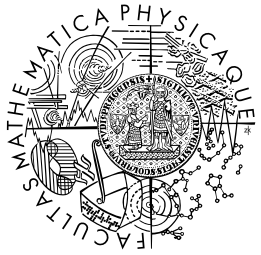

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

Habilitation Thesis

Dr. Andreas Emil Feldmann

Parameterized Approximation Algorithms in Network Design and Clustering

Department of Applied Mathematics (KAM)

Prague 2020

Preface

This habilitation thesis gives an overview of some recent results obtained by the author together with various collaborators in the area of parameterized approximation algorithms. This research area in the intersection of fixed-parameter tractability and approximation algorithms has gained growing attention in recent years. While a large variety of algorithmic topics have seen advancements using the paradigm of parameterized approximations (for an overview see the recent survey in [9]), the author's contributions are mostly concentrated on the topics of network design and clustering. Accordingly, this thesis presents the author's results on these two topics in separate sections. Each of these two sections gives a brief overview of the obtained results, after which the used techniques are presented in more detail.

The results appear in several papers listed below, which were published between 2015 and 2020. The papers are also attached to the appendix of this thesis. Some of the attached papers are extended abstracts of conference proceedings, and thus do not contain the full details due to strict page limitations. Therefore a link to a full version of each paper is provided in the list below.

Please note that several passages of this thesis are taken verbatim (with some modifications) from the author's publications. In particular, Section 1 uses parts of [9], Sections 2.1 and 2.2 contain excerpts from [3; 9], Section 2.3 includes some of [10], Section 3.1 contains parts of [5; 8; 9; 11], Section 3.2 lends from [4; 12], and Section 3.3 uses content from [7; 11].

[1] R. Chitnis and A. E. Feldmann. 'FPT Inapproximability of Directed Cut and Connectivity Problems'. In: *Proceedings of the 14th International Symposium on Parameterized and Exact Computation (IPEC)*. Vol. 148. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 8:1–8:20. DOI: 10.4230/LIPIcs.IPEC.2019.8. URL: https://arxiv.org/abs/1910.01934.

[2] R. Chitnis, A. E. Feldmann, M. T. Hajiaghayi and D. Marx. 'Tight Bounds for Planar Strongly Connected Steiner Subgraph with Fixed Number of Terminals (and Extensions)'. *SIAM Journal on Computing* 49.2 (2020), pp. 318–364. DOI: 10.1137/18M122371X. URL: https://arxiv.org/abs/1911.13161.

[3] 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. DOI: 10.4230/LIPIcs.ESA.2018.20. URL: https://arxiv.org/abs/1707.06499.

[4] V. Cohen-Addad, A. E. Feldmann and D. Saulpic. 'Near-Linear Time Approximations Schemes for Clustering in Doubling Metrics'. In: *Proceedings of the 60th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 2019, pp. 540–559. DOI: 10.1109/FOCS.2019.00041. URL: https://arxiv.org/abs/1812.08664.

[5] Y. Disser, A. E. Feldmann, M. Klimm and J. Könemann. 'Travelling on Graphs with Small Highway Dimension'. In: *Proceedings of Graph-Theoretic Concepts in Computer Science - 45th International Workshop (WG)*. Vol. 11789. Springer, 2019, pp. 175–189. DOI: 10.1007/978-3-030-30786-8_14. URL: https://arxiv.org/abs/1902.07040.

[6] 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. DOI: 10.4230/LIPIcs.STACS.2018.26. URL: https://arxiv.org/abs/1710.00668.

[7] 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, 2015, pp. 588–600. DOI: 10.1007/978-3-662-47666-6_47. URL: https://arxiv.org/abs/1605.02530.

[8] A. E. Feldmann, W. S. Fung, J. Könemann and I. Post. 'A $(1+\varepsilon)$-Embedding of Low Highway Dimension Graphs into Bounded Treewidth Graphs'. *SIAM Journal on Computing* 47.4 (2018), pp. 1667–1704. DOI: 10.1137/16M1067196. URL: https://arxiv.org/abs/1502.04588.

[9] A. E. Feldmann, C. S. Karthik, E. Lee and P. Manurangsi. 'A Survey on Approximation in Parameterized Complexity: Hardness and Algorithms'. *Algorithms* 13.6 (June 2020), p. 146. ISSN: 1999-4893. DOI: 10.3390/a13060146. URL: https://arxiv.org/abs/2006.04411.

[10] A. E. Feldmann and D. Marx. 'The Complexity Landscape of Fixed-Parameter Directed Steiner Network Problems'. In: *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*. 2016, 27:1–27:14. DOI: 10.4230/LIPIcs.ICALP.2016.27. URL: https://arxiv.org/abs/1707.06808.

[11] A. E. Feldmann and D. Marx. 'The Parameterized Hardness of the k-Center Problem in Transportation Networks'. *Algorithmica* 82.7 (2020), pp. 1989–2005. DOI: 10.1007/s00453-020-00683-w. URL: https://arxiv.org/abs/1802.08563.

[12] A. E. Feldmann and D. Saulpic. 'Polynomial Time Approximation Schemes for Clustering in Low Highway Dimension Graphs'. In: *Proceedings of the 28th Annual European Symposium on Algorithms (ESA)*. Vol. 173. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 46:1–46:22. DOI: 10.4230/LIPIcs.ESA.2020.46. URL: https://arxiv.org/abs/2006.12897.

Contents

Parameterized Approximation Algorithms in Network Design and Clustering

1 Parameterized approximation

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 *α-approximation*, i.e., a solution that is at most a factor α 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 α-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 $f : \mathbb{N} \to \mathbb{N}$ 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*. Again, such an algorithm may be randomized.

Approximation and FPT algorithms have been studied extensively for the past few decades, and this has lead to a rich literature on algorithmic techniques and deep links to other research

fields within mathematics. 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 small approximation factors, or can be shown to be solvable optimally with only a limited 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 mathematical 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 α-approximation algorithm*, which computes an α-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]), and we refer the readers to a survey of Marx [Mar08] for discussions on earlier results in the area, and our survey in [9] for an overview of more recent developments.

The aim of this thesis is to present the contributions of the author to the field of parameterized approximation algorithms. The main focus of the author's work has been on problems arising in network design and clustering. These are well-studied areas for both parameterized and approximation algorithms and therefore constitute natural starting points to develop a theory of parameterized approximation algorithms. In Section 2 we present the results in network design, while Section 3 gives an overview of those for clustering. Each section begins with a brief overview of the obtained results by the author and several collaborators, after which a more detailed account of the used techniques to obtain these results is given. Before this however, in Section 1.1 we give a more formal introduction of several concepts used in the field.

1.1 Preliminaries

In this section, we review several notions relevant to the study of parameterized approximations, and how they relate to previously studied concepts in the more classic fields of parameterized and approximation algorithms. We will not review common graph theoretic notions such as planarity or treewidth in this introduction, and instead refer to the literature [Cyg+15; Die12] and also later chapters in the appendix of this thesis, which contain some such definitions.

Parameterized approximation algorithms. As already defined above, an *FPT algorithm* computes the optimum solution in $f(k)n^{O(1)}$ time for some parameter k and computable function $f : \mathbb{N} \to \mathbb{N}$ on inputs of size n. An algorithm that computes the optimum solution in $f(k)n^{g(k)}$ time for some parameter k and computable functions $f, g : \mathbb{N} \to \mathbb{N}$, is called a *slice-wise polynomial (XP) algorithm*. If the parameter is the approximation factor, i.e., the algorithm computes a $(1+\varepsilon)$-approximation in $f(\varepsilon)n^{g(\varepsilon)}$ time, then it is called a *polynomial-time approximation scheme (PTAS)*. The latter type of algorithm has been studied *avant la lettre* for quite some time, where it is assumed that ε is a constant and thus the runtime is polynomial. Also the corresponding FPT algorithm has been studied before, which computes a $(1+\varepsilon)$-approximation in $f(\varepsilon)n^{O(1)}$ time, and is referred to as an *efficient polynomial-time approximation scheme (EPTAS)*. While it may be unusual to view these algorithms from the perspective of parametrizations, we will specifically do so in this thesis in order to obtain a more nuanced view of the complexity of the studied problems (especially in Section 3.2).

As also mentioned above, a *parameterized α-approximation algorithm* computes an α-approximation in $f(k)n^{O(1)}$ time for some parameter k on inputs of size n. If α 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 ε is constant, and a PAS is thus the corresponding notion to a PTAS. If we forbid this and consider ε 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)*, which is the corresponding notion to an EPTAS.

Kernelization. A 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 [Cyg+15]. 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 to compress the instance prior to solving the kernel. This also provides an additional dimension to the parameterized complexity landscape, and kernelization has therefore been developed into a research area in its own right.

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. The notion we will be concerned with here was introduced by Lokshtanov et al. [Lok+17]. They define an *α-approximate kernelization algorithm*, which computes a kernel I' such that any β-approximation for 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 *α-approximate kernel*. Analogous to exact kernels, any problem has a parameterized α-approximation algorithm if and only if it admits an α-approximate kernel for the same parameter [Lok+17], which however might be of exponential size in the parameter. As before, studying the existence of polynomial-sized approximate kernels adds an additional dimension to the complexity landscape of parameterized approximation algorithms, and approximate kernels are hence interesting to study in their own right.

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

Kernelization (and FPT) algorithms often come with a set of *reduction rules*, which roughly speaking constitute steps to simplify the input and are applied repeatedly until some core instance (typically a kernel) is left. Formally, a reduction rule is a polynomial time algorithm, which takes an instance I and a parameter k as input, and outputs a new instance I' and parameter k' (but in contrast to a kernelization algorithm, the size of the new instance is not necessarily bounded after only one application of a reduction rule). Furthermore, in the context of approximate kernels, a reduction rule is said to be *strictly α-safe* if there exists a polynomial

time algorithm to convert a β-approximation to I' into a $\max\{\alpha, \beta\}$-approximation to I.[1] A set of reduction rules is then applied repeatedly until some desired property of the resulting instance is met (for kernelizations this property typically is that the size of the final instance is bounded as a function of the input parameter k).

Complexity-theoretic assumptions. We assume that the reader is familiar with common complexity classes used to prove algorithmic lower bounds for parameterizations, approximations, and kernelizations, such as P, NP, W[t], APX, and coNP/poly. For sake of brevity we will not define these classes here and instead refer to the literature [Cyg+15; FG06; WS11].

The *Exponential Time Hypothesis (ETH)* is often used to obtain concrete runtime lower bounds for parameterized problems. It assumes that 3SAT cannot be solved in $2^{o(n)}$ time, where n is the number of variables. A less known assumption is the *Gap Exponential Time Hypothesis (Gap-ETH)*, which is stronger than ETH and useful to obtain lower bounds for parameterized approximations. It assumes that there exists some constant $\delta > 0$ such that there is no $2^{o(n)}$ time algorithm to decide whether all or at most a $(1 - \delta)$-fraction of the clauses of a given 3SAT formula are satisfiable. Here the assumed algorithms may be deterministic or randomized (the latter being the stronger assumption).

2 Network design

In network design the task is to connect some set of vertices in an edge-weighted graph in the cheapest possible way. To give an example, a prominent problem of this type is the Steiner Tree problem. Here 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 graph. This fundamental problem and its variants have been widely studied in the past, both on undirected and directed input graphs.

In Section 2.1 we focus on undirected input graphs. Our main results here are a PAS and a PSAKS for Steiner Tree parameterized by the number of non-terminals (*Steiner vertices*) contained in the optimum solution. In Section 2.2 we turn to directed graphs, where we first discuss the Directed Steiner Tree problem for the same parameter. In summary, we show that a PAS only exists in the unweighted case, but a PSAKS does not exist even then. Next we consider the more standard parameter given by the number of terminals, for which a different directed variant of Steiner Tree called Strongly Connected Steiner Subgraph is known to have a parameterized 2-approximation. We give a lower bound showing that this is best possible. For the more general Directed Steiner Network problem we prove that a PAS and a PSAKS exist on planar directed graphs that are also bidirected, which means that for every edge the reverse edge exists as well and has the same weight. We then present several hardness results showing that our PAS and PSAKS for this problem on planar bidirected graphs are in a sense best possible.

Finally, in Section 2.3 we present a dichotomy result on computing exact solutions for Directed Steiner Network parameterized by the number of terminals. We summarize several interesting consequences of the algorithm due to this result, including its application to derive the above-mentioned PAS for planar bidirected graphs.

[1] The strictness refers to the fact that the approximation factor is bounded by $\max\{\alpha, \beta\}$ instead of $\alpha\beta$. This is needed in order to apply reduction rules repeatedly without losing the guaranteed approximation factor. See [Lok+17] for more details.

2.1 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 δ). The unweighted Steiner Tree problem also admits a $2^k n^{O(1)}$ time algorithm [Ned09] using a different technique based on subset convolution. On the other hand, no exact polynomial-sized kernel exists [DLS14] for the Steiner Tree problem, unless NP$\subseteq$coNP/poly. Interestingly though, a PSAKS can be obtained [Lok+17].

This approximate 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, and on which several of our results are based as well. On a high level, 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 $C_1, \ldots, C_\ell$ 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 $T_1, \ldots, T_\ell$, 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. 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. The resulting graph is thus a PSAKS parameterized by the number k of terminals.

A natural alternative parameter 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, we were able to show that both an EPAS and a PSAKS are obtainable when parametrizing by the number of Steiner vertices p in the optimum.

To obtain both of these results, in [6] we 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 add a large star to the

solution, such that the star has terminal leaves and small cost in the current graph. In case (1) we would like to add 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 and contract it, 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 p and the desired approximation ratio $1+\varepsilon$. Once the number of terminals is bounded by a function of p and ε, the Dreyfus and Wagner [DW71] algorithm can be applied on the remaining instance, or a kernel can be computed using the PSAKS of Lokshtanov et al. [Lok+17]. We prove that our reduction rule does not distort solutions by much as long as the threshold is large enough, which results in the following theorem.

Theorem 2.1 ([6]). *For the* Steiner Tree *problem a* $(1+\varepsilon)$*-approximation can be computed in* $2^{O(p^2/\varepsilon^4)}n^{O(1)}$ *time for any* $\varepsilon > 0$*, where* p *is the number of non-terminals in the optimum solution. Moreover, a* $(1+\varepsilon)$*-approximate kernel of size* $(p/\varepsilon)^{2^{O(1/\varepsilon)}}$ *can be computed in polynomial time.*

A natural question is whether this theorem is generalizable to other variants of Steiner Tree in undirected graphs, for instance 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. Parameterized by the number of terminals k it is not hard to show that Steiner Forest is FPT, since we may guess a partition of the terminals such that each set of the partition is contained in the same connected component of the optimum Steiner Forest solution. Since each connected component forms a tree, an FPT algorithm for Steiner Tree can then be used to compute a solution for each terminal set separately, which leads to a runtime of $k^{O(k)}n^{O(1)}$. Also a PSAKS can be obtained for this parameter, using the same techniques as in [Lok+17] for Steiner Tree (cf. [6]).

If however the parameter is the number p of Steiner vertices in the optimum solution, then neither a PAS nor a PSAKS exists unless P=NP. This can be easily seen, since any Steiner vertex v of a Steiner Forest instance can be promoted to a trivial terminal pair that both equal v. Now any solution to the new instance corresponds to a solution in the original instance of the same weight, and vice versa. As the new instance contains no Steiner vertices, a PAS or PSAKS for parameter p would imply a PTAS for Steiner Forest. However, the problem is APX-hard [CC08], and thus a PTAS would imply P=NP.

Nonetheless, we showed in [6] that using the same techniques as for Steiner Tree, it is possible to generalize Theorem 2.1 to the Steiner Forest problem, if the parameter p is combined with the number c of connected components of the optimum solution. This yields a PAS with runtime $2^{O((p+c)^2/\varepsilon^4)}n^{O(1)}$ and a PSAKS of size $((p+c)/\varepsilon)^{2^{O(1/\varepsilon)}}$.

It is also natural to ask whether Theorem 2.1 is generalizable to variants of Steiner Tree in directed graphs, which we turn to next.

2.2 Directed Graphs

The Directed Steiner Tree problem takes as input a terminal set with a special terminal called the *root* in a directed edge-weighted graph. The task is to compute a directed tree of minimum weight that contains a path from each terminal to the root. For the parameterization by the number k of terminals, the Directed Steiner Tree problem is FPT, using the same algorithms as for the undirected version [DW71; Fuc+07; Ned09]. In contrast to the undirected case however, this problem is much harder to approximate. It was shown [HK03] that no $O(\log^{2-\varepsilon} k)$-approximation can be computed in polynomial time, unless NP-hard problems can be solved in expected quasi-polynomial time. Moreover, for the parameterization by the number p

of Steiner vertices in the optimum solution we proved in [6] that no reasonable approximation can be computed in FPT time. This can be shown using a simple reduction from the Dominating Set problem via a hardness result in [CL16], resulting in the following formal statement.

Theorem 2.2 ([6]). *The* Directed Steiner Tree *problem has no $g(p)$-approximation algorithm with runtime $f(p)n^{O(1)}$, for any computable functions f and g, where p is the number of Steiner vertices in the optimum solution, unless* W[1]=FPT.

A notable special case is the *unweighted* Directed Steiner Tree problem. Here we showed [6] that a PAS is again obtainable using the number p of Steiner vertices in the optimum as a parameter. Similar as for the undirected case, the algorithm uses a reduction rule that contracts components containing many terminals (depending on p and ε), and then uses an FPT algorithm parameterized by the number of terminal to solve the remaining instance. One caveat however is that contractions in directed graphs are tricky, as they may introduce new paths that were non-existent before. For Directed Steiner Tree this issue can be circumvented by making sure that the contracted component always contains the root. For this however, a path from the root to the contracted terminals needs to be included in the contracted component, which may contain many Steiner vertices. However in the unweighted case, the number of vertices of a path directly translates to the weight of the path. This effectively implies that the reduction rule does not distort solutions by much, and thus we obtain a PAS, as stated formally below.

A natural question then becomes whether, as in the undirected case, a PSAKS exists for the unweighted Directed Steiner Tree problem. Recall that the PSAKS for Steiner Tree relies on the result by Borchers and Du [BD97], which decomposes an undirected solution into full-components with a small number of terminals each, such that the full-components overlap very little. It is not hard to see however, that this cannot work in directed graphs. More generally, we prove [6] that in contrast to the undirected case, no polynomial-sized $(2-\varepsilon)$-approximate kernelization exists for unweighted Directed Steiner Tree, unless NP$\subseteq$coNP/poly. It remains an intriguing question whether a polynomial-sized 2-approximate kernel exists.

Theorem 2.3 ([6]). *For the* unweighted Directed Steiner Tree *problem a $(1+\varepsilon)$-approximation can be computed in $2^{p^2/\varepsilon}n^{O(1)}$ time for any $\varepsilon > 0$, where p is the number of non-terminals in the optimum solution. However, no polynomial-sized $(2-\varepsilon)$-approximate kernelization exists, unless* NP$\subseteq$coNP/poly.

We now turn back to the well-studied parameterization by the number k of terminals, and consider other directed variants of Steiner Tree. One example is the Strongly Connected Steiner Subgraph problem, where a terminal set needs to be strongly connected in the cheapest possible way. In contrast to Directed Steiner Tree this problem is W[1]-hard parameterized by the number of terminals [GNS11], and again no $O(\log^{2-\varepsilon} n)$-approximation can be computed in polynomial time [HK03], unless NP $\subseteq$ ZTIME($n^{\text{polylog}(n)}$). However, a 2-approximation can be computed in FPT time [CHK13] using the parameter k.

Interestingly, in [3] we showed that no improvement over this 2-approximation is possible when parameterizing by k. To obtain this hardness result we modified the reduction of Guo et al. [GNS11], who showed W[1]-hardness of the problem. In particular, this reduction was from the W[1]-hard Clique problem. As a starting point we instead use the approximation variant of Clique, namely the Densest k-Subgraph problem, and a recent inapproximability result for the latter [DM18]. Additionally, we introduce appropriate edge weights in the reduction of Guo et al. [GNS11]. Together with the positive result of Chitnis et al. [CHK13] we obtain the following theorem, which to date is the only known tight parameterized approximation result for a problem that can be approximated better in FPT time than in polynomial time, but where the parameterized algorithm is not an approximation scheme.

Theorem 2.4 ([3; 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[2] problem, 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 we showed in [3] that both a PAS and a PSAKS exist for the special case when the input graph is planar[3] and *bidirected*, i.e., for every directed edge uv the reverse edge vu exists and has the same cost.

To obtain these two algorithms, in [3] we generalize the theorem of Borchers and Du [BD97] for Steiner Tree to the Directed Steiner Network problem on planar bidirected graphs. That is, we show that a planar optimum 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. Similar to how Borchers and Du [BD97] exploit the tree structure of a solution to Steiner Tree, in our case the planarity is exploited to make sure that the covering graphs contain few terminals while at the same time do not cost much more than the optimum. However, to make sure that the union of these covering graphs constitutes a feasible solution, we may need to add 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.

To formally state our contribution, we encode the demands of a Directed Steiner Network instance using a *pattern graph* H: the vertex set of H is the terminal set of the input graph G, and H contains the directed edge st if and only if (s,t) is a demand. Hence the Directed Steiner Network problem asks for a minimum cost network $N \subseteq G$ having an $s \to t$ path for each edge st of H. In the following, $\mathrm{cost}(N)$ denotes the cost of a graph (solution) N, i.e., the sum of its edge weights.

Theorem 2.5 ([3]). *Let* G *be a bidirected graph, and* H *a pattern graph on the terminal set* R *of* G*. Let* $N \subseteq G$ *be the cheapest planar solution to pattern* H*. For any* $\varepsilon > 0$*, there exists a set of patterns* $\mathcal{H}$ *such that*

1. $V(H') \subseteq R$ *with* $|V(H')| \leq 2^{1+\lceil 1/\varepsilon \rceil}$ *for each* $H' \in \mathcal{H}$*,*
2. *given any feasible solutions* $N_{H'} \subseteq G$ *for all* $H' \in \mathcal{H}$*, the union* $\bigcup_{H' \in \mathcal{H}} N_{H'}$ *of the these solutions forms a feasible solution to* H*, and*
3. *there exist feasible planar solutions* $N^*_{H'} \subseteq G$ *for all* $H' \in \mathcal{H}$ *such that* $\sum_{H' \in \mathcal{H}} \mathrm{cost}(N^*_{H'}) \leq (1+\varepsilon) \cdot \mathrm{cost}(N)$*.*

Analogous to Steiner Tree, we now compute solutions for every possible list of ordered pairs (i.e., pattern graphs) of at most $2^{1+\lceil 1/\varepsilon \rceil}$ terminals. In contrast to Steiner Tree however, it is unlikely that Directed Steiner Network on planar bidirected graphs is FPT parameterized by the number of terminals (see below). Instead we use an XP algorithm with runtime $2^{O(k^{3/2}\log k)}n^{O(\sqrt{k})}$, which can be obtained by exploiting our insights on computing optimum solutions to the Directed Steiner Network problem presented in [10] (cf. Section 2.3).

[2] sometimes also called Directed Steiner Forest; note however that the optimum is not necessarily a forest.
[3] a directed graph is planar if its underlying undirected graph is.

Since each considered pattern graph has at most $2^{1+\lceil 1/\varepsilon \rceil}$ terminals, the time needed to compute solutions for all of them can be bounded by $n^{2^{O(1/\varepsilon)}}$, which is polynomial if ε is constant.

To obtain a PSAKS, 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, a dynamic program can be used to search for a solution set that is a $(1+\varepsilon)$-approximation among the precomputed solutions. This step takes $2^{O(k^2)}k^{2^{O(1/\varepsilon)}}n^{O(1)}$ time, and so the overall runtime of the algorithm can be upper bounded by $2^{O(k^2)}n^{2^{O(1/\varepsilon)}}$. This results in the following theorem.

Theorem 2.6 ([3]). *For the* Directed Steiner Network *problem on planar bidirected graphs a $(1+\varepsilon)$-approximation can be computed in $2^{O(k^2)}n^{2^{O(1/\varepsilon)}}$ 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.*

Given that the planar bidirected instances considered for Theorem 2.6 are rather restricted, a natural question becomes (a) whether the runtime can be improved (possibly even to polynomial time, and maybe even an optimum solution can be computed in FPT time), and (b) whether similar algorithms exist for any of the two natural generalizations, i.e., either planar graphs or bidirected graphs. We give partial answers to this question. In particular, the algorithms of Theorem 2.6 are in fact slightly more general than stated: they work even for non-planar bidirected graphs if we want to approximate the optimum planar solution. That is, even if the input graph is bidirected but otherwise unrestricted, the solutions are at most a $(1+\varepsilon)$-factor more expensive than the cheapest among all planar solutions. Note though that the computed solutions may be non-planar (and could thus even turn out to be cheaper than the optimum planar solution). Considering this more general setting might at first seem rather exotic. However, it turns out that several algorithms found in the literature for Directed Steiner Network on special graph classes have this quality. That is, even if they are stated as algorithms for some input graph class $\mathcal{K}$, they can be used to compute solutions in otherwise unrestricted graphs, while the solution quality is compared to the optimum solution from $\mathcal{K}$. We give some more examples of this in Section 2.3.

We give negative answers to the above questions in the more general setting just described. This can be interpreted as saying that if these questions can be answered positively in the original setting of planar bidirected graphs, then new algorithmic techniques need to be developed, which are different from those typically found in the literature to date. For the first question on improving the runtime, first off in [3] we prove that it is APX-hard to compute the planar optimum in a bidirected graph. Then, note that the algorithm of Theorem 2.6 is not an EPAS, i.e., the degree of the polynomial factor depends on the approximation factor ε. As we prove in [3], unless FPT=W[1], this dependence is necessary for bidirected inputs where we want to approximate the planar optimum. This also rules out an FPT algorithm to compute the optimum planar solution in bidirected graphs (and hence we used an XP algorithm to obtain Theorem 2.6). For the second question on generalizing the algorithms, we prove in [3] that under Gap-ETH no PAS exists for bidirected input graphs without any further restriction (i.e., when approximating the overall optimum). In [1] we also show that the other obvious generalization, where the input consists of any directed graph and we approximate the planar optimum, has no $(2-\varepsilon)$-approximation for any $\varepsilon > 0$, under Gap-ETH. As summarized in the following, these results contrast Theorem 2.6.

Theorem 2.7 ([1; 3]). *For the* Directed Steiner Network *problem the following hardness results hold for any computable function f:*

1. *computing the planar optimum in bidirected input graphs is* APX*-hard,*

2. *there is no* $f(k,\varepsilon)n^{O(1)}$ *time algorithm that computes a* $(1+\varepsilon)$*-approximation of the planar optimum in bidirected input graphs where* $\varepsilon > 0$ *is part of the input, unless* FPT=W[1],

3. *there exists a constant* $\alpha > 1$ *such that there is no* $f(k)n^{O(1)}$ *time algorithm that computes an* α*-approximation to the (overall) optimum in bidirected input graphs, under* Gap-ETH,

4. *there is no* $f(k)n^{O(1)}$ *time algorithm that computes a* $(2-\varepsilon)$*-approximation of the planar optimum in general input graphs, for any* $\varepsilon > 0$*, under* Gap-ETH.

2.3 Computing exact solutions in directed graphs

In this section we digress slightly from our main topic of parameterized approximation algorithms, and present some results on computing optimum solutions for special cases of the Directed Steiner Network problem. These results are related to the parameterized approximation algorithms presented in Section 2.2, and are partially also used as subroutines for the latter.

In [10] we analysed the dependence of the parameterized complexity of Directed Steiner Network on the structure of the pattern graphs, as introduced for Theorem 2.5. We proved that the problem is FPT whenever the pattern graphs are restricted to "almost-caterpillars" and W[1]-hard otherwise, i.e., we show a dichotomy on the complexity w.r.t. the patterns. Formally these almost-caterpillars are defined as follows, where an *out-star* or *in-star* is a directed star for which all edges point away from the center vertex or towards the center vertex, respectively.

Definition 2.8. A λ_0*-caterpillar* graph is constructed as follows. Take a directed path $(v_1,\dots,v_{\lambda_0})$ from v_1 to v_{λ_0}, and let $W_1,\dots,W_{\lambda_0}$ be pairwise disjoint vertex sets such that $v_i \in W_i$ for each $i \in \{1,\dots,\lambda_0\}$. Now add edges such that either every W_i forms an out-star with root v_i, or every W_i forms an in-star with root v_i. A 0-caterpillar is the empty graph. The class $\mathcal{C}_{\lambda,\delta}$ contains all directed graphs H such that there is a set of edges $F \subseteq E(H)$ of size at most δ for which the remaining edges $E(H) \setminus F$ span a λ_0-caterpillar for some $\lambda_0 \leq \lambda$. We say that two pattern graphs are *transitively equivalent* if their transitive closures are isomorphic, and denote by $\mathcal{C}^*_{\lambda,\delta}$ the class of patterns that are transitively equivalent to some pattern of $\mathcal{C}_{\lambda,\delta}$.

For example, for the Directed Steiner Tree problem all pattern graphs are in-stars and thus belong to the class $\mathcal{C}^*_{1,0}$. For the Strongly Connected Steiner Subgraph problem the patterns can be seen as complete graphs. In this case it turns out that no constants λ and δ exist for which the patterns to this problem would belong to some class $\mathcal{C}^*_{\lambda,\delta}$. Therefore the following theorem in particular recovers the known [DW71; GNS11] complexity results for these two problems. It is much more general though, as it gives a complete dichotomy of the tractability of Directed Steiner Network depending on the structure of the pattern graphs.

Theorem 2.9 ([10])**.** *Let* $\mathcal{H}$ *be a recursively enumerable class of patterns and let* k *be the number of terminals of a given instance.*

1. *If there are constants* λ *and* δ *such that* $\mathcal{H} \subseteq \mathcal{C}^*_{\lambda,\delta}$*, then* Directed Steiner Network *restricted to patterns from* $\mathcal{H}$ *is FPT for parameter* k*, and can be solved in* $2^{O(k+\tau\omega\log\omega)}n^{O(\omega)}$ *time, where* $\omega = (1+\lambda)(\lambda+\delta)$ *and* τ *is the vertex cover number of the given input pattern* $H \in \mathcal{H}$*.*

2. *Otherwise, if there are no such constants* λ *and* δ*, then the problem is* W[1]*-hard for parameter* k*.*

To obtain the algorithm of the first part of this theorem, in [10] we show that any optimal solution to a pattern in $\mathcal{C}^*_{\lambda,\delta}$ has treewidth[4] at most $7(1+\lambda)(\lambda+\delta)$. The algorithm is then

[4] a directed graph has treewidth ω if its underlying undirected graph does.

implied by the following useful theorem, which we obtained in [10] via a dynamic programming approach.

Theorem 2.10 ([10]). *Let an instance of* Directed Steiner Network *be given by a graph with n vertices, and a pattern H on k terminals with vertex cover number τ. The cheapest among all solution to H with treewidth ω can be computed in $2^{O(k+\tau\omega\log\omega)}n^{O(\omega)}$ time.*

This theorem has several consequences, which we now elaborate on. An arbitrary pattern graph H with d edges belongs to the class $\mathcal{C}^*_{0,d}$. Consequently, the optimum solution has treewidth at most $7d$ by our results in [10], and the algorithm of Theorem 2.10 can be used to compute the optimum to H in $2^{O(kd\log d)}n^{O(d)}$ time, i.e., we obtain an XP algorithm parameterized by the number of terminals k, since $d < k^2$. It was actually first shown by Feldman and Ruhl [FR06] that Directed Steiner Network is in XP, and thus our results in [10] recover this fact. Feldman and Ruhl [FR06] however obtain a faster $n^{O(d)}$ time XP algorithm. Measured in the stronger parameter k, this is an $n^{O(k^2)}$ time algorithm. As shown by Eiben et al. [Eib+19], this is essentially best possible as no $f(k)n^{o(k^2/\log k)}$ time algorithm exists for this problem for any computable function f, under ETH. However, as summarized below, in special cases it is possible to beat this lower bound, using Theorem 2.10.

In [3] we show that the treewidth of an optimum planar solution in a bidirected graph is $O(\sqrt{k})$, which then implies a faster XP algorithm with runtime $2^{O(k^{3/2}\log k)}n^{O(\sqrt{k})}$, as also mentioned in Section 2.2. We also prove [3] that there is no $f(k)n^{o(\sqrt{k})}$ time algorithm to compute the planar optimum in bidirected graphs, under ETH. For directed planar input graphs, we show in [2] that under ETH no $f(k)n^{o(k)}$ time algorithm can compute the optimum Directed Steiner Network solution (note that this is a stronger hardness result as the previous one, since here the input graph is planar). Eiben et al. [Eib+19] show that an optimum solution of genus g has treewidth $2^{O(g)}k$ and thus Theorem 2.10 implies an XP algorithm with runtime $2^{O(k^2\log k)}n^{O(k)}$ for solutions of constant genus, matching the previous runtime lower bound. However, for the special case of the Strongly Connected Steiner Subgraph problem, we prove in [2] that the optimum planar solution again has treewidth $O(\sqrt{k})$, leading to an XP algorithm with runtime $2^{O(k)}n^{O(\sqrt{k})}$. We also obtain [2] a runtime lower bound of $f(k)n^{o(\sqrt{k})}$ for this problem on planar graphs (which again is a stronger hardness result than previously). Note that the two algorithms for planar solutions, the one for bounded genus solutions, but also the algorithm of Theorem 2.10, have the quality mentioned in Section 2.2 that they compute optimum planar, bounded-genus, or bounded-treewidth solutions in graphs that have unbounded genus and treewidth.

Another interesting application of Theorem 2.10 is the Strongly Connected Steiner Subgraph problem on bidirected input graphs. While this problem remains NP-hard, in [3] we show that it is FPT parameterized by k, which is in contrast to general input graphs where the problem is W[1]-hard [GNS11] (as also implied by Theorem 2.9). To show this result, it is not enough to bound the treewidth of a solution and then apply Theorem 2.10 directly, as above for planar optima. In fact, in [3] we give examples in which the optimum solution to Strongly Connected Steiner Subgraph on bidirected graphs has treewidth $\Theta(k)$. Instead we provide a decomposition of optimum solutions, similar to the theorem of Borchers and Du [BD97] for Steiner Tree or our generalization in Theorem 2.5 for Directed Steiner Network. While the latter two results find sub-graphs that cover a solution (i.e., the sub-graphs may not be edge-disjoint), for Strongly Connected Steiner Subgraph on bidirected graphs we obtain a stronger result, in the sense that a solution can be decomposed into non-overlapping (i.e., edge-disjoint) sub-graphs. Each of these sub-graphs is a solution to some pattern graph H on the terminals of the input instance, and is a *poly-tree*, i.e., a directed graph whose underlying

undirected graph is a tree. However, in contrast to Theorem 2.5 the number of terminals in each poly-tree is not bounded by any constant.

The algorithm now proceeds similar to the PAS of Theorem 2.6: it first computes all optimum poly-tree solutions for every possible pattern graph, and then finds the best solution strongly connecting the terminal set by combining the poly-trees using a dynamic program. Since a poly-tree has treewidth 1, Theorem 2.10 can be used to compute an optimum solution to any pattern graph in $2^{O(k)}n^{O(1)}$ time. Furthermore, as the poly-trees of the decomposition are non-overlapping, the algorithm computes an optimum solution to Strongly Connected Steiner Subgraph on bidirected graphs. Note that in contrast to the algorithm of Theorem 2.6 however, computing the solutions to the patterns takes FPT time, as there is no constant bound on the number of terminals in each poly-tree. This also means that no polynomial-sized kernel is implied by this decomposition.

Theorem 2.11 ([3]). *The* Strongly Connected Steiner Subgraph *problem on bidirected graphs is* NP*-hard, but can be solved in* $2^{k^2+O(k)}n^{O(1)}$ *time where* k *is the number of terminals.*

3 Clustering

For clustering problems the task is to group the vertices of a metric (V, dist) into sets such that vertices that are close are in the same group, where the closeness is given by some measure depending on the distance function $\mathrm{dist} : V \times V \to \mathbb{R}^+$. Some prominent examples include the k-Median, k-Center, and Facility Location problems. For each of these, we need to select a subset $F \subseteq V$ of the vertices, called *centers* or *facilities*, which act as representatives for the groups. For k-Median and k-Center the set F can only contain k vertices and we need to minimize $\sum_{v\in V} \mathrm{dist}(v, F)$ and $\max_{v\in V} \mathrm{dist}(v, F)$, respectively, where $\mathrm{dist}(v, F) = \min_{f\in F} \mathrm{dist}(v, f)$. The Facility Location problem essentially is the Lagrangian relaxation of k-Median, i.e., there is no bound on the number of facilities but instead each vertex $v \in V$ comes with an opening cost $c(v) \in \mathbb{R}^+$, and we need to minimize $\sum_{f\in F} c(f) + \sum_{v\in V} \mathrm{dist}(v, F)$.

For this thesis we focus on metrics that model transportation networks, given that clustering problems arise in many applications of logistics where, for instance, we would like to place a limited number warehouses or hospitals on a map such that every point is close to one of them. In Section 3.1 we introduce several parameters modelling transportation networks, including the doubling and highway dimensions.

Algorithmically, the k-Median and Facility Location problems behave quite differently from the k-Center problem. Therefore, we present our results for these problems separately in Sections 3.2 and 3.3, respectively. Our main results in Section 3.2 include a near-linear time approximation scheme for k-Median and Facility Location parameterized by the doubling dimension, a PTAS for these problems parameterized by the highway dimension, and also some complementing hardness results for the latter parameter. Additionally, we present some results on metric embeddings of low highway dimension graphs into bounded treewidth graphs. These embeddings imply slower approximation schemes running in quasi-polynomial time for k-Median and Facility Location, but in return are applicable to a wider range of problems, including for instance Steiner Tree.

In Section 3.3 we begin with some hardness results for the k-Center problem, which show that using either k, the doubling dimension, or the highway dimension as a parameter is unlikely to yield better approximation factors than those obtainable in polynomial time. We then consider the combination of the parameter k with either the doubling dimension or the highway dimension, and show that in both cases it is possible to beat the previous lower bounds. Finally, we also show that even when combining all models of transportation networks presented in Section 3.1,

no exact solution for k-Center can be computed in FPT time, under standard complexity assumptions.

3.1 Metrics modelling transportation networks

In this section we present the metrics and parameters used for our results of the following sections, which are mainly based on the structure of transportation networks. For instance, a natural model for road networks is to assume that the given metric is the shortest-path metric of a planar graph, since overpasses and tunnels are relatively rare.

Another reasonable model is to assume that the metric is given by the Euclidean plane, since a road network is embedded on a large sphere (namely the Earth). In cities, where blocks of buildings form a grid of streets, it is reasonable to assume that the distances are given by the Manhattan plane. More generally, one might assume that a transportation network is given by some ℓ_q-norm in D-dimensional space for some small value of D. That is, the vertices of the given metric (V, dist) are points in $\mathbb{R}^D$ and the distance function is given by $\mathrm{dist}(u,v) = (\sum_{i=1}^{D} |u_i - v_i|^q)^{1/q}$.

The dimension D of such a metric space has been studied as a parameter from the parameterized approximation point-of-view *avant la lettre* for quite some time. While it may be unusual to see these results in the light of parameterized algorithms, we specifically do so here in order to obtain a more nuanced view of the complexity of the problems. For instance, it was shown [GGJ77; GI03] that in Euclidean metrics both the k-Median and Steiner Tree problems are paraNP-hard for this parameter (since they are NP-hard even if $D = 2$), and they are APX-hard in general metrics [CC08; JMS02]. However, EPASs for both the Steiner Tree and the k-Median problems in Euclidean metrics were shown to exist in the works of Arora [Aro98] and Kolliopoulos and Rao [KR07], respectively, who showed that a $(1+\varepsilon)$-approximation can be computed in $D^{O(\sqrt{D}/\varepsilon)^{D-1}} n^2$ time for Steiner Tree and $2^{O((\log(1/\varepsilon)/\varepsilon)^{D-1})} D^{O(D)} n^2$ time for k-Median.[5]

A related setting is the parameterization by the *doubling dimension* of the underlying metric, which is the smallest integer d such that any *ball* $B_v(r) = \{u \in V \mid \mathrm{dist}(u,v) \leq r\}$ of radius r in the metric can be covered by at most 2^d balls of half the radius $r/2$. Any point set in a D-dimensional ℓ_q-metric has doubling dimension $O(D)$, and thus the latter parameter generalizes the former. By a result of Talwar [Tal04], there are *quasi-polynomial time approximation schemes (QPTASs)* for Steiner Tree, k-Median, and Facility Location in metrics of constant doubling dimension $d \in O(1)$, i.e., they compute a $(1+\varepsilon)$-approximation in $2^{(\log n)^{f(d,\varepsilon)}}$ time for some function f. In the jargon of parameterized algorithms one could classify such an algorithm as a *slice-wise quasi-polynomial time approximation scheme*. The techniques used to obtain this algorithm for doubling metrics generalize those used for low dimensional ℓ_q-metrics. Since our algorithms presented in Section 3.2 build on these techniques as well, we will introduce them later. Using an entirely different local search technique it is possible to compute a $(1+\varepsilon)$-approximation in $n^{(d/\varepsilon)^{O(d)}}$ time [FRS19]. This is a PTAS assuming constant doubling dimension $d \in O(1)$, or in the jargon of parameterized algorithms, it is a *slice-wise polynomial time approximation scheme*.

A number of our results presented in Sections 3.2 and 3.3 are focussed on the *highway dimension*, which is a graph parameter specifically formalizing structural properties of transportation networks. We say that a metric has highway dimension h if it is the shortest-path metric of a graph of highway dimension h. The following definition is taken from [8].

[5] In [Aro98; KR07] the runtimes of these algorithms are stated as $O(n(\log n)^{O((\sqrt{D}/\varepsilon)^{D-1})})$ and $2^{O((\log(1/\varepsilon)/\varepsilon)^{D-1})} n \log^{D+6} n$, respectively, which can be shown to be upper bounded by $D^{O((\sqrt{D}/\varepsilon)^{D-1})} n^2$ and $2^{O((\log(1/\varepsilon)/\varepsilon)^{D-1})} D^{O(D)} n^2$ (see e.g. [KLP19, Lemma 1]).

Definition 3.1. The *highway dimension* of a graph G is the smallest integer h such that, for some universal constant $c \geq 4$, for every $r \in \mathbb{R}^+$, and every ball $B_v(cr)$ of radius cr, there are at most h vertices in $B_v(cr)$ hitting all shortest paths of length more than r that lie in $B_v(cr)$.

The highway dimension was originally defined by Abraham et al. [Abr+10], who specifically restricted the balls to have radius $4r$ in Definition 3.1. They also point out though that the choice of the constant c is somewhat arbitrary. In [8] we prove that when choosing any constant c strictly larger than 4 in Definition 3.1 we obtain additional properties for these graphs, which can be exploited algorithmically (see Section 3.2). Note though that increasing the constant c in Definition 3.1 restricts the class of graphs further. Moreover, as we shown in [8], the highway dimension of a graph according to Definition 3.1 can grow arbitrarily large by just a small change in the constant c. Indeed, for any c there is a graph of highway dimension 1 when using c in Definition 3.1, which however has highway dimension $\Omega(n)$ for any constant larger than c.

Since the original definition of the highway dimension in [Abr+10], several other definitions (including the above one for larger values of c) have been proposed, which we present next. We refer to [Blu19; 8] for detailed discussions.

In a follow-up paper to [Abr+10], Abraham et al. [Abr+16] define a much stronger definition of the highway dimension, which implies that the graphs also have bounded doubling dimension. Hence for this definition, any algorithm that uses the doubling dimension as a parameter can also be used as an algorithm for the highway dimension. Definition 3.1 on the other hand implies metrics of large doubling dimension as noted by Abraham et al. [Abr+10]: a star with unit edge lengths has highway dimension 1 (by using the center vertex to hit all paths), but its doubling dimension is unbounded. While it may be reasonable to assume that road networks have low doubling dimension (which are the main concern in the works of Abraham et al. [Abr+16; Abr+11; Abr+10]), there are metrics modelling transportation networks, for which it can be argued that the doubling dimension is large, while the highway dimension should be small, and thus rather adhere to Definition 3.1: in networks arising from public transportation, longer connections are serviced by larger and sparser stations (such as train stations and airports). More concretely, the so-called hub-and-spoke networks that can typically be seen in air traffic networks is much closer to a star-like network and is unlikely to have bounded doubling dimension, while still having small highway dimension. Thus in these examples it is reasonable to assume that the doubling dimension is a lot larger than the highway dimension.

All definitions of the highway dimension mentioned above imply the existence of sparse *shortest path covers*, as also introduced by Abraham et al. [Abr+10]. As done in following definition, these can thus be used to define an even more general notion of the highway dimension, which we also use for some of the results presented in Section 3.3. We also show in [8] that this is a strictly more general class, since there are graphs of highway dimension 1 according to Definition 3.2 below, which have unbounded highway dimension according to Definition 3.1 (using the same universal constant c). We also note that, as Definition 3.1, the following definition becomes more restrictive the larger the constant c is.

Definition 3.2. Let $c \geq 4$ be a universal constant. For a graph G, and $r \in \mathbb{R}^+$, a *shortest path cover* is a set $\textsc{spc}(r) \subseteq V$ of so-called *hubs* that hit all shortest paths of length in $(r, cr/2]$ of G. Such a cover is called *locally s-sparse* for scale r, if no ball $B_v(cr/2)$ of radius $cr/2$ contains more than s vertices from $\textsc{spc}(r)$. The *highway dimension* of G is the smallest integer h such that G has a locally h-sparse shortest path cover $\textsc{spc}(r)$ for every $r \in \mathbb{R}^+$.

Most of our results for low highway dimension metrics presented in the following sections exploit the structure obtained for graphs adhering to the stronger Definition 3.1 using a universal constant $c > 4$. It remains an interesting open question to determine whether there is any

difference in the algorithmic complexity for the studied problems between graphs of highway dimension according to Definition 3.1 and the larger class of graphs adhering to Definition 3.2. Especially on the algorithmic side it is often unclear how to obtain comparable results to those using Definition 3.1 when instead using Definition 3.2.

It should be further noted that (unless otherwise stated above) all the classes of metrics presented in this section are in general incomparable, i.e., they are not contained in one another. In fact, this is even true when comparing to other structural parameters such as shortest-path metrics of graphs with low tree- or pathwidth (cf. [Blu19; 8]).

3.2 The k-Median problem and its variants

Before turning to our results for the parameters introduced in Section 3.1, we mention some interesting known results for general metrics. Here we also consider the k-Means problem, which is similar to k-Median but the objective function squares the distances to the centers, i.e., we need to minimize $\sum_{v \in V}(\mathrm{dist}(v, F))^2$. The best approximation ratios achieved by polynomial time algorithms are $2.611 + \varepsilon$ for k-Median [Byr+14], and $9 + \varepsilon$ for k-Means [Kan+04]. From the hardness side, it is NP-hard to approximate k-Median [JMS02] within a factor $1 + 2/e - \varepsilon \approx 1.73 - \varepsilon$, and k-Means [Awa+15] within a factor $1 + 8/e - \varepsilon \approx 3.94 - \varepsilon$. While there are some gaps between these results for k-Median and k-Means, it is an interesting question to ask how the natural parameterization by k changes the approximation ratios for both problems. Cohen-Addad et al. [Coh+19] studied this question and gave exact answers. They 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 algorithms, giving corresponding upper and lower bounds for this parameter.

In the remainder of this section, we will first present our results for the parametrization by the doubling dimension, where we also give an overview of previous techniques on which ours build. These techniques are then refined for the parametrization by the highway dimension, to which we turn thereafter. Finally, we will also present some alternative techniques to solve problems on low doubling and low highway dimension metrics, which yield slower algorithms but can in return be applied to a wider range of problems.

3.2.1 Low doubling metrics

The starting point of many approximation algorithms for doubling metrics (including Euclidean spaces) is a decomposition of the metric, as presented in the following lemma. Here, a *hierarchical decomposition* $\mathcal{D}$ of a metric (V, dist) is a set of partitions $\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_\lambda$ of V, where $\mathcal{A}_i$ refines $\mathcal{A}_{i+1}$, i.e., every part $A \in \mathcal{A}_i$ is contained in some part of $\mathcal{A}_{i+1}$. Moreover, in $\mathcal{A}_0$ every part contains a singleton vertex, while $\mathcal{A}_\lambda$ contains only one part, namely V. For a point $v \in V$ and a radius $r > 0$, we say that the ball $B_v(r)$ is *cut by* $\mathcal{D}$ *at level* i if i is the largest integer for which the ball $B_v(r)$ is not contained in a single part of $\mathcal{A}_i$. The aspect ratio of a metric (V, dist) is the largest distance divided by the shortest distance of any points in V.

Lemma 3.3 (Reformulation of [BG13; Tal04])**.** *For any metric (V, dist) of doubling dimension d and aspect ratio α, and for any $\rho > 0$, there exists a polynomial-time computable randomized hierarchical decomposition $\mathcal{D} = \{\mathcal{A}_0, \ldots, \mathcal{A}_{\lceil \log_2 \alpha \rceil}\}$ such that:*

1. ***Scaling probability**: for any $v \in V$, radius r, and level i, we have*

$$\Pr[\mathcal{D} \text{ cuts } B_v(r) \text{ at level } i] \leq 2^{O(d)} \cdot r/2^i.$$

2. ***Portal set:** every part $A \in \mathcal{A}_i$ where $\mathcal{A}_i \in \mathcal{D}$ comes with a set of* portals $P_A \subseteq A$ *that is*

(a) ***concise:*** *the size of the portal set is bounded by* $|P_A| \leq 1/\rho^d$*, and*

(b) ***precise:*** *for every node* $u \in A$ *there is a portal* $p \in P_A$ *with* $\text{dist}(u,p) \leq \rho 2^{i+1}$.

We briefly sketch the standard use of this decomposition (see [Aro98; Mit99; Tal04]). For clustering problems, one can show that there exists a *portal-respecting solution* with near-optimal cost. In this structured solution, each client connects to a facility via a *portal-respecting path* that enters and leaves any part A of $\mathcal{D}$ only through a node of the portal set P_A. These portals therefore act as separators of the metric. A standard dynamic program approach can then compute the best portal respecting solution.

To ensure that there is a portal-respecting solution with near-optimal cost, one uses the preciseness property of the portal set: the additional distance (referred to as *distortion*) of connecting a client c with a facility f through portals instead of directly is bounded as follows. Let i be the level at which $\mathcal{D}$ *cuts* c *and* f, meaning that i is the maximum integer for which c and f lie in different parts of $\mathcal{A}_i$. At every level $j \leq i$, the distortion incurred by making a detour to the closest portal is $O(\rho 2^j)$, due to the triangle inequality and the preciseness of the portal set. Hence the total distortion is $\sum_{j \leq i} O(\rho 2^j) = O(\rho 2^i)$. Now, the bound on the scaling probability of the decomposition ensures that c and f are cut at level i with probability $2^{O(d)}\text{dist}(c,f)/2^i$. Hence combining these two bounds over all $\lceil \log_2 \alpha \rceil + 1$ levels ensures that, in expectation, the distortion between c and f is bounded by $2^{O(d)}\text{dist}(c,f) \cdot \rho \lceil \log_2 \alpha \rceil$. Using standard preprocessing techniques one can ensure that the aspect ratio is $\alpha = O(n/\varepsilon)$ when aiming for a $(1+\varepsilon)$-approximation. Hence choosing $\rho = \frac{\varepsilon}{2^{O(d)} \log(n/\varepsilon)}$ gives a distortion of $\varepsilon \cdot \text{dist}(c,f)$. Summing over all clients proves that there exists a near-optimal portal-respecting solution.

The issue with this approach is that to obtain this level of preciseness, according to the conciseness property the number of needed portals for each part A of the decomposition $\mathcal{D}$ is $(\frac{\log(n/\varepsilon)}{\varepsilon})^{O(d)}$, and the dynamic program has a runtime that is exponential in this number. As Talwar [Tal04] showed, the resulting algorithm runs in $2^{(\frac{d \log n}{\varepsilon})^{O(d)}}$ time, i.e., for any constant $d \in O(1)$ we obtain a QPTAS. However, in some cases one can lower the number of portals per part needed and thus obtain a PTAS. In Euclidean space for example, the celebrated "patching lemma" [Aro98] shows that only a constant number (depending on ε) of portals are needed for Steiner Tree. Similarly, Kolliopoulos and Rao [KR07] showed that for k-Median in Euclidean space only a constant number of portal are needed, if one uses a slightly different decomposition of the metric. Surprisingly, obtaining such a result for doubling metrics is much more challenging.

A second challenge occurs when trying to solve problems such as k-Means, where the objective function squares the distances to the centers. In this case, the analysis of Arora [Aro98], Mitchell [Mit99], and Talwar [Tal04] does not apply: if two points are separated at a high level of the decomposition, then making a detour to the closest portal may incur an expected cost much higher than the cost of the optimal solution.

In [4] we show how to circumvent these issues for clustering problems in low doubling metrics. Our contribution can be viewed as a "patching lemma" for problems such as k-Median, k-Means, and Facility Location. Namely, we present an approach which (1) reduces the number of portals to a constant, (2) works for any clustering objective which is defined as the sum of distances to some constant q (with k-Median, Facility Location, and k-Means as prominent special cases), and (3) works not only for Euclidean but also for doubling metrics.

To achieve this, in [4] we show how to reduce the number of levels on which a client can be cut from its facility. For this, we present a processing step of the instance that helps deal with clients cut from their facility at a high level. Roughly speaking, our algorithm first computes a constant factor approximation L, and a client c is called *badly-cut* if the decomposition $\mathcal{D}$ cuts it from its closest facility of L at a level larger than $\log(\text{dist}(c,L)/\varepsilon)$. Every badly-cut client is moved to its

closest facility of L. Moreover, every client at distance less than $\varepsilon \cdot \mathrm{dist}(c, L)$ of its closest facility of L can be moved to it as well. It is then shown that this new instance $\mathcal{I}_\mathcal{D}$ has *small distortion*, which essentially means that any solution to $\mathcal{I}_\mathcal{D}$ can be converted to a solution of the original instance $\mathcal{I}$ while only losing a $(1+\varepsilon)$-factor in quality. In this instance $\mathcal{I}_\mathcal{D}$, all clients are cut from their closest facility of L at some level between $\log(\varepsilon \cdot \mathrm{dist}(c, L))$ and $\log(\mathrm{dist}(c, L)/\varepsilon)$. Using this property, we show that c and its closest center in the optimal solution are also cut at a level in that range. As there are only $O(\log(1/\varepsilon))$ levels in this range, by the previous arguments it now suffices to set $\rho = \frac{\varepsilon}{2^{O(d)} \log(1/\varepsilon)}$ for the hierarchical decomposition in order to get a near-optimal portal-respecting solution. As this implies a number of portals that only depends on the doubling dimension d and the approximation factor ε, according to the conciseness property we obtain the following randomized approximation schemes parameterized by d and ε. In particular, we get a significant improvement from the previously fastest known slice-wise polynomial time approximation schemes as given by [FRS19], to randomized EPASs with near-linear running time.

Theorem 3.4 ([4]). *For the* k-Median, k-Means, *and* Facility Location *problems on metrics of doubling dimension* d *a* $(1+\varepsilon)$*-approximation can be computed in* $\widetilde{O}(2^{(1/\varepsilon)^{O(d^2)}} n)$ *time with success probability* $1 - O(\varepsilon)$ *for any* $\varepsilon > 0$.

It is interesting to note that the double-exponential dependence on d in the runtime cannot be improved to single-exponential, since any metric has doubling dimension $O(\log n)$ but the problems are APX-hard [Awa+15; GK99; JMS02] in general metrics. As mentioned before, Theorem 3.4 also holds for the corresponding problems where the distances in the objective functions are raised to the power of any integer q, so that k-Median and k-Means for instance are the special cases where $q = 1$ and $q = 2$, respectively. Furthermore, the techniques can be generalized to obtain bicriteria approximation schemes with similar running times for prize-collecting and outlier versions of the problems (cf. [4]).

3.2.2 Low highway dimension graphs

We now turn to the parametrization by the highway dimension, for which in [12] we obtain slower approximation schemes for clustering problems than those given by Theorem 3.4 for the doubling dimension. Nevertheless, our algorithms for the highway dimension utilize the techniques described above for the doubling dimension, in addition to some structural insights we obtained for low highway dimension graphs in [8].

More concretely, the above arguments for doubling metrics hold thanks to the hierarchical decomposition given by Lemma 3.3. It is therefore tempting to try to devise a similar decomposition for metrics of low highway dimension. However, it turns out that while these metrics have some similarities to low doubling metrics, they behave very differently, so that we were not able to obtain a decomposition with the properties given by Lemma 3.3. Instead, in [8] we introduce the following *town decomposition* of low highway dimension metrics, which gives a formal connection to doubling metrics. We obtain its properties specifically for Definition 3.1 of the highway dimension using any universal constant c strictly larger than 4. In the following, a *child part* of a part $A \in \mathcal{A}_i$ of some hierarchical decomposition $\mathcal{D} = \{\mathcal{A}_0, \ldots, \mathcal{A}_\lambda\}$ is a part $A' \in \mathcal{A}_{i-1}$ on the level below i for which $A' \subseteq A$.

Theorem 3.5 ([8]). *Given* $\rho > 0$ *and a shortest-path metric* (V, dist) *of highway dimension* h *according to Definition 3.1 for any universal constant* $c > 4$*, there exists a polynomial-time computable deterministic hierarchical decomposition* $\mathcal{T}$*, called the* town decomposition*, such that every part* $T \in \mathcal{T}$*, called a* town*, has a set of* hubs[6] $X_T \subseteq T$ *with the following properties:*

[6] called *approximate core hubs* in [8].

a. ***doubling**: the doubling dimension of X_T is $d = O(\log(h \log(1/\rho)))$, and*

b. ***precise**: for any two vertices u and v in different child parts of T, there is a hub $x \in X_T$ such that $\operatorname{dist}(u, x) + \operatorname{dist}(x, v) \leq (1 + 2\rho) \cdot \operatorname{dist}(u, v)$.*

The hub set X_T is similar to the portal set of Lemma 3.3, but has some fundamental differences. First note that the preciseness property of Theorem 3.5 is different from the preciseness of Lemma 3.3: in the former, some hub x is guaranteed to be close to a shortest path between u and v (and could thus be far from each of u and v), while in Lemma 3.3 the portals lie close to all vertices. Secondly, the town decomposition is deterministic, and so it may happen that a client and its facility are cut at a very high level relative to their distance — something that happens only with small probability in the doubling setting thanks to the scaling probability. Another main difference is that the size of X_T might be unbounded. As a consequence, it cannot be directly used as a portal set in a dynamic program with sub-exponential runtime. To deal with this, in [12] we combine the town decomposition with a hierarchical decomposition of each set X_T according to Lemma 3.3, to build a decomposition of low highway dimension graphs more akin to Lemma 3.3, as stated in the following lemma.

Lemma 3.6 ([12])**.** *Given $\rho > 0$ and a metric (V, dist) with aspect ratio α where (V, dist) is a shortest-path metric of a graph with highway dimension h according to Definition 3.1 for any universal constant $c > 4$, there exists a polynomial-time computable randomized hierarchical decomposition $\mathcal{D} = \{\mathcal{A}_0, \ldots, \mathcal{A}_{\lceil \log_2 \alpha \rceil}\}$ of V such that:*

1. ***Scaling probability**: for any $v \in V$, radius r, and level i, we have*

$$\Pr[\mathcal{D} \text{ cuts } B_v(r) \text{ at level } i] \leq (h \log(1/\rho))^{O(1)} \cdot r/2^i.$$

2. ***Interface**: for any $A \in \mathcal{A}_i$ on level $i \geq 1$ there exists an interface $I_A \subseteq V$, which is*

   (a) ***concise**: $|I_A| \leq (h/\rho)^{O(1)}$, and*

   (b) ***precise**: for any $u, v \in A$ such that u and v are cut by $\mathcal{D}$ at level $i - 1$, there exists $p \in I_A$ with $\operatorname{dist}(u, p) + \operatorname{dist}(p, v) \leq \operatorname{dist}(u, v) + 34 \cdot \rho 2^i$.*

As a consequence of using the town decomposition of Theorem 3.5 to construct the hierarchical decomposition of Lemma 3.6, a notable difference to the portals of Lemma 3.3 is that the preciseness property of the interface in Lemma 3.6 is weaker: as for Theorem 3.5, the hubs can be far from some vertices as long as they lie close to the shortest path. As a consequence, no analogue of near-optimal portal-respecting paths exist as was the case for portals. Instead, when connecting a client c with a facility f we need to use the interface point $p \in I_A$ of a part A containing both c and f, such that p lies close to the shortest path between c and f. This shifts the perspective from externally connecting vertices of a part to vertices outside a part, as done for portals, to internally connecting vertices of parts, as needs to be done for interfaces.

As a consequence, in [12] we develop a dynamic program, which follows more or less standard techniques as for instance given in [ARR98; KR07], but needs to handle the weaker preciseness property of the interface. The main idea is to guess the distances from interface points to facilities while recursing on the decomposition $\mathcal{D}$ of Lemma 3.6. The runtime of this algorithm is thus exponential in the number of interface points. Thanks to our techniques developed in [4] as described above, we can assume that this number is constant for clustering problems. However, due to the shifted perspective towards internally connecting vertices of parts, the runtime of the dynamic program also is exponential in the total number of levels. As for doubling metrics the aspect ratio can be reduced [8] to $\alpha = O(n/\varepsilon)$ when aiming for a $(1 + \varepsilon)$-approximation, and so the number of levels of the decomposition is logarithmic in the input size. This implies that the runtime is polynomial, and we obtain the following theorem.

Theorem 3.7 ([12]). *For the* k-Median, k-Means, *and* Facility Location *problems on metrics of highway dimension* h *according to Definition 3.1 for any universal constant* $c > 4$, *a* $(1+\varepsilon)$*-approximation can be computed in* $n^{(h/\varepsilon)^{O(1)}}$ *time with success probability* $1 - O(\varepsilon)$ *for any* $\varepsilon > 0$.

Since the techniques for Theorem 3.4 are also used for Theorem 3.7, similar to the former the algorithm of Theorem 3.7 generalizes to the corresponding problems where the distances in the objective function are raised to the power of any integer q, and bicriteria approximation schemes with similar runtimes can be obtained for prize-collecting and outlier versions. Note though that in contrast to Theorem 3.4 the algorithm of Theorem 3.7 is not an EPAS, but rather a slice-wise polynomial time approximation scheme. Whether an EPAS or even a PAS exists for the parameterization by the highway dimension as well, remains an intriguing open question.

In [12] we show that, unless P=NP, no polynomial time algorithm exists to compute optimum solutions, even in the most restrictive case when the highway dimension is 1. In fact, we obtained similar results in [5] for the Steiner Tree and Travelling Salesman problems, where for the latter we are given a metric and we need to find the shortest tour that visits all vertices. These hardness results are valid regardless of which definition of the highway dimension is used (in particular Definition 3.1 can be made arbitrarily strong by using any universal constant $c \geq 4$.).

Theorem 3.8 ([5; 12]). *The* k-Median, k-Means, Facility Location, Steiner Tree, *and* Travelling Salesman *problems are* NP*-hard on metrics of highway dimension* 1 *according to Definition 3.1 for any universal constant* $c \geq 4$.

3.2.3 Metric embeddings

We now slightly digress from the topic of this section by considering not just clustering problems but other problems as well. We present an alternative view on hierarchical decompositions leading to so-called embeddings. This tool is rather general and can be used to obtain approximation schemes for clustering problems, but also others such as Steiner Tree and Travelling Salesman, albeit with in larger running times compared to the algorithms presented so far. The idea is to map a given metric into another metric on the same vertex set, which on one hand slightly distorts the distances, but on the other hand introduces some structural properties that can be exploited algorithmically. If the problem at hand can then be solved on the latter metric, then the approximation factor is determined by the distortion when mapping the solution back to the input metric. We will specifically be focussing on mappings into shortest-path metrics of graphs with small treewidth, since plenty of algorithms are known for such graphs. Such a mapping can be probabilistic and is defined as follows.

Definition 3.9. Let (V, dist) be a metric and let $\mathcal{E}$ be a distribution over metrics (V, dist') on the same vertex set V. If for all $u, v \in V$, $\mathrm{dist}(u,v) \leq \mathrm{dist}'(u,v)$ for each $\mathrm{dist}' \in \mathcal{E}$, and $\mathbb{E}_{\mathrm{dist}' \in \mathcal{E}}[\mathrm{dist}'(u,v)] \leq a \cdot \mathrm{dist}(u,v)$, then $\mathcal{E}$ is an *embedding* with (expected) *stretch* or *distortion* a. If every $\mathrm{dist}' \in \mathcal{E}$ is the shortest-path metric of some graph class $\mathcal{G}$, then $\mathcal{E}$ is a *(probabilistic) embedding into* $\mathcal{G}$.

It was noted by Talwar [Tal04] that Lemma 3.3 implies a polynomial-time computable probabilistic embedding of any metric of doubling dimension d and aspect ratio α into graphs of treewidth $(d \log(\alpha)/\varepsilon)^{O(d)}$ with expected distortion $1+\varepsilon$. More concretely, to compute a graph of low treewidth from a given metric, first the hierarchical decomposition $\mathcal{D} = \{\mathcal{A}_0, \ldots, \mathcal{A}_{\lceil \log_2 \alpha \rceil}\}$ of Lemma 3.3 is computed. The graph then contains all edges between portals P_A for each part A, and also all edges connecting the portals P_A with the portals $P_{A'}$ of each child part A' of A. The weight of such an edge uv is simply the distance between u and v given by the metric.

In particular, any portal-respecting path of the metric exists in the graph, and so the distortion is $1+\varepsilon$ if ρ is set to an appropriate value for Lemma 3.3, as previously argued in Section 3.2.1. At the same time, a tree decomposition of the graph can be obtained by using the tree structure of $\mathcal{D}$ (i.e., every part corresponds to a node connected to the nodes of its child parts) and introducing a bag for each part A that contains its portal set P_A and the portal sets $P_{A'}$ of all its child parts A'. For doubling metrics it can be ensured [4; Tal04] that the portal sets are *nested*, which means that every portal of A which happens to be in one of its child parts A' is also a portal of A', i.e., $P_A \cap A' \subseteq P_{A'}$. Due to this, we can prove that we obtain a valid tree decomposition. Furthermore, it can be shown that each part has at most $2^{O(d)}$ child parts, and thus the treewidth is bounded by the conciseness property of Lemma 3.3, which bounds the size of the portal sets.

In [8] we build on this construction to show that also low highway dimension metrics can be embedded into graphs of bounded treewidth. A key ingredient for this is again our structural insight into low highway dimension graphs given by Theorem 3.5. On a high level, a graph of bounded treewidth is constructed by computing an embedding of each hub set X_T for every town T of a town decomposition of the given metric. Since Theorem 3.5 guarantees that each set X_T has bounded doubling dimension, each individual embedding has bounded treewidth, as argued above. The challenge now is to combine all of these embeddings into one, while making sure that both the distortion and the treewidth are still small. One problem for instance is that hub sets X_T of different towns are not nested, i.e., it may happen that a hub of a town T is not a hub of the child town, but then is again a hub of some lower-level descendent town of T. Furthermore, there is no bound on the number of child towns of a given town. For this and other reasons that distinguish low highway from low doubling dimension metrics, compared to low doubling metrics a lot more work goes into proving the following theorem.

Theorem 3.10 ([8])**.** *Let* (V, dist) *be a metric with aspect ratio* α *where* (V, dist) *is a shortest-path metric of a graph with highway dimension* h *according to Definition 3.1 for any universal constant* $c > 4$*. For any* $\varepsilon > 0$*, there is a polynomial-time computable probabilistic embedding of* (V, dist) *with expected distortion* $1+\varepsilon$ *into graphs of treewidth* $(\log \alpha)^{O(\log^2(h/\varepsilon))}$*.*

Using known algorithms for bounded treewidth graphs on the embedding given by Theorem 3.10, we obtain randomized approximation schemes for metrics of low highway dimension for problems such as k-Median and Facility Location, but also Steiner Tree and Travelling Salesman. The expected approximation guarantee is given by the distortion of the distances. To bound the runtime, as previously we may preprocess the metric (cf. [8]) so that its aspect ratio is $O(n/\varepsilon)$, which means that the treewidth bound of Theorem 3.10 is poly-logarithmic for metrics of constant highway dimension $h \in O(1)$ and for constant approximation factors $\varepsilon \in \Theta(1)$. Since algorithms for bounded treewidth graphs have running times exponential in the treewidth, this gives slice-wise quasi-polynomial runtimes, i.e., we obtain QPTASs for the given problems via the embedding of Theorem 3.10. It remains open whether QPTASs also exist for these problems when using the more general Definition 3.2 for the highway dimension.

Since the embedding for low doubling metrics by Talwar [Tal04] was constructed using the hierarchical decomposition of Lemma 3.3, a natural question is whether the above embedding for low highway dimension metrics can be simplified by using the corresponding decomposition of Lemma 3.6. This seems plausible, and might even yield improved bounds on the treewidth compared to Theorem 3.10. However this still needs to be explored and, as of writing this thesis, is left for future work. Similar to obtaining the PTAS of Theorem 3.7 based on Lemma 3.6, one main challenge for such an embedding using Lemma 3.6 is the non-existence of portal-respecting paths, which are used in the construction for Theorem 3.10 to bound the distortion.

The algorithms that are used on the metric embedding resulting from Theorem 3.10 to obtain

QPTASs for low highway dimension graphs can be any FPT or XP algorithms parameterized by the treewidth t, of which the literature provides plenty. Specifically for Steiner Tree and Travelling Salesman, there are rather efficient single-exponential $2^{O(t)}n^{O(1)}$ time FPT algorithms for this parameter [Bod+15]. We show in [5] that this implies the curious fact that these problems are *weakly* NP-hard on graphs of the smallest possible highway dimension, i.e., on graphs of highway dimension 1 the problems are NP-hard due to Theorem 3.8, but they also admit *fully polynomial time approximation schemes (FPTASs)*, which compute a $(1+\varepsilon)$-approximation with a runtime that is polynomial in both the input size and ε.

We prove this in [5] as follows. First we show that graphs of highway dimension 1 have treewidth bounded in their aspect ratio α (which can be seen as a trivial embedding with distortion 1), even when using the more general Definition 3.2. As before, we may reduce the aspect ratio to $O(n/\varepsilon)$, and thereby lose a $(1+\varepsilon)$-factor in the solution quality for Steiner Tree and Travelling Salesman. Thus according to the following theorem, the single-exponential time FPT algorithms for parameter treewidth [Bod+15] run in $2^{O(\log(n/\varepsilon))}n^{O(1)} = (n/\varepsilon)^{O(1)}$ time on the reduced instances.

Theorem 3.11 ([5])**.** *Let G be a graph with aspect ratio α and highway dimension 1 according to Definition 3.2 for any universal constant $c \geq 4$. The treewidth of G is $O(\log \alpha)$.*

In general, one might hope to prove similar bounds on the treewidth of graphs with of highway dimension larger than 1, i.e., one might conjecture that any graph of highway dimension h has treewidth, say, $(h \log \alpha)^{O(1)}$ or $O(\log^h \alpha)$. Such a bound would make it possible to circumvent the rather involved construction of the embedding given by Theorem 3.10. Also, depending on the quality of the bound this might imply faster approximation schemes for Steiner Tree and Travelling Salesman due to the single-exponential FPT algorithms for parameter treewidth. However, we can exclude at least some such general treewidth bounds for graphs of low highway dimension using a result we obtained in [7], which consists of an embedding of low doubling metrics into graphs of bounded highway dimension. Since this embedding is used to obtain lower bounds, it is interesting to note that it can be applied with the more restrictive Definition 3.1 of the highway dimension, and moreover the embedding is deterministic.

Theorem 3.12 ([7])**.** *Let (V, dist) be a metric with aspect ratio α and doubling dimension d. For any $\varepsilon > 0$, there is a polynomial-time computable deterministic embedding of (V, dist) with distortion $1+\varepsilon$ into a graph of highway dimension $O((\log(\alpha)/\varepsilon)^d)$ according to Definition 3.1 for any universal constant $c \geq 4$.*

This embedding implies a lower bound excluding a treewidth of $(h \log \alpha)^{O(1)}$ for graphs of highway dimension h and aspect ratio α, as follows. We start from a metric given by a regular $k \times k$ grid in the plane endowed with the ℓ_1-norm, which has doubling dimension 2 and aspect ratio k. Using Theorem 3.12, from this we obtain a graph G of highway dimension $O((\log(k)/\varepsilon)^2)$. Now, consider two neighbouring nodes (at distance 1) in the input grid, and note that connecting them using any additional nodes gives a path of length at least 3 due to the ℓ_1-norm. Thus if two neighbouring nodes end up not being connected by an edge in G, then any path between these vertices in G has length at least 3. Setting $\varepsilon < 2$ to obtain a distortion of less than 3 ensures that G contains a $k \times k$ grid as a subgraph, since the embedding of Theorem 3.12 is deterministic so that every pair of neighbouring nodes of the grid must end up being connected by an edge in G. Hence for instance setting $\varepsilon = 1$ we obtain a graph with treewidth $\Omega(k)$, aspect ratio $O(k)$, and highway dimension $O(\log^2 k)$. This excludes a treewidth bound of the form $(h \log \alpha)^{O(1)}$.

Note that a treewidth of $O(\log^h \alpha)$ is not excluded by the above argument. However it still seems unlikely due to the following. In [7] we use Theorem 3.12 to prove that, unless P=NP, the k-Center problem has no polynomial time $(2-\varepsilon)$-approximation algorithm on graphs

with highway dimension $O(\log^2 n)$ (cf. Section 3.3). However, we conjecture that the same inapproximability should hold for graphs of highway dimension $O(1)$. If this is true, then we could again use standard preprocessing to reduce the aspect ratio to $O(n/\varepsilon)$, from which we would obtain a treewidth bound of $O(\log^h \alpha) = O(\text{polylog}(n/\varepsilon))$ for such graphs while distorting the distances by a factor of $1+\varepsilon$. Now, for k-Center Katsikarelis et al. [KLP19] obtain an EPAS with runtime $t^{O(t/\varepsilon)}n^{O(1)}$ parameterized by the treewidth t, which would run in quasi-polynomial time on graphs with treewidth $O(\text{polylog}(n/\varepsilon))$. Consequently, a treewidth bound of $O(\log^h \alpha)$ for graphs of highway dimension h in combination with a reduction to show APX-hardness for graphs of highway dimension $O(1)$, would imply quasi-polynomial time algorithms to solve NP-hard problems. However under standard complexity assumptions this is not possible.

3.3 The k-Center problem

The k-Center problem is harder to approximate than k-Median and its variants. In particular, by a result of Hochbaum and Shmoys [HS86], the k-Center problem on general input graphs has a polynomial time 2-approximation algorithm, but this approximation factor is also best possible, unless P=NP. The lower bound of 2 on the approximation factor is rather notorious as it remains valid for many special cases and parameterizations of the problem, including those introduced in Section 3.1 as models for transportation networks. We summarize this in the following theorem.

Theorem 3.13. *The k-Center problem has no $(2-\varepsilon)$-approximation algorithms for any $\varepsilon > 0$ in the following cases and runtimes:*

- *on general graphs in $f(k)n^{O(1)}$ time [7] (i.e., parameterized by the number of centers k) for any computable function f, unless* W[2]=FPT,
- *on planar graphs in polynomial time [Ple80], unless* P=NP,
- *on two-dimensional Manhattan metrics (which have doubling dimension 2) in polynomial time [FG88], unless* P=NP,
- *on graphs of highway dimension $O(\log^2 n)$ according to Definition 3.1 for any universal constant $c \geq 4$ in polynomial time [7], unless* P=NP,
- *on graphs of highway dimension h according to Definition 3.1 for any universal constant $c \geq 4$ in $2^{2^{o(\sqrt{h})}} \cdot n^{O(1)}$ time [7], under* ETH.

The last two lower bounds of Theorem 3.13 for the highway dimension are based on the embedding of Theorem 3.12, which we use in [7] to reduce the problem on metrics of doubling dimension 2 to graphs of highway dimension $O(\log^2 n)$. The hardness of approximation for the former (as also stated in Theorem 3.13) thus carries over to the latter. Note that the hardness for graph of highway dimension $O(\log^2 n)$ does not rule out $(2-\varepsilon)$-approximation algorithms parameterized by the highway dimension, and this is left as an open problem. However, under ETH, the same reduction implies that if such an algorithm exists, then its running time must be enormous, as it must be at least doubly exponential, as stated in Theorem 3.13.

In conclusion of Theorem 3.13 it seems that considering any model of Section 3.1 for transportation networks or even the parametrization by the number of centers k, does not help to overcome the polynomial-time $(2-\varepsilon)$-inapproximability that the k-Center problem exhibits for general inputs. However, combining k as a parameter with any of the models it is possible to beat this lower bound. For instance, by a result of Fox-Epstein et al. [FKS19] the k-Center problem on edge-weighted planar graphs admits an *efficient polynomial-time bicriteria approximation scheme*, which for any $\varepsilon > 0$ and some function f computes a solution in $f(\varepsilon)n^{O(1)}$ time that uses at most $(1+\varepsilon)k$ centers and approximates the optimum with at most k centers within a factor of $1+\varepsilon$. This algorithm implies an EPAS for parameter k on planar graphs, since

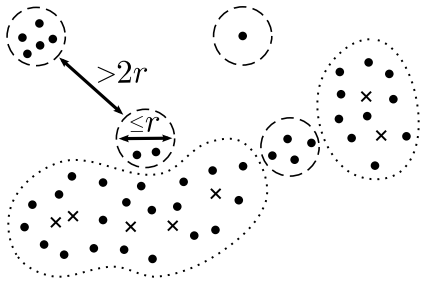

Figure 1: Clusters (dashed circles) are far from hubs (crosses). They have small diameter and are far from each other.

for instance setting $\varepsilon = \min\{\varepsilon', \frac{1}{2k}\}$ forces the algorithm to compute a $(1+\varepsilon')$-approximation in $f(k,\varepsilon')n^{O(1)}$ time using at most $(1+\varepsilon)k \leq k + \frac{1}{2}$ centers, i.e., at most k centers as k is an integer.

Another example is given by Agarwal and Procopiuc [AP02] who showed that for any ℓ_q metric in D dimensions, the k-Center problem has an EPAS when combining k and D as parameters. In [11] we generalize the latter to any metric of doubling dimension d. Our algorithm first guesses the optimum cost C. It then greedily computes a so-called *δ-net*, which is a subset of the vertices of the given metric such that every vertex is at distance at most δ from some net point, while all net points are at distance more than δ from each other. Any δ-net of a metric with doubling dimension d can be shown to have size at most $k/\varepsilon^{O(d)}$ when setting $\delta = \Theta(\varepsilon C)$. Thus we may compute an optimum k-Center solution of the vertices of the δ-net by brute-force in FPT time. The distance properties of the δ-net for $\delta = \Theta(\varepsilon C)$ imply that this solution is a $(1+\varepsilon)$-approximation in the input metric, and thus we obtain the following theorem.

Theorem 3.14 ([11])**.** *Given $\varepsilon > 0$ and a metric of doubling dimension d, a $(1+\varepsilon)$-approximation for k-Center can be computed in $(k^k/\varepsilon^{O(kd)})n^{O(1)}$ time.*

For low highway dimension graphs, Becker et al. [BKS18] used the structural results found in Theorem 3.5 to show that when using Definition 3.1 for any universal constant $c > 4$ there is an EPAS for k-Center parameterized by k and the highway dimension h. For the more general Definition 3.2 of the highway dimension it is not known whether a PAS exists for this parameterization. However, in [7] we present a 3/2-approximation for k-Center, which combines k and this notion of highway dimension as parameters. Hence also for the more general class of graphs given by Definition 3.2 the notorious $(2-\varepsilon)$-inapproximability of the k-Center problem can be beaten.

Theorem 3.15 ([7])**.** *Given $\varepsilon > 0$ and a metric of highway dimension dimension h according to Definition 3.2 for any universal constant $c \geq 4$, a 3/2-approximation for k-Center can be computed in $2^{O(kh\log h)}n^{O(1)}$ time.*

As we show in [8] some fundamental properties needed to prove the structure given by Theorem 3.5 break down when using the more general Definition 3.2 for the highway dimension. Thus, in contrast to the EPAS of Becker et al. [BKS18], we cannot rely on Theorem 3.5 to prove Theorem 3.15. Using the weaker Definition 3.2 for the highway dimension still implies some interesting structure though: fixing any value r and a shortest path cover $\text{SPC}(r)$, we show in [7] that the vertices of the metric are either at distance at most r from some hub of $\text{SPC}(r)$, or they lie in clusters[7] of diameter at most r that are at distance more than $2r$ from each other

[7] in fact these clusters are similar to (but not quite the same as) the towns in the town decomposition of Theorem 3.5 (cf. [8]).

(see Figure 1). Hence, given the cost C of the optimum k-Center solution, for $r = C/2$ a center that resides in a cluster cannot cover any vertices of some other cluster. In this sense the clusters are "independent" of each other. At the same time we are able to bound the number of hubs in $\text{SPC}(C/2)$ in terms of k and the highway dimension. Roughly, this is comparable to graphs with small vertex cover, since the vertices that are not part of a vertex cover form an independent set. In this sense the highway dimension is a generalization of the vertex cover number, and this is in fact the reason why computing the highway dimension is NP-hard, as we show in [8].

At the same time the k-Center problem is a generalization of the Dominating Set problem, where we need to select the smallest number of vertices in a graph such that every vertex is at hop-distance at most 1 from a selected vertex. This problem is W[2]-hard [DF13], but it is FPT using the vertex cover number as the parameter [Alb+02]. This is one of the reasons why combining the two parameters k and h yields a parameterized 3/2-approximation algorithm for k-Center. In fact the similarity seems so striking at first that one is tempted to reduce the problem of finding a 3/2-approximation for k-Center on low highway dimension graphs to solving Dominating Set on a graph of low vertex cover number. However, it is unclear how this can be made to work. Instead, in [7] we devise an involved algorithm that is driven by the intuition that the two problems are similar to obtain Theorem 3.15.

Given the above parameterized approximation algorithms for models of transportation networks when combining with the parameter k, a natural question is whether it is actually necessary to approximate in these cases. That is, can we hope to compute the optimum k-Center solution in comparable running times? In [11] we study this question and conclude that under standard complexity assumptions this is not possible, even if we combine all the models found in Section 3.1, as formalized by the following theorem. It is interesting to note that all the mentioned graph classes and parameters are incomparable to each other, as we show in [8] and discussed in more detail by Blum [Blu19].

Theorem 3.16 ([11])**.** *Even on edge-weighted planar graphs of doubling dimension $O(1)$, the k-Center problem is* W[1]*-hard for the combined parameter (k, p, h), where p is the pathwidth and h the highway dimension according to Definition 3.1 for any universal constant $c \geq 4$. Moreover, there is no $f(k, p, h) \cdot n^{o(p+\sqrt{k+h})}$ time algorithm[8] for the same restriction on the input graphs, for any computable function f, under ETH.*

Note that in this theorem we also add the pathwidth as a parameter, which arguably is not very useful to model transportation networks, since road networks of large cities (especially on the American continent) can contain large grids, which implies that the pathwidth will be rather large. We include this well-studied parameter here nonetheless, since by a result of Katsikarelis et al. [KLP19] it is known that an EPAS exists when parameterizing by the pathwidth (or even the tree- or cliquewidth). Thus Theorem 3.16 complements not only this result, but also the above ones for planar, low doubling, and low highway dimension graphs, by showing that approximations are necessary in each case. But furthermore, even if one were to combine all the models presented in Section 3.1 and assume that a transportation network is planar, is embeddable into some metric of constant doubling dimension, has bounded highway dimension, and even has bounded pathwidth, the k-Center problem cannot be solved efficiently, unless FPT=W[1]. Thus it seems unavoidable to approximate the problem in transportation networks when developing fast algorithms. A recent result by Blum [Blu20] also shows that the *skeleton dimension*, which is yet another graph parameter for transportation networks, can be added to the list of parameters in Theorem 3.16 while still obtaining W[1]-hardness.

To obtain Theorem 3.16, in [11] we give a reduction from the Grid Tiling with Inequality problem, which was introduced by Marx and Sidiropoulos [MS14] and is defined as follows. Given

[8] here $o(p + \sqrt{k+h})$ means $g(p + \sqrt{k+h})$ for any function g such that $g(x) \in o(x)$.

κ^2 non-empty sets $S_{i,j} \subseteq [n]^2$ of pairs of integers, where $i, j \in [\kappa]$, the task is to select one pair $s_{i,j} \in S_{i,j}$ for each set such that

- if $s_{i,j} = (a, b)$ and $s_{i+1,j} = (a', b')$ for $i \leq \kappa - 1$ then $a \leq a'$, and
- if $s_{i,j} = (a, b)$ and $s_{i,j+1} = (a', b')$ for $j \leq \kappa - 1$ then $b \leq b'$.

The Grid Tiling with Inequality problem is W[1]-hard for parameter κ, and moreover, under ETH has no $f(\kappa) \cdot n^{o(\kappa)}$ time algorithm for any computable function f.

This problem is typically used to show hardness of various problems on planar graphs or ℓ_q-metrics in the plane. On a high level, we can think of each set $S_{i,j}$ of an instance of Grid Tiling with Inequality as the contents of a cell with coordinates (i, j) in a $\kappa \times \kappa$ grid. For a reduction, we may introduce a gadget for each cell, which encodes the integer pairs of the corresponding set $S_{i,j}$, and the gadgets are then arranged in a grid-like fashion. If each gadget is a planar graph then so is the overall constructed graph. For ℓ_q-metrics in the plane, each gadget is given by a point set contained in a square corresponding to a cell of the grid. In our reduction for Theorem 3.16, ideally we would like to combine both these approaches in order to obtain a planar graph of constant doubling dimension. Thus our gadgets are edge-weighted planar graphs, but it is not clear whether the vertices can be mapped to points in the plane so that for some q the ℓ_q-distances are the same as distances in the resulting planar graph. Instead of mapping the constructed graph into the plane, we exploit the grid-like structure of the graph and use the intuition that the distances of a regular grid in the plane abide to the ℓ_1-norm. In particular, we show that every ball in the graph can be covered by at most 324 balls of half the radius, and thus the doubling dimension is constant.

We then go on to bound all parameters in our reduction for Theorem 3.16 in terms of κ. For k we show that an instance of Grid Tiling with Inequality has a solution if and only if the constructed graph has a k-Center solution of a certain bounded cost, in which each gadget needs exactly 5 centers, i.e., $k = 5\kappa^2$. To bound the pathwidth p and the highway dimension h in our reduction we again use the grid structure of the constructed graph. In particular, it is well-known that a $\kappa \times \kappa$ grid graph has pathwidth $\Theta(\kappa)$, and we are able to modify a corresponding path decomposition so that each bag only contains a constant number of additional vertices to incorporate the gadgets. For the highway dimension, Abraham et al. [Abr+10] note that in a regular $\kappa \times \kappa$ grid graph a ball of radius $\Theta(\kappa)$ contains $\Theta(\kappa)$ vertical (and horizontal) vertex disjoint paths of length $\Omega(\kappa)$, each of which will need to be hit by a hub. Thus the highway dimension of a $\kappa \times \kappa$ grid graph is $\Omega(\kappa)$. In our constructed graph for the reduction, two gadgets of neighbouring cells of the $\kappa \times \kappa$ grid are connected by only one path. Therefore we can bound the highway dimension from above by placing a hub on each of these connecting paths (i.e., between grid cells), in order to hit long vertical and horizontal paths. Additionally, we are able to place hubs in each gadget such that any ball of a given radius contains only a constant number of them. As there are $\Theta(\kappa^2)$ points where gadgets connect, this implies a highway dimension of $O(\kappa^2)$ for the constructed graph.

References to related work

[Abr+16] I. Abraham, D. Delling, A. Fiat, A. V. Goldberg and R. F. Werneck. 'Highway dimension and provably efficient shortest path algorithms'. *Journal of the ACM* 63.5 (2016), p. 41.

[Abr+11] I. Abraham, D. Delling, A. Fiat, A. Goldberg and R. Werneck. 'VC-dimension and shortest path algorithms'. In: *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*. 2011, pp. 690–699.

[Abr+10] I. Abraham, A. Fiat, A. V. Goldberg and R. F. Werneck. 'Highway dimension, shortest paths, and provably efficient algorithms'. In: *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2010, pp. 782–793.

[AP02] P. K. Agarwal and C. M. Procopiuc. 'Exact and approximation algorithms for clustering'. *Algorithmica* 33.2 (2002), pp. 201–226.

[Alb+02] J. Alber, H. L. Bodlaender, H. Fernau, T. Kloks and R. Niedermeier. 'Fixed parameter algorithms for dominating set and related problems on planar graphs'. *Algorithmica* 33.4 (2002), pp. 461–493.

[Aro98] S. Arora. 'Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems'. *Journal of the ACM* 45.5 (1998), pp. 753–782.

[ARR98] S. Arora, P. Raghavan and S. Rao. 'Approximation Schemes for Euclidean K-medians and Related Problems'. In: *Proceedings of the ACM Symposium on Theory of Computing (STOC)*. ACM, 1998, pp. 106–113.

[Awa+15] P. Awasthi, M. Charikar, R. Krishnaswamy and A. K. Sinop. 'The Hardness of Approximation of Euclidean k-Means'. In: *Proceedings of the 31st International Symposium on Computational Geometry (SOCG)*. 2015, pp. 754–767.

[BG13] Y. Bartal and L.-A. Gottlieb. 'A Linear Time Approximation Scheme for Euclidean TSP'. In: *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*. 2013, pp. 698–706.

[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: *Proceedings of the European Symposium on Algorithms (ESA)*. Vol. 112. 2018, 8:1–8:15.

[Blu19] J. Blum. 'Hierarchy of transportation network parameters and hardness results'. In: *In Proceedings of the International Symposium on Parameterized and Exact Computation (IPEC)*. 2019, 4:1–4:15.

[Blu20] J. Blum. 'W[1]-Hardness of the k-Center Problem Parameterized by the Skeleton Dimension'. In: *In Proceedings of the 26th International Computing and Combinatorics Conference (COCOON)*. Accepted for publication. 2020.

[Bod+15] H. L. Bodlaender, M. Cygan, S. Kratsch and J. Nederlof. 'Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth'. *Inf. Comput.* 243 (2015), pp. 86–111.

[BD97] A. Borchers and D.-Z. Du. 'The k-Steiner Ratio in Graphs'. *SIAM journal on Computing* 26.3 (1997), pp. 857–869.

[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 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2014, pp. 737–756.

[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.

[CL16] Y. Chen and B. Lin. 'The constant inapproximability of the parameterized dominating set problem'. In: *In Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2016, pp. 505–514.

[CHK13] R. Chitnis, M. Hajiaghayi and G. Kortsarz. 'Fixed-Parameter and Approximation Algorithms: A New Look'. In: *In Proceedings of the International Symposium on Parameterized and Exact Computation (IPEC)*. 2013, pp. 110–122.

[CC08] M. Chlebík and J. Chlebíková. 'The Steiner tree problem on graphs: Inapproximability results'. *Theoretical Computer Science* 406.3 (2008), pp. 207–214.

[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.

[Coh+19] V. Cohen-Addad, A. Gupta, A. Kumar, E. Lee and J. Li. 'Tight FPT Approximations for k-Median and k-Means'. In: *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*. Vol. 132. 2019, 42:1–42:14.

[Cyg+15] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.

[Die12] R. Diestel. *Graph Theory, 4th Edition*. Vol. 173. Graduate texts in mathematics. Springer, 2012. ISBN: 978-3-642-14278-9.

[DM18] I. Dinur and P. Manurangsi. 'ETH-Hardness of Approximating 2-CSPs and Directed Steiner Network'. In: *Proceedings of the Innovations in Theoretical Computer Science (ITCS) conference*. 2018, 36:1–36:20.

[DLS14] M. Dom, D. Lokshtanov and S. Saurabh. 'Kernelization Lower Bounds Through Colors and IDs'. *ACM Transactions on Algorithms* 11.2 (Oct. 2014), pp. 1–20.

[DF13] R. G. Downey and M. R. Fellows. *Fundamentals of parameterized complexity*. Vol. 4. Springer, 2013.

[DW71] S. E. Dreyfus and R. A. Wagner. 'The Steiner problem in graphs'. *Networks* 1.3 (1971), pp. 195–207.

[Edm65] J. Edmonds. 'Paths, Trees, and Flowers'. *Canadian journal of Mathematics* 17 (1965), pp. 449–467.

[Eib+19] E. Eiben, D. Knop, F. Panolan and O. Suchý. 'Complexity of the Steiner Network Problem with Respect to the Number of Terminals'. In: *Proceedings of the 36th International Symposium on Theoretical Aspects of Computer Science (STACS)*. 2019, 25:1–25:17.

[FG88] T. Feder and D. Greene. 'Optimal algorithms for approximate clustering'. In: *Proceedings of the ACM Symposium on Theory of Computing (STOC)*. 1988, pp. 434–444.

[FR06] J. Feldman and M. Ruhl. 'The Directed Steiner Network Problem is Tractable for a Constant number of Terminals'. *SIAM J. Comput.* 36.2 (2006), pp. 543–561.

[FG06] J. Flum and M. Grohe. *Parameterized complexity theory*. Springer, 2006.

[FKS19] 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: *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2019, pp. 1069–1088.

[FRS19] Z. Friggstad, M. Rezapour and M. R. Salavatipour. 'Local Search Yields a PTAS for k-Means in Doubling Metrics'. *SIAM J. Comput.* 48.2 (2019), pp. 452–480.

[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.

[GK99] S. Guha and S. Khuller. 'Greedy strikes back: Improved facility location algorithms'. *Journal of algorithms* 31.1 (1999), pp. 228–248.

[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.

[GI03] V. Guruswami and P. Indyk. 'Embeddings and non-approximability of geometric problems.' In: *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Vol. 3. 2003, pp. 537–538.

[HK03] E. Halperin and R. Krauthgamer. 'Polylogarithmic inapproximability'. In: *Proceedings of the ACM Symposium on Theory of Computing (STOC)*. 2003, pp. 585–594.

[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.

[JMS02] K. Jain, M. Mahdian and A. Saberi. 'A new greedy approach for facility location problems'. In: *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. 2002, pp. 731–740.

[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.

[KLP19] 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.

[KR07] S. G. Kolliopoulos and S. Rao. 'A nearly linear-time approximation scheme for the Euclidean k-median problem'. *SIAM Journal on Computing* 37.3 (2007), pp. 757–782.

[Lok+17] D. Lokshtanov, F. Panolan, M. Ramanujan and S. Saurabh. 'Lossy Kernelization'. In: *Proceedings of the ACM Symposium on Theory of Computing (STOC)*. 2017, pp. 224–237.

[MS14] D. Marx and A. Sidiropoulos. 'The limited blessing of low dimensionality: when $1 - 1/d$ is the best possible exponent for d-dimensional geometric problems'. In: *Proceedings of the International Symposium on Computational Geometry (SOCG)*. 2014, p. 67.

[Mar08] D. Marx. 'Parameterized complexity and approximation algorithms'. *The Computer journal* 51.1 (2008), pp. 60–78.

[Mit99] J. S. Mitchell. 'Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems'. *SIAM Journal on computing* 28.4 (1999), pp. 1298–1309.

[Ned09] J. Nederlof. 'Fast Polynomial-Space Algorithms Using Möbius Inversion: Improving on Steiner Tree and Related Problems'. In: *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP)*. 2009, pp. 713–725.

[Ple80] J. Plesnık. 'On the computational complexity of centers locating in a graph'. *Aplikace matematiky* 25.6 (1980), pp. 445–452.

[Tal04] K. Talwar. 'Bypassing the embedding: algorithms for low dimensional metrics'. In: *Proceedings of the ACM Symposium on Theory of Computing (STOC)*. 2004, pp. 281–290.

[Vaz01] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.

[WS11] D. P. Williamson and D. B. Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.