even contain a  $g$ -biclique, the KST Theorem ensures us that any  $k$ -vertex subgraph has density at most  $O(k^{-1/g})$ . This indeed gives a gap of  $O(k^{1/g})$  in terms of approximation Densest  $k$ -Subgraph and finishes the proof sketch for Theorem 3.19.

Unfortunately, Theorem 3.19 does not yet resolve the FPT approximability of DENSEST  $k$ -SUBGRAPH. In particular, while the hardness is only of the form  $k^{o(1)}$ , the best known algorithm (which is the same as that of the multicolored version discussed above) only gives an approximation ratio of  $k$ . Hence, we may ask whether this can be improved:

**Open Question 3.20.** *Is there an  $o(k)$ -FPT-approximation algorithm for DENSEST  $k$ -SUBGRAPH?*

This should be contrasted with Theorem 3.16, for which the FPT approximability of MULTICOLORED DENSEST  $k$ -SUBGRAPH is essentially resolved (up to lower order terms).

## 4 Algorithms

In this section we survey some of the developments on the algorithmic side in recent years. The organization of this section is according to problem types. We begin with basic packing and covering problems in Sections 4.1 and 4.2. We then move on to clustering in Section 4.3, network design in Section 4.4, and cut problems in Section 4.5. In Section 4.6 we present width reduction problems.

The algorithm of the above mentioned sections compute approximate solutions to problems that are W[1]-hard. Therefore it is necessary to approximate, even when using parameterization. However, one may also aim to obtain faster parameterized runtimes than the known FPT algorithms, by sacrificing in the solution quality. We present some results of this type in Section 4.7.

### 4.1 Packing problems

For a packing problem the task is to select as many combinatorial objects of some mathematical structure (such as a graph or a set system) as possible under some constraint, which restricts some objects to be picked if others are. A basic example is the INDEPENDENT SET problem, for which a maximum sized set of vertices of a graph needs to be found, such that none of them are adjacent to each other.

#### 4.1.1 Independent Set

The INDEPENDENT SET problem is notoriously hard in general. Not only is there no polynomial time  $n^{1-\varepsilon}$ -approximation algorithm [Zuc06] for any constant  $\varepsilon > 0$ , unless  $P=NP$ , but also, under Gap-ETH, no  $g(k)$ -approximation can be computed in  $f(k)n^{O(1)}$  time [Cha+17] for any computable functions  $f$  and  $g$ , where  $k$  is the solution size. On the other hand, for planar graphs a PTAS exists [Bak94]. Hence a natural question is how the problem behaves for graphs that are “close” to being planar.

One way to generalize planar graphs is to consider minor-free graphs, because planar graphs are exactly those excluding  $K_5$  and  $K_{3,3}$  as minors. When parameterizing by the size of an excluded minor, the INDEPENDENT SET problem is paraNP-hard, since the problem is NP-hard on planar graphs [JG79]. Nevertheless a PAS can be obtained for this parameter [DH04].

**Theorem 4.1** ([DH04; GKR13]). *Let  $H$  be a fixed graph. For  $H$ -minor-free graphs, INDEPENDENT SET admits an  $(1 + \varepsilon)$ -approximation algorithm that runs in  $f(H, \varepsilon)n^{O(1)}$  time for some function  $f$ .*

This result is part of the large framework of “bidimensionality theory” where any graph in an appropriate minor-closed class has treewidth bounded above in terms of the problem’s solution value, typically by the square root of that value. These properties lead to efficient, often subexponential, fixed-parameter algorithms, as well as polynomial-time approximation schemes, for bidimensional problems in many minor-closed graph classes. The bidimensionality theory is based on algorithmic and combinatorial extensions to parts of the Robertson-Seymour Graph Minor Theory, in particular initiating a parallel theory of graph contractions. The foundation of this work is the topological theory of drawings of graphs on surfaces. We refer the reader to the survey of [DH08] and more recent papers [DHK11; Fom+11; GKR13].

A different way to generalize planar graphs is to consider a *planar deletion set*, i.e., a set of vertices in the input graph whose removal leaves a planar graph. Taking the size of such a set as a parameter, INDEPENDENT SET is again paraNP-hard [JG79]. However, by first finding a minimum sized planar deletion set, then guessing the intersection of this set with the optimum solution to INDEPENDENT SET, and finally using the PTAS for planar graphs [Bak94], a PAS can be obtained parameterized by the size of a planar deletion set [Mar08].

**Theorem 4.2** ([Mar08]). *For the INDEPENDENT SET problem a  $(1 + \varepsilon)$ -approximation can be computed in  $2^k n^{O(1/\varepsilon)}$  time for any  $\varepsilon > 0$ , where  $k$  is the size of a minimum planar deletion set.*

Ideas using linear programming allow us to generalize and handle larger noise at the expense of worse dependence on  $\varepsilon$ . Bansal et al. [BRU17] showed that given a graph obtained by adding  $\delta n$  edges to some planar graph, one can compute a  $(1 + O(\varepsilon + \delta))$ -approximate independent set in time  $n^{O(1/\varepsilon^4)}$ , which is faster than the  $2^k n^{O(1/\varepsilon)}$  running time of Theorem 4.2 for large  $k = \delta n$ . Magen and Moharrami [MM09] showed that for every graph  $H$  and  $\varepsilon > 0$ , given a graph  $G = (V, E)$  that can be made  $H$ -minor-free after at most  $\delta n$  deletions and additions of vertices or edges, the size of the maximum independent set can be approximately computed within a factor  $(1 + \varepsilon + O(\delta|H|\sqrt{\log|H|}))$  in time  $n^{f(\varepsilon, H)}$ . Note that this algorithm does not find an independent set. Recently, Demaine et al. [Dem+19] presented a general framework to obtain better approximation algorithms for various problems including INDEPENDENT SET and CHROMATIC NUMBER, when the input graph is close to well-structured graphs (e.g., bounded degeneracy, degree, or treewidth).

It is also worth noting here that INDEPENDENT SET problem can be generalized to the  $d$ -SCATTERED SET problem where we are given an (edge-weighted) graph and are asked to select at least  $k$  vertices, so that the distance between any pair is at least  $d$  [KLP18]. Recently in [KLP19a] some lower and upper bounds on the approximation of the  $d$ -SCATTERED SET problem have been provided.

A special case of INDEPENDENT SET is the INDEPENDENT SET OF RECTANGLES problem, where a set of axis-parallel rectangles is given in the two-dimensional plane, and the task is to find a maximum sized subset of non-intersecting rectangles. This is a special case, since pairwise intersections of rectangles can be encoded by edges in a graph for which the vertices

are the rectangles. Parameterized by the solution size, the problem is  $W[1]$ -hard [Mar05], and while a QPTAS is known [AW13], it is a challenging open question whether a PTAS exists. It was shown [GKW19] however that both a PAS and a PSAKS exist for INDEPENDENT SET OF RECTANGLES parameterized by the solution size, even for the weighted version.

The runtime of this PAS is  $f(k, \varepsilon)n^{g(\varepsilon)}$  for some functions  $f$  and  $g$ , where  $k$  is the solution size. Note that the dependence on  $\varepsilon$  in the degree of the polynomial factor of this algorithm cannot be removed, unless  $FPT=W[1]$ , since any *efficient* PAS with runtime  $f(k, \varepsilon)n^{O(1)}$  could be used to compute the optimum solution in FPT time by setting  $\varepsilon$  to  $\frac{1}{k+1}$  in the  $W[1]$ -hard unweighted version of the problem [Mar05]. However, in the so-called *shrinking model* an efficient PAS can be obtained [PLW17] for INDEPENDENT SET OF RECTANGLES. The parameter in this case is a factor  $0 < \delta < 1$  by which every rectangle is shrunk before computing an approximate solution, which is compared to the optimum solution without shrinking.

**Theorem 4.3** ([GKW19; PLW17]). *For the INDEPENDENT SET OF RECTANGLES problem a  $(1 + \varepsilon)$ -approximation can be computed in  $k^{O(k/\varepsilon^8)}n^{O(1/\varepsilon^8)}$  time for any  $\varepsilon > 0$ , where  $k$  is the size of the optimum solution, or in  $f(\delta, \varepsilon)n^{O(1)}$  time for some computable function  $f$  and any  $\varepsilon > 0$  and  $0 < \delta < 1$ , where  $\delta$  is the shrinking factor. Moreover, a  $(1 + \varepsilon)$ -approximate kernel with  $k^{O(1/\varepsilon^8)}$  rectangles can be computed in polynomial time.*

Another special case of INDEPENDENT SET is the INDEPENDENT SET ON UNIT DISK GRAPH problem, where given set of  $n$  unit disks in the Euclidean plane, the task is to determine if there exists a set of  $k$  non-intersecting disks. The problem is NP-hard [CCJ90] but admits a PTAS [III+98]. Marx [Mar05] showed that, when parameterized by the solution size, the problem is  $W[1]$ -hard; this also rules out EPTAS (and even *efficient* PAS) for the problem, assuming  $FPT \neq W[1]$ . On the other hand, in [AF04] the authors give an FPT algorithm for a special case of INDEPENDENT SET ON UNIT DISK GRAPH when there is a lower bound on the distance between any pair of centers.

#### 4.1.2 Vertex Coloring

A problem related to INDEPENDENT SET is the VERTEX COLORING problem, for which the vertices need to be colored with integer values, such that no two adjacent vertices have the same color (which means that each color class forms an independent set in the graph). The task is to minimize the number of used colors. For planar graphs the problem has a polynomial time  $4/3$ -approximation algorithm [Mar08] via the celebrated Four Color Theorem, and a better approximation is not possible in polynomial time [Sto73]. Using this algorithm, a  $7/3$ -approximation can be computed in FPT time when parameterizing by the size of a planar deletion set [Mar08]. When generalizing planar graphs by excluding any fixed minor, and taking its size as the parameter, a 2-approximation can be computed in FPT time [DHK05]. Due to the NP-hardness for planar graphs [Sto73], neither of these two parameterizations admits a PAS, unless  $P=NP$ .

**Theorem 4.4** ([DHK05; Mar08]). *For the VERTEX COLORING problem*

- *a  $7/3$ -approximation can be computed in  $k^k n^{O(1)}$  time, where  $k$  is the size of a minimum planar deletion set, and*
- *a 2-approximation can be computed in  $f(k)n^{O(1)}$  time for some function  $f$ , where  $k$  is the size of an excluded minor of the input graph.*

One way to generalize VERTEX COLORING is to see each color class as an induced graph of degree 0. The DEFECTIVE COLORING problem<sup>10</sup> correspondingly asks for a coloring of the vertices, such that each color class induces a graph of maximum degree  $\Delta$ , for some given  $\Delta$ . The aim again is to minimize the number of used colors. In contrast to VERTEX COLORING, the DEFECTIVE COLORING problem is W[1]-hard [BLM18] parameterized by the *treewidth*. This parameter measures how “tree-like” a graph is, and is defined as follows.

**Definition 4.5.** A tree decomposition of a graph  $G = (V, E)$  is a tree  $T$  for which every node is associated with a bag  $X \subseteq V$ , such that the following properties hold:

1. the union of all bags is the vertex set  $V$  of  $G$ ,
2. for every edge  $(u, v)$  of  $G$ , there is a node of  $T$  for which the associated bag contains  $u$  and  $v$ , and
3. for every vertex  $u$  of  $G$ , all nodes of  $T$  for which the associated bags contain  $u$ , induce a connected subtree of  $T$ .

The width of a tree decomposition is the size of the largest bag minus 1 (which implies that a tree has a decomposition of width 1 where each bag contains the endpoints of one edge). The treewidth of a graph is the smallest width of any of its tree decompositions.

Treewidth is fundamental parameter of a graph and will be discussed more elaborately in Section 4.6.1. However, it is worth mentioning here that VERTEX COLORING is in FPT when parameterized by treewidth.

The strong polynomial-time approximation lower bound of  $n^{1-\varepsilon}$  for VERTEX COLORING [Zuc06] naturally carries over to the more general DEFECTIVE COLORING problem. A much improved approximation factor of 2 is possible though in FPT time if the parameter is the treewidth [BLM18]. It can be shown however, that a PAS is not possible in this case, as there is no  $(3/2 - \varepsilon)$ -approximation algorithm for any  $\varepsilon > 0$  parameterized by the treewidth [BLM18], unless  $\text{FPT} = \text{W}[1]$ . A natural question is whether the bound  $\Delta$  of DEFECTIVE COLORING can be approximated instead of the number of colors. For this setting, a bicriteria PAS parameterized by the treewidth exists [BLM18], which computes a solution with the optimum number of colors where each color class induces a graph of maximum degree at most  $(1 + \varepsilon)\Delta$ .

**Theorem 4.6 ([BLM18]).** For the DEFECTIVE COLORING problem, given a tree decomposition of width  $k$  of the input graph,

- a solution with the optimum number of colors where each color class induces a graph of maximum degree  $(1 + \varepsilon)\Delta$  can be computed in  $(k/\varepsilon)^{O(k)} n^{O(1)}$  time for any  $\varepsilon > 0$ ,
- a 2-approximation (of the optimum number of colors) can be computed in  $k^{O(k)} n^{O(1)}$  time, but
- no  $(3/2 - \varepsilon)$ -approximation (of the optimum number of colors) can be computed in  $f(k)n^{O(1)}$  time for any  $\varepsilon > 0$  and computable function  $f$ , unless  $\text{FPT} = \text{W}[1]$ .

The algorithms of the previous theorem build on the techniques of [Lam14] using approximate addition trees in combination with dynamic programs that yield XP algorithms for these problems. This technique can be applied to various problems (cf. Section 4.2), including a different generalization of VERTEX COLORING called EQUITABLE COLORING. Here the aim is to color the vertices of a graph with as few colors as possible, such that every two adjacent vertices receive different colors, and all color classes contain the same number of vertices. It is a generalization of VERTEX COLORING, since one may add a sufficiently large independent set (i.e., a set of isolated vertices) to a graph such that the number of colors needed for an optimum VERTEX COLORING solution is the same as for an optimum EQUITABLE COLORING solution.

<sup>10</sup>sometimes called IMPROPER COLORING.



The EQUITABLE COLORING problem is  $W[1]$ -hard even when combining the number of colors needed and the treewidth of the graph as parameters [Fel+11]. On the other hand, a PAS exists [Lam14] if the parameter is the *cliquewidth* of the input graph. This is a weaker parameter than treewidth, as the cliquewidth of a graph is bounded as a function of its treewidth. But while bounded treewidth graphs are sparse, cliquewidth also allows for dense graphs (such as complete graphs). Formally, a graph of cliquewidth  $\ell$  can be constructed using the following recursive operations using  $\ell$  labels on the vertices:

1. Introduce( $x$ ): create a graph containing a singleton vertex labelled  $x \in \{1, \dots, \ell\}$ .
2. Union( $G_1, G_2$ ): return the disjoint union of two vertex-labelled graphs  $G_1$  and  $G_2$ .
3. Join( $G, x, y$ ): add all edges connecting a vertex of label  $x$  with a vertex of label  $y$  to the vertex-labelled graph  $G$ .
4. Rename( $G, x, y$ ): change the label of every vertex of  $G$  with label  $x$  to  $y \in \{1, \dots, \ell\}$ .

A *cliquewidth expression* with  $\ell$  labels is a recursion tree describing how to construct a graph using the above four operations using only labels from the set  $\{1, \dots, \ell\}$ . Notice that the cliquewidth of a complete graph is two and therefore we have graphs of bounded clique-width but unbounded treewidth. Also, as stated earlier the cliquewidth of a graph is bounded above exponentially in its treewidth and this dependence is tight for some graph families [CR05].

The PAS for EQUITABLE COLORING will compute a coloring using at most  $k$  colors such that the ratio between the sizes of any two color classes is at most  $1 + \varepsilon$ . In this sense it is a *bicriteria* approximation algorithm.

**Theorem 4.7** ([Lam14]). *For the EQUITABLE COLORING problem, given a cliquewidth expression with  $\ell$  labels for the input graph, a solution with optimum number of colors where the ratio between the sizes of any two color classes is at most  $1 + \varepsilon$ , can be computed in  $(k/\varepsilon)^{O(k\ell)} n^{O(1)}$  time<sup>11</sup> for any  $\varepsilon > 0$ , where  $k$  is the optimum number of colors.*

A variant of VERTEX COLORING is the MIN SUM COLORING problem, where instead of minimizing the number of colors, the aim is to minimize the sum of (integer) colors, where the sum is taken over all vertices. This problem is FPT parameterized by the treewidth [Sal03], but the related MIN SUM EDGE COLORING problem is NP-hard [Mar09] on graphs of treewidth 2 (while being polynomial time solvable on trees [GK00]). For this problem the edges need to be colored with integer values, so that no two edges sharing a vertex have the same color, and the aim again is to minimize the total sum of colors. Despite being APX-hard [Mar09] and also paraNP-hard for parameter treewidth, MIN SUM EDGE COLORING admits a PAS for this parameter [Mar04].

**Theorem 4.8** ([Mar04]). *For the MIN SUM EDGE COLORING problem a  $(1 + \varepsilon)$ -approximation can be computed in  $f(k, \varepsilon)n$  time for any  $\varepsilon > 0$ , where  $k$  is the treewidth of the input graph.*

### 4.1.3 Subgraph Packing

A special family of packing problems can be obtained by *subgraph packing*. Let  $H$  be a fixed “pattern” graph. The  $H$ -PACKING problem, given the “host” graph  $G$ , asks to find the maximum number of vertex-disjoint copies of  $H$ . One can also let  $H$  be a family of graphs and ask the analogous problem. There is another choice whether each copy of  $H$  is required to be an induced subgraph or a regular subgraph. We focus on the regular subgraph case here.

<sup>11</sup>In [Lam14] the runtime of these algorithms is stated as  $(\log n/\varepsilon)^{O(k)} 2^{k\ell} n^{O(1)}$ , which can be shown to be upper bounded by  $(k/\varepsilon)^{O(k\ell)} n^{O(1)}$  (see e.g. [KLP19b, Lemma 1]).

When  $H$  is a single graph with  $k$  vertices, a simple greedy algorithm that finds an arbitrary copy of  $H$  and adds it to the packing, guarantees a  $k$ -approximation in time  $f(H, n) \cdot n$ . Here  $f(H, n)$  denotes the time to find a copy of  $H$  in an  $n$ -vertex graph. Following a general result for  $k$ -SET PACKING, a  $(k + 1 + \varepsilon)/3$ -approximation algorithm that runs in polynomial time for fixed  $k, \varepsilon$  exists. [Cyg13]. When  $H$  is 2-vertex-connected or a star graph, even for fixed  $k$ , it is NP-hard to approximate the problem better than a factor  $\Omega(k/\text{polylog}(k))$  [GL15]. There is no known connected  $H$  that admits an FPT (or even XP) algorithm achieving a  $k^{1-\delta}$ -approximation for some  $\delta > 0$ ; in particular, the parameterized approximability of  $k$ -PATH PACKING is wide open. It is conceivable that  $k$ -PATH PACKING admits an  $o(k)$ -parameterized approximation algorithm, given an  $O(\log k)$ -approximation algorithm for  $k$ -PATH DELETION [Lee17b] and an improved kernel for INDUCED  $P_3$  PACKING [Fom+19].

When  $H$  is the family of all cycles, the problem becomes the VERTEX CYCLE PACKING problem, for which the largest number of vertex-disjoint cycles of a graph needs to be found. No polynomial time  $O(\log^{1/2-\varepsilon} n)$ -approximation is possible for this problem [FS07] for any  $\varepsilon > 0$ , unless every problem in NP can be solved in randomized quasi-polynomial time. Furthermore, despite being FPT [Lok+19] parameterized by the solution size, VERTEX CYCLE PACKING does not admit any polynomial-sized exact kernel for this parameter [BTY11], unless  $\text{NP} \subseteq \text{coNP}/\text{poly}$ . Nevertheless, a PSAKS can be found [Lok+17].

**Theorem 4.9** ([Lok+17]). *For the VERTEX CYCLE PACKING problem a  $(1 + \varepsilon)$ -approximate kernel of size  $k^{O(1/(\varepsilon \log \varepsilon))}$  can be computed in polynomial time, where  $k$  is the solution size.*

#### 4.1.4 Scheduling

Yet another packing problem on graphs, which however has applications in scheduling and bandwidth allocation, is the UNSPLITTABLE FLOW ON A PATH problem. Here a path with edge capacities is given together with a set of tasks, each of which specifies a start and an end vertex on the path and a demand value. The goal is to find the largest number of tasks such that for each edge on the path the total demand of selected tasks for which the edge lies between its start and end vertex, does not exceed the capacity of the edge. This problem admits a QPTAS [Bat+15], but it remains a challenging open question whether a PTAS exists. When parameterizing by the solution size, UNSPLITTABLE FLOW ON A PATH is W[1]-hard [Wie17]. However a PAS exists [Wie17] for this parameter.

**Theorem 4.10** ([Wie17]). *For the UNSPLITTABLE FLOW ON A PATH problem a  $(1 + \varepsilon)$ -approximation can be computed in  $2^{O(k \log k)} n^{g(\varepsilon)}$  time for some computable function  $g$  and any  $\varepsilon > 0$ , where  $k$  is the solution size.*

Another scheduling problem is FLOW TIME SCHEDULING, for which a set of jobs is given, each of which is specified by a processing time, a release date, and a weight. The jobs need to be scheduled on a given number of machines, such that no job is processed before its release date and a job only runs on one machine at a time. Given a schedule, the *flow time* of a job is the weighted difference between its completion time and release date, and the task for the FLOW TIME SCHEDULING problem is to minimize the sum of all flow times. Two types of schedules are distinguished: in a *preemptive* schedule a job may be interrupted on one machine and then resumed on another, while in a *non-preemptive* schedule every job runs on one machine until its completion once it was started. If preemptive schedules are allowed, FLOW TIME SCHEDULING has no polynomial time  $O(\log^{1-\varepsilon} p)$ -approximation algorithm [GKM08], unless

$P = NP$ , where  $p$  is the maximum processing time. For the more restrictive non-preemptive setting, no  $O(n^{1/2-\epsilon})$ -approximation can be computed in polynomial time [KTW99], unless  $P = NP$ , where  $n$  is the number of jobs. The latter lower bound is in fact even valid for only one machine, and thus parameterizing FLOW TIME SCHEDULING by the number of machines will not yield any better approximation ratio in this setting. A natural parameter for FLOW TIME SCHEDULING is the maximum over all processing times and weights of the given jobs. It is not known whether the problem is FPT or W[1]-hard for this parameter. However, when combining this parameter with the number of machines, a PAS can be obtained [Wie18] despite the strong polynomial time approximation lower bounds.

**Theorem 4.11** ([Wie18]). *For the FLOW TIME SCHEDULING problem a  $(1 + \epsilon)$ -approximation can be computed in  $(mk)^{O(mk^3/\epsilon)}n^{O(1)}$  time in the preemptive setting, and in  $(mk/\epsilon)^{O(mk^5)}n^{O(1)}$  time in the non-preemptive setting, for any  $\epsilon > 0$ , where  $m$  is the number of machines and  $k$  is an upper bound on every processing time and weight.*

## 4.2 Covering Problems

For a covering problem the task is to select a set of  $k$  combinatorial objects in a mathematical structure, such as a graph or set system (i.e., hypergraph), under some constraints that demands certain other objects to be intersected/covered. A basic example is the SET COVER problem where we are given a set system, which is simply a collection of subsets of a universe. The goal is to determine whether there are  $k$  subsets whose union cover the whole universe.

There are two ways define optimization based on covering problems. First, we may view the covering demands as strict constraints and aim to find a solution that minimize the constraint/cost while covering all objects (i.e. relaxing the size- $k$  constraint); this results in a minimization problem. Second, we may view the size constraint as a strict constraint and aim to find a solution that covers as many objects as possible; this results in a maximization problem. We divide our discussion mainly into two parts, based on these two types of optimization problems. In Subsection 4.2.3, we discuss problems related to covering that fall into neither category.

### 4.2.1 Minimization Variants

We start out discussion with the minimization variants. For brevity, we overload the problem name and use the same name for the minimization variant (e.g. we use SET COVER instead of the more cumbersome MIN SET COVER). Later on, we will use different names for the maximization versions; hence, there will be no confusion.

**Set Cover, Dominating Set and Vertex Cover.** As discussed in detail in Section 3.1.2, SET COVER and equivalently DOMINATING SET are very hard to approximate in the general case. Hence, special cases where some constraints are placed on the set system are often considered. Arguably the most well-studied special case of SET COVER is the VERTEX COVER problem, in which the set system is a graph. That is, we would like to find the smallest set of vertices such that every edge has at least one endpoint in the selected set (i.e. the edge is “covered”). VERTEX COVER is well known to be FPT [BG93] and admit a linear-size kernel [NJ75]. A generalization of VERTEX COVER on  $d$ -uniform hypergraph, where the input is now a hypergraph and the goal is to find the smallest set of vertices such that every hyperedge contain at least one vertex from

the set, is also often referred to as  $d$ -HITTING SET in the parameterized complexity community. However, we will mostly use the nomenclature VERTEX COVER on  $d$ -uniform hypergraph because many algorithms generalize well from VERTEX COVER in graphs to hypergraphs. Indeed, branching algorithms for VERTEX COVER on graphs can be easily generalized to hypergraphs, and hence the latter is also FPT. Polynomial-size kernels are also known for VERTEX COVER on  $d$ -uniform hypergraphs [Abu10].

While VERTEX COVER both on graphs and  $d$ -uniform hypergraphs are already tractable, approximation can still help make algorithms even faster and kernels even smaller. We defer this discussion to Section 4.7.

**Connected Vertex Cover.** A popular variant of VERTEX COVER that is the CONNECTED VERTEX COVER problem, for which the computed solution is required to induce a connected subgraph of the input. Just as VERTEX COVER, the problem is FPT [Cyg12]. However, unlike VERTEX COVER, CONNECTED VERTEX COVER does not admit a polynomial-time kernel [DLS14], unless  $\text{NP} \subseteq \text{coNP}/\text{poly}$ . In spite of this, a PSAKS for CONNECTED VERTEX COVER exists:

**Theorem 4.12 ([Lok+17]).** *For any  $\varepsilon > 0$ , an  $(1 + \varepsilon)$ -approximate kernel with  $k^{O(1/\varepsilon)}$  vertices can be computed in polynomial time.*

The ideas behind [Lok+17] is quite neat and we sketch it here. There are two reduction rules: (i) if there exists a vertex with degree more than  $\Delta := 1/\varepsilon$  just “select” the vertex and (ii) if we see a vertex with more than  $k$  false twins, i.e. vertices with the same set of neighbors, then we simply remove it from the graph. An important observation for (i) is that, since we have to either pick the vertex or all  $\geq \Delta$  neighbors anyway, we might as well just select it even in the second case because it affects the size of the solution by a factor of at most  $\frac{1+\Delta}{\Delta} = 1 + \varepsilon$ . For (ii), it is not hard to see that we either select one of the false twins or all of them; hence, if a vertex has more than  $k$  false twins, then it surely cannot be in the optimal solution. Roughly speaking, these two observations show that this is an  $(1 + \varepsilon)$ -kernel. Of course, in the actual proof, “selecting” a vertex needs to be defined more carefully, but we will not do it here. Nonetheless, imagine the end step when we cannot apply these two reduction rules anymore. Essentially speaking, we end up with a graph where some (less than  $(1 + \varepsilon)k$ ) vertices are marked as “selected” and the remaining vertices have degree at most  $\Delta$ . Now, every vertex is either inside the solution, or all of its neighbors must lie in the solution. There are only (at most)  $k$  vertices in the first case. For the second case, note that these vertices have degree at most  $\Delta$  and they have at most  $k$  false twins, meaning that there are at most  $k^{1+\Delta} = k^{1+1/\varepsilon}$  such vertices. In other words, the kernel is of size  $k^{O(1/\varepsilon)}$  as desired. This constitutes the main ideas in the proof; let us stress again that the actual proof is of course more complicated than this since we did not define rule (i) formally.

Recently, Krithika et al. [KMR18] considered the following structural parameters beyond the solution size: split deletion set, clique cover and cluster deletion set. In each case, the authors provide a PSAKS for the problem. We will not fully define these parameters here, but we note that the first parameter (split deletion set) is always no larger than the size of the minimum vertex cover of the graph. In another very recent work, Majumdar et al. [DM20] give a PSAKS for each of the following parameters, each of which is always no larger than the solution size: the deletion distance of the input graph to the class of cographs, the class of bounded treewidth graphs, and the class of all chordal graphs. Hence, these results may be viewed as a generalization of the aforementioned PSAKS from [Lok+17].

**Connected Dominating Set.** Similar to CONNECTED VERTEX COVER, the CONNECTED DOMINATING SET problem is the variant of DOMINATING SET for which the solution additionally needs to induce a connected subgraph of the input graph. When placing no restriction on the input graph, the problem is as hard to approximate as DOMINATING SET. However, for some special classes of graphs, PSAKS or bi-PSAKS<sup>12</sup> are known; these include graphs with bounded expansion, nowhere dense graphs, and  $d$ -biclique-free graphs [Eib+18].

**Covering Problems parameterized by Graph Width Parameters.** Several works in literature also study the approximability of variants of VERTEX COVER and DOMINATING SET parameterized by graph widths [Ang+16; Lam14]. These variants include:

- POWER VERTEX COVER (PVC). Here, along with the input graph, each edge has an integer demand and we have to assign (power) values to vertices, such that each edge has at least one endpoint with a value at least its demand. The goal is to minimize the total assigned power. Note that this generalizes VERTEX COVER, where edges have unit demands.
- CAPACITATED VERTEX COVER (CVC). The problem is similar to VERTEX COVER, except that each vertex has a capacity which limits the number of edges that it can cover. Once again, VERTEX COVER is a special case of CVC where each vertex's capacity is  $\infty$ .
- CAPACITATED DOMINATING SET (CDS). Analogous to CVC, this is a generalization of DOMINATING SET where each vertex has a capacity and it can only cover/dominate at most that many other vertices.

All problems above are FPT under standard parameter (i.e. the optimum) [Ang+16; Dom+08]. However, when parameterizing by the treewidth<sup>13</sup>, all three problems become W[1]-hard [Ang+16]. (This is in contrast to VERTEX COVER and DOMINATING SET, both of which admit straightforward dynamic programming FPT algorithms parameterized by treewidth.) Despite this, good FPT approximation algorithms are known for the problem. In particular, a PAS is known for PVC [Ang+16]. For CVC and CDS, a bicriteria PAS exists for the problem [Lam14], which in this case computes a solution of size at most the optimum, so that no vertex capacity is violated by more than a factor of  $1 + \varepsilon$ .

The approximation algorithms for CVC and CDS are a result of a more general approach of Lampis [Lam14]. The idea is to execute an “approximate” version of dynamic programming in tree decomposition instead of the exact version; this helps reduce the running time from  $n^{O(w)}$  to  $(\log n / \varepsilon)^{O(w)}$ , which is FPT. The approach is quite flexible: several approximation for graphs problems including covering problems can be achieved via this method and it also applies to clique-width. Please refer to [Lam14] for more details.

**Packing-Covering Duality and Erdős-Pósa Property.** Given a set system  $(V, \mathcal{C})$  where  $V$  is the universe and  $\mathcal{C} = \{C_1, \dots, C_m\}$  is a collection of subsets of  $V$ , HITTING SET is the problem of computing the smallest  $S \subseteq V$  that intersects every  $C_i$ , and SET PACKING is the problem of computing the largest subcollection  $\mathcal{C}' \subseteq \mathcal{C}$  such that no two sets in  $\mathcal{C}'$  intersect. It can also be observed that the optimal value for HITTING SET is at least the optimal value for SET PACKING, while the standard LP relaxations for them (*covering LP* and *packing LP*) have the

---

<sup>12</sup>Recall that a *bi-kernel* is similar to a kernel except that its the output need not be an instance of the original problem. Bi-PSAKS can be defined analogously to PSAKS, but with bi-kernel instead of kernel. In the case of CONNECTED DOMINATING SET, the bi-kernel outputs an instance of an annotated variant of CONNECTED DOMINATING SET, where some vertices are marked and do not need to be covered by the solution.

<sup>13</sup>see Definition 4.5 for the definition of the treewidth.

same optimal value by strong duality. Studying the other direction of the inequality (often called the *packing-covering duality*) for natural families of set systems has been a central theme in combinatorial optimization. The gap between the covering optimum and packing optimum is large in general (e.g., DOMINATING SET / INDEPENDENT SET), but can be small for some families of set systems (e.g.,  $s$ - $t$  CUT /  $s$ - $t$  DISJOINT PATHS and VERTEX COVER / MATCHING especially in bipartite graphs).

One notion that has been important for both parameterized and approximation algorithms is the Erdős-Pósa property [EP65]. A family of set systems is said to have the Erdős-Pósa property when there is a function  $f : \mathbb{N} \rightarrow \mathbb{N}$  such that for any set system in the family, if the packing optimum is  $k$ , the covering optimum is at most  $f(k)$ . This immediately implies that the multiplicative gap between these two optima is at most  $f(k)/k$ , and constructive proofs for the property for various set systems have led to  $(f(k)/k)$ -approximation algorithms. Furthermore, for some problems including CYCLE PACKING, the Erdős-Pósa property gives an immediate parameterized algorithm. We refer the reader to a recent survey [RT17] and papers [KK18; Lok+19; VB+19].

The original paper of Erdős and Pósa [EP65] proved the property for set systems  $(V, \mathcal{C})$  when there is an underlying graph  $G = (V, E)$  and  $\mathcal{C}$  is the set of cycles, which corresponds to the pair CYCLE PACKING / FEEDBACK VERTEX SET; every graph either has at least  $k$  vertex-disjoint cycles or there is a feedback vertex set of size at most  $O(k \log k)$ . Many subsequent papers also studied natural set systems arising from graphs where  $V$  is the set of vertices or edges and  $\mathcal{C}$  denotes a collection of subgraphs of interest. For those set systems, Erdős-Pósa Properties are closely related to Set Packing introduced in Section 4.1.3 and  $\mathcal{F}$ -DELETION problems introduced in Section 4.6.

## 4.2.2 Maximization Variants

We now move on to the maximization variants of covering problems. To our knowledge, these covering problems are much less studied in the context of parameterized approximability compared to their minimization counterparts. In particular, we are only aware of works on the maximization variants of SET COVER and VERTEX COVER, which are typically called MAX  $k$ -COVERAGE and MAX  $k$ -VERTEX COVER respectively.

**Max  $k$ -Coverage.** Recall that here we are given a set system and the goal is to select  $k$  subsets whose union is maximized. It is well known that the simple greedy algorithm yields an  $(\frac{e}{e-1})$ -approximation [CNW80]. Furthermore, Fiege shows, in his seminal work [Fei98], show that this is tight:  $(\frac{e}{e-1} - \varepsilon)$ -approximation is NP-hard for any constant  $\varepsilon > 0$ . In fact, recently it has been shown that this inapproximability applies also to the parameterized setting. Specifically, under Gap-ETH,  $(\frac{e}{e-1} - \varepsilon)$ -approximation cannot be achieved in FPT time [Coh+19] or even  $f(k) \cdot n^{o(k)}$  time [Man]. In other words, the trivial algorithm is tight in terms of running time, the greedy algorithm is tight in terms of approximation ratio, and there is essentially no trade-off possible between these two extremes. We remark here that this hardness of approximation is also the basis of hardness for  $k$ -MEDIAN and  $k$ -MEANS [Coh+19] (see Section 4.3).

Due to the strong inapproximability result for the general case of MAX  $k$ -COVERAGE, different parameters have to be considered in order to obtain a PAS for MAX  $k$ -COVERAGE. An interesting positive result here is when the parameters are  $k$  and the VC dimension of the set system, for which a PAS exists while the exact version of the problem is W[1]-hard [BKL12].

**Max  $k$ -Vertex Cover.** Another special case of MAX  $k$ -COVERAGE is the restriction when each element belongs to at most  $d$  subsets in the system. This corresponds exactly to the maximization variant of the VERTEX COVER problem on  $d$ -uniform hypergraph, which will refer to as MAX  $k$ -VERTEX COVER. Note here that, for such set systems, their VC-dimensions are also bounded by  $\log d + 1$  and hence the aforementioned PAS of [BKL12] applies here as well. Nonetheless, MAX  $k$ -VERTEX COVER admits a much simpler PAS (and even PSAKS) compared to MAX  $k$ -COVERAGE parameterized by  $k$  and VC-dimension, as we will discuss more below.

MAX  $k$ -VERTEX COVER was first studied in the context of parameterized complexity by Guo et al. [GNW07] who showed that the problem is W[1]-hard. Marx, in his survey on parameterized approximation algorithms [Mar08], gave a PAS for the problem with running time  $2^{\tilde{O}(k^3/\varepsilon)}$ . Later, Lokshtanov et al. [Lok+17] shows that Marx’s approach can be used to give a PSAKS of size  $O(k^5/\varepsilon^2)$ . Both of these results mainly focus on graphs. Later, Skowron and Faliszewski [SF17]<sup>14</sup> gave a more general argument that both works generally for any  $d$ -uniform hypergraph and improves the running time and kernel size:

**Theorem 4.13** ([SF17]). *For the MAX  $k$ -VERTEX COVER problem in  $d$ -uniform hypergraphs, an  $(1 + \varepsilon)$ -approximation can be computed in  $O^*((d/\varepsilon)^k)$  time for any  $\varepsilon > 0$ . Moreover, an  $(1 + \varepsilon)$ -approximate kernel with  $O(dk/\varepsilon)$  vertices can be computed in polynomial time.*

The main idea of the above proof is simple and elegant, and hence we will include it here. For convenience, we will only discuss the graph case, i.e.,  $d = 2$ . It suffices to just give the  $O(k/\varepsilon)$ -vertex kernel; the PAS immediately follows by running the brute force algorithm on the output instance from the kernel. The kernel is as simple as it gets: just keep  $2k/\varepsilon$  vertices with highest degrees and throw the remaining vertices away! Note that there is a subtle point here, which is that we do not want to throw away the edges linking from the kept vertices to the remaining vertices. If self-loops are allowed in graph, this is not an issue since we may just add a self-loop to each vertex for each edge adjacent to it with the other endpoint being discarded. When self-loops are not allowed, it is still possible to overcome this issue but with slightly larger kernel; we refer the readers to Section 3.2 of [Man18a] for more detail.

Having defined the kernel, let us briefly discuss the intuition as to why it works. Let  $V_{2k/\varepsilon}$  denote the set of  $2k/\varepsilon$  highest-degree vertices. The main argument of the proof is that, if there is an optimal solution  $S$ , then we may modify it to be entirely contained in  $V_{2k/\varepsilon}$  while preserving the number of covered edges to within  $(1 + \varepsilon)$  factor. The modification is simple: for every vertex that is outside of  $V_{2k/\varepsilon}$ , we replace it with a random vertex from  $V_{2k/\varepsilon}$ . Notice here that we always replace a vertex with a higher-degree vertex. Naturally, this should be good in terms of covering more edges, but there is a subtle point here: it is possible that the high degree vertices are “double counted” if a particular edge is covered by both endpoints. The size  $2k/\varepsilon$  is selected exactly to combat this issue; since the set is large enough, “double counting” is rare for random vertices. This finishes our outline for the intuition.

We end by remarking that MAX  $k$ -VERTEX COVER on graphs is already APX-hard [Pet94], and hence the PASes mentioned above once again demonstrate additional power of FPT approximation algorithms over polynomial-time approximation algorithms.

<sup>14</sup>The argument of [SF17] was later independently rediscovered in [Man18a] as well.

### 4.2.3 Other Related Problems

There are several other covering-related problems that do not fall into the two categories we discussed so far. We discuss a couple such problems below.

**Min  $k$ -Uncovered.** The first is the MIN  $k$ -UNCOVERED problem, where the input is a set system and we would like to select  $k$  sets as to minimize the number of uncovered elements. When we are concerned with exact solutions, this is of course the SET COVER. However, the optimization version becomes quite different from MAX  $k$ -COVERAGE. In particular, since it is hard to determine whether we can find  $k$  subsets that cover the whole universe, the problem is not approximable at all in the general case. However, if restrict ourselves to graphs and hypergraphs (for which we refer to the problem as MIN  $k$ -VERTEX COVER), it is possible to get a (randomized) PAS for the problem [SF17]:

**Theorem 4.14** ([SF17]). *For the MIN  $k$ -VERTEX UNCOVERED problem in  $d$ -uniform hypergraphs, an  $(1 + \varepsilon)$ -approximation can be computed in  $O^*((d/\varepsilon)^k)$  time for any  $\varepsilon > 0$ .*

The algorithm is based on the following simple randomized branching: pick a random uncovered element and branch on all possibilities of selecting a subset that contains it. Notice that since an element belongs to only  $d$  subsets, the branching factor is at most  $d$ . The key intuition in the approximation proof is that, when the number of elements we have covered so far is still much less than that in the optimal solution, there is a relatively large probability (i.e.  $\varepsilon$ ) that the random element is covered in the optimal solution. If we always pick such a “good” element in most branching steps, then we would end up with a solution close to the optimum. Skowron and Faliszewski [SF17] formalizes this intuition by showing that the algorithm outputs an  $(1 + \varepsilon)$ -approximate solution with probability roughly  $\varepsilon^k$ . Hence, by repeating the algorithm  $(1/\varepsilon)^k$  time, one arrives at the claimed PAS. To the best of our knowledge, it is unknown whether a PSAKS exists for the problem.

**Min  $k$ -Coverage.** Another variant of the SET COVER problem studied is the MIN  $k$ -COVERAGE<sup>15</sup>, where we would like to select  $k$  subsets that *minimizes* the number of covered elements. We stress here that this problem is *not* a relaxation of SET COVER but rather is much more closely related to graph expansion problems (see [CDM17]).

It is known that, when there is no restriction on the input set system, the problem is (up to a polynomial factor) as hard to approximate as the DENEST  $k$ -SUBGRAPH problem [Chl+18]. Hence, by the inapproximability of the latter discussed earlier in the survey (Theorem 3.19), we also have that there is no  $k^{o(1)}$ -approximation algorithm for the problem that runs in FPT time.

Once again, the special case that has been studied in literature is when the input set system is a graph, in which case we refer to the problem as MIN  $k$ -VERTEX COVER. Gupta, Lee and Li [GLL18a; GLL18b] used the technique of Marx [Mar08] to give a PAS for the problem with running time  $O^*((k/\varepsilon)^{O(k)})$ . The running time was later improved in [Man18a] to  $O^*((1/\varepsilon)^{O(k)})$ . The algorithm there is again based on branching, but the rules are more delicate and we will not discuss them here. An interesting aspect to note here is that, while both MAX  $k$ -VERTEX COVER and MIN  $k$ -VERTEX COVER have PSAKS of the (asymptotically) same running time, the former

---

<sup>15</sup>The problem has also been referred to as MIN  $k$ -UNION and SMALL SET BIPARTITE VERTEX EXPANSION in literature [CDM17; Chl+18].



admits a PSAKS whereas the latter does not (assuming a variant of the Small Set Expansion Conjecture) [Man18a].

To the best of our knowledge, MIN  $k$ -VERTEX COVER has not been explicitly studied on  $d$ -uniform hypergraphs before, but we suspect that the above results should carry over from graphs to hypergraphs as well.

### 4.3 Clustering

Clustering is a representative task in unsupervised machine learning that has been studied in many fields. In combinatorial optimization communities, it is often formulated as the following: Given a set  $P$  of points and a set  $F$  of candidate centers (also known as facilities), and a metric on  $X \supseteq P \cup F$  given by the distance  $\rho : X \times X \rightarrow \mathbb{R}^+ \cup \{0\}$ , choose  $k$  centers  $C \subseteq F$  to minimize some objective function  $\text{cost} := \text{cost}(P, C)$ . To fully specify the problem, the choices to make are the following. Let  $\rho(C, p) := \min_{c \in C} \rho(c, p)$ .

- Objective function: Three well-studied objective functions are
  - $k$ -MEDIAN ( $\text{cost}(P, C) := \sum_{p \in P} \rho(C, p)$ ).
  - $k$ -MEANS ( $\text{cost}(P, C) := \sum_{p \in P} \rho(C, p)^2$ ).
  - $k$ -CENTER ( $\text{cost}(P, C) := \max_{p \in P} \rho(C, p)$ ).
- Metric space: The ambient metric space  $X$  can be
  - A general metric space explicitly given by the distance  $\rho : X \times X \rightarrow \mathbb{R}^+ \cup \{0\}$ .
  - The Euclidean space  $\mathbb{R}^d$  equipped with the  $\ell_2$  distance.
  - Other structured metric spaces including metrics with *bounded doubling dimension* or *bounded highway dimension*.

While many previous results on clustering focused on non-parameterized polynomial time, there are at least three natural parameters one can parameterize: The number of clusters  $k$ , the dimension  $d$  (if defined), and the approximation accuracy parameter  $\varepsilon$ . In general metric spaces, parameterized approximation algorithms (mainly with parameter  $k$ ) were considered very recently, but in Euclidean spaces, many previous results already give parameterized approximation algorithms with parameters  $k, d$ , and  $\varepsilon$ .

#### 4.3.1 General metric space

We can assume  $X = P \cup F$  without loss of generality. Let  $n := |X|$  and note that the distance  $\rho : X \times X \rightarrow \mathbb{R}^+ \cup \{0\}$  is explicitly specified by  $\Theta(n^2)$  numbers. A simple exact algorithm running in time  $O(n^{k+1})$  can be achieved by enumerating all  $k$  centers  $c_1, \dots, c_k \in F$  and assign each point  $p$  to the closest center. In this setting, the best approximation ratios achieved by polynomial time algorithms are  $2.611 + \varepsilon$  for  $k$ -MEDIAN [Byr+14],  $9 + \varepsilon$  for  $k$ -MEANS [Kan+04], and 3 for  $k$ -CENTER [Gon85].<sup>16</sup> From the hardness side, it is NP-hard to approximate  $k$ -MEDIAN within a factor  $1 + 2/e - \varepsilon \approx 1.73 - \varepsilon$ ,  $k$ -MEDIAN within a factor  $1 + 8/e - \varepsilon \approx 3.94 - \varepsilon$ ,  $k$ -CENTER within a factor  $3 - \varepsilon$  [GK99].

While there are some gaps between the best algorithms and the best hardness results for  $k$ -MEDIAN and  $k$ -MEANS, it is an interesting question to ask how parameterization by  $k$  changes the approximation ratios for both problems. Cohen-Addad et al. [Coh+19] studied this question and gave exact answers.

<sup>16</sup>A special case that has received significant attention assumes  $P = F$ . In this case, the best approximation ratio for  $k$ -CENTER becomes 2.

**Theorem 4.15** ([Coh+19]). *For any  $\varepsilon > 0$ , there is an  $(1 + 2/e + \varepsilon)$ -approximation algorithm for  $k$ -MEDIAN, and an  $(1 + 8/e + \varepsilon)$ -approximation algorithm for  $k$ -MEANS, both running in time  $(O(k \log k / \varepsilon^2))^{kn^{O(1)}}$ .*

*There exists a function  $g : \mathbb{R}^+ \rightarrow \mathbb{R}^+$  such that assuming the Gap-ETH, for any  $\varepsilon > 0$ , any  $(1 + 2/e - \varepsilon)$ -approximation algorithm for  $k$ -MEDIAN, and any  $(1 + 8/e - \varepsilon)$ -approximation algorithm for  $k$ -MEANS, must run in time at least  $n^{k^g(\varepsilon)}$ .*

These results show that if we parameterize by  $k$ ,  $1 + 2/e$  (for  $k$ -MEDIAN) and  $1 + 8/e$  (for  $k$ -MEANS) are the exact limits of approximation for parameterized approximation algorithms. Similar reductions also show that no parameterized approximation algorithm can achieve  $(3 - \varepsilon)$ -approximation for  $k$ -CENTER for any  $\varepsilon > 0$  (only assuming  $W[2] \neq \text{FPT}$ ), so the power of parameterized approximation is exactly revealed for all three objective functions.

**Algorithm for  $k$ -MEDIAN.** We briefly describe ideas for the algorithm for  $k$ -MEDIAN in [Theorem 4.15](#). The main technical tool that the algorithm uses is a *coreset*, which will be also frequently used for Euclidean subspaces in the next subsection.

When  $S$  is a set of points with weight functions  $w : S \rightarrow \mathbb{R}^+$ , let us extend the definition of the objective function  $\text{cost}(S, C)$  such that

$$\text{cost}(S, C) := \sum_{p \in S} w(p) \cdot \rho(C, p).$$

Given a clustering instance  $(P, F, \rho, k)$  and  $\varepsilon > 0$ , a subset  $S \subseteq P$  with weight functions  $w : S \rightarrow \mathbb{R}^+$  is called a (strong) coreset if for any  $k$  centers  $C = \{c_1, \dots, c_k\} \subseteq F$ ,

$$|\text{cost}(S, C) - \text{cost}(P, C)| \leq \varepsilon \cdot \text{cost}(P, C).$$

For a general metric space, Chen [Che06] gave a coreset of cardinality  $\tilde{O}(k^2 \log^2 n / \varepsilon^2)$ . (In this subsection,  $\tilde{O}(\cdot)$  hides  $\text{poly}(\log \log n, \log k, \log(1/\varepsilon))$ .) This was improved by Feldman and Langeberg [FL11] to  $O(k \log n / \varepsilon^2)$ . We introduce high-level ideas of [Che06] later.

Given the coreset, it remains to give a good parameterized approximation algorithm for the problem for a much smaller (albeit weighted) point set  $|P| = O(\text{poly}(\log n, k))$ . Note that  $|F|$  can be still as large as  $n$ , so naively choosing  $k$  centers from  $F$  will take  $n^k$  time and exhaustively partitioning  $P$  into  $k$  sets will take  $k^{|P|} = n^{\text{poly}(k)}$  time. (Indeed, exactly solving this small case will give an EPAS, which will contradict the Gap-ETH.)

Fix an optimal solution, and let  $C^* = \{c_1^*, \dots, c_k^*\}$  are the optimal centers and  $P_i^* \subseteq P$  is the cluster assigned to  $c_i^*$ . One information we *can* guess is, for each  $i \in [k]$ , the point  $p_i \in P_i^*$  closest to  $c_i^*$  and (approximate)  $\rho(c_i^*, p_i)$ . Since  $|P| = \text{poly}(k \log n)$ , guessing them only takes time  $(k \log n)^{O(k)}$ , which can be made FPT by separately considering the case  $(\log n)^k \leq n$  and the case  $(\log n)^k \geq n$ , in which  $k = \Omega(\log n / \log \log n)$  and  $(\log n)^k = (k \log k)^{O(k)}$ .

Let  $F_i \subseteq F$  be the set of candidate centers that are at distance approximately  $r_i$  from  $p_i$ , so that  $c_i^* \in F_i$  for each  $i$ . The algorithm chooses  $k$  centers  $C \subseteq F$  such that  $|C \cap F_i| \geq 1$  for each  $i \in [k]$ . Let  $c_i \in C \cap F_i$ . For any point  $p \in P$  (say  $p \in P_j^*$ , though the algorithm doesn't need to know  $j$ ), then

$$\rho(C, p) \leq \rho(c_j, p) \leq \rho(c_j, p_j) + \rho(p_j, c_j^*) + \rho(c_j^*, p) \leq 3\rho(c_j^*, p),$$

where  $\rho(c_j, p_j) \approx \rho(p_j, c_j^*) \leq \rho(c_j^*, p)$  by the choice of  $p_j$ .

This immediately gives a 3-approximation algorithm in FPT time, which is worse than the best polynomial time approximation algorithm. To get the optimal  $(1 + 2/e)$ -approximation algorithm, we further reduce the job of finding  $c_i \in F_i$  to *maximizing a monotone submodular function with a partition matroid constraint*, which is known to admit an optimal  $(1 - 1/e)$ -approximation algorithm [Cal+11]. Then we can ensure that for  $(1 - 1/e)$  fraction of points, the distance to the chosen centers is shorter than in the optimal solution, and for the remaining  $1/e$  fraction, the distance is at most three times the distance in the optimal solution. We refer the reader to [Coh+19] for further details.

**Constructing a coreset.** As discussed above, a coreset is a fundamental building block for optimal parameterized approximation algorithms for  $k$ -MEDIAN and  $k$ -MEANS for general metrics. We briefly describe the construction of Chen [Che06] that gives a coreset of cardinality  $\tilde{O}(k^2 \log^2 n / \varepsilon^2)$  for  $k$ -MEDIAN. Similar ideas can be also used to obtain an EPAS for Euclidean spaces parameterized by  $k$ , though better specific constructions are known in Euclidean spaces.

We first partition  $P$  into  $P_1, \dots, P_\ell$  such that  $\ell = O(k \log n)$  and

$$\sum_{i=1}^{\ell} |P_i| \text{diam}(P_i) = O(\text{OPT}).$$

Such a partition can be obtained by using a known (bicriteria) constant factor approximation algorithm for  $k$ -MEDIAN. Next, let  $t = \tilde{O}(k \log n)$ , and for each  $i = 1, \dots, \ell$ , we let  $S_i = \{s_1, \dots, s_t\}$  be a random subset of  $t$  points of  $P_i$  where each  $s_j$  is an independent and uniform sample from  $P_i$  and is given weight  $|P_i|/t$ . (If  $|P_i| \leq t$ , we simply let  $S_i = P_i$  with weights 1.) The final coreset  $S$  is the union of all  $S_i$ 's.

To prove that it works, we simply need to show that for any set of  $k$  centers  $C \subseteq F$  with  $|C| = k$ ,

$$\Pr[|\text{cost}(S, C) - \text{cost}(P, C)| > \varepsilon \cdot \text{cost}(P, C)] \leq o(1/n^k),$$

so that the union bound over  $\binom{n}{k}$  choices of  $C$  works. Indeed, we show that for each  $i = 1, \dots, \ell$ ,

$$\Pr[|\text{cost}(S_i, C) - \text{cost}(P_i, C)| > \varepsilon \cdot |P_i| \text{diam}(P_i)] \leq o(1/(\ell \cdot n^k)), \quad (1)$$

so that we can also union bound and sum over  $i \in [\ell]$ , using the fact that  $\sum_i |P_i| \text{diam}(P_i) = O(\text{OPT}) \leq O(\text{cost}(P, C))$ .

It is left to prove (1). Fix  $C$  and  $i$  (let  $P_i = \{p_1, \dots, p_{|P_i|}\}$ ), and recall that

$$\text{cost}(P_i, C) = \sum_{j=1}^{|P_i|} \rho(C, p_j).$$

When  $|P_i| \leq t$ ,  $S_i = P_i$ , so (1) holds. Otherwise, recall that  $S_i = \{s_1, \dots, s_t\}$  where each  $s_j$  is an independent and uniform sample from  $P_i$  with weight  $w := |P_i|/t$ . For  $j = 1, \dots, t$ , let  $X_j := w \cdot \rho(C, s_j)$ . Note that  $\text{cost}(S_i, C) = \sum_j X_j$  and  $\text{cost}(P_i, C) = t \cdot \mathbb{E}[X_j] = \mathbb{E}[\text{cost}(S_i, C)]$ . A crucial observation is that that  $|\rho(C, p_j) - \rho(C, p_{j'})| \leq \text{diam}(P_i)$  for any  $j, j' \in [|P_i|]$ , so that  $|X_j - X_{j'}| \leq w \cdot \text{diam}(P_i)$  for any  $j, j' \in [t]$ . If we let  $X_{\min} := \min_{j \in [|P_i|]} (w\rho(C, p_j))$  and  $Y_j := (X_j - X_{\min}) / (w \cdot \text{diam}(P_i))$ ,  $Y_j$ 's are  $t$  i.i.d. random variables that are supported in  $[0, 1]$ .

The standard Chernoff-Hoeffding inequality gives

$$\Pr \left[ \left| \sum_j X_j - tE[X_j] \right| > \varepsilon |P_i| \text{diam}(P_i) \right] = \Pr \left[ \left| \sum_j Y_j - tE[Y_j] \right| > \varepsilon t \right] \leq \exp(O(\varepsilon^2 t)) \leq o(1/(\ell \cdot n^k)),$$

proving (1) for  $t = \tilde{O}(k \log n)$  and finishing the proof. A precise version of this argument was stated in Haussler [Hau92].

### 4.3.2 Euclidean space

For Euclidean spaces, we assume that  $X = F = \mathbb{R}^d$  for some  $d \in \mathbb{N}$ , endowed with the standard  $\ell_2$  metric. Let  $n = |P|$  in this subsection. Now we have  $k$  and  $d$  as natural structural parameters of clustering tasks. Many previous approximation algorithms in EUCLIDEAN  $k$ -MEDIAN and EUCLIDEAN  $k$ -MEANS in Euclidean spaces, without explicit mention to parameterized complexity, are parameterized approximation algorithms parameterized by  $k$  or  $d$  (or both). The highlight of this subsection is that for both EUCLIDEAN  $k$ -MEDIAN and EUCLIDEAN  $k$ -MEANS, an EPAS exists with only one of  $k$  and  $d$  as a parameter. Without any parameterization, both EUCLIDEAN  $k$ -MEDIAN and  $k$ -MEANS are known to be APX-hard [CK19; LSW17]. We introduce these results in the chronological order, highlighting important ideas.

**EUCLIDEAN  $k$ -MEDIAN with parameter  $d$ .** The first PTAS for EUCLIDEAN  $k$ -MEDIAN in Euclidean spaces with fixed  $d$  appears in Arora et al. [ARR98]. The techniques extend Arora's previous PTAS for the EUCLIDEAN TRAVELING SALESMAN problem in Euclidean spaces [Aro98], first proving that there exists a near-optimal solution that interacts with a *quadtrees* (a geometric division of  $\mathbb{R}^d$  into a hierarchy of square regions) in a restricted sense, and finally finding such a tour using dynamic programming. The running time is  $n^{O(1/\varepsilon)}$  for  $d = 2$  and  $n^{(\log n/\varepsilon)^{d-2}}$  for  $d > 2$ . Kolliopoulos and Rao [KR99] improved the running time to  $2^{O((\log(1/\varepsilon)/\varepsilon)^{d-1})} n \log^{d+6} n$ , which is an EPAS with parameter  $d$ .

**EUCLIDEAN  $k$ -MEDIAN and EUCLIDEAN  $k$ -MEANS with parameter  $k$ .** An EPAS for EUCLIDEAN  $k$ -MEANS even with parameters both  $k$  and  $d$  took longer to be discovered, and first appeared when Matoušek [Mat00] gave an approximation scheme that runs in time  $O(n\varepsilon^{2k^2d} \log^k n)$ . After this, several improvements on the running time followed Bădoiu et al. [BHPI02], De La Vega et al. [DLV+03] and Har-Peled and Mazumdar [HPM04].

Kumar et al. [KSS04; KSS05] gave approximation schemes for both  $k$ -MEDIAN and  $k$ -MEANS, running in time  $2^{(k/\varepsilon)^{O(1)}} dn$ . This shows that an EPAS can be obtained by using only  $k$  as a parameter. Using this result and improved coresets, more improvements followed [Che06; FL11; FMS07]. The current best runtime to get  $(1 + \varepsilon)$ -approximation is  $O(ndk + d \cdot \text{poly}(k/\varepsilon) + 2^{\tilde{O}(k/\varepsilon)})$  for  $k$ -MEANS [FMS07], and  $O(ndk + 2^{\text{poly}(1/\varepsilon, k)})$  time for  $k$ -MEDIAN [FL11].

A crucial property of the Euclidean space that allows an EPAS with parameter  $k$  (which is ruled out for general metrics by Theorem 4.15) is the *sampling property*, which says that for any set  $Q \subseteq \mathbb{R}^d$  as one cluster, there is an algorithm that is given only  $g(1/\varepsilon)$  samples from  $Q$  and outputs  $h(1/\varepsilon)$  candidate centers such that one of them is  $\varepsilon$ -close to the optimal center for the entire cluster  $Q$  for some functions  $g, h$ . (E.g., for  $k$ -MEANS, the mean of  $O(1/\varepsilon)$  random samples  $\varepsilon$ -approximates the actual mean with constant probability.) This idea leads

to an  $(1 + \varepsilon)$ -approximation algorithm running in time  $|P|^{f(\varepsilon, k)}$ . Together even with a general coreset construction of size  $\text{poly}(k, \log n, 1/\varepsilon)$ , one already gets an EPAS with parameter  $k$ . Better coresets construction are also given in Euclidean spaces. Recent developments [Bec+19; HV20; SW18] construct core-sets of size  $\text{poly}(k, 1/\varepsilon)$  (no dependence on  $n$  or  $d$ ), which is further extended to the shortest-path metric of an excluded-minor graph [Bra+20].

**EUCLIDEAN  $k$ -MEANS with parameter  $d$ .** Cohen-Addad et al. [CKM19] and Friggstad et al. [FRS19] recently gave approximation schemes running in time  $n^{f(d, \varepsilon)}$  using local search techniques. These results were improved to an EPAS in [Coh18], and also extended to doubling metrics [CFS19].

**Other metrics and  $k$ -CENTER.** For the  $k$ -CENTER problem an EPAS exists when parametrizing by both  $k$  and the doubling dimension [FM18], and also for planar graphs there is an EPAS for parameter  $k$ , which is implied by the EPTAS of Fox-Epstein et al. [FEKS19] (cf. [FM18]).

There are also parameterized approximation schemes for metric spaces with *bounded highway dimension* [BKS18; Fel15; FM18] and various graph width parameters [KLP19b].

**Capacitated clustering and other variants.** Another example where the parameterization by  $k$  helps is CAPACITATED  $k$ -MEDIAN, where each possible center  $c \in F$  has a capacity  $u_c \in \mathbb{N}$  and can be assigned at most  $u_c$  points. It is not known whether there exists a constant-factor approximation algorithm, and known constant factor approximation algorithms either open  $(1 + \varepsilon)k$  centers [Li16] or violate capacity constraints by an  $(1 + \varepsilon)$  factor [DL16]. Adamczyk et al. [Ada+18] gave a  $(7 + \varepsilon)$ -approximation algorithm in  $f(k, \varepsilon)n^{O(1)}$  time, showing that a constant factor parameterized approximation algorithm is possible. The approximation ratio was soon improved to  $(3 + \varepsilon)$  [CL19b]. For CAPACITATED EUCLIDEAN  $k$ -MEANS, [XZZ19] also gave a  $(69 + \varepsilon)$ -approximation algorithm for in  $f(k, \varepsilon)n^{O(1)}$  time.

While the capacitated versions of clustering look much harder than their uncapacitated counterparts, there is no known theoretical separation between the capacitated version and the uncapacitated version in any clustering task. Since the power of parameterized algorithms for uncapacitated clustering is well understood, it is a natural question to understand the “capacitated VS uncapacitated question” in the FPT setting.

**Open Question 4.16.** *Does CAPACITATED  $k$ -MEDIAN admit an  $(1 + 2/e)$ -approximation algorithm in FPT time with parameter  $k$ ? Do CAPACITATED EUCLIDEAN  $k$ -MEANS /  $k$ -MEDIAN admit an EPAS with parameter  $k$  or  $d$ ?*

Since clustering is a universal task, like capacitated versions, many variants of clustering tasks have been studied including  $k$ -MEDIAN /  $k$ -MEANS WITH OUTLIERS [KLS18] and MATROID / KNAPSACK MEDIAN [Swa16]. While no variant is proved to be harder than the basic versions, it would be interesting to see whether they all have the same parameterized approximability with the basic versions.

## 4.4 Network design

In network design the task is to connect some set of vertices in a metric, which is often given by the shortest-path metric of an edge-weighted graph. Two very prominent problems of this type

are the TRAVELLING SALESPERSON (TSP) and STEINER TREE problems. For TSP all vertices need to be connected in a closed walk (called a *route*), and the length of the route needs to be minimized.<sup>17</sup> For STEINER TREE a subset of the vertices (called *terminals*) is given as part of the input, and the objective is to connect all terminals by a tree of minimum weight in the metric (or graph). Both of these are fundamental problems that have been widely studied in the past, both on undirected and directed input graphs.

**Undirected graphs.** A well-studied parameter for STEINER TREE is the number of terminals, for which the problem has been known to be FPT since the early 1970s due to the work of Dreyfus and Wagner [DW71]. Their algorithm is based on dynamic programming and runs in  $3^k n^{O(1)}$  time if  $k$  is the number of terminals. Faster algorithms based on the same ideas with runtime  $(2 + \delta)^k n^{O(1)}$  for any constant  $\delta > 0$  exist [Fuc+07] (here the degree of the polynomial depends on  $\delta$ ). The unweighted STEINER TREE problem also admits a  $2^k n^{O(1)}$  time algorithm [Ned09] using a different technique based on subset convolution. Given any of these exact algorithms as a subroutine, a faster PAS can also be found [Fel+12] (cf. Section 4.7). On the other hand, no exact polynomial-sized kernel exists [DLS14] for the STEINER TREE problem, unless  $\text{NP} \subseteq \text{coNP}/\text{poly}$ . Interestingly though, a PSAKS can be obtained [Lok+17].

This kernel is based on a well-known fact proved by Borchers and Du [BD97], which is very useful to obtain approximation algorithms for the STEINER TREE problem. It states that any Steiner tree can be covered by smaller trees containing few terminals, such that these trees do not overlap much. More formally, a *full-component* is a subtree of a Steiner tree, for which the leaves coincide with its terminals. For the optimum Steiner tree  $T$  and any  $\varepsilon > 0$ , there exist full-components  $C_1, \dots, C_\ell$  of  $T$  such that

1. each full-component  $C_i$  contains at most  $2^{\lceil 1/\varepsilon \rceil}$  terminals (leaves),
2. the sum of the weights of the full-components is at most  $1 + \varepsilon$  times the cost of  $T$ , and
3. taking any collection of Steiner trees  $T_1, \dots, T_\ell$ , such that each tree  $T_i$  connects the subset of terminals that forms the leaves of full-component  $C_i$ , the union  $\bigcup_{i=1}^{\ell} T_i$  is a feasible solution to the input instance.

Not knowing the optimum Steiner tree, it is not possible to know the subsets of terminals of the full-components corresponding to the optimum. However, it is possible to compute the optimum Steiner tree for every subset of terminals of size at most  $2^{\lceil 1/\varepsilon \rceil}$  using an FPT algorithm for STEINER TREE. The time to compute all these solutions is  $k^{O(2^{1/\varepsilon})} n^{O(1)}$ , using for instance the Dreyfus and Wagner [DW71] algorithm. Now the above three properties guarantee that the graph given by the union of all the computed Steiner trees, contains a  $(1 + \varepsilon)$ -approximation for the input instance. In fact, the best polynomial time approximation algorithm known to date [Byr+13] uses an iterative rounding procedure to find a  $\ln(4)$ -approximation of the optimum solution in the union of these Steiner trees. To obtain a kernel, the union needs to be sparsified, since it may contain many Steiner vertices and also the edge weights might be very large. However, Lokshtanov et al. [Lok+17] show that the number of Steiner vertices can be reduced using standard techniques, while the edge weights can be encoded so that their space requirement is bounded in the parameter and the cost of any solution is distorted by at most a  $1 + \varepsilon$  factor.

**Theorem 4.17 ([Fel+12; Lok+17]).** *For the STEINER TREE problem a  $(1 + \varepsilon)$ -approximation can be computed in  $(2 + \delta)^{(1-\varepsilon/2)} n^{O(1)}$  time for any constant  $\delta > 0$  (and in  $2^{(1-\varepsilon/2)} n^{O(1)}$  time in the*

<sup>17</sup>Sometimes also the non-metric version of TSP is considered, which however is much harder than the metric one. We only consider the metric version here.

unweighted case) for any  $\varepsilon > 0$ , where  $k$  is the number of terminals. Moreover, a  $(1 + \varepsilon)$ -approximate kernel of size  $(k/\varepsilon)^{O(2^{1/\varepsilon})}$  can be computed in polynomial time.

A natural alternative to the number of terminals is to consider the vertices remaining in the optimum tree after removing the terminals: a folklore result states that STEINER TREE is W[2]-hard parameterized by the number of non-terminals (called *Steiner vertices*) in the optimum solution. At the same time, unless P=NP there is no PTAS for the problem, as it is APX-hard [CC08]. However an approximation scheme is obtainable when parametrizing by the number of Steiner vertices  $k$  in the optimum, and also a PSAKS is obtainable under this parameterization.

To obtain both of these results, Dvořák et al. [Dvo+18] devise a reduction rule that is based on the following observation: if the optimum tree contains few Steiner vertices but many terminals, then the tree must contain (1) a large component containing only terminals, or (2) a Steiner vertex that has many terminal neighbours. Intuitively, in case (2) we would like to identify a large star with terminal leaves and small cost in the current graph, while in case (1) we would like to find a cheap edge between two terminals. Note that such a single edge also is a star with terminal leaves. The reduction rule will therefore find the star with minimum weight per contained terminal, which can be done in polynomial time. This rule is applied until the number of terminals, which decreases after each use, falls below a threshold depending on the input parameter  $k$  and the desired approximation ratio  $1 + \varepsilon$ . Once the number of terminals is bounded by a function of  $k$  and  $\varepsilon$ , the Dreyfus and Wagner [DW71] algorithm can be applied on the remaining instance, or a kernel can be computed using the PSAKS of Theorem 4.17. It can be shown that the reduction rule does not distort the optimum solution by much as long as the threshold is large enough, which implies the following theorem.

**Theorem 4.18** ([Dvo+18]). *For the STEINER TREE problem a  $(1 + \varepsilon)$ -approximation can be computed in  $2^{O(k^2/\varepsilon^4)} n^{O(1)}$  time for any  $\varepsilon > 0$ , where  $k$  is the number of non-terminals in the optimum solution. Moreover, a  $(1 + \varepsilon)$ -approximate kernel of size  $(k/\varepsilon)^{2^{O(1/\varepsilon)}}$  can be computed in polynomial time.*

This theorem is also generalizable to the STEINER FOREST problem, where a list of terminal pairs is given and the task is to find a minimum weight forest in the input graph connecting each pair. In this case though, the parameter has to be combined with the number of connected components of the optimum forest [Dvo+18].

A variation of the STEINER FOREST problem is the SHALLOW-LIGHT STEINER NETWORK (SLSN) problem. Here a graph with both edge costs and edge lengths is given, together with a set of terminal pairs and a length threshold  $L$ . The task is to compute a minimum cost subgraph, which connects each terminal pair with a path of length at most  $L$ . For this problem a dichotomy result was shown [BDZ18] in terms of the pattern given by the terminal pairs. More precisely, the terminal pairs are interpreted as edges in a graph for which the vertices are the terminals: if  $\mathcal{C}$  is some class of graphs, then  $\text{SLSN}_{\mathcal{C}}$  is the SHALLOW-LIGHT STEINER NETWORK problem restricted to sets of terminal pairs that span some graph in  $\mathcal{C}$ . Let  $\mathcal{C}^*$  denote the class of all stars, and  $\mathcal{C}_{\lambda}$  the class of graphs with at most  $\lambda$  edges. The  $\text{SLSN}_{\mathcal{C}^*}$  problem is APX-hard [CC08], as it is a generalization of STEINER TREE (where  $L = \infty$ ). At the same time, both the  $\text{SLSN}_{\mathcal{C}^*}$  and  $\text{SLSN}_{\mathcal{C}_{\lambda}}$  problems parameterized by the number of terminals are paraNP-hard [Has92], since they are generalizations of the RESTRICTED SHORTEST PATH problem (where there is exactly one terminal pair). A PAS can however be obtained for both of these problems (whenever  $\lambda$  is a constant), but for no other class  $\mathcal{C}$  of demand patterns [BDZ18].

**Theorem 4.19** ([BDZ18]). *For any constant  $\lambda > 0$ , there is an FPTAS for the  $\text{SLSN}_{\mathcal{C}_\lambda}$  problem. For the  $\text{SLSN}_{\mathcal{C}^*}$  problem a  $(1 + \varepsilon)$ -approximation can be computed in  $4^k(n/\varepsilon)^{O(1)}$  time for any  $\varepsilon > 0$ , where  $k$  is the number of terminal pairs. Moreover, under Gap-ETH no  $(5/3 - \varepsilon)$ -approximation for  $\text{SLSN}_{\mathcal{C}}$  can be computed in  $f(k)n^{O(1)}$  time for any  $\varepsilon > 0$  and computable function  $f$ , whenever  $\mathcal{C}$  is a recursively enumerable class for which  $\mathcal{C} \not\subseteq \mathcal{C}^* \cup \mathcal{C}_\lambda$  for every constant  $\lambda$ .*

A notable special case is when all edge lengths are 1 but edge costs are arbitrary. Then  $\text{SLSN}_{\mathcal{C}_\lambda}$  is polynomial time solvable for any constant  $\lambda$ , while  $\text{SLSN}_{\mathcal{C}^*}$  is FPT parameterized by the number of terminals [BDZ18]. At the same time the parameterized approximation lower-bound of Theorem 4.19 is still valid for this case. It is not known however, whether constant approximation factors can be obtained for  $\text{SLSN}_{\mathcal{C}}$  when  $\mathcal{C}$  is a class different from  $\mathcal{C}_\lambda$  and  $\mathcal{C}^*$ . More generally we may ask the following question.

**Open Question 4.20.** *Given some class of graphs  $\mathcal{C} \not\subseteq \mathcal{C}^* \cup \mathcal{C}_\lambda$ , which approximation factor  $\alpha_{\mathcal{C}}$  can be obtained in FPT time for  $\text{SLSN}_{\mathcal{C}}$  parameterized by the number of terminals?*

Turning to the TSP problem, a generalization of TSP introduces deadlines until which vertices need to be visited by the computed tour. A natural parameterization in this setting is the number of vertices that have deadlines. It can be shown [Boc+07] that no approximation better than 2 can be computed when using this parameter. Nevertheless, a 2.5-approximation can be computed in FPT time [Boc+07]. The algorithm will guess the order in which the vertices with deadlines are visited by the optimum solution. It then computes a  $3/2$ -approximation for the remaining vertices using Christofides algorithm [Vaz01]. The approximation ratio follows, since the optimum tour can be thought of as two tours, of which one visits only the deadline vertices, while the other contains all remaining vertices. The approximation algorithm incurs a cost of  $\text{Opt}$  for the former, and a cost of  $\frac{3}{2} \cdot \text{Opt}$  for the latter part of the optimum tour.

**Theorem 4.21** ([Boc+07]). *For the DLTSP problem a 2.5-approximation can be computed in  $O(k! \cdot k) + n^{O(1)}$  time, if the number of vertices with deadlines is  $k$ . Moreover, no  $(2 - \varepsilon)$ -approximation can be computed in  $f(k)n^{O(1)}$  time for any  $\varepsilon > 0$  and computable function  $f$ , unless  $P=NP$ .*

**Low dimensional metrics.** Just as for clustering problems, another well-studied parameter in network design is the dimension of the underlying geometric space. A typical setting is when the input is assumed to be a set of points in some  $k$ -dimensional  $\ell_p$ -metric, where distances between points  $x$  and  $y$  are given by a function  $\text{dist}(x, y) = (\sum_{i=1}^k |x_i - y_i|^p)^{1/p}$ . Two prominent examples are Euclidean metrics (where  $p = 2$ ) and Manhattan metrics (where  $p = 1$ ). The dimension  $k$  of the metric space has been studied as a parameter from the parameterized approximation point-of-view *avant la lettre* for quite a while. It was shown [GGJ77; Pap77] that both STEINER TREE and TSP are paraNP-hard for this parameter (since they are NP-hard even if  $k = 2$ ), and that they are APX-hard in general metrics [CC08; KLS15]. However, a PAS for Euclidean metrics both for the STEINER TREE and the TSP problems were shown to exist in the seminal work of Arora [Aro98].<sup>18</sup> The techniques are similar to those used for clustering, and we refer to Section 4.3.2 for an overview.

**Theorem 4.22** ([Aro98]). *For the STEINER TREE and TSP problems a  $(1 + \varepsilon)$ -approximation can be computed in  $k^{O(\sqrt{k}/\varepsilon)^{k-1}} n^2$  time for any  $\varepsilon > 0$ , if the input consists of  $n$  points in  $k$ -dimensional Euclidean space.*

---

<sup>18</sup>In [Aro98] the runtime of these algorithms is stated as  $O(n(\log n)^{O(\sqrt{k}/\varepsilon)^{k-1}})$ , which can be shown to be upper bounded by  $k^{O(\sqrt{k}/\varepsilon)^{k-1}} n^2$  (see e.g. [KLP19b, Lemma 1]).



This result also holds for the  $t$ -MST and  $t$ -TSP problems [Aro98], where the cheapest tree or tour, respectively, on at least  $t$  nodes needs to be found. In this case the runtime has to be multiplied by  $t$  however.

A related setting is the parameterization by the *doubling dimension* of the underlying metric. That is, when the parameter  $k$  is the smallest integer such that any ball in the metric can be covered by  $2^k$  balls of half the radius. Any point set in a  $k$  dimensional  $\ell_p$ -metric has doubling dimension  $O(k)$ , and thus the latter parameter generalizes the former. For the TSP problem the above theorem can be generalized [Got15] to a PAS parameterized by the doubling dimension.

**Theorem 4.23** ([Got15]). *For the TSP problem a  $(1 + \varepsilon)$ -approximation can be computed in  $2^{(k/\varepsilon)^{O(k^2)}} n \log^2 n$  time for any  $\varepsilon > 0$ , if the input consists of  $n$  points with doubling dimension  $k$ .*

Given that a PAS exists for STEINER TREE in the Euclidean case, it is only natural to ask whether this is also possible for low doubling metrics. Only a QPTAS is known so far [Tal04]. Moreover, a related parameter is the *highway dimension*, which is used to model transportation networks. As shown by Feldmann et al. [Fel+18] the techniques of Talwar [Tal04] for low doubling metrics can be generalized to the highway dimension to obtain a QPTAS as well. But again, it is quite plausible to assume that a PAS exists.

**Open Question 4.24.** *Is there a PAS for STEINER TREE parameterized by the doubling dimension? Is there a PAS for either STEINER TREE or TSP parameterized by the highway dimension?*

**Directed Graphs.** When considering directed input graphs (asymmetric metrics), the DIRECTED STEINER TREE problem takes as input a terminal set and a special terminal called the *root*. The task is to compute a directed tree of minimum weight that contains a path from each terminal to the root. In general no  $f(k)$ -approximation can be computed in FPT time for any computable function  $f$ , when the parameter  $k$  is the number of Steiner vertices in the optimum solution [Dvo+18]. A notable special case is the *unweighted* DIRECTED STEINER TREE problem, which for this parameter admits a PAS. The techniques here are the same as those used to obtain Theorem 4.18 for the undirected case. However, in contrast to the undirected case which admits a PSAKS, no polynomial-sized  $(2 - \varepsilon)$ -approximate kernelization exists for DIRECTED STEINER TREE [Dvo+18], unless  $\text{NP} \subseteq \text{coNP}/\text{poly}$ . It is an intriguing question whether a 2-approximate kernel exists.

**Open Question 4.25.** *Is there a polynomial-sized 2-approximate kernel for the unweighted DIRECTED STEINER TREE problem parameterized by the number of Steiner vertices in the optimum solution?*

If the parameter is the number of terminals, the (weighted) DIRECTED STEINER TREE problem is FPT, using the same algorithm as for the undirected version [DW71; Fuc+07]. A different variant of STEINER TREE in directed graphs is the STRONGLY CONNECTED STEINER SUBGRAPH problem, where a terminal set needs to be strongly connected in the cheapest possible way. This problem is W[1]-hard parameterized by the number of terminals [GNS11], and no  $O(\log^{2-\varepsilon} n)$ -approximation can be computed in polynomial time [HK03], unless  $\text{NP} \subseteq \text{ZTIME}(n^{\text{polylog}(n)})$ . However, a 2-approximation can be computed in FPT time [CHK13].

The crucial observation for this algorithm is that in any strongly connected solution, fixing some terminal as the root, every terminal can be reached from the root, while at the same time the root can be reached from each terminal. Thus the optimum solution is the union of two directed trees, of which one is directed towards the root and the other is directed away from the

root, and the leaves of both trees are terminals. Hence it suffices to compute two solutions to the DIRECTED STEINER TREE problem, which can be done in FPT time, to obtain a 2-approximation for STRONGLY CONNECTED STEINER SUBGRAPH. Interestingly, no better approximation is possible with this runtime [CFM18].

**Theorem 4.26** ([CFM18; CHK13]). *For the STRONGLY CONNECTED STEINER SUBGRAPH problem a 2-approximation can be computed in  $(2 + \delta)^k n^{O(1)}$  time for any constant  $\delta > 0$ , where  $k$  is the number of terminals. Moreover, under Gap-ETH no  $(2 - \varepsilon)$ -approximation can be computed in  $f(k)n^{O(1)}$  time for any  $\varepsilon > 0$  and computable function  $f$ .*

A generalization of both DIRECTED STEINER TREE and STRONGLY CONNECTED STEINER SUBGRAPH is the DIRECTED STEINER NETWORK problem<sup>19</sup>, for which an edge-weighted directed graph is given together with a list of ordered terminal pairs. The aim is to compute the cheapest subgraph that contains a path from  $s$  to  $t$  for every terminal pair  $(s, t)$ . If  $k$  is the number of terminals, then for this problem no  $k^{1/4-o(1)}$ -approximation can be computed in  $f(k)n^{O(1)}$  time [DM18] for any computable function  $f$ , under Gap-ETH. But both a PAS and a PSAKS exist [CFM18] for the special case when the input graph is planar and *bidirected*, i.e., for every directed edge  $uv$  the reverse edge  $vu$  exists and has the same cost.

Similar to the PSAKS for the STEINER TREE problem, these two algorithms are based on a generalization of Borchers and Du [BD97]. That is, Chitnis et al. [CFM18] show that a planar solution in a bidirected graph can be covered by planar graphs with at most  $2^{O(1/\varepsilon)}$  terminals each, such that the sum of their costs is at most  $1 + \varepsilon$  times the cost of the solution. These covering graphs may need to contain edges that are reverse to those in the solution, but are themselves not part of the solution. For this the underlying graph needs to be bidirected. Analogous to STEINER TREE, to obtain a kernel it then suffices to compute solutions for every possible list of ordered pairs of at most  $2^{O(1/\varepsilon)}$  terminals. In contrast to STEINER TREE however, there is no FPT algorithm for this. Instead, an XP algorithm with runtime  $2^{O(k^{3/2} \log k)} n^{O(\sqrt{k})}$  needs to be used, which runs in polynomial time for  $k \leq 2^{O(1/\varepsilon)}$  terminals with  $\varepsilon$  being a constant. After taking the union of all computed solutions, the number of Steiner vertices and the encoding length of the edge weights can be reduced in a similar way as for the STEINER TREE problem. To obtain a PAS, the algorithm will guess how the planar optimum can be covered by solutions involving only small numbers of terminals. It will then compute solutions on these subsets of at most  $2^{O(1/\varepsilon)}$  terminals using the same XP algorithm.

**Theorem 4.27** ([CFM18]). *For the DIRECTED STEINER NETWORK problem on planar bidirected graphs a  $(1 + \varepsilon)$ -approximation can be computed in  $\max\{2^{k^{2^{O(1/\varepsilon)}}}, n^{2^{O(1/\varepsilon)}}\}$  time for any  $\varepsilon > 0$ , where  $k$  is the number of terminals. Moreover, a  $(1 + \varepsilon)$ -approximate kernel of size  $(k/\varepsilon)^{2^{O(1/\varepsilon)}}$  can be computed in polynomial time.*

## 4.5 Cut problems

Starting from Menger's theorem and the corresponding algorithm for  $s$ - $t$  CUT, graph cut problems have always been at the heart of combinatorial optimization. While many natural generalizations of  $s$ - $t$  CUT are NP-hard, further study of these cut problems yielded beautiful techniques such as flow-cut gaps and metric embeddings in approximation algorithms [ARV09; LR99], and also important separators and randomized contractions in parameterized algorithms [Chi+16; Cyg+19; Mar06; MR14].

<sup>19</sup>sometimes also called DIRECTED STEINER FOREST; note however that the optimum is not necessarily a forest.

### 4.5.1 Multicut

An instance of **UNDIRECTED MULTICUT** (resp. **DIRECTED MULTICUT**) is an undirected (resp. directed) graph  $G = (V, E)$  with  $k$  pairs of vertices  $(s_1, t_1), \dots, (s_k, t_k)$ . The goal is to remove the minimum number of edges such that there is no path from  $s_i$  to  $t_i$  for every  $i \in [k]$ . **UNDIRECTED MULTIWAY CUT** (resp. **DIRECTED MULTIWAY CUT**) is a special case of **UNDIRECTED MULTICUT** (resp. **DIRECTED MULTICUT**) where  $k$  vertices are given as terminals and the goal is to make sure there is no path between any pair of terminals. They have been actively studied from both approximation and parameterized algorithms perspectives. We survey parameterized approximation algorithms for these problems with parameters  $k$  and the solution size  $\text{OPT}$ .

**Undirected Multicut.** **UNDIRECTED MULTICUT** admits an  $O(\log k)$ -approximation algorithm [GVY96] in polynomial time, and is NP-hard to approximate within any constant factor assuming the Unique Games Conjecture [Cha+06]. **UNDIRECTED MULTIWAY CUT** admits an 1.2965-approximation algorithm [SV13] in polynomial time, and is NP-hard to approximate within a factor 1.20016 [Bér+19]. **UNDIRECTED MULTICUT** (and thus **UNDIRECTED MULTIWAY CUT**) admits an exact algorithm parametrized by  $\text{OPT}$  [Mar06; MR14].

With  $k$  as a parameter, we cannot hope for an exact algorithm or an approximation scheme, since even **UNDIRECTED MULTIWAY CUT** with 3 terminals is NP-hard to approximate within a factor  $12/11 - \varepsilon$  for any  $\varepsilon > 0$  under the Unique Games Conjecture. However, for **UNDIRECTED MULTICUT** with  $k$  pairs  $(s_1, t_1), \dots, (s_k, t_k)$ , one can reduce it to  $k^{O(k)}$  instances of **UNDIRECTED MULTIWAY CUT** with at most  $2k$  terminals, by guessing a partition of these  $s_1, t_1, \dots, s_k, t_k$  according to the connected components containing them in the optimal solution (e.g.,  $s_i$  and  $t_i$  should be always in different groups), merging the vertices in the same group into one vertex, and solving **UNDIRECTED MULTIWAY CUT** with the merged vertices as terminals. This shows an 1.2965-approximation algorithm for **UNDIRECTED MULTICUT** that runs in time  $k^{O(k)} n^{O(1)}$ .

Some recent results improve or generalize this observation. For graphs with bounded genus  $g$ , Cohen-Addad et al [CDVDM18] gave an EPAS running in time  $f(g, k, \varepsilon) \cdot n \log n$ . Chekuri and Madan [CM17] considered the *demand graph*  $H$ , which is the graph formed by  $k$  edges  $(s_1, t_1), \dots, (s_k, t_k)$ . When  $t$  is the smallest integer such that  $H$  does not contain  $t$  disjoint edges as an induced subgraph, they presented a 2-approximation algorithm that runs in time  $k^{O(t)} n^{O(1)}$ .

**Directed Multicut.** Generally, **DIRECTED MULTICUT** is a much harder computational task than **UNDIRECTED MULTICUT** in terms of both approximation and parameterized algorithms. **DIRECTED MULTICUT** admits a  $\min(k, \tilde{O}(n^{11/23}))$ -approximation algorithm [AAC07]. It is NP-hard to approximate within a factor  $k - \varepsilon$  for any  $\varepsilon > 0$  for fixed  $k$  [Lee17a] under the Unique Games Conjecture, or  $2^{\Omega(\log^{1-\varepsilon} n)}$  for any  $\varepsilon > 0$  [CK09] for general  $k$ . **DIRECTED MULTIWAY CUT** admits a 2-approximation algorithm [NZ97], which is tight even when  $k = 2$  [Lee17a]. Parameterizing by  $\text{Opt}$ , **DIRECTED MULTICUT** is FPT for  $k = 2$ , but **DIRECTED MULTICUT** is W[1]-hard even when  $k = 4$  [PW18]. **DIRECTED MULTIWAY CUT** on the other hand is in FPT [Chi+16].

Since it is hard to improve the trivial  $k$ -approximation algorithm even for fixed  $k$  [Lee17a], parameterizing by  $k$  does not yield a better approximation algorithm. Chitnis and Feldmann [CF19] gave a  $k/2$ -approximation algorithm that runs in time  $2^{O(\text{Opt}^2)} n^{O(1)}$ , and also proved that the problem under the same parameterization is still hard to approximate within a

factor  $59/58$  with  $k = 4$ .

**Open Question 4.28.** *What is the best approximation ratio (as a function of  $k$ ) achieved by a parameterized algorithm (with parameter  $\text{Opt}$ )? Will it be close to  $O(1)$  or  $\Omega(k)$ ?*

#### 4.5.2 Minimum Bisection and Balanced Separator.

Given a graph  $G = (V, E)$ , MINIMUM EDGE BISECTION (resp. MINIMUM VERTEX BISECTION) asks to remove the fewest number of edges such that the graph is partitioned into two parts  $A$  and  $B$  with  $||A| - |B|| \leq 1$ . BALANCED EDGE SEPARATOR (resp. BALANCED VERTEX SEPARATOR) is a more relaxed version of the problem where the goal is to bound the size of the largest component by  $\alpha n$  for some  $1/2 < \alpha < 1$ . It has been actively studied from approximation algorithms, culminating in  $O(\sqrt{\log n})$ -approximation algorithms for both BALANCED EDGE SEPARATOR and BALANCED VERTEX SEPARATOR [ARV09; FHL08], and an  $O(\log n)$ -approximation algorithm for MINIMUM EDGE BISECTION [Răc08].

If we parameterize by the size of optimal separator  $k$ , MINIMUM EDGE BISECTION admits an exact parameterized algorithm [Cyg+19]. While MINIMUM VERTEX BISECTION is  $W[1]$ -hard [Mar06], Feige and Mahdian [FM06] gave an algorithm that given  $2/3 \leq \alpha < 1$  and  $\varepsilon > 0$ , in time  $2^{O(k)}n^{O(1)}$  returns an  $(\alpha + \varepsilon)$  separator of size at most  $k$ .

#### 4.5.3 $k$ -Cut

Given an undirected graph  $G = (V, E)$  and an integer  $k \in \mathbb{N}$ , the  $k$ -CUT problem asks to remove the smallest number of edges such that  $G$  is partitioned into at least  $k$  non-empty connected components. The edge contraction algorithm by Karger and Stein [KS96] yields a randomized exact XP algorithm running in time  $O(n^{2k})$ , which was made deterministic by Thorup [Tho08]. There were recent improvements to the running time [GLL18b; GLL19]. Also there is an exact parameterized algorithm with parameter  $\text{OPT}$  [Chi+16; KT11]. For general  $k$ , it admits a  $(2 - 2/k)$ -approximation algorithm [SV95], and is NP-hard to approximate within a factor  $(2 - \varepsilon)$  for any  $\varepsilon > 0$  under the Small Set Expansion Hypothesis [Man18b].

A simple reduction shows that  $k$ -CUT captures  $(k - 1)$ -CLIQUE, so an exact FPT algorithm with parameter  $k$  is unlikely to exist. Gupta et al. [GLL18a] gave an  $(2 - \delta)$ -approximation algorithm for a small universal constant  $\delta > 0$  that runs in time  $f(k) \cdot n^{O(1)}$ . The approximation ratio was improved to 1.81 in [GLL18b], and further to 1.66 [KL20]. Very recently, Lokshantov et al. [LSS20] gave a PAS that runs in time  $(k/\varepsilon)^{O(k)}n^{O(1)}$ , thereby (essentially) resolving the parameterized approximability of  $k$ -CUT.

## 4.6 $\mathcal{F}$ -DELETION problems

Let  $\mathcal{F}$  be a vertex-hereditary family of undirected graphs, which means that if  $G \in \mathcal{F}$  and  $H$  is a vertex-induced subgraph of  $G$ , then  $H \in \mathcal{F}$  as well.  $\mathcal{F}$ -DELETION is the problem where given a graph  $G = (V, E)$ , we are supposed to find  $S \subseteq V$  such that the subgraph induced by  $V \setminus S$  (denoted by  $G \setminus S$ ) belongs to  $\mathcal{F}$ . The goal is to minimize  $|S|$ . The natural *weighted version*, where there is a non-negative weight  $w(v)$  for each vertex  $v$  and the goal is to minimize the sum of the weights of the vertices in  $S$ , is called WEIGHTED  $\mathcal{F}$ -DELETION.

$\mathcal{F}$ -DELETION captures numerous combinatorial optimization problems, including VERTEX COVER (when  $\mathcal{F}$  includes all graphs with no edges), FEEDBACK VERTEX SET (when  $\mathcal{F}$  is the set

of all forests), and ODD CYCLE TRANSVERSAL (when  $\mathcal{F}$  is the set of all bipartite graphs). There are a lot more interesting graph classes  $\mathcal{F}$  studied in structural and algorithmic graph theory. Some famous examples include planar graphs, perfect graphs, chordal graphs, and graphs with bounded treewidth.

In addition to beautiful structural results that give multiple equivalent characterizations, these graph classes often admit very efficient algorithms for some tasks that are believed to be hard in general graphs. Therefore, a systematic study of  $\mathcal{F}$ -DELETION for more graph classes is not only an interesting algorithmic task by itself, but also a way to obtain better algorithms for other optimization problems when the given graph  $G$  is *close* to a nice class  $\mathcal{F}$  (i.e., deleting few vertices from  $G$  makes it belong to  $\mathcal{F}$ .) Indeed, some algorithms for INDEPENDENT SET for noisy planar / minor-free graphs discussed in Section 4.1 use an algorithm for  $\mathcal{F}$ -DELETION as a subroutine [BRU17].

For the maximization version where the goal is to *maximize*  $|V \setminus S|$ , a powerful but pessimistic characterization is known. Lund and Yannakakis [LY93] showed that whenever  $\mathcal{F}$  is vertex-hereditary and *nontrivial* (i.e., there are infinitely many graphs in  $\mathcal{F}$  and out of  $\mathcal{F}$ ), the maximization version is hard to approximate within a factor  $2^{\log^{1/2-\epsilon} n}$  for any  $\epsilon > 0$ . So no nontrivial  $\mathcal{F}$  is likely to admit even a polylogarithmic approximation algorithm. But the situation is different for the minimization problem, since VERTEX COVER admits a 2-approximation algorithm, while ODD CYCLE TRANSVERSAL [Kho02] and PERFECT DELETION [Heg+11] are NP-hard to approximate within any constant factor approximation algorithm. (The first result assumes the Unique Games Conjecture.) It indicates that a characterization of approximabilities for the minimization versions will be more complex and challenging.

There are two (closely related) frameworks to capture large graph classes.

- Choose a *graph width parameter* (e.g., treewidth, pathwidth, cliquewidth, rankwidth, etc.) and  $k \in \mathbb{N}$ . Let  $\mathcal{F}$  be the set of graphs  $G$  with the chosen width parameter at most  $k$ . The parameter of  $\mathcal{F}$ -DELETION is  $k$ .
- Choose a notion of *subgraph* (e.g., subgraph, induced subgraph, minor, etc.) and a finite family of forbidden graphs  $\mathcal{H}$ . Let  $\mathcal{F}$  be the set of graphs  $G$  that do not have any graph in  $\mathcal{H}$  as the chosen notion of subgraph. The parameter of  $\mathcal{F}$ -DELETION is  $|\mathcal{H}| := \sum_{H \in \mathcal{H}} |V(H)|$ .

Many interesting classes are captured by the above frameworks. For example, to express FEEDBACK VERTEX SET, we can take  $\mathcal{F}$  to be the set of graphs with treewidth at most 1, or equivalently, the set of graphs that does not have the triangle graph  $K_3$  as a minor. In the rest of the subsection, we introduce known results of  $\mathcal{F}$ -DELETION under the above two parameterizations. Note that under these two parameterizations, the need for approximation is inherent since the simplest problem in both frameworks, VERTEX COVER, already does not admit a polynomial-time  $(2 - \epsilon)$ -approximation algorithm under the Unique Games Conjecture.

Finally, we mention that the parameterization by the size of the optimal solution has been studied more actively from the parameterized complexity community, where many important problems are shown to be in FPT [CM14; Fom+12; Mar10].

#### 4.6.1 Treewidth and Planar Minor Deletion

The treewidth of a graph (see Definition 4.5) is arguably the most well-studied graph width parameter with numerous structural and algorithmic applications. It is one of the most important concepts in the graph minor project of Robertson and Seymour. Algorithmically, Courcelle's theorem [Cou90] states that every problem expressible in the monadic second-order logic of

graphs can be solved in FPT time parameterized by treewidth. We refer the reader to the survey of Bodlaender [Bod07]. Computing treewidth is NP-hard in general [ACP87], but if we parameterize by treewidth, it can be done in FPT time [Bod96], and there is a faster constant-factor parameterized approximation algorithm [Bod+16].

Let  $k \in \mathbb{N}$  be the parameter. TREEWIDTH  $k$ -DELETION (also known as TREEWIDTH  $k$ -MODULATOR in the literature) is a special case of  $\mathcal{F}$ -DELETION where  $\mathcal{F}$  is the set of all graphs with treewidth at most  $k$ . Note the case  $k = 0$  yields VERTEX COVER and  $k = 1$  yields FEEDBACK VERTEX SET.

Fomin et al. [Fom+12] gave a randomized  $f(k)$ -approximation algorithm that runs in  $g(k) \cdot nm$  for some computable functions  $f$  and  $g$ . The approximation ratio was improved by Gupta et al. [Gup+19] that gave a deterministic  $O(\log k)$ -approximation algorithm that runs in  $f(k) \cdot n^{O(1)}$  some  $f$ .

This result has immediate applications to minor deletion problems. Let  $\mathcal{H}$  be a finite set of graphs, and consider  $\mathcal{H}$ -MINOR DELETION, which is a special case of  $\mathcal{F}$ -DELETION when  $\mathcal{F}$  is the set of all graphs that do not have any graph in  $\mathcal{H}$  as a minor. Its parameterized and kernelization complexity (with parameter  $\text{Opt}$ ) for family  $\mathcal{H}$  has been actively studied [DJ19; Fom+12; JP18a].

When  $\mathcal{H}$  contains a planar graph  $H$  (also known as PLANAR  $\mathcal{H}$ -DELETION in the literature), by the polynomial grid-minor theorem [CC16], any graph  $G \in \mathcal{F}$  has treewidth at most  $k := \text{poly}(|V(H)|)$ . Therefore, in order to solve  $\mathcal{H}$ -MINOR DELETION, one can first solve TREEWIDTH  $k$ -DELETION to reduce the treewidth to  $k$  and then solve  $\mathcal{H}$ -MINOR DELETION optimally using Courcelle's theorem [Cou90]. Combined with the above algorithm for TREEWIDTH  $k$ -DELETION [Gup+19], this strategy yields an  $O(\log k)$ -approximation algorithm that runs in  $f(|\mathcal{H}|) \cdot n^{O(1)}$  time.

Beyond PLANAR  $\mathcal{H}$ -DELETION, there are not many results known for  $\mathcal{H}$ -MINOR DELETION. The case  $\mathcal{H} = \{K_5, K_{3,3}\}$  is called MINIMUM PLANARIZATION and was recently shown to admit an  $O(\log^{O(1)} n)$ -approximation algorithm in  $n^{O(\log n / \log \log n)}$  time [KS17].

While the unweighted versions of TREEWIDTH  $k$ -DELETION and PLANAR  $\mathcal{H}$ -DELETION admit an approximation algorithm whose approximation ratio only depends on  $k$  not  $n$ , such an algorithm is not known for WEIGHTED TREEWIDTH  $k$ -DELETION or WEIGHTED PLANAR  $\mathcal{H}$ -DELETION. Agrawal et al. [Agr+18] gave a randomized  $O(\log^{1.5} n)$ -approximation algorithm and a deterministic  $O(\log^2 n)$ -approximation algorithm that run in polynomial time for fixed  $k$ , i.e., the degree of the polynomial depends on  $k$ . Bansal et al. [BRU17] gave an  $O(\log n \log \log n)$ -approximation algorithm for the edge deletion version. The only graphs  $H$  whose weighted minor deletion problem is known to admit a constant factor approximation algorithm are single edge (WEIGHTED VERTEX COVER), triangle (WEIGHTED FEEDBACK VERTEX SET), and diamond [FJP10]. Also for the weighted versions, no hardness beyond VERTEX COVER is known.

**Open Question 4.29.** *Does WEIGHTED TREEWIDTH  $k$ -DELETION admit an  $f(k)$ -approximation algorithm with parameter  $k$  for some function  $f$ ? Does TREEWIDTH  $k$ -DELETION admit a  $c$ -approximation algorithm with parameter  $k$  for some universal constant  $c$ ?*

**Algorithms for TREEWIDTH  $k$ -DELETION.** Here we present high-level ideals of [Gup+19] and [Agr+18] for TREEWIDTH  $k$ -DELETION and WEIGHTED TREEWIDTH  $k$ -DELETION respectively. These two algorithms share the following two important ingredients:

1. Graphs with bounded treewidth admit good *separators*.
2. There are good approximation algorithms to find such separators.

Given an undirected and vertex-weighted graph  $G = (V, E)$  and an integer  $k \in \mathbb{N}$ , let (WEIGHTED)  $k$ -VERTEX SEPARATOR be the problem whose goal is to remove the vertices of minimum total weight so that each connected component has at most  $k$  vertices. An algorithm is called an  $\alpha$ -bicriteria approximation algorithm if it returns a solution whose total weight is at most  $\alpha \cdot \text{OPT}$  and each connected component has at most  $1.1k$  vertices.<sup>20</sup> The case  $k = 2n/3$  is called BALANCED SEPRATOR and has been actively studied in the approximation algorithms community, and the best approximation algorithm achieves  $O(\sqrt{\log n})$ -bicriteria approximation [FHL08]. When  $k$  is small,  $O(\log k)$ -bicriteria approximation is also possible [Lee17b].

**WEIGHTED TREewidth  $k$ -DELETION.** Agrawal et al. [Agr+18] achieves an  $O(\log^{1.5} n)$ -approximation for WEIGHTED TREewidth  $k$ -DELETION in time  $n^{O(k)}$ . It would be interesting to see whether the running time can be made FPT with parameter  $k$ .

The main structure of their algorithm is *top-down recursive*. Deleting the optimal solution  $S^*$  from  $G$  reduces the treewidth of  $G \setminus S^*$  to  $k$ , so from the forest decomposition of  $G \setminus S^*$ , there exists a set  $M^* \subseteq G \setminus S^*$  with at most  $k + 1$  vertices such that each connected component of  $G \setminus (M^* \cup S^*)$  has at most  $2n/3$  vertices. While we do not know  $S^*$ , we can exhaustively try every possible  $M \subseteq V$  with  $|M| \leq k + 1$  and use the bicriteria approximation algorithm for BALANCED SEPRATOR to find  $M$  and  $S$  such that (1)  $|M| \leq k + 1$ , (2)  $w(S) \leq O(\sqrt{\log n})\text{OPT}$ , and (3)  $G \setminus (M \setminus S)$  has at most  $1.1 \cdot (2n/3) \leq 3n/4$  vertices.

Let  $G_1, \dots, G_t$  be the resulting connected components of  $G \setminus (S \cap M)$ . We solve each  $G_i$  recursively to compute  $S_i$  such that each  $G_i \setminus S_i$  has treewidth at most  $k$ . The weight of  $S$  was already bounded in terms of  $\text{OPT}$ , but the weight of  $M$  was not, so we finally need to consider the graph induced by  $M \cup V(G_1) \cup \dots \cup V(G_t)$  and delete vertices of small weight to ensure small treewidth. But this task is easy since since the treewidth of each  $G_i$  is bounded by  $k$  and  $|M| \leq k + 1$ , which bounds the treewidth of the considered graph by  $2k + 1$ . So we can fetch the algorithm for small treewidth graphs to solve the problem optimally. Note that the total weight of removed vetices in this recursive call is at most  $(O(\sqrt{\log n}) + 1)\text{OPT}$ . Since  $\sum_i \text{OPT}(G_i) \leq \text{OPT}(G)$  and the recursion depth is at most  $O(\log n)$ , the total approximation ratio is  $O(\log^{1.5} n)$ .

**TREewidth  $k$ -DELETION.** Gupta et al. [Gup+19] give an  $O(\log k)$ -approximation algorithm that runs in time  $f(k) \cdot n^{O(1)}$  for the unweighted version of TREewidth  $k$ -DELETION. The main structure of this algorithm is *bottom-up iterative refinement*. The algorithm maintains a feasible solution  $S \subseteq V$  (we can start with  $S = V$ ), and iteratively uses  $S$  to obtain another feasible solution  $S'$ . If the new solution is not smaller (i.e.  $|S'| \geq |S|$ ), then  $|S| \leq O(\log k) \cdot \text{OPT}$ .

Let us focus on one refinement step with the current feasible solution  $S$ . Let  $S^*$  be the optimal solution, so that  $G \setminus S^*$  has treewidth at most  $k$ . We use the following simple lemma showing the existence of a good separator of  $G$  in a finer scale than before.

**Lemma 4.30** ([Fom+11; Gup+19]). *Let  $H$  be a graph with treewidth at most  $k$ ,  $T \subseteq V(H)$  be any subset of vertices, and  $\varepsilon > 0$ . There exists  $R \subseteq V(H)$  such that (1)  $|R| \leq \varepsilon|T|$  and (2) every connected component of  $H \setminus R$  has at most  $O(k/\varepsilon)$  vertices from  $T$ .*

<sup>20</sup>1.1 here can be replaced by  $1 + \varepsilon$  for any constant  $\varepsilon > 0$ .

Plugging  $H \leftarrow G \setminus S^*$ ,  $T \leftarrow S$  in the above lemma and letting  $S' = R \cup S^*$ , we can conclude that there exists  $S' \subseteq V$  such that  $|S'| \leq |R| + |S^*| \leq \varepsilon|S| + \text{OPT}$  and each connected component of  $G \setminus S'$  has at most  $O(k/\varepsilon)$  vertices from  $S$ .

How can we find such a set  $S'$  efficiently? Note that if  $S = V$ , then  $S'$  is an  $O(k/\varepsilon)$ -vertex separator of  $G$ . Lee [Lee17b] defined a generalization of  $k$ -VERTEX SEPARATOR called  $k$ -SUBSET VERTEX SEPARATOR, where the input consists of  $G = (V, E)$ ,  $S \subseteq V$ ,  $k \in \mathbb{N}$ , and the goal is to remove the smallest number of vertices so that each connected component has at most  $k$  vertices from  $S$ , and gave an  $O(\log k)$ -bicriteria approximation algorithm.

Since the above lemma guarantees that  $\text{OPT}$  for  $O(k/\varepsilon)$ -SUBSET VERTEX SEPARATOR is at most  $\text{OPT}$  for TREEWIDTH  $k$ -DELETION plus  $\varepsilon|S|$ , applying this bicriteria approximation algorithm yields  $S'$  such that  $|S'| \leq O(\log k)(\text{OPT} + \varepsilon|S|)$  and each connected component of  $G \setminus S'$  has at most  $O(k/\varepsilon)$  vertices from  $S$ . Since  $S$  is a feasible solution, it implies that the treewidth of each connected component is bounded by  $O(k/\varepsilon)$ , so we can solve each component optimally in time  $f(k/\varepsilon) \cdot n^{O(1)}$ . By setting  $\varepsilon = 0.5$ , we can see the size of new solution is strictly decreased unless  $|S| = O(\log k) \cdot \text{OPT}$ , finishing the proof.

## 4.6.2 Subgraph Deletion

Let  $H$  be a fixed *pattern graph* with  $k$  vertices. Given a *host graph*  $G$ , deciding whether  $H$  is a subgraph of  $G$  (in the usual sense) is known as SUBGRAPH ISOMORPHISM, whose parameterized complexity with various parameters (e.g.,  $k$ ,  $\text{tw}(H)$ ,  $\text{genus}(G)$ , etc.) was studied by Marx and Pilipczuk [MP14].

Guruswami and Lee [GL15] studied the corresponding vertex deletion problem  $H$ -SUBGRAPH DELETION (called  $H$ -TRANSVERSAL in the paper), which is a special case of  $\mathcal{F}$ -DELETION where  $\mathcal{F}$  is the set of graphs that do not have  $H$  as a subgraph. Note that the problem admits a simple  $k$ -approximation algorithm that runs in time  $O(n \cdot f(n, H))$ , where  $f(n, H)$  denotes time to solve SUBGRAPH ISOMORPHISM with the pattern graph  $H$  and a host graph with  $n$  vertices. Their main hardness result states that assuming the Unique Games Conjecture, whenever  $H$  is 2-vertex connected, for any  $\varepsilon > 0$ , no polynomial time algorithm (including algorithms running in time  $n^{f(k)}$  for any  $f$ ) can achieve a  $(k - \varepsilon)$ -approximation. (Without the UGC, they still ruled out a  $(k - 1 - \varepsilon)$ -approximation.)

Among  $H$  that are not 2-vertex-connected, there is an  $O(\log k)$ -approximation algorithm when  $H$  is a star (in time  $n^{O(1)}$ ) or a path (in time  $f(k)n^{O(1)}$ ) [EKS; GL15; Lee17b]. The algorithm for  $k$ -path follows from the result for TREEWIDTH  $k$ -DELETION, because any graph without a  $k$ -path has treewidth at most  $k$ . Whenever  $H$  is a tree with  $k$  vertices, detecting a copy of  $H$  in  $G$  with  $n$  vertices can be done in  $2^{O(k)}n^{O(1)}$  time [AYZ95], and it is open whether there is an  $O(\log k)$ -approximation algorithm for  $H$ -SUBGRAPH DELETION in time  $f(k) \cdot n^{O(1)}$ .

## 4.6.3 Other Deletion Problems

**Chordal graphs.** A graph is *chordal* if it does not have an induced cycle of length  $\geq 4$ . Chordal graphs form a subclass of perfect graphs that have been actively studied. Initially motivated by efficient kernels, approximation algorithms for CHORDAL DELETION have been developed recently. The current best results are a  $\text{poly}(\text{OPT})$ -approximation [JP18b; KK18] and a  $O(\log^2 n)$ -approximation [Agr+18].



**Edge versions.** While this subsection focused on the vertex deletion problem, there are some results on the *edge deletion*, *edge addition*, and *edge modification* versions. (Edge modification allows both addition and deletion.) Cao and Sandeep [CS17] studied MINIMUM FILL-IN, whose goal is to add the minimum number of edges to make a graph chordal. They gave new inapproximability results implying improved time lower bounds for parameterized algorithms. Giannopoulou et al. [Gia+17] gave  $O(1)$ -approximation algorithms for PLANAR  $\mathcal{H}$ -IMMERSION DELETION parameterized by  $\mathcal{H}$ . Bliznets et al. [Bli+18] considered  $H$ -free edge modification for a forbidden induced subgraph  $H$  and give an almost complete characterization on its approximability depending on  $H$ .

**Directed graphs.** There is also a large body of work on parameterized algorithms for vertex deletion problems in *directed graphs*. While many of the known problems (including DIRECTED FEEDBACK VERTEX SET [CLL07]) admit an exact FPT algorithm, Lokshtanov et al. [Lok+20] studied DIRECTED ODD CYCLE TRANSVERSAL, and proved that it is  $W[1]$ -hard and is unlikely to admit a PAS under the Parameterized Inapproximability Hypothesis (or Gap-ETH). They complemented the result by showing a 2-approximation algorithm running in time  $f(\text{OPT})n^{O(1)}$ .

## 4.7 Faster Algorithms and Smaller Kernels via Approximation

The focus of this section so far has been on problems for which its exact version is intractable (i.e.  $W[1]$ / $W[2]$ -hard) and the goal is to obtain good approximations in FPT time. In this subsection, we shift our focus slightly by asking: *does approximation allow us to find faster algorithms for problems already known to be in FPT?*

To illustrate this, let us consider VERTEX COVER. It is of course well-known that the *exact* version of the problem can be solved in FPT time, with the current best running time being  $O^*(1.2738^k)$  [CKX06]. The question here would be: if we are allowed to output an  $(1 - \varepsilon)$ -approximate solution, instead of just an exact one, can we speed up the algorithm?

To the best of our knowledge, such a question was tackled for the first time by Bourgeois et al. [BEP09] and revisited quite a few times in the literature [Ban+19; BF10; BF13; Fel+12; MT18]. As one might have suspected, the answer to this question is a YES, as stated below.

**Theorem 4.31** ([Fel+12]). *Let  $\delta > 0$  be such that there exists an  $O^*(\delta^k)$ -time algorithm for VERTEX COVER (e.g.  $\delta = 1.2738$ ). Then, for any  $\varepsilon > 0$ , there is an  $(1 + \varepsilon)$ -approximation for VERTEX COVER that runs in  $O^*(\delta^{(1-\varepsilon)k})$  time.*

The main idea of the algorithm is inspired by the “local ratio” method in the approximation algorithms literature (see e.g. [Bar+04]) and we sketch it here. The algorithm works in two stages. In the first stage, we run the greedy algorithm: as long as we have picked less than  $2\varepsilon k$  vertices so far and not all edges are covered, pick an uncovered edge and add both endpoints to our solution. In the second stage, we run the exact algorithm on the remaining part of the graph to find a VERTEX COVER of size  $(1 - \varepsilon)k$ . Since the first stage runs in polynomial time, the running time of the entire algorithm is dominated by the second stage, whose running time is  $O^*(\delta^{(1-\varepsilon)k})$  as desired. The correctness of the algorithm follows from the fact that, for each selected edge in the first step, the optimal solution still needs to pick at least one endpoint. As a result, the optimal solution must pick at least  $\varepsilon k$  vertices with respect to the first stage (compared to  $2\varepsilon k$  picked by the algorithm). Thus, when the optimal solution is of size at most  $k$ ,

there must be a solution in the second stage of size at most  $(1 - \varepsilon)k$ , meaning that the algorithm finds such a solution and outputs a vertex cover of size  $(1 + \varepsilon)k$  as claimed.

The above “approximate a small fraction and brute force the rest” approach of Fellows et al. [Fel+12] generalizes naturally to problems beyond VERTEX COVER. Fellows et al. [Fel+12] formalized the method in terms of  $\alpha$ -fidelity kernelization and apply it to several problems, including CONNECTED VERTEX COVER,  $d$ -HITTING SET and STEINER TREE. For these problems, the method gives an  $(1 + \varepsilon)$ -approximation algorithm that runs in time  $O^*(\delta^{(1-\Omega(\varepsilon))k})$ , where  $\delta > 0$  denotes a constant for which a  $O^*(\delta^k)$ -time algorithm is known for the exact version of the corresponding problem. The approach, in some form or another, is also applicable both to other parameterized problems [BPS16; Esc+15] and to non-parameterized problems (e.g. [BEP09]); since the latter is out-of-scope for the survey, we will not discuss the specifics here.

An intriguing question related to this line of work is whether it must be the case that the running time of  $(1 + \varepsilon)$ -approximation algorithms is of the form  $O^*(\delta^{(1-\Omega(\varepsilon))k})$ . That is, can we get an  $(1 + o(1))$ -approximation for these problems in time  $O^*(\lambda^k)$  where  $\lambda$  is a constant strictly smaller than  $\delta$ ? More specifically, we may ask the following:

**Open Question 4.32.** *Let  $\delta > 0$  be the smallest (known) constant such that an  $O^*(\delta^k)$ -time exact algorithm exists for VERTEX COVER. Is there an algorithm that, for any  $\varepsilon > 0$ , runs in time  $f(1/\varepsilon) \cdot O^*(\lambda^k)$  for some constant  $\lambda < \delta$ ?*

Of course, the question applies not only for VERTEX COVER but other problems in the list as well. The informal crux of this question is whether, in the regime of very good approximation factors (i.e.  $1 + o(1)$ ), approximation can still be exploited in such a way that the algorithm works significantly better than the approach “approximate a  $o(1)$  fraction and then brute force”.

Turning back once again to our running example of VERTEX COVER, it turns out that algorithms faster than “approximate a small fraction and then brute force” are known [Ban+19; BF10; MT18] but only for the regime of large approximation ratios. In particular, Brankovic and Fernau [BF10] give faster algorithms than in Theorem 4.31 already for approximation ratios as small as  $3/2$ . The algorithms in [Ban+19; MT18] focus on the case of “barely non-trivial”  $(2 - \rho)$ -approximation factors. (Recall the greedy algorithm yields a 2-approximation and, under the Unique Games Conjecture, the problem is NP-hard to approximate to within any constant factor less than 2.) The algorithm in [Ban+19] has a running time of  $O^*(2^{k/2^{\Omega(1/\rho)}})$ , which was later improved in [MT18] to  $O^*(2^{k/2^{\Omega(1/\rho^2)}})$ . These running times should be contrasted with that of “approximate a small fraction and then brute force” (i.e., applying Theorem 4.31 directly with  $\varepsilon = 1 - \rho$ ) which gives an algorithm with running time  $O^*(2^{k\rho})$ . In other words, [Ban+19] and [MT18] improves the “saving factor” from  $1/\rho$  to  $2^{\Omega(1/\rho)}$  and  $2^{\Omega(1/\rho^2)}$  respectively. It should be noted however that, since the known  $(2 - o(1))$ -factor hardness of approximation is shown via the Unique Games Conjecture and unique games admit subexponential time algorithms [ABS15; BRS11], it is still entirely possible that this regime of approximating VERTEX COVER admits subexponential time algorithms as well. This is perhaps the biggest open question in the “barely non-trivial” approximation range:

**Open Question 4.33.** *Is there an algorithm that runs in  $2^{o(k)}n^{O(1)}$  time and achieves an approximation ratio of  $(2 - \rho)$  for some absolute constant  $\rho > 0$ ?*

Let us now briefly discuss the techniques used in some of the aforementioned works. The algorithms in [Ban+19; BF10] are based on branching in conjunction with certain approximation techniques. (See also [Fer12] where a similar technique is used for a related problem TOTAL

VERTEX COVER.) A key idea in [Ban+19; BF10] is that (i) if the (average or maximum) degree of the graph is small, then good polynomial-time approximation algorithms are known [Hal02] and (ii) if the degree is large, then branching algorithms are naturally already fast. The second part of [BF10] involves a delicate branching rule. However, for [Ban+19], it is quite simple: for some threshold  $d$  (to be specified), as long as there exists a vertex with degree at least  $d$ , then (1) with some probability, simply add the vertex to the vertex cover, or (2) branch on both possibilities of it being inside the cover and outside. After this branching finishes and we are left with low-degree graphs, just run the known polynomial-time approximation algorithms [Hal02] on these graphs. The point here is that the “error” incurred if option (1) is chosen will be absorbed by the approximation. By carefully selecting  $d$  and the probability, one can arrive at the desired running time and approximation guarantee. This algorithm is randomized, but can be derandomized using the sparsification lemma [IPZ01].

To the best of our knowledge, this “barely non-trivial approximation” regime has not been studied beyond VERTEX COVER. In particular, while Bansal et al. [Ban+19] apply their techniques on several problems, these are not parameterized problems and we are not aware of any other parameterized study related to the regime discussed here.

Parallel to the running time questions we have discussed so far, one may ask an analogous question in the kernelization regime: *does approximation allow us to find smaller kernels for problems that already admit polynomial-size kernels?* As is the case with exact algorithms, parameterized approximation algorithms go hand in hand with approximate kernels. Indeed, many algorithmic improvements mentioned can also be viewed as improvements in terms of the size of the kernels. In particular, recall the proof sketch of Theorem 4.31 for VERTEX COVER. If we stop and do not proceed with brute force in the second step, then we are left with an  $(1 + \epsilon)$ -approximate kernel. It is also not hard to argue that, by for instance applying the standard  $2k$ -size kernelization at the end, we are left with at most  $2(1 - \epsilon)k$  vertices. This improves upon the best known  $2k - \Theta(\log k)$  bound for the exact kernel [Lam11]. A similar improvement is known also for  $d$ -HITTING SET [Fel+12].

## 5 Future Directions

Although we have provided open questions along the way, we end this survey by zooming out and discussing some general future directions or meta-questions, which we find to be interesting and could be the basis for future work.

### Approximation factors.

The quality of a polynomial-time approximation algorithm is mainly measured by the obtainable approximation factor  $\alpha$ : the smaller it is the more feasibly solvable the problem is. Therefore, a lot of work has been invested into determining the smallest obtainable approximation factor  $\alpha$  for all kinds of computationally hard problems. In the non-parameterized (i.e. NP-hardness) world, a whole spectrum of approximability has been discovered (cf. [Vaz01; WS11]): the most feasibly solvable NP-hard problems (e.g., the KNAPSACK problem) admit a so-called *polynomial-time approximation scheme (PTAS)*, which is an algorithm computing a  $(1 + \epsilon)$ -approximation for any given constant  $\epsilon > 0$ . Some problems can be shown not to admit a PTAS (under reasonable complexity assumptions), but still allow constant approximation factors (e.g., the STEINER TREE problem). Yet others can only be approximated within a polylogarithmic factor (e.g., the

SET COVER problem), while some are even harder than this, as the best approximation factor obtainable is polynomial in the input size (e.g., the CLIQUE problem).

In contrast to polynomial-time approximation algorithms, a full spectrum of obtainable approximation ratios is still missing when allowing parameterized runtimes. Instead, only some scattered basic results are known. In particular, most of parameterized approximation problems belongs to one of the following categories:

- A *parameterized approximation scheme (PAS)* exists, i.e., for any constant  $\varepsilon > 0$  a  $(1 + \varepsilon)$ -approximation can be computed in  $f(k)n^{O(1)}$  time for some parameter  $k$ . These are currently the most prevalent types of results in the literature. To just mention one example, the STEINER TREE problem is APX-hard, but admits a PAS [Dvo+18] when parameterized by the number of non-terminals (so-called *Steiner vertices*) in the optimum solution (cf. Section 4.4).
- A lower bound excluding any non-trivial approximation factor exists. For example, under ETH the DOMINATING SET problem has no  $g(k)$ -approximation in  $f(k)n^{o(k)}$  time [KLM19] for any functions  $g$  and  $f$ , where  $k$  is the size of the largest dominating set.
- A polynomial-time approximation algorithm can achieve a similar approximation ratio, i.e., the parameterization is not very helpful. For instance, for the  $k$ -CENTER problem<sup>21</sup> a 2-approximation can be computed in polynomial time [HS86], but even when parameterizing by  $k$  no  $(2 - \varepsilon)$ -approximation is possible [Fel15] for any  $\varepsilon > 0$ , under standard complexity assumptions. A similar situation holds for MAX  $k$ -COVERAGE, which we discussed in Section 4.2.2.
- Constant or logarithmic approximation ratios can be shown, and which beat any approximation ratio obtainable in polynomial time. For instance, STRONGLY CONNECTED STEINER SUBGRAPH problem : under standard complexity assumptions, for this problem no polynomial-time  $O(\log^{2-\varepsilon} n)$ -approximation algorithm exists [HK03], and there is no FPT algorithm parameterized by the number  $k$  of terminals [GNS11]. However it is not hard to compute a 2-approximation in  $2^{O(k)}n^{O(1)}$  time [CHK13], and no  $(2 - \varepsilon)$ -approximation algorithm with runtime  $f(k)n^{O(1)}$  exists [CFM18] under Gap-ETH, for any function  $f$  and any  $\varepsilon > 0$  (cf. Section 4.4).

For many problems discussed in this survey, including DENSEST  $k$ -SUBGRAPH, STEINER TREE with bounded doubling / highway dimension, it has not been determined which category they belong. There are also a lot of problems in the final category for which asymptotically tight approximation ratios have not been found, including DIRECTED MULTICUT, TREewidth  $k$ -DELETION (both weighted and unweighted). The parameterized approximability of  $\mathcal{H}$ -MINOR DELETION for non-planar  $\mathcal{H}$  is also wide open except MINIMUM PLANARIZATION ( $\mathcal{H} = \{K_5, K_{3,3}\}$ ) [KS17]. It is an immediate but still interesting direction to prove tight parameterized approximation ratios for these (and more) problems.

Digressing, we remark that this survey does not include FPT-approximation of counting problems, such as approximately counting number of  $k$ -paths in a graph. The best  $(1 + \varepsilon)$ -multiplicative factor algorithm known [BDH18; Bjö+19] for counting number of  $k$ -paths runs in time  $4^k f(\varepsilon) \text{poly}(n)$  for some subexponential function  $f$  (cf. [Pra19]). So a natural question is: can we count  $k$ -paths approximately in time  $c^k$ , where  $c$  is as close to the base of running time of the algorithm of deciding existence of  $k$ -Path in a graph (the best currently known  $c$  is roughly 1.657 [Bjö14; Bjö+17])?

<sup>21</sup>Here we consider the version where the set of candidate centers is not separately given.

## Parameterized running times.

The quality of FPT algorithms is mainly measured in the obtainable runtime. Given a parameter  $k$ , for some problems the optimum solution can be computed in  $f(k)n^{g(k)}$  time, for some functions  $f$  and  $g$  independent of the input size  $n$  (i.e., the degree of the polynomial also depends on the parameter). If such an algorithm exists the problem is *slice-wise polynomial* (XP), and the algorithm is called an *XP algorithm*. A typical example is if a solution of size  $k$  is to be found within a data set of size  $n$ , in which case often an  $n^{O(k)}$  time exhaustive search algorithm exists. However, an FPT algorithm with runtime, say,  $O(2^k n)$  is a lot more efficient than an XP algorithm with runtime  $n^{O(k)}$ , and therefore the aim is usually to find FPT algorithms, while XP algorithms are counted as prohibitively slow. The discovery of the W-hierarchy in complexity theory has paved the way to providing evidence when an FPT algorithm is unlikely to exist. Assuming ETH, it is even possible to provide lower bounds on the runtimes obtainable by any FPT or XP algorithm. Similar to approximation algorithms, this has led to the discovery of a spectrum of tractability (cf. [Cyg+15]): starting from slightly sub-exponential  $2^{O(\sqrt{k})}n^{O(1)}$  time, through single exponential  $2^{O(k)}n^{O(1)}$  time, to double exponential  $2^{2^{O(k)}}n^{O(1)}$  time for FPT algorithms with matching asymptotic lower bounds under ETH (e.g., for the PLANAR VERTEX COVER, VERTEX COVER, and EDGE CLIQUE COVER problems, respectively, each parameterized by the solution size). Also for XP algorithms, asymptotically tight runtime bounds of the form  $n^{O(\sqrt{k})}$  and  $n^{O(k)}$  can be obtained under ETH (e.g., for the CLIQUE problem parameterized by the solution size, and the PLANAR BIDIRECTED STEINER NETWORK problem parameterized by the number of terminals [CFM18], respectively). Finally, problems that are NP-hard when the given parameter is constant do not even allow XP algorithms unless  $P=NP$  (e.g., the GRAPH COLOURING problem where the parameter is the number of colours).

In terms of tight runtime bounds, existing results on parameterized approximation algorithms are few and far between. In particular, most of them show that for a given parameter  $k$  one of the following cases applies.

- An approximation is possible in  $f(k)n^{O(1)}$  time for *some* function  $f$ . Most current results are only concerned with the existence of an algorithm with this type of runtime, i.e., they do not provide any evidence that the obtained runtime is best possible, or try to optimize it. The only lower bounds known exclude certain types of approximation schemes when a hardness result for the parameterization by the solution size exists. For instance, it is known that if some problem does not admit a  $2^{o(k)}n^{O(1)}$  time algorithm for this parameter  $k$  then it also does not admit an EPTAS with runtime  $2^{o(1/\epsilon)}n^{O(1)}$  (cf. [DF13; Mar08]).
- A certain approximation ratio cannot be obtained in  $f(k)n^{O(1)}$  time for *any* function  $f$ . For example, it is known that while a 2-approximation for the STRONGLY CONNECTED STEINER SUBGRAPH problem can be computed in  $2^{O(k)}n^{O(1)}$  time [CHK13], where  $k$  is the number of terminals, no  $(2 - \epsilon)$ -approximation can be computed in  $f(k)n^{O(1)}$  time [CFM18] for any function  $f$ , under Gap-ETH (cf. Section 4.4).

Hence, matching lower bounds on the time needed to compute an approximation are missing. For example, is the runtime of  $2^{O(k)}n^{O(1)}$  best possible to compute a 2-approximation for the STRONGLY CONNECTED STEINER SUBGRAPH problem? Could there be a  $2^{O(\sqrt{k})}n^{O(1)}$  time algorithm to compute a 2-approximation as well? But also for PASs the exact obtainable runtime is often elusive, even if certain types of approximation schemes can be excluded. For instance, for the STEINER TREE problem parameterized by the number of Steiner vertices in the optimum solution a  $(1 + \epsilon)$ -approximation can be computed in  $2^{O(k^2/\epsilon^4)}n^{O(1)}$  time [Dvo+18].

But is the dependence on  $k$  and  $\varepsilon$  best possible? Could there be a  $2^{O(k/\varepsilon^4)}n^{O(1)}$  or  $2^{O(k^2/\varepsilon)}n^{O(1)}$  time algorithm as well?

We remark that, for problems for which straightforward algorithms are known to be (essentially) the best possible in FPT time, or for which an improvement over polynomial time approximation is not possible, sometimes tight running time lower bounds are known in conjunction with tight inapproximability ratios. This includes  $k$ -DOMINATING SET (Section 3.1.2),  $k$ -CLIQUE (Section 3.2.1) and MAX  $k$ -COVERAGE (Section 4.2.2).

## Kernel sizes.

The development of *compositionality* has led to a theory from which lower bounds on the size of the smallest possible kernel of a problem can be derived (under reasonable complexity assumptions). The spectrum (cf. [Cyg+15]) here reaches from polynomial-sized kernels (e.g., for any  $q \geq 3$  and  $\varepsilon > 0$  the  $q$ -SAT problem parameterized by the number of variables  $n$  has no  $O(n^{q-\varepsilon})$ -sized kernel) to exponential-sized kernels (e.g., the STEINER TREE problem parameterized by the number of terminals does not admit any polynomial-sized kernel despite being FPT).

For approximate kernels, only a small number of publications exist, and the few known results fall into two categories:

- A *polynomial-sized approximate kernelization scheme (PSAKS)* exists, i.e., for any  $\varepsilon > 0$  there is a  $(1 + \varepsilon)$ -approximate kernelization algorithm that computes a  $(1 + \varepsilon)$ -approximate kernel of size polynomial in the parameter  $k$ . For example, the STEINER TREE problem admits a PSAKS for both the parameterization in the number of terminals [Lok+17] and in the number of Steiner vertices in the optimum [Dvo+18], even though neither of these two parameters admits a polynomial-sized (exact) kernel.
- A lower bound excluding any approximation factor for polynomial-sized kernels exists. For example, the LONGEST PATH problem parameterized by the maximum path length has no  $\alpha$ -approximate polynomial-sized kernel for any  $\alpha$  [Lok+17], despite being FPT for this parameter [Cyg+15].

Hence again the intermediate cases, for which tight constant or logarithmic approximation factors can be proved for polynomial-sized kernels, are missing. Studying approximate kernelization algorithms however is of undeniable importance to the field of parameterized approximation algorithms, as witnessed by the importance of exact kernelization to fixed-parameter tractability.

## Completeness in hardness of approximation.

A final direction we would like to highlight is to obtain more *completeness* in inapproximability results. Most of the results so far for FPT hardness of approximation either (i) rely on gap hypothesis or (ii) yield a hardness in terms of the W-hierarchy but the exact version of the problem is known to be complete on an even higher level (e.g., DOMINATING SET is known to be W[1]-hard to approximate but its exact version is W[2]-complete). We have discussed (i) extensively in Section 3.2 and some examples of (ii) in Section 3.1. There are also some examples of (ii) that are not covered here; for instance, Marx [Mar13] showed W[t]-hardness for certain monotone/anti-monotone circuit satisfiability problems and the exact versions of these problems are known to be complete for higher levels of the W-hierarchy. The situation here is

unlike that in the theory of NP-hardness of approximation; there the PCP Theorem [Aro+98; AS92] implies NP-completeness of optimization problems<sup>22</sup>.

Thus, in the parameterized inapproximability arena, the main question here is whether we can prove completeness results for hardness of approximation for the aforementioned problems. The two important examples here are: is  $k$ -CLIQUE W[1]-hard to approximate, and is  $k$ -DOMINATING SET W[2]-hard to approximate? As discussed in Section 3.2, the former is also closely related to resolving PIH.

Finally, we note that, while completeness results are somewhat rare in FPT hardness of approximation, some are known. We give two such examples here. First is the  $k$ -STEINER ORIENTATION problem, discussed in Section 3.1.3; it is W[1]-complete to approximate [Wlo19]. Second is the MONOTONE CIRCUIT SATISFIABILITY problem (without depth bound), which was proved to be W[P]-complete by Marx [Mar13]. However, it does not seem clear to us whether these techniques can be applied elsewhere, e.g., for  $k$ -CLIQUE.

## References

- [AAC07] A. Agarwal, N. Alon and M. S. Charikar. ‘Improved approximation for directed cut problems’. In: *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. ACM. 2007, pp. 671–680.
- [ABS15] S. Arora, B. Barak and D. Steurer. ‘Subexponential Algorithms for Unique Games and Related Problems’. *J. ACM* 62.5 (2015), 42:1–42:25.
- [Abu10] F. N. Abu-Khzam. ‘A kernelization algorithm for d-Hitting Set’. *J. Comput. Syst. Sci.* 76.7 (2010), pp. 524–531.
- [ACP87] S. Arnborg, D. G. Corneil and A. Proskurowski. ‘Complexity of finding embeddings in  $ak$ -tree’. *SIAM Journal on Algebraic Discrete Methods* 8.2 (1987), pp. 277–284.
- [Ada+18] M. Adamczyk, J. Byrka, J. Marcinkowski, S. M. Meesum and M. Włodarczyk. ‘Constant factor FPT approximation for capacitated  $k$ -median’. *arXiv preprint arXiv:1809.05791* (2018). To appear in ESA ’19.
- [AF04] J. Alber and J. Fiala. ‘Geometric separation and exact solutions for the parameterized independent set problem on disk graphs’. *J. Algorithms* 52.2 (2004), pp. 134–151.
- [Agr+18] A. Agrawal, D. Lokshtanov, P. Misra, S. Saurabh and M. Zehavi. ‘Polylogarithmic Approximation Algorithms for Weighted-F-Deletion Problems’. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.
- [Ajt98] M. Ajtai. ‘The Shortest Vector Problem in  $\ell_2$  is NP-hard for Randomized Reductions (Extended Abstract)’. In: *STOC*. 1998, pp. 10–19.
- [Alo+11] N. Alon, S. Arora, R. Manokaran, D. Moshkovitz and O. Weinstein. ‘Inapproximability of Densest  $k$ -Subgraph from Average Case Hardness’. Unpublished Manuscript. 2011.

---

<sup>22</sup>To be more precise, these problems need to be phrased as *promise problems* and NP-hardness is with respect to these. We will not go into details here.

- [Ang+16] E. Angel, E. Bampis, B. Escoffier and M. Lampis. ‘Parameterized power vertex cover’. In: *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 2016, pp. 97–108.
- [Aro+93] S. Arora, L. Babai, J. Stern and Z. Sweedyk. ‘The Hardness of Approximate Optima in Lattices, Codes, and Systems of Linear Equations’. In: *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*. 1993, pp. 724–733.
- [Aro98] S. Arora. ‘Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems’. *Journal of the ACM (JACM)* 45.5 (1998), pp. 753–782.
- [Aro+98] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. ‘Proof Verification and the Hardness of Approximation Problems’. *J. ACM* 45.3 (1998), pp. 501–555.
- [ARR98] S. Arora, P. Raghavan and S. Rao. ‘Approximation Schemes for Euclidean  $k$ -Medians and Related Problems.’ In: *STOC*. Vol. 98. 1998, pp. 106–113.
- [ARV09] S. Arora, S. Rao and U. Vazirani. ‘Expander flows, geometric embeddings and graph partitioning’. *Journal of the ACM (JACM)* 56.2 (2009), p. 5.
- [ARW17] A. Abboud, A. Rubinfeld and R. R. Williams. ‘Distributed PCP Theorems for Hardness of Approximation in  $P$ ’. In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*. 2017, pp. 25–36.
- [AS92] S. Arora and S. Safra. ‘Probabilistic Checking of Proofs; A New Characterization of NP’. In: *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*. 1992, pp. 2–13.
- [AW13] A. Adamaszek and A. Wiese. ‘Approximation schemes for maximum weight independent set of rectangles’. In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE. 2013, pp. 400–409.
- [AYZ95] N. Alon, R. Yuster and U. Zwick. ‘Color-coding’. *J. ACM* 42.4 (1995), pp. 844–856.
- [Bak94] B. S. Baker. ‘Approximation algorithms for NP-complete problems on planar graphs’. *Journal of the ACM (JACM)* 41.1 (1994), pp. 153–180.
- [Ban+19] N. Bansal, P. Chalermsook, B. Laekhanukit, D. Nanongkai and J. Nederlof. ‘New Tools and Connections for Exponential-Time Approximation’. *Algorithmica* 81.10 (2019), pp. 3993–4009.
- [Bar+04] R. Bar-Yehuda, K. Bendel, A. Freund and D. Rawitz. ‘Local ratio: A unified framework for approximation algorithms’. *ACM Comput. Surv.* 36.4 (2004), pp. 422–463.
- [Bat+15] J. Batra, N. Garg, A. Kumar, T. Mömke and A. Wiese. ‘New approximation schemes for unsplittable flow on a path’. In: *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2015, pp. 47–58.
- [BD97] A. Borchers and D.-Z. Du. ‘The  $k$ -Steiner Ratio in Graphs’. *SIAM journal on Computing* 26.3 (1997), pp. 857–869.



- [BDH18] C. Brand, H. Dell and T. Husfeldt. ‘Extensor-coding’. In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*. 2018, pp. 151–164.
- [BDZ18] A. Babay, M. Dinitz and Z. Zhang. ‘Characterizing Demand Graphs for (Fixed-Parameter) Shallow-Light Steiner Network’. In: *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.
- [Bec+19] L. Becchetti, M. Bury, V. Cohen-Addad, F. Grandoni and C. Schwiegelshohn. ‘Oblivious dimension reduction for k-means: beyond subspaces and the Johnson-Lindenstrauss lemma’. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 2019, pp. 1039–1050.
- [Bel+93] M. Bellare, S. Goldwasser, C. Lund and A. Russell. ‘Efficient probabilistically checkable proofs and applications to approximations’. In: *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*. 1993, pp. 294–304.
- [BEP09] N. Bourgeois, B. Escoffier and V. T. Paschos. ‘Efficient Approximation of Combinatorial Problems by Moderately Exponential Algorithms’. In: *WADS*. 2009, pp. 507–518.
- [Bér+19] K. Bérczi, K. Chandrasekaran, T. Király and V. Madan. ‘Improving the Integrality Gap for Multiway Cut’. In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 2019, pp. 115–127.
- [BF10] L. Brankovic and H. Fernau. ‘Combining Two Worlds: Parameterised Approximation for Vertex Cover’. *Algorithms and Computation* (2010), pp. 390–402.
- [BF13] L. Brankovic and H. Fernau. ‘A novel parameterised approximation algorithm for minimum vertex cover’. *Theor. Comput. Sci.* 511 (2013), pp. 85–108.
- [BG93] J. F. Buss and J. Goldsmith. ‘Nondeterminism Within P’. *SIAM J. Comput.* 22.3 (1993), pp. 560–572.
- [BGS98] M. Bellare, O. Goldreich and M. Sudan. ‘Free Bits, PCPs, and Nonapproximability-Towards Tight Results’. *SIAM J. Comput.* 27.3 (1998), pp. 804–915.
- [Bha+17] A. Bhangale, R. Gandhi, M. T. Hajiaghayi, R. Khandekar and G. Kortsarz. ‘Bi-Covering: Covering Edges with Two Small Subsets of Vertices’. *SIAM J. Discrete Math.* 31.4 (2017), pp. 2626–2646.
- [Bha+19] A. Bhattacharyya, É. Bonnet, L. Egri, S. Ghoshal, Karthik C. S., B. Lin, P. Manurangsi and D. Marx. ‘Parameterized Intractability of Even Set and Shortest Vector Problem’. *Electronic Colloquium on Computational Complexity (ECCC)* 26 (2019), p. 115.
- [BHPI02] M. Bădoiu, S. Har-Peled and P. Indyk. ‘Approximate clustering via core-sets’. In: *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*. ACM. 2002, pp. 250–257.
- [Bjö14] A. Björklund. ‘Determinant Sums for Undirected Hamiltonicity’. *SIAM J. Comput.* 43.1 (2014), pp. 280–299.
- [Bjö+17] A. Björklund, T. Husfeldt, P. Kaski and M. Koivisto. ‘Narrow sieves for parameterized paths and packings’. *J. Comput. Syst. Sci.* 87 (2017), pp. 119–139.

- [Bjö+19] A. Björklund, D. Lokshtanov, S. Saurabh and M. Zehavi. ‘Approximate Counting of  $k$ -Paths: Deterministic and in Polynomial Space’. In: *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*. 2019, 24:1–24:15.
- [BKL12] A. Badanidiyuru, R. Kleinberg and H. Lee. ‘Approximating low-dimensional coverage problems’. In: *Proceedings of the twenty-eighth annual symposium on Computational geometry*. ACM. 2012, pp. 161–170.
- [BKS18] A. Becker, P. N. Klein and D. Saulpic. ‘Polynomial-time approximation schemes for  $k$ -center,  $k$ -median, and capacitated vehicle routing in bounded highway dimension’. In: *26th Annual European Symposium on Algorithms (ESA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.
- [Bli+18] I. Bliznets, M. Cygan, P. Komosa and M. Pilipczuk. ‘Hardness of approximation for  $H$ -free edge modification problems’. *ACM Transactions on Computation Theory (TOCT)* 10.2 (2018), p. 9.
- [BLM18] R. Belmonte, M. Lampis and V. Mitsou. ‘Parameterized (Approximate) Defective Coloring’. In: *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.
- [Boc+07] H.-J. Bockenhauer, J. Hromkovic, J. Kneis and J. Kupke. ‘The parameterized approximability of TSP with deadlines’. *Theory of Computing Systems* 41.3 (2007), pp. 431–444.
- [Bod07] H. L. Bodlaender. ‘Treewidth: Structure and algorithms’. In: *International Colloquium on Structural Information and Communication Complexity*. Springer. 2007, pp. 11–25.
- [Bod+16] H. L. Bodlaender, P. G. Drange, M. S. Dregi, F. V. Fomin, D. Lokshtanov and M. Pilipczuk. ‘A  $c^k n$  5-Approximation Algorithm for Treewidth’. *SIAM Journal on Computing* 45.2 (2016), pp. 317–378.
- [Bod96] H. L. Bodlaender. ‘A linear-time algorithm for finding tree-decompositions of small treewidth’. *SIAM Journal on computing* 25.6 (1996), pp. 1305–1317.
- [BPS16] É. Bonnet, V. T. Paschos and F. Sikora. ‘Parameterized exact and approximation algorithms for maximum  $k$ -set cover and related satisfiability problems’. *RAIRO-Theoretical Informatics and Applications* 50.3 (2016), pp. 227–240.
- [Bra+20] V. Braverman, S. H.-C. Jiang, R. Krauthgamer and X. Wu. ‘Coresets for Clustering in Excluded-minor Graphs and Beyond’. *arXiv preprint arXiv:2004.07718* (2020).
- [BRS11] B. Barak, P. Raghavendra and D. Steurer. ‘Rounding Semidefinite Programming Hierarchies via Global Correlation’. In: *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS*. 2011, pp. 472–481.
- [BRU17] N. Bansal, D. Reichman and S. W. Umboh. ‘LP-based robust algorithms for noisy minor-free and bounded treewidth graphs’. In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 1964–1979.
- [BS92] P. Berman and G. Schnitger. ‘On the Complexity of Approximating the Independent Set Problem’. *Inf. Comput.* 96.1 (1992), pp. 77–94.
- [BTY11] H. L. Bodlaender, S. Thomassé and A. Yeo. ‘Kernel bounds for disjoint cycles and disjoint paths’. *Theoretical Computer Science* 412.35 (2011), pp. 4570–4578.

- [Byr+13] J. Byrka, F. Grandoni, T. Rothvoss and L. Sanità. ‘Steiner Tree Approximation via Iterative Randomized Rounding’. *journal of the ACM* 60.1 (2013), pp. 1–33.
- [Byr+14] J. Byrka, T. Pensyl, B. Rybicki, A. Srinivasan and K. Trinh. ‘An improved approximation for k-median, and positive correlation in budgeted optimization’. In: *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*. SIAM. 2014, pp. 737–756.
- [Cai+97] L. Cai, J. Chen, R. G. Downey and M. R. Fellows. ‘Advice Classes of Parameterized Tractability’. *Ann. Pure Appl. Logic* 84.1 (1997), pp. 119–138.
- [Cal+11] G. Calinescu, C. Chekuri, M. Pál and J. Vondrák. ‘Maximizing a monotone submodular function subject to a matroid constraint’. *SIAM Journal on Computing* 40.6 (2011), pp. 1740–1766.
- [CC08] M. Chlebík and J. Chlebíková. ‘The Steiner tree problem on graphs: Inapproximability results’. *Theoretical Computer Science* 406.3 (2008), pp. 207–214.
- [CC16] C. Chekuri and J. Chuzhoy. ‘Polynomial bounds for the grid-minor theorem’. *Journal of the ACM (JACM)* 63.5 (2016), p. 40.
- [CC93] L. Cai and J. Chen. ‘On fixed-parameter tractability and approximability of NP-hard optimization problems’. In: *[1993] The 2nd Israel Symposium on Theory and Computing Systems*. IEEE. 1993, pp. 118–126.
- [CC97] L. Cai and J. Chen. ‘On fixed-parameter tractability and approximability of NP optimization problems’. *Journal of Computer and System Sciences* 54.3 (1997), pp. 465–474.
- [CCJ90] B. N. Clark, C. J. Colbourn and D. S. Johnson. ‘Unit disk graphs’. *Discret. Math.* 86.1-3 (1990), pp. 165–177.
- [CDM17] E. Chlamtác, M. Dinitz and Y. Makarychev. ‘Minimizing the Union: Tight Approximations for Small Set Bipartite Vertex Expansion’. In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*. 2017, pp. 881–899.
- [CDVDM18] V. Cohen-Addad, É. C. De Verdière and A. De Mesmay. ‘A near-linear approximation scheme for multicuts of embedded graphs with a fixed number of terminals’. In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2018, pp. 1439–1458.
- [CF19] R. Chitnis and A. E. Feldmann. ‘FPT Inapproximability of Directed Cut and Connectivity Problems’. In: *IPEC*. 2019.
- [CFM18] R. Chitnis, A. E. Feldmann and P. Manurangsi. ‘Parameterized Approximation Algorithms for Bidirected Steiner Network Problems’. In: *Proceedings of the 26th Annual European Symposium on Algorithms (ESA)*. 2018, 20:1–20:16.
- [CFS19] V. Cohen-Addad, A. E. Feldmann and D. Saulpic. ‘Near-Linear Time Approximation Schemes for Clustering in Doubling Metrics’ (2019).
- [Cha+06] S. Chawla, R. Krauthgamer, R. Kumar, Y. Rabani and D. Sivakumar. ‘On the hardness of approximating multicut and sparsest-cut’. *computational complexity* 15.2 (2006), pp. 94–114.
- [Cha16] S. O. Chan. ‘Approximation Resistance from Pairwise-Independent Subgroups’. *J. ACM* 63.3 (2016), 27:1–27:32.

- [Cha+17] P. Chalermsook, M. Cygan, G. Kortsarz, B. Laekhanukit, P. Manurangsi, D. Nanongkai and L. Trevisan. ‘From Gap-ETH to FPT-Inapproximability: Clique, Dominating Set, and More’. In: *FOCS*. 2017, pp. 743–754.
- [Che+04] J. Chen, X. Huang, I. A. Kanj and G. Xia. ‘Linear FPT reductions and computational lower bounds’. In: *STOC*. 2004, pp. 212–221.
- [Che+06] J. Chen, X. Huang, I. A. Kanj and G. Xia. ‘Strong computational lower bounds via parameterized complexity’. *J. Comput. Syst. Sci.* 72.8 (2006), pp. 1346–1367.
- [Che06] K. Chen. ‘On k-median clustering in high dimensions’. In: *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. Society for Industrial and Applied Mathematics. 2006, pp. 1177–1185.
- [Che+07] J. Chen, X. Huang, I. A. Kanj and G. Xia. ‘Polynomial time approximation schemes and parameterized complexity’. *Discrete Applied Mathematics* 155.2 (2007), pp. 180–193.
- [Chi+16] R. Chitnis, M. Cygan, M. Hajiaghayi, M. Pilipczuk and M. Pilipczuk. ‘Designing FPT algorithms for cut problems using randomized contractions’. *SIAM Journal on Computing* 45.4 (2016), pp. 1171–1229.
- [CHK13] R. Chitnis, M. Hajiaghayi and G. Kortsarz. ‘Fixed-Parameter and Approximation Algorithms: A New Look’. In: *IPEC*. 2013, pp. 110–122.
- [Chl+18] E. Chlamtác, M. Dinitz, C. Konrad, G. Kortsarz and G. Rabanca. ‘The Densest k-Subhypergraph Problem’. *SIAM J. Discrete Math.* 32.2 (2018), pp. 1458–1477.
- [CK09] J. Chuzhoy and S. Khanna. ‘Polynomial flow-cut gaps and hardness of directed cut problems’. *Journal of the ACM (JACM)* 56.2 (2009), p. 6.
- [CK19] V. Cohen-Addad and Karthik C. S. ‘Inapproximability of Clustering in  $L_p$ -metrics’. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science*. IEEE. 2019.
- [CKM19] V. Cohen-Addad, P. N. Klein and C. Mathieu. ‘Local search yields approximation schemes for k-means and k-median in euclidean and minor-free metrics’. *SIAM Journal on Computing* 48.2 (2019), pp. 644–667.
- [CKN13] M. Cygan, G. Kortsarz and Z. Nutov. ‘Steiner Forest Orientation Problems’. *SIAM J. Discrete Math.* 27.3 (2013), pp. 1503–1513.
- [CKX06] J. Chen, I. A. Kanj and G. Xia. ‘Improved parameterized upper bounds for vertex cover’. In: *International symposium on mathematical foundations of computer science*. Springer. 2006, pp. 238–249.
- [CL19a] Y. Chen and B. Lin. ‘The Constant Inapproximability of the Parameterized Dominating Set Problem’. *SIAM J. Comput.* 48.2 (2019). Preliminary version in *FOCS* 2016, pp. 513–533.
- [CL19b] V. Cohen-Addad and J. Li. ‘On the Fixed-Parameter Tractability of Capacitated Clustering’. In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2019.
- [CLL07] J. Chen, Y. Liu and S. Lu. ‘Directed feedback vertex set problem is FPT’. In: *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2007.

- [CM14] Y. Cao and D. Marx. ‘Interval deletion is fixed-parameter tractable’. In: *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. SIAM. 2014, pp. 122–141.
- [CM17] C. Chekuri and V. Madan. ‘Approximating multicut and the demand graph’. In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 855–874.
- [CNW80] G. Cornuejols, G. L. Nemhauser and L. A. Wolsey. ‘Worst-case and probabilistic analysis of algorithms for a location problem’. *Operations Research* 28.4 (1980), pp. 847–858.
- [Cob64] A Cobham. ‘The intrinsic computational difficulty of functions’. In: *Proceedings of the 1964 Congress for Logic, Methodology, and the Philosophy of Science*. 1964, pp. 24–30.
- [Coh18] V. Cohen-Addad. ‘A fast approximation scheme for low-dimensional k-means’. In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2018, pp. 430–440.
- [Coh+19] V. Cohen-Addad, A. Gupta, A. Kumar, E. Lee and J. Li. ‘Tight FPT Approximations for k-Median and k-Means’. In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Ed. by C. Baier, I. Chatzigiannakis, P. Flocchini and S. Leonardi. Vol. 132. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 42:1–42:14.
- [Cou90] B. Courcelle. ‘The monadic second-order logic of graphs. I. Recognizable sets of finite graphs’. *Information and computation* 85.1 (1990), pp. 12–75.
- [CR05] D. G. Corneil and U. Rotics. ‘On the Relationship Between Clique-Width and Treewidth’. *SIAM J. Comput.* 34.4 (2005), pp. 825–847.
- [CS17] Y. Cao and R. Sandeep. ‘Minimum fill-in: Inapproximability and almost tight lower bounds’. In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 875–880.
- [CT97] M. Cesati and L. Trevisan. ‘On the efficiency of polynomial time approximation schemes’. *Information Processing Letters* 64.4 (1997), pp. 165–171.
- [Cyg12] M. Cygan. ‘Deterministic parameterized connected vertex cover’. In: *Scandinavian Workshop on Algorithm Theory*. Springer. 2012, pp. 95–106.
- [Cyg13] M. Cygan. ‘Improved approximation for 3-dimensional matching via bounded pathwidth local search’. In: *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE. 2013, pp. 509–518.
- [Cyg+15] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.
- [Cyg+16] M. Cygan, D. Lokshtanov, M. Pilipczuk, M. Pilipczuk and S. Saurabh. ‘Lower Bounds for Approximation Schemes for Closest String’. In: *15th Scandinavian Symposium and Workshops on Algorithm Theory*. 2016.
- [Cyg+19] M. Cygan, D. Lokshtanov, M. Pilipczuk, M. Pilipczuk and S. Saurabh. ‘Minimum Bisection is fixed-parameter tractable’. *SIAM Journal on Computing* 48.2 (2019), pp. 417–450.

- [Dem+19] E. D. Demaine, T. D. Goodrich, K. Kloster, B. Lavalley, Q. C. Liu, B. D. Sullivan, A. Vakilian and A. van der Poel. ‘Structural Rounding: Approximation Algorithms for Graphs Near an Algorithmically Tractable Class’. In: *27th Annual European Symposium on Algorithms (ESA 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2019.
- [DF13] R. G. Downey and M. R. Fellows. *Fundamentals of parameterized complexity*. Vol. 4. Springer, 2013.
- [DH04] E. D. Demaine and M. Hajiaghayi. ‘Equivalence of local treewidth and linear local treewidth and its algorithmic applications’. In: *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2004, pp. 840–849.
- [DH08] E. D. Demaine and M. Hajiaghayi. ‘The bidimensionality theory and its algorithmic applications’. *The Computer Journal* 51.3 (2008), pp. 292–302.
- [DHK05] E. D. Demaine, M. T. Hajiaghayi and K.-i. Kawarabayashi. ‘Algorithmic graph minor theory: Decomposition, approximation, and coloring’. In: *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS’05)*. IEEE. 2005, pp. 637–646.
- [DHK11] E. D. Demaine, M. Hajiaghayi and K.-i. Kawarabayashi. ‘Contraction decomposition in H-minor-free graphs and algorithmic applications’. In: *Proceedings of the forty-third annual ACM symposium on Theory of computing*. ACM. 2011, pp. 441–450.
- [Din07] I. Dinur. ‘The PCP theorem by gap amplification’. *J. ACM* 54.3 (2007), p. 12.
- [Din16] I. Dinur. ‘Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover’. *Electronic Colloquium on Computational Complexity (ECCC)* 23 (2016), p. 128.
- [DJ19] H. Donkers and B. M. Jansen. ‘A Turing Kernelization Dichotomy for Structural Parameterizations of  $F$ -Minor-Free Deletion’. In: *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 2019, pp. 106–119.
- [DL16] G. Demirci and S. Li. ‘Constant Approximation for Capacitated  $k$ -Median with  $(1 + \epsilon)$ -Capacity Violation’. *arXiv preprint arXiv:1603.02324* (2016).
- [DLS14] M. Dom, D. Lokshtanov and S. Saurabh. ‘Kernelization Lower Bounds Through Colors and IDs’. *ACM Transactions on Algorithms* 11.2 (2014), pp. 1–20.
- [DLV+03] W. F. De La Vega, M. Karpinski, C. Kenyon and Y. Rabani. ‘Approximation schemes for clustering problems’. In: *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*. ACM. 2003, pp. 50–58.
- [DM18] I. Dinur and P. Manurangsi. ‘ETH-Hardness of Approximating 2-CSPs and Directed Steiner Network’. In: *ITCS*. 2018, 36:1–36:20.
- [DM20] S. S. Diptapriyo Majumdar M. S. Ramanujan. ‘On the Approximate Compressibility of Connected Vertex Cover’. *Algorithmica* (2020).
- [DN17] I. Dinur and I. L. Navon. ‘Exponentially Small Soundness for the Direct Product Z-Test’. In: *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*. 2017, 29:1–29:50.

- [Dom+08] M. Dom, D. Lokshantov, S. Saurabh and Y. Villanger. ‘Capacitated domination and covering: A parameterized perspective’. In: *International Workshop on Parameterized and Exact Computation*. Springer. 2008, pp. 78–90.
- [Dow+99] R. G. Downey, M. R. Fellows, A. Vardy and G. Whittle. ‘The Parametrized Complexity of Some Fundamental Problems in Coding Theory’. *SIAM J. Comput.* 29.2 (1999), pp. 545–570.
- [Dvo+18] P. Dvořák, A. E. Feldmann, D. Knop, T. Masařík, T. Toufar and P. Veselý. ‘Parameterized Approximation Schemes for Steiner Trees with Small Number of Steiner Vertices’. In: *Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science (STACS)*. 2018, 26:1–26:15.
- [DW71] S. E. Dreyfus and R. A. Wagner. ‘The Steiner problem in graphs’. *Networks* 1.3 (1971), pp. 195–207.
- [EB81] P. van Emde-Boas. *Another NP-complete partition problem and the complexity of computing short vectors in a lattice*. Report. Department of Mathematics. University of Amsterdam. Department, Univ., 1981.
- [Edm65] J. Edmonds. ‘Paths, Trees, and Flowers’. *Canadian journal of Mathematics* 17 (1965), pp. 449–467.
- [Eib+18] E. Eiben, M. Kumar, A. E. Mouawad, F. Panolan and S. Siebertz. ‘Lossy Kernels for Connected Dominating Set on Sparse Graphs’. In: *STACS 2018*. Vol. 96. 2018, 29:1–29:15.
- [EKS] T. Ebenlendr, P. Kolman and J. Sgall. ‘An Approximation Algorithm for Bounded Degree Deletion’. *Manuscript* ().
- [EP65] P. Erdős and L. Pósa. ‘On Independent Circuits Contained in a Graph’. *Canadian Journal of Mathematics* 17 (1965), 347–352.
- [Esc+15] B. Escoffier, J. Monnot, V. T. Paschos and M. Xiao. ‘New results on polynomial inapproximability and fixed parameter approximability of edge dominating set’. *Theory of Computing Systems* 56.2 (2015), pp. 330–346.
- [Fei+96] U. Feige, S. Goldwasser, L. Lovász, S. Safra and M. Szegedy. ‘Interactive Proofs and the Hardness of Approximating Cliques’. *J. ACM* 43.2 (1996), pp. 268–292.
- [Fei98] U. Feige. ‘A threshold of  $\ln n$  for approximating set cover’. *Journal of the ACM (JACM)* 45.4 (1998), pp. 634–652.
- [FEKS19] E. Fox-Epstein, P. N. Klein and A. Schild. ‘Embedding Planar Graphs into Low-Treewidth Graphs with Applications to Efficient Approximation Schemes for Metric Problems’. In: *SODA*. 2019.
- [Fel+11] M. R. Fellows, F. V. Fomin, D. Lokshantov, F. Rosamond, S. Saurabh, S. Szeider and C. Thomassen. ‘On the complexity of some colorful problems parameterized by treewidth’. *Information and Computation* 209.2 (2011), pp. 143–153.
- [Fel+12] M. R. Fellows, A. Kulik, F. Rosamond and H. Shachnai. ‘Parameterized approximation via fidelity preserving transformations’. In: *ICALP*. 2012, pp. 351–362.
- [Fel15] A. E. Feldmann. ‘Fixed Parameter Approximations for k-Center Problems in Low Highway Dimension Graphs’. In: *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP)*. Springer Berlin Heidelberg, 2015, pp. 588–600.

- [Fel+18] A. E. Feldmann, W. S. Fung, J. Könemann and I. Post. ‘A  $(1+\epsilon)$ -Embedding of Low Highway Dimension Graphs into Bounded Treewidth Graphs’. *SIAM Journal on Computing* 47.4 (2018), pp. 1667–1704.
- [Fer12] H. Fernau. ‘Saving on Phases: Parameterized Approximation for Total Vertex Cover’. In: *Combinatorial Algorithms, 23rd International Workshop, IWOCA 2012, Tamil Nadu, India, July 19-21, 2012, Revised Selected Papers*. 2012, pp. 20–31.
- [FG06] J. Flum and M. Grohe. *Parameterized complexity theory*. Springer, 2006.
- [FHL08] U. Feige, M. Hajiaghayi and J. R. Lee. ‘Improved approximation algorithms for minimum weight vertex separators’. *SIAM Journal on Computing* 38.2 (2008), pp. 629–657.
- [FJP10] S. Fiorini, G. Joret and U. Pietropaoli. ‘Hitting diamonds and growing cacti’. In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 2010, pp. 191–204.
- [FL11] D. Feldman and M. Langberg. ‘A unified framework for approximating and clustering data’. In: *Proceedings of the forty-third annual ACM symposium on Theory of computing*. ACM. 2011, pp. 569–578.
- [FM06] U. Feige and M. Mahdian. ‘Finding small balanced separators’. In: *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. ACM. 2006, pp. 375–384.
- [FM18] A. E. Feldmann and D. Marx. ‘The Parameterized Hardness of the k-Center Problem in Transportation Networks’. In: *Proceedings of the 16th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*. 2018, 19:1–19:13.
- [FMS07] D. Feldman, M. Monemizadeh and C. Sohler. ‘A PTAS for k-means clustering based on weak coresets’. In: *Proceedings of the twenty-third annual symposium on Computational geometry*. ACM. 2007, pp. 11–18.
- [Fom+11] F. V. Fomin, D. Lokshtanov, V. Raman and S. Saurabh. ‘Bidimensionality and EPTAS’. In: *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2011, pp. 748–759.
- [Fom+12] F. V. Fomin, D. Lokshtanov, N. Misra and S. Saurabh. ‘Planar F-deletion: Approximation, kernelization and optimal FPT algorithms’. In: *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*. IEEE. 2012, pp. 470–479.
- [Fom+19] F. V. Fomin, T.-N. Le, D. Lokshtanov, S. Saurabh, S. Thomassé and M. Zehavi. ‘Subquadratic kernels for implicit 3-hitting set and 3-set packing problems’. *ACM Transactions on Algorithms (TALG)* 15.1 (2019), pp. 1–44.
- [FRS19] Z. Friggstad, M. Rezapour and M. R. Salavatipour. ‘Local search yields a PTAS for k-means in doubling metrics’. *SIAM Journal on Computing* 48.2 (2019), pp. 452–480.
- [FS07] Z. Friggstad and M. R. Salavatipour. ‘Approximability of packing disjoint cycles’. In: *International Symposium on Algorithms and Computation*. Springer. 2007, pp. 304–315.



- [Fuc+07] B. Fuchs, W. Kern, D Molle, S. Richter, P. Rossmanith and X. Wang. ‘Dynamic programming for minimum Steiner trees’. *Theory of Computing Systems* 41.3 (2007), pp. 493–500.
- [GGJ77] M. R. Garey, R. L. Graham and D. S. Johnson. ‘The complexity of computing Steiner minimal trees’. *SIAM journal on applied mathematics* 32.4 (1977), pp. 835–859.
- [Gia+17] A. C. Giannopoulou, M. Pilipczuk, J.-F. Raymond, D. M. Thilikos and M. Wrochna. ‘Linear Kernels for Edge Deletion Problems to Immersion-Closed Graph Classes’. In: *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2017.
- [GK00] K. Giaro and M. Kubale. ‘Edge-chromatic sum of trees and bounded cyclicity graphs’. *Information Processing Letters* 75.1-2 (2000), pp. 65–69.
- [GK99] S. Guha and S. Khuller. ‘Greedy strikes back: Improved facility location algorithms’. *Journal of algorithms* 31.1 (1999), pp. 228–248.
- [GKK11] J. Guo, I. Kanj and S. Kratsch. ‘Safe approximation and its relation to kernelization’. In: *International Symposium on Parameterized and Exact Computation*. Springer. 2011, pp. 169–180.
- [GKM08] N. Garg, A. Kumar and V Muralidhara. ‘Minimizing Total Flow-Time: The Unrelated Case’. *Algorithms and Computation* (2008), pp. 424–435.
- [GKR13] M. Grohe, K.-i. Kawarabayashi and B. Reed. ‘A simple algorithm for the graph minor decomposition: logic meets structural graph theory’. In: *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2013, pp. 414–431.
- [GKW19] F. Grandoni, S. Kratsch and A. Wiese. ‘Parameterized Approximation Schemes for Independent Set of Rectangles and Geometric Knapsack’. In: *European Symposium on Algorithms*. 2019.
- [GL15] V. Guruswami and E. Lee. ‘Inapproximability of H-Transversal/Packing’. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2015.
- [GLL18a] A. Gupta, E. Lee and J. Li. ‘An FPT algorithm beating 2-approximation for k-cut’. In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2018, pp. 2821–2837.
- [GLL18b] A. Gupta, E. Lee and J. Li. ‘Faster exact and approximate algorithms for k-cut’. In: *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2018, pp. 113–123.
- [GLL19] A. Gupta, E. Lee and J. Li. ‘The number of minimum k-cuts: improving the Karger-Stein bound’. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. ACM. 2019, pp. 229–240.
- [GNS11] J. Guo, R. Niedermeier and O. Suchý. ‘Parameterized Complexity of Arc-Weighted Directed Steiner Problems’. *SIAM J. Discrete Math.* 25.2 (2011), pp. 583–599.
- [GNW07] J. Guo, R. Niedermeier and S. Wernicke. ‘Parameterized complexity of vertex cover variants’. *Theory of Computing Systems* 41.3 (2007), pp. 501–520.

- [Gol08] O. Goldreich. *Computational Complexity: A Conceptual Perspective*. 1st ed. New York, NY, USA: Cambridge University Press, 2008.
- [Gon85] T. F. Gonzalez. ‘Clustering to minimize the maximum intercluster distance’. *Theoretical Computer Science* 38 (1985), pp. 293–306.
- [Got15] L. Gottlieb. ‘A Light Metric Spanner’. In: *FOCS*. 2015.
- [Gup+19] A. Gupta, E. Lee, J. Li, P. Manurangsi and M. Włodarczyk. ‘Losing tree-width by separating subsets’. In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2019, pp. 1731–1749.
- [GVY96] N. Garg, V. V. Vazirani and M. Yannakakis. ‘Approximate max-flow min-(multi) cut theorems and their applications’. *SIAM Journal on Computing* 25.2 (1996), pp. 235–251.
- [Hal02] E. Halperin. ‘Improved Approximation Algorithms for the Vertex Cover Problem in Graphs and Hypergraphs’. *SIAM J. Comput.* 31.5 (2002), pp. 1608–1623.
- [Hås01] J. Håstad. ‘Some optimal inapproximability results’. *J. ACM* 48.4 (2001), pp. 798–859.
- [Has92] R. Hassin. ‘Approximation schemes for the restricted shortest path problem’. *Mathematics of Operations research* 17.1 (1992), pp. 36–42.
- [Hås96] J. Håstad. ‘Clique is Hard to Approximate Within  $n^{1-\epsilon}$ ’. In: *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*. 1996, pp. 627–636.
- [Hau92] D. Haussler. ‘Decision theoretic generalizations of the PAC model for neural net and other learning applications’. *Information and computation* 100.1 (1992), pp. 78–150.
- [Heg+11] P. Heggernes, P. Van’t Hof, B. M. Jansen, S. Kratsch and Y. Villanger. ‘Parameterized complexity of vertex deletion into perfect graph classes’. In: *International Symposium on Fundamentals of Computation Theory*. Springer. 2011, pp. 240–251.
- [Her+15] D. Hermelin, S. Kratsch, K. Soltys, M. Wahlström and X. Wu. ‘A Completeness Theory for Polynomial (Turing) Kernelization’. *Algorithmica* 71.3 (2015), pp. 702–730.
- [HK03] E. Halperin and R. Krauthgamer. ‘Polylogarithmic inapproximability’. In: *STOC*. 2003, pp. 585–594.
- [HPM04] S. Har-Peled and S. Mazumdar. ‘On coresets for k-means and k-median clustering’. In: *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. ACM. 2004, pp. 291–300.
- [HS86] D. S. Hochbaum and D. B. Shmoys. ‘A unified approach to approximation algorithms for bottleneck problems’. *Journal of the ACM* 33.3 (1986), pp. 533–550.
- [HV20] L. Huang and N. K. Vishnoi. ‘Coresets for Clustering in Euclidean Spaces: Importance Sampling is Nearly Optimal’. *arXiv preprint arXiv:2004.06263* (2020).
- [III+98] H. B. H. III, M. V. Marathe, V. Radhakrishnan, S. S. Ravi, D. J. Rosenkrantz and R. E. Stearns. ‘NC-Approximation Schemes for NP- and PSPACE-Hard Problems for Geometric Graphs’. *J. Algorithms* 26.2 (1998), pp. 238–274.

- [IKW12] R. Impagliazzo, V. Kabanets and A. Wigderson. ‘New Direct-Product Testers and 2-Query PCPs’. *SIAM J. Comput.* 41.6 (2012), pp. 1722–1768.
- [IPZ01] R. Impagliazzo, R. Paturi and F. Zane. ‘Which Problems Have Strongly Exponential Complexity?’ *J. Comput. Syst. Sci.* 63.4 (2001), pp. 512–530.
- [JG79] D. S. Johnson and M. R. Garey. *Computers and intractability: A guide to the theory of NP-completeness*. Vol. 1. WH Freeman San Francisco, 1979.
- [JP18a] B. M. Jansen and A. Pieterse. ‘Polynomial Kernels for Hitting Forbidden Minors under Structural Parameterizations’. In: *26th Annual European Symposium on Algorithms (ESA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.
- [JP18b] B. M. Jansen and M. Pilipczuk. ‘Approximation and kernelization for chordal vertex deletion’. *SIAM Journal on Discrete Mathematics* 32.3 (2018), pp. 2258–2301.
- [Kan+04] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman and A. Y. Wu. ‘A local search approximation algorithm for k-means clustering’. *Computational Geometry* 28.2-3 (2004), pp. 89–112.
- [Kan92] V. Kann. ‘On the Approximability of NP-complete Optimization Problems’. PhD thesis. Royal Institute of Technology, 1992.
- [Kho02] S. Khot. ‘On the power of unique 2-prover 1-round games’. In: *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*. ACM. 2002, pp. 767–775.
- [Kho06] S. Khot. ‘Ruling Out PTAS for Graph Min-Bisection, Dense k-Subgraph, and Bipartite Clique’. *SIAM J. Comput.* 36.4 (2006), pp. 1025–1071.
- [KK18] E. J. Kim and O.-j. Kwon. ‘Erdős-Pósa property of chordless cycles and its applications’. In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2018, pp. 1665–1684.
- [KL20] K.-I. Kawarabayashi and B. Lin. ‘A nearly  $5/3$ -approximation FPT Algorithm for Min-k-Cut’. In: *SODA*. 2020.
- [KLM19] Karthik C. S., B. Laekhanukit and P. Manurangsi. ‘On the parameterized complexity of approximating dominating set’. *J. ACM* 66.5 (2019). Preliminary version in *STOC 2018*, 33:1–33:38.
- [KLP18] I. Katsikarelis, M. Lampis and V. T. Paschos. ‘Structurally Parameterized d-Scattered Set’. In: *Graph-Theoretic Concepts in Computer Science - 44th International Workshop, WG 2018, Cottbus, Germany, June 27-29, 2018, Proceedings*. 2018, pp. 292–305.
- [KLP19a] I. Katsikarelis, M. Lampis and V. T. Paschos. ‘Improved (In-)Approximability Bounds for d-Scattered Set’. In: *Approximation and Online Algorithms - 17th International Workshop, WAOA 2019, Munich, Germany, September 12-13, 2019, Revised Selected Papers*. 2019, pp. 202–216.
- [KLP19b] I. Katsikarelis, M. Lampis and V. T. Paschos. ‘Structural parameters, tight bounds, and approximation for (k, r)-center’. *Discrete Applied Mathematics* 264 (2019), pp. 90–117.
- [KLS15] M. Karpinski, M. Lampis and R. Schmied. ‘New inapproximability bounds for TSP’. *JCSS* 81 (8 2015), pp. 1665–1677.

- [KLS18] R. Krishnaswamy, S. Li and S. Sandeep. ‘Constant approximation for k-median and k-means with outliers via iterative rounding’. In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. ACM. 2018, pp. 646–659.
- [KM19] Karthik C. S. and P. Manurangsi. ‘On Closest Pair in Euclidean Metric: Monochromatic is as Hard as Bichromatic’. In: *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*. 2019, 17:1–17:16.
- [KMR18] R Krithika, D. Majumdar and V. Raman. ‘Revisiting connected vertex cover: FPT algorithms and lossy kernels’. *Theory of Computing Systems* 62.8 (2018), pp. 1690–1714.
- [KR99] S. G. Kolliopoulos and S. Rao. ‘A nearly linear-time approximation scheme for the Euclidean k-median problem’. In: *European Symposium on Algorithms*. Springer. 1999, pp. 378–389.
- [Kra12] S. Kratsch. ‘Polynomial kernelizations for  $\text{MIN } F^+ \Pi_1$  and  $\text{MAX NP}$ ’. *Algorithmica* 63.1-2 (2012), pp. 532–550.
- [KS17] K.-i. Kawarabayashi and A. Sidiropoulos. ‘Polylogarithmic approximation for minimum planarization (almost)’. In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2017, pp. 779–788.
- [KS96] D. R. Karger and C. Stein. ‘A new approach to the minimum cut problem’. *Journal of the ACM (JACM)* 43.4 (1996), pp. 601–640.
- [KSS04] A Kumar, Y Sabharwal and S Sen. ‘A simple linear time  $(1 + \epsilon)$ -approximation algorithm for k-means clustering in any dimensions’. In: *45th Annual IEEE Symposium on Foundations of Computer Science*. IEEE. 2004, pp. 454–462.
- [KSS05] A. Kumar, Y. Sabharwal and S. Sen. ‘Linear time algorithms for clustering problems in any dimensions’. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2005, pp. 1374–1385.
- [KST54] T. Kővári, V. T. Sós and P. Turán. ‘On a problem of K. Zarankiewicz’. *Colloquium Mathematicae* 3.1 (1954), pp. 50–57.
- [KT11] K.-i. Kawarabayashi and M. Thorup. ‘The minimum k-way cut of bounded size is fixed-parameter tractable’. In: *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*. IEEE. 2011, pp. 160–169.
- [KTW99] H. Kellerer, T. Tautenhahn and G. Woeginger. ‘Approximability and nonapproximability results for minimizing total flow time on a single machine’. *SIAM Journal on Computing* 28.4 (1999), pp. 1155–1166.
- [Lai19] W. Lai. ‘The Inapproximability of k-DominatingSet for Parameterized AC 0 Circuits †’. *Algorithms* 12.11 (2019), p. 230.
- [Lam11] M. Lampis. ‘A kernel of order  $2k - c \log k$  for vertex cover’. *Inf. Process. Lett.* 111.23-24 (2011), pp. 1089–1091.
- [Lam14] M. Lampis. ‘Parameterized Approximation Schemes Using Graph Widths’. In: *ICALP*. 2014, pp. 775–786.

- [Lee17a] E. Lee. ‘Improved Hardness for Cut, Interdiction, and Firefighter Problems’. In: *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2017.
- [Lee17b] E. Lee. ‘Partitioning a graph into small pieces with applications to path transversal’. In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics. 2017, pp. 1546–1558.
- [Li16] S. Li. ‘Approximating capacitated  $k$ -median with  $(1 + \epsilon)k$  open facilities’. In: *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2016, pp. 786–796.
- [Lin18] B. Lin. ‘The Parameterized Complexity of the  $k$ -Biclique Problem’. *J. ACM* 65.5 (2018), 34:1–34:23.
- [Lin19] B. Lin. ‘A Simple Gap-Producing Reduction for the Parameterized Set Cover Problem’. In: *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*. 2019, 81:1–81:15.
- [Lok+17] D. Lokshtanov, F. Panolan, M. Ramanujan and S. Saurabh. ‘Lossy Kernelization’. In: *STOC*. 2017, pp. 224–237.
- [Lok+19] D. Lokshtanov, A. E. Mouawad, S. Saurabh and M. Zehavi. ‘Packing Cycles Faster Than Erdos–Posa’. *SIAM Journal on Discrete Mathematics* 33.3 (2019), pp. 1194–1215.
- [Lok+20] D. Lokshtanov, M. S. Ramanujan, S. Saurabh and M. Zehavi. ‘Parameterized Complexity and Approximability of Directed Odd Cycle Transversal’. In: *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*. 2020, pp. 2181–2200.
- [LR99] T. Leighton and S. Rao. ‘Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms’. *Journal of the ACM (JACM)* 46.6 (1999), pp. 787–832.
- [LSS20] D. Lokshtanov, S. Saurabh and V. Surianarayanan. ‘A Parameterized Approximation Scheme for Min  $k$ -Cut’. *arXiv preprint arXiv:2005.00134* (2020).
- [LSW17] E. Lee, M. Schmidt and J. Wright. ‘Improved and simplified inapproximability for  $k$ -means’. *Information Processing Letters* 120 (2017), pp. 40–43.
- [LY93] C. Lund and M. Yannakakis. ‘The approximation of maximum subgraph problems’. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 1993, pp. 40–51.
- [Man] P. Manurangsi. ‘Tight Running Time Lower Bounds for Strong Inapproximability of Maximum  $k$ -Coverage, Unique Set Cover and Related Problems (via  $t$ -Wise Agreement Testing Theorem)’. To appear in SODA 2020.
- [Man17] P. Manurangsi. ‘Almost-polynomial ratio ETH-hardness of approximating densest  $k$ -subgraph’. In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*. 2017, pp. 954–961.

- [Man18a] P. Manurangsi. ‘A Note on Max  $k$ -Vertex Cover: Faster FPT-AS, Smaller Approximate Kernel and Improved Approximation’. In: *2nd Symposium on Simplicity in Algorithms (SOSA 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2018.
- [Man18b] P. Manurangsi. ‘Inapproximability of maximum biclique problems, minimum  $k$ -cut and densest at-least- $k$ -subgraph from the small set expansion hypothesis’. *Algorithms* 11.1 (2018), p. 10.
- [Mar04] D. Marx. ‘Minimum sum multicoloring on the edges of planar graphs and partial  $k$ -trees’. In: *International Workshop on Approximation and Online Algorithms*. Springer. 2004, pp. 9–22.
- [Mar05] D. Marx. ‘Efficient approximation schemes for geometric problems?’ In: *European Symposium on Algorithms*. Springer. 2005, pp. 448–459.
- [Mar06] D. Marx. ‘Parameterized graph separation problems’. *Theoretical Computer Science* 351.3 (2006), pp. 394–406.
- [Mar08] D. Marx. ‘Parameterized complexity and approximation algorithms’. *The Computer journal* 51.1 (2008), pp. 60–78.
- [Mar09] D. Marx. ‘Complexity results for minimum sum edge coloring’. *Discrete Applied Mathematics* 157.5 (2009), pp. 1034–1045.
- [Mar10] D. Marx. ‘Chordal deletion is fixed-parameter tractable’. *Algorithmica* 57.4 (2010), pp. 747–768.
- [Mar13] D. Marx. ‘Completely inapproximable monotone and antimonotone parameterized problems’. *J. Comput. Syst. Sci.* 79.1 (2013), pp. 144–151.
- [Mat00] J. Matoušek. ‘On approximate geometric  $k$ -clustering’. *Discrete & Computational Geometry* 24.1 (2000), pp. 61–84.
- [MM09] A. Magen and M. Moharrami. ‘Robust algorithms for on minor-free graphs based on the Sherali-Adams hierarchy’. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Springer, 2009, pp. 258–271.
- [Mos15] D. Moshkovitz. ‘The Projection Games Conjecture and the NP-Hardness of  $\ln$ -Approximating Set-Cover’. *Theory of Computing* 11 (2015), pp. 221–235.
- [MP14] D. Marx and M. Pilipczuk. ‘Everything you always wanted to know about the parameterized complexity of Subgraph Isomorphism (but were afraid to ask)’. In: *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2014.
- [MR14] D. Marx and I. Razgon. ‘Fixed-parameter tractability of multicut parameterized by the size of the cutset’. *SIAM Journal on Computing* 43.2 (2014), pp. 355–388.
- [MR17] P. Manurangsi and P. Raghavendra. ‘A Birthday Repetition Theorem and Complexity of Approximating Dense CSPs’. In: *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*. 2017, 78:1–78:15.

- [MT18] P. Manurangsi and L. Trevisan. ‘Mildly Exponential Time Approximation Algorithms for Vertex Cover, Balanced Separator and Uniform Sparsest Cut’. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2018, August 20-22, 2018 - Princeton, NJ, USA*. 2018, 20:1–20:17.
- [Ned09] J. Nederlof. ‘Fast Polynomial-Space Algorithms Using Möbius Inversion: Improving on Steiner Tree and Related Problems’. In: *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*. 2009, pp. 713–725.
- [NJ75] G. L. Nemhauser and L. E. T. Jr. ‘Vertex packings: Structural properties and algorithms’. *Math. Program.* 8.1 (1975), pp. 232–248.
- [NZ97] J. Naor and L. Zosin. ‘A 2-approximation algorithm for the directed multiway cut problem’. In: *Proceedings 38th Annual Symposium on Foundations of Computer Science*. IEEE. 1997, pp. 548–553.
- [Pap77] C. H. Papadimitriou. ‘The Euclidean travelling salesman problem is NP-complete’. *Theoretical computer science* 4.3 (1977), pp. 237–244.
- [Pet94] E. Petrank. ‘The hardness of approximation: Gap location’. *Computational Complexity* 4.2 (1994), pp. 133–157.
- [PLW17] M. Pilipczuk, E. J. van Leeuwen and A. Wiese. ‘Approximation and Parameterized Algorithms for Geometric Independent Set with Shrinking’. In: *42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2017.
- [Pra19] K. Pratt. ‘Waring Rank, Parameterized and Exact Algorithms’. In: *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2019, pp. 806–823.
- [PW18] M. Pilipczuk and M. Wahlström. ‘Directed Multicut is  $W[1]$ -hard, Even for Four Terminal Pairs’. *TOCT* 10.3 (2018), 13:1–13:18.
- [PY91] C. H. Papadimitriou and M. Yannakakis. ‘Optimization, Approximation, and Complexity Classes’. *J. Comput. Syst. Sci.* 43.3 (1991), pp. 425–440.
- [Räc08] H. Räcke. ‘Optimal hierarchical decompositions for congestion minimization in networks’. In: *Proceedings of the fortieth annual ACM symposium on Theory of computing*. ACM. 2008, pp. 255–264.
- [Raz98] R. Raz. ‘A Parallel Repetition Theorem’. *SIAM J. Comput.* 27.3 (1998), pp. 763–803.
- [RS10] P. Raghavendra and D. Steurer. ‘Graph expansion and the unique games conjecture’. In: *Proceedings of the forty-second ACM symposium on Theory of computing*. ACM. 2010, pp. 755–764.
- [RS97] R. Raz and S. Safra. ‘A Sub-Constant Error-Probability Low-Degree Test, and a Sub-Constant Error-Probability PCP Characterization of NP’. In: *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*. 1997, pp. 475–484.
- [RT17] J.-F. Raymond and D. M. Thilikos. ‘Recent techniques and results on the Erdős–Pósa property’. *Discrete Applied Mathematics* 231 (2017), pp. 25–43.

- [RW19] A. Rubinfeld and V. V. Williams. ‘SETH vs Approximation’. *SIGACT News* 50.4 (2019), pp. 57–76.
- [Sal03] M. R. Salavatipour. ‘On sum coloring of graphs’. *Discrete Applied Mathematics* 127.3 (2003), pp. 477–488.
- [SF17] P. Skowron and P. Faliszewski. ‘Chamberlin–Courant Rule with Approval Ballots: Approximating the MaxCover Problem with Bounded Frequencies in FPT Time’. *Journal of Artificial Intelligence Research* 60 (2017), pp. 687–716.
- [Sto73] L. Stockmeyer. ‘Planar 3-colorability is NP-complete’. *ACM Sigact News* 5.3 (1973), pp. 19–25.
- [SV13] A. Sharma and J. Vondrák. ‘Multiway cut, pairwise realizable distributions, and descending thresholds’. *arXiv preprint arXiv:1309.2729* (2013).
- [SV95] H. Saran and V. V. Vazirani. ‘Finding  $k$  cuts within twice the optimal’. *SIAM Journal on Computing* 24.1 (1995), pp. 101–108.
- [SW18] C. Sohler and D. P. Woodruff. ‘Strong coresets for  $k$ -median and subspace approximation: Goodbye dimension’. In: *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2018, pp. 802–813.
- [Swa16] C. Swamy. ‘Improved approximation algorithms for matroid and knapsack median problems and applications’. *ACM Transactions on Algorithms (TALG)* 12.4 (2016), p. 49.
- [Tal04] K. Talwar. ‘Bypassing the embedding: algorithms for low dimensional metrics’. In: *STOC*. 2004, pp. 281–290.
- [Tho08] M. Thorup. ‘Minimum  $k$ -way cuts via deterministic greedy tree packing’. In: *Proceedings of the fortieth annual ACM symposium on Theory of computing*. ACM. 2008, pp. 159–166.
- [Vaz01] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.
- [VB+19] W. C. Van Batenburg, T. Huynh, G. Joret and J.-F. Raymond. ‘A tight Erdős-Pósa function for planar minors’. In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, pp. 1485–1500.
- [Wie17] A. Wiese. ‘A  $(1 + \epsilon)$ -approximation for Unsplittable Flow on a Path in fixed-parameter running time’. In: *ICALP*. 2017, 67:1–67:13.
- [Wie18] A. Wiese. ‘Fixed-Parameter approximation schemes for weighted flowtime’. In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018)*. 2018.
- [Wlo19] M. Włodarczyk. ‘Inapproximability within  $W[1]$ : the case of Steiner Orientation’. *CoRR* abs/1907.06529 (2019). arXiv: [1907.06529](https://arxiv.org/abs/1907.06529).
- [WS11] D. P. Williamson and D. B. Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.
- [XZZ19] Y. Xu, Y. Zhang and Y. Zou. ‘A constant parameterized approximation for hard-capacitated  $k$ -means’. *arXiv preprint arXiv:1901.04628* (2019).
- [Zuc06] D. Zuckerman. ‘Linear degree extractors and the inapproximability of max clique and chromatic number’. In: *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. ACM. 2006, pp. 681–690.



[Zuc96] D. Zuckerman. 'Simulating BPP Using a General Weak Random Source'. *Algorithmica* 16.4/5 (1996), pp. 367–391.