Complexity Framework for Forbidden Subgraphs IV: The Steiner Forest Problem

Hans L. Bodlaender[1], Matthew Johnson[2], Barnaby Martin[2], Jelle J. Oostveen[1 [0009−0009−4419−3143]] ⋆, Sukanya Pandey[1 [0000−0001−5728−1120]], Daniël Paulusma[2 [0000−0001−5945−9287]], Siani Smith[3], and Erik Jan van Leeuwen[1 [0000−0001−5240−7257]]

[1] Dept. Information & Computing Sciences, Utrecht University, The Netherlands
`h.l.bodleander,j.j.oostveen,s.pandey1,e.j.vanleeuwen@uu.nl`
[2] Durham University, Durham, United Kingdom
`matthew.johnson2,barnaby.d.martin,daniel.paulusma@durham.ac.uk`
[3] University of Bristol and Heilbronn Institute for Mathematical Research, Bristol, United Kingdom `siani.smith@bristol.ac.uk`

Abstract. We study Steiner Forest on H-subgraph-free graphs, that is, graphs that do not contain some fixed graph H as a (not necessarily induced) subgraph. We are motivated by a recent framework that completely characterizes the complexity of many problems on H-subgraph-free graphs. However, in contrast to, e.g. the related Steiner Tree problem, Steiner Forest falls outside this framework. Hence, the complexity of Steiner Forest on H-subgraph-free graphs remained tantalizingly open. We make significant progress on this open problem: our main results are four novel polynomial-time algorithms for different excluded graphs H that are central to further understand its complexity. Along the way, we study the complexity of Steiner Forest for graphs with a small c-deletion set, that is, a small set X of vertices such that each component of $G - X$ has size at most c. Using this parameter, we give two algorithms that we later employ as subroutines. First, we present a significantly faster parameterized algorithm for Steiner Forest parameterized by $|X|$ when $c = 1$ (i.e. the vertex cover number), which by a recent result is best possible under ETH [Feldmann and Lampis, arXiv 2024]. Second, we prove that Steiner Forest is polynomial-time solvable for graphs with a 2-deletion set of size at most 2. The latter result is tight, as the problem is NP-complete for graphs with a 3-deletion set of size 2.

Keywords: Steiner forest · forbidden subgraph · complexity dichotomy · vertex cover number · deletion set

1 Introduction

We consider the complexity of a classical graph problem, Steiner Forest, restricted to graphs that do not contain some fixed graph H as a subgraph. Such

⋆ J.J. Oostveen is supported by the NWO grant OCENW.KLEIN.114 (PACAN).

graphs are said to be H-*subgraph-free*, that is, they cannot be modified to H by a sequence of edge deletions and vertex deletions. A graph G is H-*free* if G cannot be modified into H by a sequence of vertex deletions only. Even though H-free graphs are more widely studied in the literature, H-subgraph-free graphs are also highly interesting, as was recently shown through the introduction of a large, general framework for the subgraph relation [15,16,17,19].

For a set of graphs $\mathcal{H}$, a graph G is $\mathcal{H}$-*subgraph-free* if G is H-subgraph-free for every $H \in \mathcal{H}$. In order to unify known classifications for Independent Set [1], Dominating Set [1], List Colouring [14], Long Path [1] and Max-Cut [18] on $\mathcal{H}$-subgraph-free graphs (for finite $\mathcal{H}$), a systematic approach was developed in [15]. We first explain this framework.

For $k \geq 1$, the k-*subdivision* of an edge $e = uv$ of a graph replaces e by a path of length $k+1$ with endpoints u and v (and k new vertices). The k-*subdivision* of a graph G is the graph obtained from G after k-subdividing each edge. For a graph class $\mathcal{G}$ and an integer k, let $\mathcal{G}^k$ consist of the k-subdivisions of the graphs in $\mathcal{G}$. A graph problem Π is NP-complete *under edge subdivision of subcubic graphs* if for every integer $j \geq 1$, there is an integer $\ell \geq j$ such that: if Π is NP-complete for the class $\mathcal{G}$ of subcubic graphs (graphs with maximum degree at most 3), then Π is NP-complete for $\mathcal{G}^\ell$. Now, Π is a *C123-problem* if:

C1. Π is polynomial-time solvable for every graph class of bounded treewidth,
C2. Π is NP-complete for the class of subcubic graphs, and
C3. Π is NP-complete under edge subdivision of subcubic graphs.

A *subdivided* claw is a graph obtained from a *claw* (4-vertex star) by subdividing each of its three edges zero or more times. The *disjoint union* of two vertex-disjoint graphs G_1 and G_2 is the graph $(V(G_1) \cup V(G_2), E(G_1) \cup E(G_2))$. The set $\mathcal{S}$ consists of all graphs that are disjoint unions of subdivided claws and paths. We can now state the complexity classification of [15].

Theorem 1 ([15]). *Let Π be a C123-problem. For a finite set $\mathcal{H}$, the problem Π on $\mathcal{H}$-subgraph-free graphs is polynomial-time solvable if $\mathcal{H}$ contains a graph from $\mathcal{S}$ (or equivalently, if the class of $\mathcal{H}$-subgraph-free graphs has bounded treewidth) and* NP-*complete otherwise.*

See the (long) table of problems in [15], for examples of C123-problems other than the ones above and for examples of problems that do not satisfy C2 or C3. There are also problems that only satisfy C2 and C3 but not C1. For example, Subgraph Isomorphism is NP-complete even for input pairs of path-width 1. A few years ago, Bodlaender et al. [6] settled the complexity of Subgraph Isomorphism for H-subgraph-free graphs apart from essentially two open cases. ($H = P_5$ and $H = 2P_5$). Hence, the following question is challenging:

How do C23-problems*, i.e., problems that satisfy C2 and C3 but not C1, behave for H-subgraph-free graphs? Can we still classify their computational complexity?*

We consider this question for Steiner Forest. A *Steiner forest* of an undirected graph G, with a set $T = \{(s_1, t_1), \ldots, (s_p, t_p)\}$ of specified pairs of vertices called *terminals*, is a subgraph F of G, such that s_i and t_i, for every $i \in \{1, \ldots, p\}$, belong to the same connected component of F. This leads to the problem:

Steiner Forest
Instance: A graph G, a set T of terminal pairs and an integer k.
Question: Does (G, T) have a Steiner forest F with $\lvert E(F)\rvert \leq k$?

Steiner Forest generalizes the C123-problem Steiner Tree [15], which is to decide if for a given integer k, a graph G with some specified set T of vertices has a tree S with $|E(S)| \leq k$ containing every vertex of T: take all pairs of vertices of T as terminal pairs to obtain an equivalent instance of Steiner Forest.

For a constant c, a *c-deletion set* of a graph $G = (V, E)$ is a set $T \subseteq V$ such that each connected component of $G - T$ has size at most c. The *c-deletion set number* (or $(c+1)$-*component order connectivity*) of G is the size of a smallest c-deletion set (see also [3,7,8,9,11,12]). The following theorem is a crucial result of Bateni, Hajiaghayi and Marx [4] and plays an important role in our paper:

Theorem 2 ([4]). Steiner Forest *is polynomial-time solvable for graphs of treewidth at most* 2*, but* NP*-complete for graphs of treewidth* 3*, tree-depth* 4*, and* 3*-deletion set number* 2*.*

This shows that Steiner Forest does not satisfy C1, unlike Steiner Tree [2]. As Steiner Tree satisfies C2 and C3 [15], Steiner Forest satisfies C2 and C3 and is a C23-problem, unlike Steiner Tree which is C123 [15]. This leaves the complexity of Steiner Forest on H-subgraph-free graphs open.

Our Results. Let $K_{a,b}$ be the complete bipartite graph with a vertices on one side and b on the other. Let $S_{a,b,c}$ be the graph obtained from the claw ($K_{1,3}$) by subdividing its three edges $a-1$, $b-1$ and $c-1$ times, respectively. Let P_r be the path on r vertices. For two graphs H_1 and H_2, we write $H_1 \subseteq H_2$ if H_1 is a subgraph of H_2, i.e., $V(H_1) \subseteq V(H_2)$ and $E(H_1) \subseteq E(H_2)$. We write $H_1 + H_2$ for the disjoint union of H_1 and H_2 and sH_1 to denote the disjoint union of s copies of H_1. Our results on Steiner Forest for H-subgraph-free graphs are:

Theorem 3. *For a graph H,* Steiner Forest *on H-subgraph-free graphs is*

- *polynomial-time solvable if $H \subseteq 2K_{1,3} + P_3 + sP_2, 2P_4 + P_3 + sP_2, P_9 + sP_2$ or $S_{1,1,4} + sP_2$ for each $s \geq 0$, and*
- NP*-complete if $H \supseteq 3K_{1,3}, 2K_{1,3} + P_4, K_{1,3} + 2P_4, 3P_4$ or if $H \notin \mathcal{S}$.*

The gap between the easy and hard cases could be significantly reduced if we could resolve an intriguing open problem (see Section 5). As graphs of tree-depth 3 are P_8-subgraph-free, Theorems 2 and 3 yield the following dichotomy:

Corollary 1. *For a constant t,* Steiner Forest *on graphs of tree-depth t is polynomial-time solvable if $t \leq 3$ and* NP*-complete if $t \geq 4$.*

The NP-hardness part of Theorem 3 follows from the gadget from Theorem 2 and NP-completeness of Steiner Forest when $H \notin \mathcal{S}$ [5], as shown in Section 2. For the polynomial part, proven in Section 4, we first make some useful observations in Section 3.

Gima et al. [13] showed that Steiner Forest has an $n^{O(k)}$-time algorithm in the weighted case and an $(2k^2 2^k k^{2^k})^{O(k)} n^{O(1)}$-time algorithm in the unweighted case, where k is the vertex cover number. We need their result as a subroutine for several of our algorithms, but we were able to significantly improve on it, as we show in the following result (proof omitted).

Theorem 4. Steiner Forest *has a* $2^{O(k \log k)} n^{O(1)}$ *time algorithm, where* k *is the vertex cover number of the input graph, even in the weighted case.*

Afterwards, Feldmann and Lampis [10] showed our $2^{O(k \log k)} n^{O(1)}$ algorithm is best possible under the Exponential Time Hypothesis. They also gave an alternative algorithm with the same runtime that relies on an algorithm of Bateni et al. [4]. In contrast, our algorithm is self-contained.

Another important subroutine for some of our algorithms is the polynomial-part of the following dichotomy. We prove this part in Lemma 3 in Section 3, while the NP-completeness part is taken from Theorem 2, which is due to [4].

Theorem 5. *For a constant* c, Steiner Forest *on graphs with a* c*-deletion set of size at most* 2 *is polynomial-time solvable if* $c \leq 2$ *and* NP*-complete if* $c \geq 3$.

2 NP-Completeness Results

Bateni, Hajiaghayi and Marx [4] explicitly proved that Steiner Forest is NP-complete for graphs of treewidth 3, see also Theorem 2. The additional properties in the lemma below can be easily verified from inspecting their gadget, which is displayed in Figure 1.

Lemma 1. Steiner Forest *is* NP*-complete for* $(3K_{1,3}, 2K_{1,3} + P_4, K_{1,3} + 2P_4, 3P_4)$*-subgraph-free graphs of tree-depth* 4 *with* 3*-deletion set number* 2.

We can now show the NP-completeness part of Theorem 3. It is known that Steiner Tree, and thus Steiner Forest, is NP-complete for H-subgraph-free graphs if $H \notin \mathcal{S}$ [5]. The NP-completeness part of Theorem 3 now follows immediately from this observation and Lemma 1.

3 Polynomial Subroutines

A *minimum* Steiner forest for an instance (G, T) is one with the smallest number of edges. Denote the number of edges of such a forest by $\mathsf{sf}(G, T)$. We assume that for any terminal pair $(s, t) \in T$, s and t are distinct (as any pair where $s = t$ can be removed without affecting the feasibility or the size of a minimum solution). A vertex v is a *terminal vertex* if there is a pair $(s, t) \in T$ with $v = s$

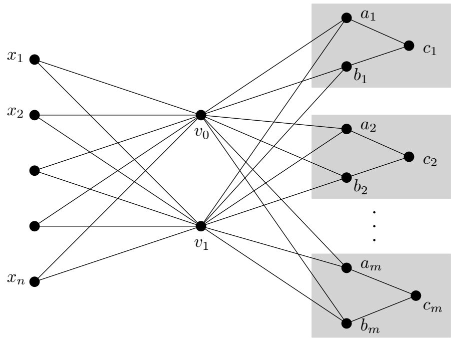

Fig. 1. The graph G (gadget from [4]) used in the proof of Lemma 1. On the left there are vertices $x_1 \dots x_n$ representing the boolean variables. On the right, each P_3 represents a clause of the given R-formula ϕ.

or $v = t$. A *cut vertex* of a connected graph is a vertex whose removal yields a graph with at least two connected components. A graph with no cut vertices is 2-*connected*. A 2-*connected component* is a maximal subgraph that is 2-connected. A graph class is *hereditary* if it is closed under deleting vertices.

Lemma 2. *For every hereditary graph class $\mathcal{G}$, if* Steiner Forest *is polynomial-time solvable for the subclass of* 2*-connected graphs of $\mathcal{G}$, then it is polynomial-time solvable for $\mathcal{G}$.*

Proof (Sketch). Assume G is connected. If G is not 2-connected, then let G_1 be a 2-connected component of G with only one cut-vertex v of G. Let $G_2 = G-(V(G_1)\setminus\{v\})$. For $i = 1, 2$, let T_i be the set of all terminal pairs in G_i with both terminals in $V(G_i)$, to which we add the pair (s_j, v) for each $(s_j, t_j) \in T$ such that $s_j \in V(G_i)$ and $t_j \notin V(G_i)$. Similarly, if $(s_j, t_j) \in T$ such that $s_j \notin V(G_i)$ and $t_j \in V(G_i)$, then we add (v, t_j) to T_i. Now apply the algorithm for 2-connected graphs to (G_1, T_1), and recurse on (G_2, T_2). □

The *contraction* of an edge $e = (u, v)$ in a graph G replaces u and v by a new vertex w that is adjacent to all former neighbours of u and v in G. Our next result shows the polynomial part of Theorem 5.

Lemma 3. Steiner Forest *is polynomial-time solvable for graphs with a* 2*-deletion set of size at most* 2*.*

Proof. Let G be an n-vertex graph that, together with a set T of terminal pairs, is an instance of Steiner Forest. If G has a 2-deletion set of size 1, then this vertex forms a cut vertex of G. Hence, we can apply Lemma 2 to reduce to the case of graphs of size 3, which is trivial. So, we assume that G has a 2-deletion set C of size 2, say $C = \{u, v\}$. By Lemma 2 we may assume that G is 2-connected.

First suppose u and v are adjacent. If the edge uv is part of the solution, we may contract uv to form a 2-deletion set of size 1. We remember the size of the

Steiner Forest found. If uv is not part of the solution, we reduce to the case when u and v are not adjacent. From now on, we assume u and v are non-adjacent.

We first search for a minimum solution for (G, T) over all solutions with one connected component. We compute a minimum solution over all solutions that contain u but not v, and a minimum solution over all solutions that contain v but not u. This takes polynomial time, as we reduce to the case of graphs of size 3 by applying Lemma 2. We now compute a minimum solution for (G, T) over all solutions with one connected component that contain both u and v. As G is 2-connected, there exists a path from u to v in G. As C is a 2-deletion set, every path from u to v has at most two inner vertices. We check each of the $O(n^2)$ paths from u to v (with at most two inner vertices). For each choice of P, we contract the path P to a single vertex and apply Lemma 2 to reduce to the case of graphs of of size 3 by applying Lemma 2.

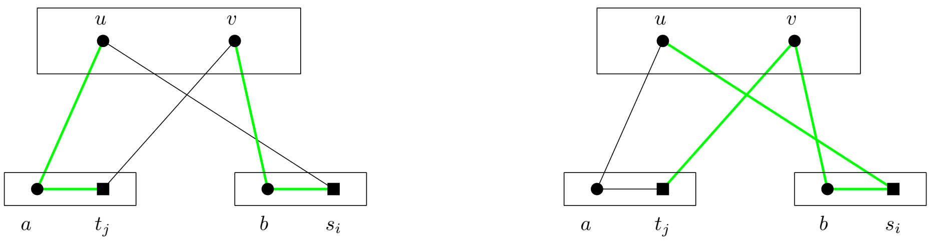

Fig. 2. Left: a solution containing two non-terminal vertices a and b. Right: the conversion to an equivalent solution containing b as the only non-terminal vertex.

It remains to find a minimum solution for (G, T) over all solutions with two connected components, one containing u and the other one v, and to check this minimum solution with the minimum solutions of the other types found above.

If some connected component D of $G \setminus \{u, v\}$ does not contain any terminal vertex, we can safely delete D. Hence, we assume that every connected component of $G \setminus \{u, v\}$ contains at least one terminal vertex. We now show that we can restrict ourselves to solutions that contain at most one non-terminal vertex.

Suppose that there exists an optimal solution F with two non-terminal vertices a and b. We may assume without loss of generality that v is adjacent to b, and that b is adjacent to terminal s_i. Hence, F contains the edges vb and bs_i. As G is 2-connected, the edge us_i must exist in G as well. Consider another connected component of $G \setminus \{u, v\}$, say at_j, where t_j is a terminal. First assume that $au \in E(G)$. As G is 2-connected, we also have $vt_j \in E(G)$. As $a \in V(F)$, the edges ua and at_j must belong to $E(F)$. We now remove the edges ua and at_j from F and add the edges us_i and vt_j to F. This yields a minimum Steiner Forest F' with fewer non-terminal vertices. If $au \notin E(G)$, we have that $av \in E(G)$. In this case we can make a similar replacement and come to the same conclusion.

Due to the above, we may restrict ourselves to finding minimum solutions for (G, T) with two connected components that contain at most one non-terminal vertex. Hence, we branch by considering all $O(n)$ options for the set of non-terminal vertices used in a minimum solution. We consider each of the $O(n)$ branches separately and as follows.

First, we remove all non-terminal vertices that we did not guess to be in the solution. For the guessed non-terminal vertex we do as follows. By construction, such a guessed non-terminal vertex z is adjacent to u or v, and we contract the edge zu or zv, respectively. Afterwards, we apply Lemma 2 again such that we may assume that the resulting instance, which we denote by (G, T) again, is 2-connected. Now every vertex of $V(G) \setminus \{u, v\}$ is a terminal vertex. Hence, every vertex of $V(G) \setminus \{u, v\}$ will be in the solution we are trying to construct.

For every connected component D of $G - \{u, v\}$ we do as follows. As C is a 2-deletion set, D consists of at most two vertices. We only consider D if D has exactly two vertices, x and y. If $(x, y) \in T$ and neither x nor y appears in any other terminal pair, then we add the edge xy to the solution; remove the vertices x and y from G; and also remove the pair (x, y) from T. This is indeed optimal, as edges from x and y to u or v would not be involved in connecting any other terminal pairs. So we may assume that $(x, y) \notin T$, or x or y appears in some other pair than (x, y) in T.

We now apply the following operation on the component D, depending on its adjacency to u and v. First suppose x and y are both adjacent to both u and v. Then we may remove the edge xy for the following reason. Recall that x and y are terminal vertices such that $(x, y) \notin T$, or x or y appears in some other pair than (x, y) in T. Hence, we need to connect x to either u or v, and we also need to connect y to either u or v. For doing this, we do not need to use the edge xy. Now suppose one of x, y, say x, is adjacent to u and v, whereas the other one, y, is adjacent to only one of u and v, say to u. For the same reasons as before, we need to connect x to either u or v, and we also need to connect y to either u or v. If we use xu, then we must also use yu, and in that case we can replace xu by xy. Hence, we may remove xu from G.

So, afterwards, we reduced the instance in polynomial time to a new instance, which we will also denote by (G, T), with the following properties. Every connected component of $G - \{u, v\}$ has at most two vertices. By 2-connectivity, for every connected component that contains exactly one vertex z we have the edges uz and vz. Moreover, for every connected component that contains exactly two vertices x and y, we have the edges ux and vy but not uy and not vx.

As every vertex in $V(G) \setminus \{u, v\}$ is a terminal vertex and we search for a minimum solution for (G, T) with two connected components, we need one edge in the solution for each 1-vertex connected component of $G - \{u, v\}$ and two edges for each 2-vertex connected component. First, suppose one of u, v, say u, is not a terminal vertex. Then $G - \{v\}$ is a minimum solution, so we can stop (note that we already found this solution before).

Now suppose that both u and v are terminal vertices. We discard the branch if u and v represent terminals of the same pair (as then the solution must be connected). Else we do as follows. If u represents s_i, then we connect the terminal vertex that represents t_i to u. This can only be done in one way: if the terminal vertex representing t_i is not adjacent to u, it contains a unique neighbour adjacent to u. Afterwards, we remove the terminal pairs that we connected in this way from G. We repeat these steps for v. If it was not possible to connect some terminal pair, then we discard the branch. Otherwise, we apply Lemma 2 again, such that the resulting instance, which we denote by (G, T) again, is 2-connected. The other properties are maintained, and neither u nor v is a terminal vertex. Hence, we can take as solution for (G, T) either $G - \{u\}$ or $G - \{v\}$. We now construct a solution for the original graph and terminal set, and out of all the solutions found take a minimum one. □

For some of our polynomial-time results we need to extend Lemma 3 as follows (proof omitted).

Lemma 4. Steiner Forest *is polynomial-time solvable for graphs G with a set $X \subseteq E(G)$ of bounded size such that $G - X$ has a 2-deletion set C of size at most 2 and each end-point of every edge of X is either an isolated vertex of $G - X$ or a vertex of C.*

4 Polynomial Cases

To prove the polynomial part of Theorem 3, we start with a general lemma. To prove it we use Theorem 4; we omit the details.

Lemma 5. *For a graph H, if* Steiner Forest *can be solved in polynomial time on the class of H-subgraph-free graphs, then* Steiner Forest *can be solved in polynomial time on the class of $(H + P_2)$-subgraph-free graphs.*

We now consider specific graphs H. The first one is the case $H = 2K_{1,3}$. We omit the proof.

Lemma 6. Steiner Forest *is polynomial-time solvable for $2K_{1,3}$-subgraph-free graphs.*

By using Lemma 4 (which generalized Lemma 3) we can extend Lemma 6 (again we omit the proof details).

Lemma 7. Steiner Forest *is polynomial-time solvable for $(2K_{1,3} + P_3)$-subgraph-free graphs.*

We now consider the case $H = S_{1,1,4}$. We omit the proof.

Lemma 8. Steiner Forest *is polynomial-time solvable for $S_{1,1,4}$-subgraph-free graphs.*

We use Lemma 8 to prove the case $H = P_9$. We also need Lemma 3 again.

Lemma 9. Steiner Forest *is polynomial-time solvable for P_9-subgraph-free graphs.*

Proof. Let $G = (V, E)$ be a P_9-subgraph-free graph that is part of an instance of Steiner Forest. By Lemma 2 we may assume that G is 2-connected. Let $P = u_1 \cdots u_r$, for some $r \geq 2$, be a longest (not necessarily induced) path in G. As G is P_9-subgraph-free, we have that $r \leq 8$. If $r \leq 5$, then G is P_6-subgraph-free, and thus $S_{1,1,4}$-subgraph-free, and we can apply Lemma 8. Hence, $r \in \{6, 7, 8\}$.

Case 1. $r = 6$.
Then G is P_7-subgraph-free. Suppose $G - V(P)$ has a connected component D with more than one vertex. As G is P_7-subgraph-free and $|V(D)| \geq 2$, no vertex of D is adjacent to u_1, u_2, u_5 or u_6. As G is connected, at least one of u_3 or u_4, say u_3, has a neighbour v in D. Suppose $w \in V(D)$ is adjacent to u_4 (where $w = v$ is possible). Let vQw be a path from v to w in D; note that Q might be empty. Now the path $u_1u_2u_3vQwu_4u_5u_6$ has at least seven vertices, contradicting that G is P_7-subgraph-free. Hence, no vertex of D is adjacent to u_4. This means that u_3 is a cut-vertex of G, contradicting the 2-connectivity. We conclude that every connected component of $G - V(P)$ consists of one vertex. In other words, $\{u_1, \ldots, u_6\}$ is a vertex cover of G, and we can apply Theorem 4.

Case 2. $r = 7$.
Then G is P_8-subgraph-free. Suppose $G - V(P)$ has a connected component D with more than one vertex. As G is P_8-subgraph-free and $|V(D)| \geq 2$, no vertex of D is adjacent to u_1, u_2, u_6 or u_7. As G is connected, at least one of u_3, u_4 or u_5 has a neighbour v in D.

First assume that u_3 or u_5, say u_3, has a neighbour v in D. As $|V(D)| \geq 2$, v has a neighbour w in D. If w has a neighbour $x \neq v$ in D, then $xwvu_3u_4u_5u_6u_7$ is a path on eight vertices, contradicting that G is P_8-subgraph-free. Hence, v is the only neighbour of w in D. As G is 2-connected, w has a neighbour on P. Recall that no vertex of D is not adjacent to u_1, u_2, u_6 or u_7. If w is adjacent to u_4, then $u_1u_2u_3vwu_4u_5u_6$ is a path on eight vertices. If w is adjacent to u_5, then $u_1u_2u_3vwu_5u_6u_7$ is a path on eight vertices. Hence, w must be adjacent to u_3 (and u_3 is the only neighbour of w on P). We now find that w is the only neighbour of v in D, as otherwise, if v has a neighbour $w' \neq w$ on D, then $w'vwu_3u_4u_5u_6u_7$ is a path on eight vertices. In other words, $V(D) = \{v, w\}$. By the same arguments, but now applied on v, we find that u_3 is the only neighbour of v on P. Hence, u_3 is a cut-vertex of G, contradicting the 2-connectivity of G.

From the above we conclude that no vertex of D is adjacent to u_3. By the same reason, no vertex of D is adjacent to u_5. We find that u_4 disconnects D from the rest of G, contradicting the 2-connectivity of G. We conclude that every connected component of $G - V(P)$ consists of one vertex. In other words, $\{u_1, \ldots, u_7\}$ is a vertex cover of G, and we can apply Theorem 4.

Case 3. $r = 8$.
Recall that G is P_9-subgraph-free, so this is the last case to consider. If every connected component of $G - V(P)$ consists of one vertex, then $\{u_1, \ldots, u_8\}$ is a vertex cover of G, and we can apply Theorem 4. Now suppose that $G - V(P)$ has a connected component D with more than one vertex. As G is P_9-subgraph-free

and $|V(D)| \geq 2$, no vertex of D is not adjacent to u_1, u_2, u_7 or u_8. As G is connected, at least one of u_3, u_4, u_5 or u_6 has a neighbour in D.

First, suppose that neither u_3 nor u_6 has a neighbour in D. Then u_4 or u_5, say u_4, has a neighbour v in D. As G is 2-connected, there exists a path vQu_5 from v to u_5 that does not contain u_4. As no vertex from $\{u_1, u_2, u_3, u_6, u_7, u_8\}$ has a neighbour in D, the vertices of Q belong to D. Now, $u_1u_2u_3u_4vQu_5u_6u_7u_8$ is a path on nine vertices, contradicting that G is P_9-subgraph-free.

Hence, at least one of u_3 or u_6, say u_3, has a neighbour v in D. As $|V(D)| \geq 2$, we find that v has a neighbour w in D. If w has a neighbour $x \neq v$ in D, then $xwvu_3u_4u_5u_6u_7u_8$ is a path on nine vertices, contradicting that G is P_9-subgraph-free. Hence, v is the only neighbour of w in D. As G is 2-connected, this means that w has a neighbour on P. Recall that no vertex of D is adjacent to u_1, u_2, u_7 or u_8. If w is adjacent to u_4, then $u_1u_2u_3vwu_4u_5u_6u_7$ is a path on nine vertices. If w is adjacent to u_5, then $u_1u_2u_3vwu_5u_6u_7u_8$ is a path on nine vertices. Hence, w must be adjacent to either or both u_3 and u_6 (and w has no other neighbours on P). We now find that w is the only neighbour of v in D for the following reason. Suppose v has a neighbour $w' \neq w$ on D. If w is adjacent to u_3, then $w'vwu_3u_4u_5u_6u_7u_8$ is a path on nine vertices. If w is adjacent to u_6, then $w'vwu_6u_5u_4u_3u_2u_1$ is a path on nine vertices. In other words, $V(D) = \{v, w\}$. By the same arguments, but now applied on v, we find that apart from u_3, it holds that v may have only u_6 as a neighbour of P.

We have proven for any path $Q = q_1 \dots q_8$ on eight vertices that every connected component of size at least 2 in $G-Q$ has size exactly 2, and moreover, q_3 and q_6 are the only vertices of Q with neighbours in such a connected component. We may assume without loss of generality that v is adjacent to u_3 and w is adjacent to u_6, as otherwise one of u_3, u_6, v, w is a cut-vertex of G, contradicting the 2-connectivity of G. By replacing P with $P' = u_1u_2u_3vwu_6u_7u_8$, we find that u_4 and u_5 have no neighbours outside $\{u_3, u_6\}$. By replacing P with $P' = u_5u_4u_3vwu_6u_7u_8$, we find that u_2 has no neighbours outside $\{u_3, u_6\}$ (just like u_1). By symmetry, u_7 has no neighbours outside $\{u_3, u_6\}$ (just like u_8). Hence, every connected component of $G - \{u_3, u_6\}$ has at most two vertices. Thus, $\{u_3, u_6\}$ is a 2-deletion set of size at most 2, and we can apply Lemma 3. □

We use Lemma 9 to show the case $H = 2P_4 + P_3$ (proof details are omitted).

Lemma 10. Steiner Forest *is polynomial-time solvable for* $(2P_4+P_3)$*-subgraph-free graphs.*

We now prove the polynomial part of Theorem 3 by combining Lemma 5 with each of the other lemmas in this section.

5 Conclusions

Our aim was to increase our understanding of the complexity of Steiner Forest and more generally, C23-problems, that is, graph problems that are not only NP-complete on subcubic graphs (C2) and under edge division of subcubic graphs

(C3), but also NP-complete for graphs of bounded treewidth (not C1). Therefore, we studied Steiner Forest for H-subgraph-free graphs. We significantly narrowed the number of open cases, thereby proving a number of boundary cases (see Theorem 3). So far, we could not generalize Lemma 5 from P_2 to P_3:

Open Problem 1 *Let H be a graph. Is* Steiner Forest *polynomial-time solvable on $(H + P_3)$-subgraph-free graphs if it is so for H-subgraph-free graphs?*

An affirmative answer to this question would reduce the number of open cases in Theorem 3 to a finite number. However, this requires a polynomial-time algorithm for Steiner Forest on graphs with a 2-deletion set of size d for any constant d.

The question of whether such an algorithm exists turned out to be challenging. One important attempt that we made to solve this question was to reduce instances to highly structured instances. In particular, we were able to reduce it to the case where the vertices of the deletion set itself belong to different connected components of a minimum Steiner forest, and all vertices not in the deletion set are terminals. However, even solving such highly structured instances seems difficult. We managed to reduce it to a Constraint Satisfaction Problem (CSP). Interestingly, this CSP can be solved in polynomial time for 2-deletion sets of size 2 (cf. Lemma 3). The same CSP is NP-complete when we consider deletion sets of size 3. Unfortunately, our reduction is only one way, so this does not directly imply NP-completeness of Steiner Forest in this case. Still, this hints that the problem might be NP-complete for $H = sP_3$ for some $s \geq 4$.

Finally, the C123 problem Steiner Tree is also classified with respect to the induced subgraph relation: it is polynomial-time solvable for H-free graphs if $H \subseteq_i sP_1 + P_4$ for some $s \geq 0$ and NP-complete otherwise [5]. The hardness part of this result immediately carries over to Steiner Forest. However, we do not know the complexity of Steiner Forest on $(sP_1 + P_4)$-free graphs.

Acknowledgments. We thank Daniel Lokshtanov for pointing out a possible relationship between Steiner Forest and CSP, as discussed in Section 5. We also thank the anonymous reviewers of earlier versions of this paper for their helpful comments and suggestions.

References

1. Alekseev, V.E., Korobitsyn, D.V.: Complexity of some problems on hereditary graph classes. Diskretnaya Matematika **4**, 34–40 (1992)
2. Arnborg, S., Lagergren, J., Seese, D.: Easy problems for tree-decomposable graphs. Journal of Algorithms **12**, 308–340 (1991)
3. Barefoot, C.A., Entringer, R., Swart, H.: Vulnerability in graphs – a comparative survey. Journal of Combinatorial Mathematics and Combinatorial Computing **1**, 13–22 (1987)
4. Bateni, M., Hajiaghayi, M.T., Marx, D.: Approximation schemes for steiner forest on planar graphs and graphs of bounded treewidth. Journal of the ACM **58**, 21:1–21:37 (2011)

5. Bodlaender, H.L., Brettell, N., Johnson, M., Paesani, G., Paulusma, D., van Leeuwen, E.J.: Steiner trees for hereditary graph classes: A treewidth perspective. Theoretical Computer Science **867**, 30–39 (2021)
6. Bodlaender, H.L., Hanaka, T., Kobayashi, Y., Kobayashi, Y., Okamoto, Y., Otachi, Y., van der Zanden, T.C.: Subgraph Isomorphism on graph classes that exclude a substructure. Algorithmica **82**, 3566–3587 (2020)
7. Bulteau, L., Dabrowski, K.K., Köhler, N., Ordyniak, S., Paulusma, D.: An algorithmic framework for locally constrained homomorphisms. Proc. WG 2022, LNCS **13453**, 114–128 (2022)
8. Drange, P.G., Dregi, M.S., van 't Hof, P.: On the computational complexity of vertex integrity and component order connectivity. Algorithmica **76**, 1181–1202 (2016)
9. Dvorák, P., Eiben, E., Ganian, R., Knop, D., Ordyniak, S.: Solving integer linear programs with a small number of global variables and constraints. Proc. IJCAI 2017 pp. 607–613 (2017)
10. Feldmann, A.E., Lampis, M.: Parameterized algorithms for steiner forest in bounded width graphs. Proc. ICALP 2024, LIPIcs **to appear** (2024)
11. Fujita, S., Furuya, M.: Safe number and integrity of graphs. Discrete Applied Mathematics **247**, 398–406 (2018)
12. Fujita, S., MacGillivray, G., Sakuma, T.: Safe set problem on graphs. Discrete Applied Mathematics **215**, 106–111 (2016)
13. Gima, T., Hanaka, T., Kiyomi, M., Kobayashi, Y., Otachi, Y.: Exploring the gap between treedepth and vertex cover through vertex integrity. Theoretical Computer Science **918**, 60–76 (2022)
14. Golovach, P.A., Paulusma, D.: List coloring in the absence of two subgraphs. Discrete Applied Mathematics **166**, 123–130 (2014)
15. Johnson, M., Martin, B., Oostveen, J.J., Pandey, S., Paulusma, D., Smith, S., van Leeuwen, E.J.: Complexity framework for forbidden subgraphs I: The Framework. CoRR **abs/2211.12887** (2022)
16. Johnson, M., Martin, B., Pandey, S., Paulusma, D., Smith, S., van Leeuwen, E.J.: Edge Multiway Cut and Node Multiway Cut are NP-hard on subcubic graphs. CoRR **abs/2211.12203** (2022)
17. Johnson, M., Martin, B., Pandey, S., Paulusma, D., Smith, S., van Leeuwen, E.J.: Complexity framework for forbidden subgraphs III: When problems are polynomial on subcubic graphs. Proc. MFCS 2023, LIPIcs **272**, 57:1–57:15 (2023)
18. Kamiński, M.: Max-Cut and containment relations in graphs. Theoretical Computer Science **438**, 89–95 (2012)
19. Martin, B., Pandey, S., Paulusma, D., Siggers, M., Smith, S., van Leeuwen, E.J.: Complexity framework for forbidden subgraphs II: When hardness is not preserved under edge subdivision. CoRR **abs/2211.14214** (2022)