Subexponential Parameterized Directed Steiner Network Problems on Planar Graphs: A Complete Classification

Esther Galby
Chalmers University of Technology, Gothenburg, Sweden

Sándor Kisfaludi-Bak
Department of Computer Science, Aalto University, Finland

Dániel Marx
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany

Roohani Sharma
Department of Informatics, University of Bergen, Norway

Abstract

In the Directed Steiner Network problem, the input is a directed graph G, a set $T \subseteq V(G)$ of k terminals, and a demand graph D on T. The task is to find a subgraph $H \subseteq G$ with the minimum number of edges such that for every $(s,t) \in E(D)$, the solution H contains a directed $s \to t$ path. The goal of this paper is to investigate how the complexity of the problem depends on the demand pattern in *planar graphs*. Formally, if $\mathcal{D}$ is a class of directed graphs, then the $\mathcal{D}$-Steiner Network ($\mathcal{D}$-DSN) problem is the special case where the demand graph D is restricted to be from $\mathcal{D}$. We give a complete characterization of the behavior of every $\mathcal{D}$-DSN problem on planar graphs. We classify every class $\mathcal{D}$ closed under transitive equivalence and identification of vertices into three cases: assuming ETH, either the problem is

1. solvable in time $2^{O(k)} \cdot n^{O(1)}$, i.e., FPT parameterized by the number k of terminals, but not solvable in time $2^{o(k)} \cdot n^{O(1)}$,
2. solvable in time $f(k) \cdot n^{O(\sqrt{k})}$, but cannot be solved in time $f(k) \cdot n^{o(\sqrt{k})}$, or
3. solvable in time $f(k) \cdot n^{O(k)}$, but cannot be solved in time $f(k) \cdot n^{o(k)}$.

Our result is a far-reaching generalization and unification of earlier results on Directed Steiner Tree, Directed Steiner Network, and Strongly Connected Steiner Subgraph on planar graphs. As an important step of our lower bound proof, we discover a rare example of a genuinely planar problem (i.e., described by a planar graph and two sets of vertices) that cannot be solved in time $f(k) \cdot n^{o(k)}$: given two sets of terminals S and T with $|S| + |T| = k$, find a subgraph with minimum number of edges such that every vertex of T is reachable from every vertex of S.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability; Theory of computation → Graph algorithms analysis

Keywords and phrases Directed Steiner Network, Sub-exponential algorithm

Digital Object Identifier 10.4230/LIPIcs.ICALP.2024.67

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2208.06015` [16]

1 Introduction

Finding Steiner trees and related network design problems were intensively studied in undirected graphs, directed graphs, and planar graphs, from the viewpoint of approximation and parameterized algorithms [1–3, 5–8, 11–14, 17–19, 22, 23, 25, 27, 29, 31–33]. The simplest problem of this type is Steiner Tree, where given an undirected graph G and set $T \subseteq V(G)$ of terminals, the task is to find a tree with smallest number of edges that contains every terminal. This problem models a network-design scenario where the terminals need to be connected to each other with a network of minimum cost. Steiner Forest is the generalization where we do not require connection between every pair of terminals, but have to satisfy a given set of demands. Formally, the input of Steiner Forest is a graph G with pairs of vertices $(s_1, t_1), \ldots, (s_d, t_d)$, the task is to find a subgraph with the minimum number of edges that satisfies every request, that is, s_i and t_i are in the same component of the solution for every $i \in [d]$.

On directed graphs, Directed Steiner Tree (DST) is defined by specifying one of the terminals in T to be the root and the task is to find a subgraph with the smallest number of edges such that there is a path from the root to every terminal in the solution. This problem models a scenario where we need to construct a network where the root can broadcast to every other terminal. An equally natural network design problem on directed graphs is the Strongly Connected Steiner Subgraph (SCSS) problem, where given a directed graph G and a set $T \subseteq V(G)$ of terminals, the task is to find a subgraph with the smallest number of edges where T is in a single strongly connected component, or in other words, the solution contains a path from every terminal to every other terminal. The directed variant of Steiner Forest generalizes both of these problems: in Directed Steiner Network (DSN), the input is a digraph G with pairs of vertices $(s_1, t_1), \ldots, (s_d, t_d)$, and the task is to find a subgraph with the minimum number of edges that has an $s_i \to t_i$ path for every $i \in [d]$.

Planar graphs. A well-known phenomenon on planar graphs is that the running time of parameterized algorithms for typical NP-hard problems have exponential dependence on $O(\sqrt{k})$, where k is the parameter, and this dependence is best possible assuming the Exponential-Time Hypothesis (ETH) [8–10, 15, 20, 21, 24, 26–28, 30]. All three of DST, SCSS, and DSN remain NP-hard on planar graphs. However, they behave very differently from the viewpoint of parameterized complexity: the dependence of the running time on the number k of terminals is very different.

Our starting point

1. Planar DST can be solved in time $2^k \cdot n^{O(1)}$ [4], but cannot be solved in time $2^{o(k)} \cdot n^{O(1)}$ [27], assuming the ETH.
2. Planar SCSS can be solved in time $2^{O(k \log k)} \cdot n^{O(\sqrt{k})}$ [8], but has no algorithm with running time $f(k) \cdot n^{o(\sqrt{k})}$ for any function f, assuming the ETH [8].
3. Planar DSN can be solved in time $f(k) \cdot n^{O(k)}$ [12], but has no algorithm with running time $f(k) \cdot n^{o(k)}$ for any function f, assuming the ETH [8].

Using the terminology of parameterized complexity, Planar DST is *fixed-parameter tractable (FPT)* parameterized by the number k of terminals, but does not admit a *subexponential* FPT algorithm, assuming the ETH. For Planar SCSS and Planar DSN, it is already a highly nontrivial result to show that there is an algorithm that runs in polynomial time for fixed values of k; such an algorithm is called an *XP algorithm.* Furthermore, Planar SCSS admits a *subexponential* XP algorithm (i.e., the exponent of n is $o(k)$), while Planar DSN has no such algorithm, assuming the ETH.

As these results show, there has been significant interest in parameterized directed connectivity problems on planar graphs and in particular tight bounds were obtained for the three problems Planar DST, Planar SCSS, Planar DSN. But what can we say about other natural variants of connectivity requirements? For example, already with the simple extension that the input contains two sets of terminals T_1 and T_2, we can define three different natural connectivity requirements:

(1) The solution has to contain a directed path from any $t \in T_i$ to any $t' \in T_i$.
(2) The solution has to contain a directed path from any $t \in T_1$ to any $t' \in T_1 \cup T_2$.
(3) The solution has to contain a directed path from any $t \in T_1$ to any $t' \in T_2$.

Do these problems behave similarly to one of the three problems listed above? The goal of this paper is to answer such questions by putting the previous results into the context of a wider landscape of directed network design problems. We systematically explore other special cases of Planar DSN and determine their behavior on planar graphs. Our main result is showing that every special case defined in a formal setting behaves similarly to one of the three problems Planar DST, Planar SCSS, Planar DSN: assuming the ETH, the best possible running time is of the form $2^{O(k)} \cdot n^{O(1)}$, $f(k) \cdot n^{O(\sqrt{k})}$, or $f(k) \cdot n^{O(k)}$. Furthermore, we provide an exact combinatorial characterization of the problems belonging to the three classes. In particular, we can use these results to show that variants (1) and (2) behave similarly to Planar SCSS, while variant (3) behaves similarly to Planar DSN. Therefore, (3) is a rare (perhaps first) example of a planar problem with k terminals where the best possible running time is $n^{O(k)}$ and the input can be described in a purely planar way (by the two sets T_1 and T_2) that does not contain any extra information violating the planarity of the instance. This is in stark contrast with the general Planar DSN problem, where the lower bound showing the optimality of the $n^{O(k)}$ running time requires that the input contain an arbitary list of pairs of vertices, giving a highly nonplanar input.

Dichotomy for general graphs. We explore the different special cases of Directed Steiner Network on planar graphs in a framework similar to how Feldmann and Marx [14] treated the problem on general graphs. We can define various special cases of Directed Steiner Network by looking at what kind of graph the connection demands define on the terminals: it is an out-star for Directed Steiner Tree, a bidirected clique for Strongly Connected Steiner Subgraph, and a matching for Directed Steiner Network. More generally, for every class $\mathcal{D}$ of directed graphs, we investigate the problem where the pattern of demands has to belong to the class $\mathcal{D}$. Our goal is to understand how the graph-theoretic properties of the members of $\mathcal{D}$ influence the resulting special case of Directed Steiner Network.

Formally, for every class $\mathcal{D}$, Feldmann and Marx [14] defined the restriction of the problem in the following way.

$\mathcal{D}$-Steiner Network
Input: Digraph G, a set of k terminals $T \subseteq V(G)$, and a demand digraph $D \in \mathcal{D}$ with vertex set T.
Question: What is the minimum number of edges in a subgraph H of G where for each $(u, v) \in E(D)$ there is a $u \to v$ path in H?

One can define also the weighted version of the problem: the input contains weights on the edges and the goal is to minimize the total weight of the subgraph H. Typically, the weighted generalization does not make the problem harder (polynomially bounded integer weights can be easily simulated by subdivided edges, but the algorithmic results in this paper and earlier work allows integer weights in binary as well). Feldmann and Marx [14] characterized those classes $\mathcal{D}$ where $\mathcal{D}$-Steiner Network is fixed-parameter tractable (FPT) parameterized by the number of terminals, that is, can be solved in time $f(k) \cdot n^{O(1)}$. The characterization can be stated in a clean way in terms of five hard families of patterns if we observe the following closure properties of the problem. Observe first that only the transitive closure of D matters for the problem: if D_1 and D_2 have the same transitive closure, then having D_1 or D_2 in the input results in exactly the same problem. Therefore, it makes sense to consider only classes $\mathcal{D}$ that are *closed under transitive equivalence,* that is, if D_1 and D_2 have the same transitive closure and $D_1 \in \mathcal{D}$, then $D_2 \in \mathcal{D}$ as well. Moreover, we may assume that $\mathcal{D}$ is *closed under identifying vertices:* that is, if $G \in \mathcal{D}$ and G' is obtained by merging two vertices $x, y \in V(G)$ to a single vertex whose in- and out-neighbors are the union of the in- and out-neighbors of x and y, respectively, then G' is also in $\mathcal{D}$. Feldmann and Marx [14, Lemma 5.2] showed that if $\mathcal{D}$-Steiner Network is FPT, then it is FPT also for the closure $\mathcal{D}'$ of $\mathcal{D}$ under identifying vertices, that is, adding further demand patterns obtained by identifying vertices does not make the problem any harder. Intuitively, if D' is obtained from $D \in \mathcal{D}$ by identifying x and y to w, then an instance with demand pattern D' can be simulated by an instance with demand pattern D if we replace w with the two terminals x and y connected by 0-weight edges in both direction (or something similar in case of unweighted graphs[1]).

These arguments show that it is sufficient to obtain a characterization for classes closed under transitive equivalence and identifying vertices. Under these assumptions, Feldmann and Marx [14] identified five graph classes that prevent the problem from being FPT. A *pure out-diamond* is a complete bipartite graph $K_{2,t}$ directed from the 2-element side to the t-element side. A *flawed out-diamond* has in addition a vertex v and edges going from v to the 2-element side. The pure in-diamond and flawed in-diamond are defined similarly by reversing edge orientations. Let us denote by $\mathcal{A}_1$, $\mathcal{A}_2$, ..., $\mathcal{A}_5$ the class of all pure out-diamonds, flawed out-diamonds, pure in-diamonds, flawed in-diamonds, and directed cycles, respectively.

▶ **Theorem 1** (Feldmann and Marx [14]). *Let $\mathcal{D}$ be a class of graphs closed under transitive equivalence and identifying vertices.*

1. ***FPT:*** *If $\mathcal{A}_i \not\subseteq \mathcal{D}$ for any $i \in [5]$, then $\mathcal{D}$-*Steiner Network *can be solved in time $2^{O(k)} n^{O(1)}$, where k is the number of terminals.*
2. ***Hard:*** *If $\mathcal{A}_i \subseteq \mathcal{D}$ for some $i \in [5]$, then $\mathcal{D}$-*Steiner Network *is W[1]-hard parameterized by the number k of terminals.*

The first part of Theorem 1 was proved by a combination of an algorithm that solves the problem in time $2^{O(kw \log w)} \cdot n^{O(w)}$ if there is an optimum solution with treewidth w and a combinatorial result showing that if $\mathcal{D}$ is not the superset of $\mathcal{A}_i$ for any $i \in [5]$, then there is a constant bound on the treewidth of optimum solutions. The second part follows from a W[1]-hardness result for each of the five classes $\mathcal{A}_i$.

Our result: trichotomy for planar graphs. Our main result classifies Planar $\mathcal{D}$-Steiner Network (the special case of the problem restricted to planar digraphs G) into three levels of complexity: $2^{O(k)} \cdot n^{O(1)}$, $f(k) \cdot n^{O(\sqrt{k})}$, or $f(k) \cdot n^{O(k)}$ time. In light of Theorem 1 and the earlier results on planar graphs, there are three natural questions that arise:

1. Are there cases that are FPT on planar graphs, but W[1]-hard on general graphs?
2. Are there subexponential FPT cases on planar graphs, that is, where the running time is $2^{o(k)} \cdot n^{O(1)}$?
3. Are there W[1]-hard cases where the optimal running time is neither $f(k) \cdot n^{O(\sqrt{k})}$ nor $f(k) \cdot n^{O(k)}$? If not, where is the boundary line between these two cases?

[1] As mentioned above, polynomially bounded integer weights can be simulated by subdivision of edges. If there are C edges of weight 0, then let us consider every original weight-1 edge to have weight $C + 1$ (i.e., a path of length $C + 1$) and every weight-0 to have weight 1. Then the original instance has a solution of weight at most x if and only if the new instance has a solution of weight at most $x(C + 1) + C$.

We answer the first question negatively: the hard cases remain hard on planar graphs. The answer to the second question is also negative: we show that every (nontrivial) case of PLANAR $\mathcal{D}$-DSN is at least as hard as DIRECTED STEINER TREE, hence a known lower bound [27] shows that there is no subexponential FPT algorithm, assuming the ETH.

The answer to the third question is positive, but arguably for the wrong reason. Consider the following artificial case. Let $\mathcal{D} = \{D_1, D_2, \dots\}$ be defined the following way: D_i consists of the disjoint union of an out-star of i edges and a directed matching of $\lceil \log_2 i \rceil$ edges. Using the results of Feldmann and Marx [14, Theorems 1.4 and 1.5], it can be shown that PLANAR $\mathcal{D}$-DSN is solvable in time $f(k)n^{O(\log k)}$, as the graph D_i can be interpreted as a "1-caterpillar with $\lceil \log_2 i \rceil$ extra edges" (see the definition in [14]) and hence there is an optimum solution of treewidth $O(\log k)$. On the other hand, a PLANAR DSN problem with $\lceil \log_2 i \rceil$ terminal pairs can be reduced to PLANAR $\mathcal{D}$-DSN with pattern D_i: we can effectively "ignore" the terminals in the out-star by putting these terminals on a directed cycle of 0-weight edges. Thus the lower bound ruling out $f(k)n^{o(k)}$ algorithms for PLANAR DSN [8] translates into a lower bound ruling out $f(k)n^{o(\log k)}$ algorithms for PLANAR $\mathcal{D}$-DSN, assuming the ETH. Therefore, the optimal exponent of n in the running time is $O(\log k)$. This is somewhat counterintuitive: as there is a simple reduction from PLANAR DSN to PLANAR $\mathcal{D}$-DSN, it feels that the latter problem should be harder. Formally, however, this is not the case: the reduction introduced a new, irrelevant, trivial part of the problem (the terminals on the cycle of 0-weight edges), which increased the parameter significantly.

One could argue that such trivial features of the instance should not influence the way we measure the complexity of the problem. Terminals that are in the same strongly connected component of 0-weight edges can be effectively treated as a single terminal. Therefore, instead of the parameter k being the number of terminals, one could consider the parameter to be the number of strongly connected components of the 0-weight edges that contain terminals, or in other words, the number of terminals after contracting every directed cycle of weight 0. With this parameterization, the reduction from PLANAR DSN increases the parameter only by 1 and shows that PLANAR $\mathcal{D}$-DSN has no $f(k)n^{o(k)}$ time algorithm, satisfying the expectation that the problem should be at least as hard as PLANAR DSN. Equivalently, we can consider the closure $\mathcal{D}'$ of $\mathcal{D}$ under identifying vertices: then $\mathcal{D}'$ contains every directed matching, hence the problem is clearly at least as hard as PLANAR SCSS. Assuming that the pattern class $\mathcal{D}$ is closed under identifying vertices is a clean way of formalizing the intention that we want to consider terminals in a strongly connected component of weight-0 edges as a single terminal that contributes only 1 to the parameter. In order to obtain meaningful classifications, we assume in the rest of the paper that the class of patterns has this closure property. This way, we avoid pathological examples similar to the one described above.

Under the assumption that $\mathcal{D}$ is closed under transitive equivalence and identifying vertices, we can answer the third question in the negative and map the boundary line between the $f(k) \cdot n^{O(\sqrt{k})}$ and the $f(k) \cdot n^{O(k)}$ cases. We define a finite number $\kappa \leq 300000$ of classes $\mathcal{C}_i$, $i \in [\kappa]$, and show that these are precisely the classes of patterns that prevent subexponential $f(k) \cdot n^{O(\sqrt{k})}$ time algorithms. Note that for every $i \in [\kappa]$, it is easy to show that arbitrary large strongly connected graphs can be obtained from $\mathcal{C}_i$ by identifying vertices. That is, if $\mathcal{D}$ is closed under transitive equivalence and identifying vertices, and $\mathcal{C}_i \subseteq \mathcal{D}$, then $\mathcal{A}_5 \subseteq \mathcal{C}_i \subseteq \mathcal{D}$, i.e., $\mathcal{D}$ contains every directed cycle. Thus we have three different cases depending on whether $\mathcal{D}$ contains (1) none of the $\mathcal{A}_i$'s, (2) some $\mathcal{A}_i$, but no $\mathcal{C}_i$, or (3) some $\mathcal{C}_i$.

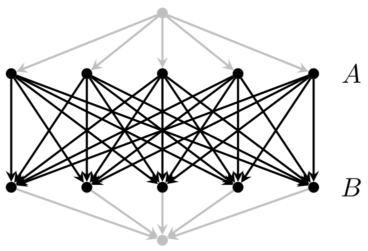

Figure 1 The 5-hard biclique patterns: each gray vertex may or may not be present.

Our main result

▶ **Theorem 2.** *Let $\mathcal{D}$ be a class of directed graphs closed under transitive equivalence and identifying vertices where the number of edges is not bounded.*

1. ***FPT:*** *If $\mathcal{A}_i \not\subseteq \mathcal{D}$ for any $i \in [5]$, then* Planar $\mathcal{D}$-Steiner Network
   - **(i)** *can be solved in time $2^{O(k)} \cdot n^{O(1)}$,*
   - **(ii)** *but has no $2^{o(k)} \cdot n^{O(1)}$ time algorithm assuming the ETH.*
2. ***Subexponential XP:*** *If $\mathcal{A}_i \subseteq \mathcal{D}$ for some $i \in [5]$, but $\mathcal{C}_i \not\subseteq \mathcal{D}$ for any $i \in [\kappa]$, then* Planar $\mathcal{D}$-Steiner Network
   - **(iii)** *can be solved in time $f(k) \cdot n^{O(\sqrt{k})}$,*
   - **(iv)** *but has no $f(k) \cdot n^{o(\sqrt{k})}$ time algorithm assuming the ETH.*
3. ***Hard XP:*** *If $\mathcal{C}_i \subseteq \mathcal{D}$ for some $i \in [\kappa]$, then* Planar $\mathcal{D}$-Steiner Network
   - **(v)** *can be solved in time $f(k) \cdot n^{O(k)}$,*
   - **(vi)** *but has no $f(k) \cdot n^{o(k)}$ time algorithm assuming the ETH.*

We remark that the algorithms work also for weighted graphs, while the lower bounds hold already for unweighted graphs.

Hard classes. Let us define now the graph classes $\mathcal{C}_i$ representing the hard-patterns. Given a digraph G and a set $X \subseteq V(G)$, an X*-source* is a vertex $s \in V(G) \setminus X$ such that $N^+(s) = X$. Similarly, an X*-sink* is a vertex $t \in V(G) \setminus X$ such that $N^-(t) = X$. The first 4 classes $\mathcal{C}_1$, ..., $\mathcal{C}_4$ are defined by extending a biclique.

▶ **Definition 3** (t-hard-biclique-pattern)**.** *A* t-hard-biclique-pattern *is an (acyclic) digraph D constructed in the following way. We start with two disjoint sets A and B with $|A| = |B| = t$ and introduce every edge from A to B. Furthermore, we introduce into D any combination of the following items (see Figure 1):*

1. *an A-source;*
2. *a B-sink.*

In particular, there are $2 \cdot 2$ types of t-hard-biclique patterns: we let $\mathcal{C}_1, \ldots, \mathcal{C}_4$ be the 4 classes that each contain all the t-hard-biclique-patterns of a specific type for every t.

The following definition specifies the remaining classes.

▶ **Definition 4** (t-hard-matching-pattern)**.** *A* t-hard-matching-pattern *is an (acyclic) digraph D constructed the following way. We start with disjoint vertex sets $W = \{w_1, \ldots, w_t\}$, $X = \{x_1, \ldots, x_t\}$, $Y = \{y_1, \ldots, y_t\}$ and $Z = \{z_1, \ldots, z_t\}$ and introduce the edges w_ix_i and y_iz_i for every $i \in [t]$. Furthermore, we introduce into D any combination of the following items:*

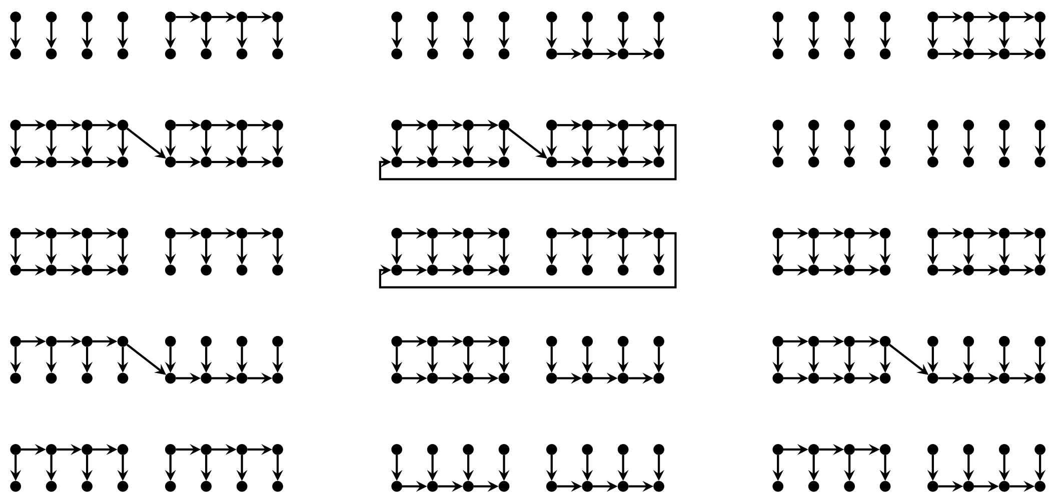

Figure 2 The 4-hard matching patterns (without source, sink, r_{WZ}, or r_{YX}).

1. *either the directed path* $w_1 \to w_2 \to \ldots \to w_t \to z_1 \to z_2 \to \ldots \to z_t$, *or any of the directed paths* $w_1 \to w_2 \to \ldots \to w_t$ *and* $z_1 \to z_2 \to \ldots \to z_t$;
2. *either the directed path* $y_1 \to y_2 \to \ldots \to y_t \to x_1 \to x_2 \to \ldots \to x_t$, *or any of the directed paths* $x_1 \to x_2 \to \ldots \to x_t$ *and* $y_1 \to y_2 \to \ldots \to y_t$;
3. *a vertex* s *such that for exactly one* $S \in \{W, X, Y, Z, W \cup Y, X \cup Z, X \cup Y, W \cup Z\}$, $N^+(s) \cap (W \cup X \cup Y \cup Z) = S$;
4. *a vertex* t *such that for exactly one* $S \in \{W, X, Y, Z, W \cup Y, X \cup Z, X \cup Y, W \cup Z\}$, $N^-(t) \cap (W \cup X \cup Y \cup Z) = S$;
5. *a vertex* r_{WZ} *such that* $N^-(r_{WZ}) \setminus \{s\} = W$ *and* $N^+(r_{WZ}) \setminus \{t\} = Z$;
6. *a vertex* r_{YX} *such that* $N^-(r_{YX}) \setminus \{s\} = Y$ *and* $N^+(r_{YX}) \setminus \{t\} = X$;
7. *arc* $s \to r_{WZ}$ *if* $N^+(s) \cap W = \emptyset$, *or arc* $s \to r_{YX}$ *if* $N^+(s) \cap Y = \emptyset$, *or both if* $N^+(s) \cap (W \cup Y) = \emptyset$;
8. *arc* $r_{WZ} \to t$ *if* $N^-(t) \cap Z = \emptyset$, *or arc* $r_{YX} \to t$ *if* $N^-(t) \cap X = \emptyset$, *or both if* $N^-(t) \cap (Z \cup X) = \emptyset$
9. *arc* $s \to t$ *if* s *cannot already reach* t.

In particular, there are less than $5 \cdot 5 \cdot 9 \cdot 9 \cdot 2 \cdot 2 \cdot 4 \cdot 4 \cdot 2$ *types of* t*-hard matching patterns: we let* $\mathcal{C}_5, \ldots, \mathcal{C}_\kappa$ *, where* $\kappa \leq 259200$*, be the classes that each contain all the* t*-hard-matching-patterns of a specific type for every* t*.*

Note that some of these classes are isomorphic. For example, adding the path $x_1 \to x_t$ or the path $z_1 \to z_t$ lead to isomorphic graphs. If we just consider the graph classes where we choose not to add a source, sink, vertex r_{WZ}, or vertex r_{YX}, then we have 15 non-isomorphic classes, as shown in Figure 2. One could think of t-hard-matching-patterns as (the transitive closure of) one of these graphs, potentially extended by appropriate sources, sinks, and r_{WZ}/r_{YX} vertices.

Finally, we define a t-hard pattern as any of the patterns defined above.

▶ **Definition 5** (t-hard-pattern). *A* t-hard-pattern *is either a* t*-hard-biclique-pattern or* t*-hard-matching-pattern, that is, a pattern that belongs to one of the* κ *classes* $\mathcal{C}_1, \ldots, \mathcal{C}_\kappa$ *defined in Definitions 3 and 4.*

1.1 Overview of our main result

Observe that Theorem 2 consists of six statements. Let us briefly discuss how these six statements are proved. Note that some of these statements follow from known results, while for others we need to do a substantial amount of new technical work. The proofs of statements (iii) and (vi) form the main technical part of the paper (see Figure 4).

The main problem studied in this paper, obtaining tight upper and lower bounds for different families of network design problems, is a genuine computer science question. Due to the nature of the question we are asking, our results are at the intersection of computational complexity, algorithms, and combinatorics (graph theory). The lower bounds are obtained using the standard method of computational complexity: by reductions from problems for which (conditional) lower bounds were already established. In particular, we are using the known lower bounds for Planar DSN and Planar SCSS on planar graphs [8]. However, the majority of the hardness proofs we present contain significant new ideas, new gadget constructions, and use new non-obvious global structures when connecting the gadgets.

The upper bounds are obtained using an algorithm of Feldmann and Marx [14] (see Theorem 7 below) solving the problem in time depending on the treewidth of an optimum solution. Therefore, in this paper the main technical effort is spent on the combinatorial question of bounding the treewidth of optimum solutions. Our proof uses planarity in a geometric way (arguing about faces, invoking Sperner's Lemma, etc.) and hence completely different from earlier proofs that relied only on bidimensionality of planar graphs [8]. The treewidth-based algorithm offers an abstraction that allows us to treat the upper bounds in a clean, modular manner. Thus we can focus our efforts on understanding the patterns that allow faster solutions, without having to develop details of algorithmic steps.

Furthermore, we have the purely combinatorial task of connecting the obstructions that prevent the treewidth upper bound and the hard structures. In the full version [16], we establish this connection with a heavy use of case analysis and Ramsey-theoretical arguments. The type and amount of combinatorial effort is very different from what was needed in earlier work on general graphs [14], where more elementary arguments were sufficient.

Statement (i): FPT algorithms

The FPT result (i) follows directly from Theorem 1 (here the surprising aspect is that, by statement (iv), there are no further FPT cases).

Statement (ii): no subexponential FPT algorithms

The lower bound (ii) follows by observing that every relevant class contains either all in-stars or all out-stars, hence the lower bound for Directed Steiner Tree [27] applies. To avoid triviality, we need to assume that the class contains graphs with arbitrarily large number of edges.

▶ **Lemma 6.** *Let $\mathcal{D}$ be a class of graphs closed under identifying vertices and transitive closure where the number of edges of the graphs is not bounded. Then one of the following holds:*

- *$\mathcal{D}$ contains every directed cycle,*
- *$\mathcal{D}$ contains every out-star, or*
- *$\mathcal{D}$ contains every in-star.*

In statement (ii) of Theorem 2, we assume that $\mathcal{A}_i \not\subseteq \mathcal{D}$, and $\mathcal{A}_5$ is the class of all directed cycles. Thus $\mathcal{D}$ contains either every out-star or every in-star.

Statement (iii): $f(k)n^{O(\sqrt{k})}$ algorithms

Our main technical result is proving statement (iii): the existence of an $f(k) \cdot n^{O(\sqrt{k})}$ time algorithm if $\mathcal{C}_i \not\subseteq \mathcal{D}$ for any $i \in [\kappa]$ (in the following subsection, we give a more detailed description of the proof). This algorithm is obtained by showing that the treewidth of the optimal solution is always $O(\sqrt{k})$ under these conditions. Then we can use the following result of Feldmann and Marx [14].

▶ **Theorem 7** (Theorem 1.5 of [14]). *If an instance (G, T, D) of* Directed Steiner Network *has an optimum solution H of treewidth w, then it can be solved in $2^{O(kw \log w)} \cdot n^{O(w)}$ time.*

Note that this is a slightly weaker form of the statement, with a simplified bound on the running time. With Theorem 7 at hand, our main goal is to prove that every optimum solution of Planar $\mathcal{D}$-DSN has treewidth $O(\sqrt{k})$ if $\mathcal{C}_i \not\subseteq \mathcal{D}$ for any $i \in [\kappa]$.

Towards proving this bound, we first translate the question to a problem on acyclic graphs: it is sufficient to show that if the solution is acyclic, then the total degree of the branch vertices (i.e., of degree > 2) is $O(k)$. More formally, for a vertex v of a digraph, let $d^*(v)$ denote the *branch degree* of v, defined as

$$d^*(v) = \max(d^+(v) + d^-(v) - 2, 0),$$

where $d^+(v)$ and $d^-(v)$ denotes the out- and in-degree of v, respectively. The total branch degree of a graph G is the sum of all branch degrees of the vertices of G.

We say that a feasible solution H of (G, T, D) is *edge-minimal* if for all edges $e \in E(H)$ the graph $H - e$ is not feasible. An edge e is *essential* for some demand edge $(t, t') \in E(D)$ if there is no $t \to t'$ path in $H - e$. Note that all edges of an edge-minimal graph H are essential for some demand edge of D. We say that a pattern class $\mathcal{D}$ is *c-acyclic-bounded* for some $c = O(1)$ if for any instance (G, T, D) of Planar $\mathcal{D}$-Steiner Network where G, D are acyclic, and any edge-minimal solution H, the total branch degree of H is at most $c|T|$.

The next theorem moves the problem to the domain of acyclic digraphs: what we need now is a linear bound on the total branch degree of acyclic solutions.

▶ **Theorem 8.** *If the pattern class $\mathcal{D}$ is c-acyclic-bounded for some $c = O(1)$, then for any instance of* Planar $\mathcal{D}$-Steiner Network *with $|T| = k$, the solution graph H has treewidth $O(\sqrt{k})$.*

Applying Theorem 7 implies that c-acyclic-bounded classes have the desired subexponential algorithm, but we still need to establish a link between non-c-acyclic-bounded classes and t-hard-patterns. First, we argue that if the total branch degree is too large, then a grid-like structure can be found in the solution. The grid-like structure appears in the solution to satisfy a set of edges in the demand graph D, and this set of demands forms a certain hard structure in the demand pattern that we call a t-tough-pair which we define informally here (see Definition 14 for a formal definition). We say that two edges e_1 and e_2 are *weakly independent* if there is no directed path from the head of one to the tail of the other. Edges e_1 and e_2 are *strongly independent* if, in addition to being weakly independent, there is no directed path containing the heads of both edges and there is no directed path containing the tails of both edges. An edge e is *minimal* in a digraph D if there is no path from the tail of e to the head of e avoiding e. Let $E_1 \cup E_2$ be a vertex-disjoint set of minimal edges with $|E_1| = |E_2| = t$. We say that (E_1, E_2) is a *t-tough-pair* if

- any two edges $e, e' \in E_1$ are weakly independent,
- any two edges $e, e' \in E_2$ are weakly independent, and
- any two edges $e_1 \in E_1$ and $e_2 \in E_2$ are strongly independent.

Observe that in particular the two matchings in a t-hard-matching-pattern form (vertical edges in Figure 2) a t-tough-pair. Similarly, taking two vertex-disjoint matchings of size t each in a $2t$-hard-biclique-pattern is also a t-tough-pair.

Our main structure theorem connects the total branch degree to the existence of these kind of hard structures.

▶ **Theorem 9** (Structure Theorem). *Let $\mathcal{D}$ be a class of graphs closed under identifying vertices and transitive equivalence. Then either $\mathcal{D}$ has a pattern with a t-tough-pair for each positive integer t, or it is c-acyclic-bounded for some constant c.*

Theorems 8 and Theorem 9 show that the existence of arbitrarily large t-tough-pairs is the canonical reason why the treewidth is not $O(\sqrt{k})$. The lower bounds ruling out $f(k) \cdot n^{o(k)}$ time algorithms essentially rely on the existence of t-tough-pairs. However, the existence of a t-tough-pair in a demand pattern $D \in \mathcal{D}$ is not sufficient for the lower bound: the t-tough-pair could be only a small part of the pattern D, and hence the lower bounds may not apply. We show, with heavy use of Ramsey's Theorem and other combinatorial arguments, that whenever a large t-tough-pair appears in a graph, then the graph can be "cleaned": we can identify vertices to obtain one of the t-hard-patterns. Therefore, if arbitrary large t-tough-pairs appear in the members of a class $\mathcal{D}$ closed under identifying vertices, then the class is a superset of one of the hard classes $\mathcal{C}_i$.

▶ **Theorem 10.** *Let $\mathcal{D}$ be a class of graphs closed under transitive equivalence and identifying vertices. The following two are equivalent:*

1. *For every t, there is a $D \in \mathcal{D}$ that has a t-tough pair.*
2. *$\mathcal{C}_i \subseteq \mathcal{D}$ for some $i \in [\kappa]$.*

We can conclude that if $\mathcal{D}$ is not the superset of $\mathcal{C}_i$ for any $i \in [\kappa]$, then the treewidth of the optimum solution is $O(\sqrt{k})$, implying that Planar $\mathcal{D}$-DSN can be solved in time $f(k) \cdot n^{O(\sqrt{k})}$.

Statement (iv): no $f(k)n^{o(\sqrt{k})}$ algorithms

If $\mathcal{D}$ contains $\mathcal{A}_5$ (directed cycles), then lower bounds ruling out $f(k) \cdot n^{o(\sqrt{k})}$ time algorithms follow from the known lower bound for Strongly Connected Steiner Subgraph [8]. When $\mathcal{D}$ contains one of $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3, \mathcal{A}_4$ (pure or flawed dimonds), the problem is known to be W[1]-hard on general graphs [14]. We reprove the hardness of diamonds, this time restricted to planar graphs, and observe that this W[1]-hardness proof actually rules out $f(k) \cdot n^{o(\sqrt{k})}$ time algorithms. Compared to the W[1]-hardness on general graphs, the proof for planar graphs is more involved. As it is very usual for planar problems, we establish these lower bounds by reducing from $k \times k$-Grid Tiling, which cannot be solved in time $f(k) \cdot n^{o(k)}$, assuming the ETH [9]. For statement (iv), we need to reduce from $\sqrt{k} \times \sqrt{k}$ Grid Tiling to a Planar $\mathcal{D}$-DSN with $O(k)$ terminals forming a pure/flawed in/out-diamond pattern, ruling out $f(k) \cdot n^{o(\sqrt{k})}$ algorithms for such patterns.

In all these reductions, we are reusing and extending the gadget constructions from earlier work [8]. However, the high-level structure of the reduction is substantially different and depends on the pattern class we are considering. In light of Theorem 7, we should first verify, as a sanity check, that the treewidth of the solution can be sufficiently large, that is, it can be $\Omega(\sqrt{k})$ in case of diamonds. Typically, one can expect that examples with sufficiently large treewidth shed some light on how the high-level structure of the hardness proof could look like. Figure 3 shows that treewidth can be indeed sufficiently large: a $\sqrt{k} \times \sqrt{k}$ grid can be obtained from two "interlocking combs."

Statement (v): $f(k)n^{O(k)}$ algorithms

The upper bound $f(k) \cdot n^{O(k)}$ (statement (v)) follows from the work of Eiben et al. [12], who showed that PLANAR DSN with k terminals can always be solved within this running time. Note that Feldman and Ruhl [13] presented a $n^{O(p)}$ time algorithm for DSN on general graphs where p is the number of demands. However, as the number of demands on k terminals can be $\Omega(k^2)$, their algorithm *does not* give an $f(k) \cdot n^{O(k)}$ algorithm where k is the number of terminals.

Statement (vi): no $f(k)n^{o(k)}$ algorithms

To prove statement (vi) ruling out $f(k) \cdot n^{o(k)}$ algorithms, we provide such a lower bound for each class $\mathcal{C}_i$ for $i \in [\kappa]$. Analogously to statement (iv), the proof is by reduction from $k \times k$ GRID TILING to a PLANAR $\mathcal{D}$-DSN instance with a k-hard-matching-pattern or a k-hard-biclique-pattern, ruling out $f(k) \cdot n^{o(k)}$ algorithms. Again, let us verify that the treewidth can be sufficiently large: Figure 3 shows how a $k \times k$ grid can appear in the solution to an instance with k terminals.

For t-hard-matching-patterns, the simplest case is when we have two induced matchings of size t. Then a $t \times t$ grid can arise very easily in the solution if the terminals are on the boundary of a grid. The crucial point here is that the t-hard-matching-pattern was defined in a way that all the additional paths, sources etc. do not interfere with the grid, see the figure for an example. For the t-hard-biclique-pattern, there is a non-obvious and highly delicate way of constructing an instance with $2t$ terminals where a $t \times t$ grid appears. Combining these constructions gives the lower bound.

▶ **Theorem 11.** *Let $\mathcal{D}$ be a class of graphs closed under identifying vertices and transitive equivalence. If $\mathcal{C}_i \subseteq \mathcal{D}$ for some $i \in [\kappa]$, then* PLANAR $\mathcal{D}$-STEINER NETWORK *has no $f(k) \cdot n^{o(k)}$ time algorithm assuming the ETH.*

Let us observe that if $\mathcal{D}$ consists of bicliques directed from one side to the other, then PLANAR $\mathcal{D}$-DSN corresponds to the following problem: given a planar digraph G with two sets $S, T \subseteq V(G)$ of terminals with $|S| + |T| = k$, find a subgraph with minimum number of edges such that there is a path from every vertex of S to every vertex of T. Our result shows that, assuming the ETH, this problem has no $f(k) \cdot n^{o(k)}$ time algorithm. This result is surprising, as the problem can be considered to be *genuinely planar* in the sense that the input is a planar graph with k terminals and a single bit of annotation at each terminal (and there is no extra information, such as terminal pairs, that can disregard the planarity of the instance). To our knowledge, this is the first example of a relatively natural planar problem where $f(k) \cdot n^{O(k)}$ is best possible and cannot be improved to $f(k) \cdot n^{O(\sqrt{k})}$.

1.2 Details of Statement (iii): the $f(k) \cdot n^{O(\sqrt{k})}$ algorithm

In this section, we give a more detailed overview of the technical steps of the proof of (iii) sketched above.

From treewidth to total branch degree. Theorem 8 translates the question about the treewidth of the solution in general graphs to a question about the total branch degree of the solution in acyclic graphs. Suppose that we have an edge-minimal solution H in a (not necessarily acyclic) graph G with k terminals. Let us contract the strongly connected

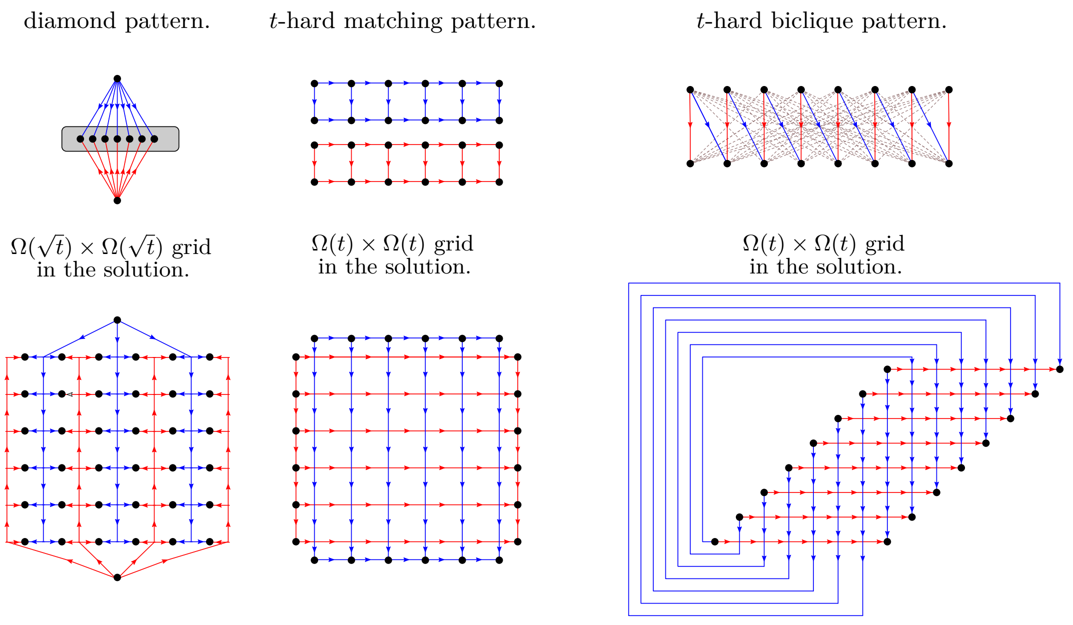
diamond pattern.
t-hard matching pattern.
t-hard biclique pattern.
$\Omega(\sqrt{t}) \times \Omega(\sqrt{t})$ grid in the solution.
$\Omega(t) \times \Omega(t)$ grid in the solution.
$\Omega(t) \times \Omega(t)$ grid in the solution.

Figure 3 Pattern graphs (top row) and example minimal solution graphs with large grid patterns and large treewidth (bottom row). The red/blue edges show how (some of the) demands are connected in the solution.

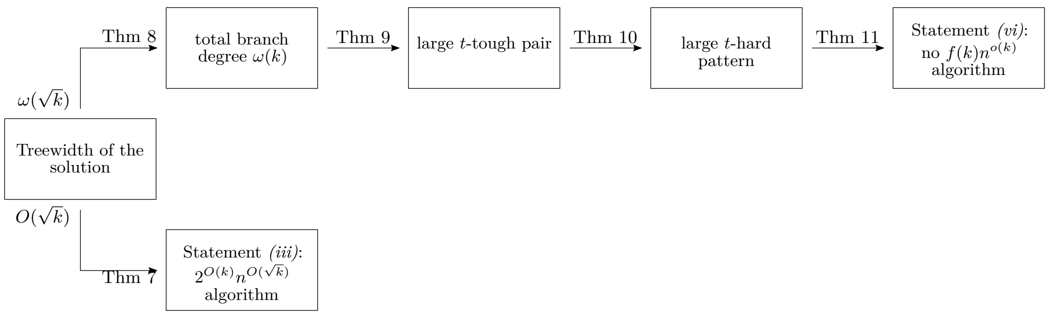
Treewidth of the solution
$\omega(\sqrt{k})$
Thm 8
total branch degree $\omega(k)$
Thm 9
large t-tough pair
Thm 10
large t-hard pattern
Thm 11
Statement *(vi)*: no $f(k)n^{o(k)}$ algorithm
$O(\sqrt{k})$
Thm 7
Statement *(iii)*: $2^{O(k)}n^{O(\sqrt{k})}$ algorithm

Figure 4 The structure of the proofs of statements (iii) and (vi).

components of H in both G and H to obtain G' and H', respectively. We can observe that H' is an acyclic graph that is the optimum solution to an instance in G' with at most k terminals. Our goal is to show that if H' has total branch degree d, then H has treewidth $O(\sqrt{d+k})$. Therefore, in the later steps of the proof, we bound the total branch degree of H' by $O(k)$, giving an $O(\sqrt{k})$ bound on the treewidth of H.

We say that a vertex of a strongly connected component of H is a *portal* if it is incident to an edge connecting it to some other component. For simplicity of discussion, let us assume here that every strongly connected component of H has at least 3 edges incident to the portals, that is, every vertex of H' has at least 3 incident edges. (If a component has less than 3 such edges and has no terminal, then it consists only of a single vertex and does not affect treewidth anyway; if it has terminals, then it can be taken into account with additional calculations.) By this assumption, the set P of portals has size at most $6d$, where d is the total branch degree of H'.

We want to bound the treewidth of H by showing that there is a set W of $O(d+k)$ vertices such that $H - W$ has treewidth at most 2. It is known that if removing a set W of vertices from a *planar* graph reduces treewidth to a constant, then the planar graph has treewidth $O(\sqrt{|W|})$. Thus the treewidth bound $O(\sqrt{d+k})$ follows from the existence of such a set W.

Let $H[V_i]$ be a strongly connected component of H that has p_i portals and contains k_i terminals. The key observation is that the only role of $H[V_i]$ in the solution is to fully connect the terminals and portals in $H[V_i]$. That is, we can assume that $H[V_i]$ is an optimum solution of a Strongly Connected Steiner Subgraph instance with $p_i + k_i$ terminals. Chitnis et al. [8] showed that we can remove a set W_i of $O(p_i + k_i)$ vertices from such an optimum solution to reduce its treewidth to 2. Therefore, taking the union of P and every W_i, we get a set W of size $O(d) + O(\sum(p_i + k_i)) = O(d + k)$ whose removal reduces treewidth to 2 (as removing P breaks the graph in a way that each component is a subset of some V_i, and the removal of W_i breaks $H[V_i]$ into components of treewidth at most 2).

Building a skeleton. Towards the proof of Theorem 9, our goal is to bound the total branch degree by $O(k)$ in an edge-minimal acyclic solution H. At some step of the proof, it will be important to assume that H is a triangulated planar graph (every face has exactly three vertices and edges), which is of course not true in general. Therefore, we introduce artificial undirected edges in the graph H to make it triangulated. As these edges do not play any role in the directed problem, it does not change the nature of the solution. Another simplification step is that we assume that there is no vertex $v \notin T$ with $d^-(v) = d^+(v) = 1$. Such a vertex has branch degree 0 and hence suppressing it (i.e., removing it and adding an edge from its in-neighbor to its out-neighbor) has no effect on the total branch degree and on the connectivity of the terminals.

We start by building a *skeleton* of the solution: a connected subgraph that contains every terminal. The skeleton is composed from *segments* of two types. A *long segment* is a directed path of H of length at least some constant L. A *short segment* is any path in the undirected sense of length at most L, possibly containing both undirected or directed edges of any orientation. Furthermore, we require that any two long segments in the skeleton are *distant,* that is, have distance at least L in the undirected sense.

A skeleton tree consisting of $O(k)$ segments and containing all the terminals can be built the following way. Initially, we start with an edgeless subgraph R containing only the k terminals. For simplicity of discussion, let us assume that the demand pattern is connected (in the undirected sense). Then there has to be a demand $t_i t_j$ such that t_i and t_j are in two different components C_i and C_j of R, respectively. This means that H has a directed path P connecting two different components of R. If P has length at most L, then we can introduce it as short segment to reduce the number of components of R. Otherwise, we can shorten P to P' such that every vertex of P' is at distance at least L from R and the two endpoints are at distance exactly L from two different components C and C' of R. Then we can reduce the number of components of R by introducing P' as a long segment and two short segments connecting the endpoints of P' to C and C'. By repeating these steps, we can reduce the number of components to 1 by introducing $O(k)$ segments in total.

Refining the faces. Our next goal is to further refine the skeleton such that every face of the skeleton has at most 35 segments on its boundary, and it is still true that the skeleton consists of $O(k)$ segments. We achieve this goal by iteratively dividing a face into two by introducing to the skeleton a new path consisting of at most 5 segments. We argue below that if the division is not very skewed in a certain sense, then the bound $O(k)$ on the number of segments can be achieved even after iterative applications of this step.

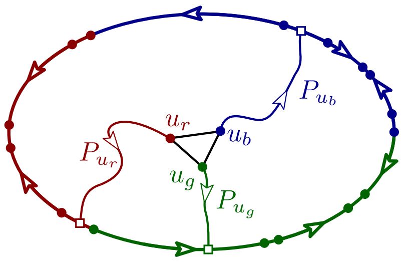

Figure 5 Finding a division that is not skewed.

Suppose that we have a face F where $x \geq 36$ segments appear on the boundary. Let P be a path between two segments of the boundary and assume that P consists of at most 5 segments. Introducing the path P into the skeleton creates two new faces F_1 and F_2 that see some number x_1 and x_2 segments on the boundary of F, plus the 5 new segments of P. We have $x_1 + x_2 \leq x + 2$: if the endpoints of P are internal vertices of segments, then we may have up to 2 segments that are now on the boundary of both F_1 and F_2.

For a face seeing $x \geq 13$ segments of the skeleton, let us define $x - 13 \geq 0$ to be the potential of the face. If we chose the path P such that $x_1, x_2 \geq 13$, then the potential of the two new faces F_1 and F_2 are defined. Moreover, the total potential of the two faces is at most

$$(x_1 + 5 - 13) + (x_2 + 5 - 13) \leq x - 14,$$

strictly less than the potential of F. This means that if we start with a face F that sees x segments of the skeleton, then repeated applications of this step can introduce only $O(x)$ new segments.

Finding a division that is not skewed. Next we show that if face F sees $x \geq 36$ segments of the skeleton, then we can find a division with $x_1, x_2 \geq 13$. Then as we have seen above, repeated applications of this step introduces $O(x)$ segments and divide F into faces that see at most 35 segments each.

Let us divide the boundary of F into three parts, red, green, and blue, each containing at least 12 segments (see Figure 5). As every vertex v inside the face F is essential for the solution, there is a directed path P_v from v to some vertex of the boundary; let us fix such a P_v for each v. This defines a color of v according to which of the three parts of the boundary contains the head of P_v. Then by Sperner's Lemma and fact that the graph is triangulated, there is a triangle u_r, u_g, u_b inside F where the three vertices have three different colors. From the assumptions that u_r, u_g, u_b are on three different parts, and each part has length at least 13, it follows that there are two vertices, say u_r and u_b, such that both subpaths of the boundary between the heads of P_{u_r} and P_{u_b} have at least 12 segments. Then putting together P_{u_r} and P_{u_b} creates a path P that divides the face F in the required way. This argument needs to be refined a bit further: as we said earlier, we want a skeleton where the long segments are distant, i.e., are at distance at least L from each other. But this can be easily achieved by appropriately shortening the long segments P_{u_r}, P_{u_b}, and then extending them by three short segments.

Many edges incident to a long path. We assume now that the skeleton has $O(k)$ faces, each seeing at most 35 segments. If we can show that the total branch degree (of the orginal solution H without the artificial edges) is a constant in each face, then we can bound by $O(k)$ the total branch degree of the solution. We can observe, using the acyclicity of the edges inside the face, that we need to bound only the number of edges incident on the boundary.

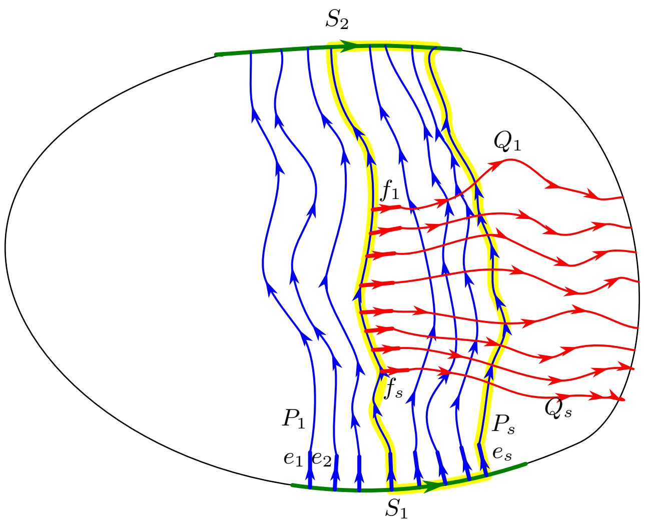

Figure 6 Finding a grid.

Let e be an edge inside the face incident to vertex v of the boundary. We say that e is *essential* for demand $t_i t_j$ if removing e breaks every path from t_i to t_j. Then we can define a path P_e the following way: let us take any path P from t_i to t_j, and let P_e be the subpath of P starting from e (which has to appear on P) to the first vertex on the boundary of F. Let us consider two edges e_1, e_2 starting from the same vertex v of the boundary. Let us observe that P_{e_1} and P_{e_2} cannot intersect: then we could bypass e.g. e_1 by starting on P_{e_2} and following it until intersection. By a similar argument, P_{e_1} and P_{e_2} cannot go to the same long segment: then one of P_{e_1} and P_{e_2} could be avoided by using the other path and part of the long segment. From these observations, it follows that the only way the boundary can have many edges incident to it is that if there are edges $e_1, \ldots, e_s$ incident to distinct vertices of a long segment S_1, with paths $P_1, \ldots, P_s$ going to distinct vertices of some other long segment S_2 (see Figure 6).

Finding a grid and a t-tough pair. Now comes the point where we use the assumption that long segments are distant. In particular, this means that the "middle path" $P_{e_{s/2}}$ is long. The internal vertices of this path have no terminals (as all the terminals are on the skeleton), hence it is not possible that $d^+(v) = d^-(v) = 1$ for any such internal vertex. Thus either there are many vertices on this path that have an edge leaving the path, or many vertices that have an edge entering the path. Assume without loss of generality the former, let $f_1, \ldots, f_s$ be these edges. Again, each edge is essential for some demand, hence the path satisfiying the demand has a subpath Q_i starting with f_i and going to the boundary. We can observe again that these paths have to be disjoint. Therefore, we can obtain a grid-like structure in the region surrounded by S_1, $P_{s/2}$, S_2, and P_s, see the region highlighted by yellow in Figure 6. (There are some other cases to consider, which we ignore here. For example, the paths Q_i may go to S_1 or S_2.) This region has $s/2 - 1$ "vertical" paths $P_{s/2}$, $\ldots$, P_{s-1}, intersected by the s "horizontal paths" $Q_1, \ldots, Q_s$.

We observe that if this grid has t horizontal and vertical paths, then we can use it to discover a t-tough pair. Each edge e_i is essential for some minimal demand; let E_1 be the set of these t demands. Similarly, we define E_2 based on choosing a minimal demand for which f_i is essential. Then we can carefully verify that (E_1, E_2) is a t-tough-pair: if there is an edge in the demand graph that is not allowed, then a careful analysis shows that there is a way of bypassing some e_i or f_i in the grid, contradicting the fact that it is essential. This concludes the proof that if we have an upper bound on the size of the largest t-tough pair appearing in the graphs of class $\mathcal{D}$, then we can bound the treewidth of the solution by $O(\sqrt{k})$.

Cleaning. To prove Theorem 10, we need to show that if arbitrary large t-tough-pairs appear in the graphs of $\mathcal{D}$, then $\mathcal{C}_i \subseteq \mathcal{D}$ for some $i \in [\kappa]$. The proof is a long combinatorial argument to show that we can find t-tough-pairs that are canonical in some sense, and then we use the assumption that $\mathcal{D}$ is closed under identifying vertices to contract the vertices outside the t-tough-pair into a small constant number of well-behaved vertices.

Suppose that there is a t-tough-pair (E_1, E_2) in a digraph D. The minimality of the edges in E_1 and the fact that they do not appear in directed cycles (as they are weakly independent to themselves) imply that for any two edges $x_iy_i, x_jy_j \in E_1$, at least one of the following holds:

1. exactly the edges x_iy_j, x_jy_i appear between $\{x_i, y_i\}$ to $\{x_j, y_j\}$,
2. there is no edge from $\{x_i, y_i\}$ to $\{x_j, y_j\}$, or
3. there is no edge from $\{x_j, y_j\}$ to $\{x_i, y_i\}$.

Let us consider a complete graph on t vertices $w_1, \ldots, w_t$, and for every $i < j$, color the edge w_iw_j according to which of the three statements hold for the edges x_iy_i and x_jy_j (if more than one statement is true, we can choose arbitrarily). By Ramsey's Theorem, there is a large subset $E_1' \subseteq E_1$ where the same statement holds for any pair of edges. We can find a similar subset $E_2' \subseteq E_2$. We consider two main cases. The first case is when Statement 1 holds either in E_1' or E_2'. Then what we have is a matching x_iy_i of minimal edges that is part of a complete bipartite graph, that is, every x_i is adjacent to every y_j (but note that x_iy_j does not have to be a minimal edge). The second case is where we have Statement 2 or 3 in both E_1' and E_2'. Then we can reorder E_1 and E_2 to have a further ordering property: there is no edge from $\{x_i, y_i\}$ to $\{x_j, y_j\}$ for $j < i$. We handle the two cases separately. With further Ramsey arguments and case distinctions, we show that identifications can be used to find a t'-hard biclique pattern or a t'-hard matching pattern appearing in a graph in $\mathcal{D}$, where t' is some unbounded function of t. It follows that if arbitrarily large t-tough pairs appear in $\mathcal{D}$, then $\mathcal{D}$ is a superclass of some $\mathcal{C}_i$.

For full details and proofs, as well as a concluding discussion and open problems, please see the full version of this article [16].

2 Formal definition of a t-tough-pair

In this section we give the formal definition of a t-tough-pair. Further definitions, that are specific to the sections are defined in the beginning of the respective sections.

Given a digraph D and an edge $e = (u, v) \in E(D)$, we say that e is a *minimal* edge of D if D has no (u, v)-path of length strictly greater than 1 in D, where the length of the path is the number of edges in it. We say that a digraph D is *reachability-minimal* if each edge of D is minimal. For an edge $e = (u, v)$ in a directed graph D, v is called the *head* of e and u is called the *tail* of e. For any $E' \subseteq E(D)$, $\texttt{head}(E')$ (resp. $\texttt{tail}(E')$) denotes the set of heads (resp. tails) of the edges in E'. Next we define weak independence and strong independence that are crucial to define the t-tough-pair formally.

▶ **Definition 12** (Weakly independent edges). *Given a digraph D and edges $e_1 = (u_1, v_1), e_2 = (u_2, v_2) \in E(D)$, we say that the pair of edges (e_1, e_2) is weakly independent in D, if $u_1 \neq v_1 \neq u_2 \neq v_2$, and D has neither a (v_1, u_2)-path nor a (v_2, u_1)-path. A set of edges $E' \subseteq E(D)$ is weakly independent if every pair of distinct edges in E' are pairwise weakly independent and for each edge $(u_i, v_i) \in E'$, there is no (v_i, u_i)-path in D.*

Informally, a pair of edges is weakly independent, if the head of one edge cannot reach the tail of the other. Therefore, if a pair of edges is weakly independent, then they cannot lie on a directed path.

▶ **Definition 13** (Strongly independent edges)**.** *Given a digraph D and edges $e_1 = (u_1, v_1), e_2 = (u_2, v_2) \in E(D)$, we say that the pair of edges (e_1, e_2) is strongly independent in D, if they are weakly independent in D, and additionally D has no (u_1, u_2)-path, no (u_2, u_1)-path, no (v_1, v_2)-path and no (v_2, v_1)-path.*

Informally, a pair of edges is strongly independent, if they are weakly independent, and the head of one cannot reach the head of the other, and the tail of one cannot reach the tail of the other. That is, the vertices of the heads (resp. vertices of tails) do not lie on any directed path.

▶ **Definition 14** (t-tough-pair)**.** *Given a digraph D, $E_1, E_2 \subseteq E(D)$, we say that (E_1, E_2) is a tough-pair in D if:*

1. $|E_1| = |E_2|$,
2. *each edge of $E_1 \cup E_2$ is a minimal edge in D,*
3. *all edges in E_i are pairwise weakly independent in D, for both $i \in \{1, 2\}$, and*
4. *for each $e_1 \in E_1$ and $e_2 \in E_2$, (e_1, e_2) are strongly independent in D.*

Further, for a positive integer t, we say that (E_1, E_2) is a t-tough-pair if $|E_1| = |E_2| = t$.

References

1 MohammadHossein Bateni, Chandra Chekuri, Alina Ene, Mohammad Taghi Hajiaghayi, Nitish Korula, and Dániel Marx. Prize-collecting Steiner Problems on Planar Graphs. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1028–1049, 2011. doi:10.1137/1.9781611973082.79.

2 MohammadHossein Bateni, Mohammad Taghi Hajiaghayi, and Dániel Marx. Approximation Schemes for Steiner Forest on Planar Graphs and Graphs of Bounded Treewidth. *J. ACM*, 58(5):21:1–21:37, 2011. doi:10.1145/2027216.2027219.

3 Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Approximation algorithms for spanner problems and directed steiner forest. *Inf. Comput.*, 222:93–107, 2013. doi:10.1016/j.ic.2012.10.007.

4 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets möbius: fast subset convolution. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 67–74. ACM, 2007. doi:10.1145/1250790.1250801.

5 Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation Algorithms for Directed Steiner Problems. *J. Algorithms*, 33(1):73–91, 1999. doi:10.1006/jagm.1999.1042.

6 Rajesh Chitnis, Hossein Esfandiari, Mohammad Taghi Hajiaghayi, Rohit Khandekar, Guy Kortsarz, and Saeed Seddighin. A Tight Algorithm for Strongly Connected Steiner Subgraph on Two Terminals with Demands. *Algorithmica*, 77(4):1216–1239, 2017. doi:10.1007/s00453-016-0145-8.

7 Rajesh Chitnis, Andreas Emil Feldmann, and Pasin Manurangsi. Parameterized Approximation Algorithms for Bidirected Steiner Network Problems. In *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, pages 20:1–20:16, 2018. doi:10.4230/LIPIcs.ESA.2018.20.

8 Rajesh Hemant Chitnis, Andreas Emil Feldmann, Mohammad Taghi Hajiaghayi, and Dániel Marx. Tight bounds for planar strongly connected steiner subgraph with fixed number of terminals (and extensions). *SIAM J. Comput.*, 49(2):318–364, 2020. doi:10.1137/18M122371X.

9 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.

10 Éric Colin de Verdière. Multicuts in planar and bounded-genus graphs with bounded number of terminals. *Algorithmica*, 78(4):1206–1224, 2017. doi:10.1007/s00453-016-0258-0.

11 Pavel Dvorák, Andreas Emil Feldmann, Dusan Knop, Tomás Masarík, Tomas Toufar, and Pavel Veselý. Parameterized Approximation Schemes for Steiner Trees with Small Number of Steiner Vertices. In *35th Symposium on Theoretical Aspects of Computer Science, STACS 2018, February 28 to March 3, 2018, Caen, France*, pages 26:1–26:15, 2018. doi:10.4230/LIPIcs.STACS.2018.26.

12 Eduard Eiben, Dusan Knop, Fahad Panolan, and Ondrej Suchý. Complexity of the Steiner Network Problem with Respect to the Number of Terminals. In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science (STACS 2019)*, volume 126 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:17, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2019.25.

13 Jon Feldman and Matthias Ruhl. The Directed Steiner Network Problem is Tractable for a Constant Number of Terminals. *SIAM J. Comput.*, 36(2):543–561, 2006. doi:10.1137/S0097539704441241.

14 Andreas Emil Feldmann and Dániel Marx. The complexity landscape of fixed-parameter directed steiner network problems. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016*, volume 55 of *LIPIcs*, pages 27:1–27:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.27.

15 Fedor V. Fomin, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Subexponential parameterized algorithms for planar and apex-minor-free graphs via low treewidth pattern covering. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 515–524. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.62.

16 Esther Galby, Sándor Kisfaludi-Bak, Dániel Marx, and Roohani Sharma. Subexponential parameterized directed steiner network problems on planar graphs: a complete classification, 2022. arXiv:2208.06015.

17 Jiong Guo, Rolf Niedermeier, and Ondrej Suchý. Parameterized complexity of arc-weighted directed steiner problems. *SIAM J. Discrete Math.*, 25(2):583–599, 2011. doi:10.1137/100794560.

18 S. Louis Hakimi. Steiner's problem in graphs and its implications. *Networks*, 1(2):113–133, 1971. doi:10.1002/net.3230010203.

19 Richard M. Karp. Reducibility Among Combinatorial Problems. In *Proceedings of a Symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, pages 85–103, 1972. doi:10.1007/978-1-4684-2001-2_9.

20 Philip N. Klein and Dániel Marx. Solving planar k-terminal cut in $O(n^{c\sqrt{k}})$ time. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 569–580. Springer, 2012. doi:10.1007/978-3-642-31594-7_48.

21 Philip N. Klein and Dániel Marx. A subexponential parameterized algorithm for subset TSP on planar graphs. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1812–1830. SIAM, 2014. doi:10.1137/1.9781611973402.131.

22 A Levin. Algorithm for the shortest connection of a group of graph vertices. *Soviet Math. Dokl.*, 12:1477–1481, 1971. doi:10.4086/toc.2010.v006a005.

23 Chung-Lun Li, S. Thomas McCormick, and David Simchi-Levi. The point-to-point delivery and connection problems: complexity and algorithms. *Discrete Applied Mathematics*, 36(3):267–292, 1992. doi:10.1016/0166-218X(92)90258-C.

24 Daniel Lokshtanov, Saket Saurabh, and Magnus Wahlström. Subexponential parameterized odd cycle transversal on planar graphs. In Deepak D'Souza, Telikepalli Kavitha, and Jaikumar Radhakrishnan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012, December 15-17, 2012, Hyderabad, India*, volume 18 of *LIPIcs*, pages 424–434. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2012. doi:10.4230/LIPIcs.FSTTCS.2012.424.

25 Dániel Marx. On the Optimality of Planar and Geometric Approximation Schemes. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 338–348, 2007. doi:10.1109/FOCS.2007.50.

26 Dániel Marx. A tight lower bound for planar multiway cut with fixed number of terminals. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 677–688. Springer, 2012. doi:10.1007/978-3-642-31594-7_57.

27 Dániel Marx, Marcin Pilipczuk, and Michal Pilipczuk. On subexponential parameterized algorithms for steiner tree and directed subset TSP on planar graphs. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 474–484. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00052.

28 Dániel Marx and Michal Pilipczuk. Optimal parameterized algorithms for planar facility location problems using voronoi diagrams. *ACM Trans. Algorithms*, 18(2):13:1–13:64, 2022. doi:10.1145/3483425.

29 Madan Natu and Shu-Cherng Fang. The Point-to-point Connection Problem - Analysis and Algorithms. *Discrete Applied Mathematics*, 78(1-3):207–226, 1997. doi:10.1016/S0166-218X(97)00010-3.

30 Jesper Nederlof. Detecting and counting small patterns in planar graphs in subexponential parameterized time. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1293–1306. ACM, 2020. doi:10.1145/3357713.3384261.

31 S. Ramanathan. Multicast tree generation in networks with asymmetric links. *IEEE/ACM Trans. Netw.*, 4(4):558–568, 1996. doi:10.1109/90.532865.

32 Hussein F. Salama, Douglas S. Reeves, and Yannis Viniotis. Evaluation of Multicast Routing Algorithms for Real-Time Communication on High-Speed Networks. *IEEE Journal on Selected Areas in Communications*, 15(3):332–345, 1997. doi:10.1109/49.564132.

33 Pawel Winter. Steiner problem in networks: A survey. *Networks*, 17(2):129–167, 1987. doi:10.1002/net.3230170203.