RAIRO-Theor. Inf. Appl. 57 (2023) 3
https://doi.org/10.1051/ita/2023002

RAIRO - Theoretical Informatics and Applications
www.rairo-ita.org

ON THE COMPUTATIONAL DIFFICULTY OF THE TERMINAL CONNECTION PROBLEM*

Alexsander A. de Melo[1],**, Celina M.H. de Figueiredo[1] and Uéverton S. Souza[2]

Abstract. A *connection tree* of a graph G for a *terminal set* W is a tree subgraph T of G such that leaves$(T) \subseteq W \subseteq V(T)$. A non-terminal vertex is called *linker* if its degree in T is exactly 2, and it is called *router* if its degree in T is at least 3. The Terminal connection problem (TCP) asks whether G admits a connection tree for W with at most ℓ linkers and at most r routers, while the Steiner tree problem asks whether G admits a connection tree for W with at most k non-terminal vertices. We prove that, if $r \geq 1$ is fixed, then TCP is polynomial-time solvable when restricted to split graphs. This result separates the complexity of TCP from the complexity of Steiner tree, which is known to be NP-complete on split graphs. Additionally, we prove that TCP is NP-complete on strongly chordal graphs, even if $r \geq 0$ is fixed, whereas Steiner tree is known to be polynomial-time solvable. We also prove that, when parameterized by clique-width, TCP is W[1]-hard, whereas Steiner tree is known to be in FPT. On the other hand, agreeing with the complexity of Steiner tree, we prove that TCP is linear-time solvable when restricted to cographs (*i.e.* graphs of clique-width 2). Finally, we prove that, even if either $\ell \geq 0$ or $r \geq 0$ is fixed, TCP remains NP-complete on graphs of maximum degree 3.

Mathematics Subject Classification. 68Q17, 68Q25, 68R10, 05C40, 05C85.

Received March 21, 2022. Accepted January 31, 2023.

1. Introduction

Steiner tree is one of the most fundamental network design problems, proved to be NP-complete by Karp in his seminal paper [20]. Besides being related to several real-world applications, Steiner tree is of great theoretical interest, and it has been extensively studied from the perspective of graph theory [4, 9, 16, 29, 33] and computational complexity [2, 8, 12, 30]. The Steiner tree problem has as input a connected graph G, a non-empty terminal set $W \subseteq V(G)$, and a non-negative integer k, and it asks whether there exists a connected subgraph T of G such that $W \subseteq V(T)$ and $|V(T) \setminus W| \leq k$. Such a connected subgraph T admits a spanning tree with at most k non-terminal vertices. In this paper, we analyse the computational complexity of a network design problem closely related to Steiner tree, called Terminal connection.

*This work was supported by the Brazilian agencies CAPES (Finance Code: 001), CNPq (Grant Numbers: 140399/2017-8, 407635/2018-1, 303726/2017-2) and FAPERJ (Grant Numbers: E-26/202.793/2017 and E-26/203.272/2017).

Keywords and phrases: Computational difficulty of problems, parameterized complexity, terminal vertices, connection tree, steiner tree, split graphs, strongly chordal graphs, cographs, bounded degree.

[1] Federal University of Rio de Janeiro, Rio de Janeiro, Brazil.
[2] Fluminense Federal University, Niterói, Brazil.

** Corresponding author: aamelo@cos.ufrj.br

Let G be a graph and $W \subseteq V(G)$ be a non-empty set. A *connection tree* T of G for W is a tree subgraph of G such that $\text{leaves}(T) \subseteq W \subseteq V(T)$, where $\text{leaves}(T)$ denotes the leaf set of T. In a connection tree T for W, the vertices belonging W are called *terminal*, and the vertices belonging to $V(T) \setminus W$ are called *non-terminal* and are classified into two types according to their respective degrees in T, namely: the non-terminal vertices of degree exactly 2 in T are called *linkers* and the non-terminal vertices of degree at least 3 in T are called *routers* *cf.* [11]. We remark that the vertex set of every connection tree can be partitioned into terminal vertices, linkers and routers. For each connection tree T, we let $\mathsf{L}(T)$ denote the linker set of T and $\mathsf{R}(T)$ denote the router set of T. Next, we present a formal definition for the TERMINAL CONNECTION problem.

TERMINAL CONNECTION (TCP)

Input:	A connected graph G, a non-empty terminal set $W \subseteq V(G)$ and two non-negative integers ℓ and r.
Question:	Does there exist a connection tree T of G for W such that $\lvert\mathsf{L}(T)\rvert \leq \ell$ and $\lvert\mathsf{R}(T)\rvert \leq r$?

TCP was introduced by Dourado *et al.* [11], having as motivation applications in information security and network routing, and it was proved to be polynomial-time solvable when the parameters ℓ and r are both fixed [11]. Nevertheless, it was proved to be NP-complete even if either $\ell \geq 0$ or $r \geq 0$ is fixed [11]. In particular, the problem was proved to be NP-complete even if $\ell \geq 0$ is fixed and the input graph has constant maximum degree [10].

There is a straightforward Turing reduction from STEINER TREE to TCP, namely: (G, W, k) is a yes-instance of STEINER TREE if and only if (G, W, ℓ, r) is a yes-instance of TCP for some pair $\ell, r \in \{0, \ldots, k\}$ such that $\ell + r = k$. An interesting aspect of this Turing reduction is the fact that it preserves the structure of the input graph. Consequently, if TCP is polynomial-time solvable on some graph class $\mathcal{G}$, then so is STEINER TREE. Analogously, if STEINER TREE is NP-complete on some graph class $\mathcal{G}$, then TCP cannot be solved in polynomial-time on $\mathcal{G}$, unless P=NP. Nevertheless, if either $\ell \geq 0$ or $r \geq 0$ is fixed, then possibly TCP is polynomial-time solvable on a graph class $\mathcal{G}$, while STEINER TREE remains NP-complete on $\mathcal{G}$. In addition, there might exist a graph class $\mathcal{G}$ on which STEINER TREE is polynomial-time solvable whereas TCP remains NP-complete.

In this work, we confirm the existence of such complexity separating classes. In Section 2, we prove that, on *split* graphs, TCP is polynomial-time solvable if $r \geq 1$ is fixed, whereas STEINER TREE is known to be NP-complete [33]. Besides, we prove that, on *strongly chordal* graphs, TCP remains NP-complete even if $r \geq 0$ is fixed, whereas STEINER TREE is known to be polynomial-time solvable [33]. Also, we prove in Section 3.1 that, parameterized by clique-width, TCP is W[1]-hard, whereas STEINER TREE is known to be in FPT [1].

On the other hand, in Section 3.2, we prove that TCP can be solved in linear-time on *cographs* (*i.e.* graphs of clique-width 2), agreeing with the computational complexity of STEINER TREE [4]. Additionally, in Section 4, we prove that TCP remains NP-complete on graphs of maximum degree 3 even if either $\ell \geq 0$ or $r \geq 0$ is fixed. It is worth mentioning that, although STEINER TREE is known to be NP-complete on graphs of maximum degree 3 [22], our NP-completeness results of TCP with either $\ell \geq 0$ or $r \geq 0$ fixed do not immediately follow from the NP-completeness of STEINER TREE.

Table 1 summarises the mentioned results.

1.1. Related works

Motivated by applications in optical networks and bandwidth consumption minimization, another variant of STEINER TREE that has been investigated is the one in which the number of *branching nodes* in the sought tree T, *i.e.* vertices (which not necessarily are non-terminal) of degree at least 3 in T, is bounded. In [17, 31, 32], the authors addressed the undirected and directed cases of this variant, for which they devised approximation and parameterized tractable algorithms, apart from obtaining some intractability results.

In addition, Dourado *et al.* introduced in [10] the *strict* variant of TCP, called STRICT TERMINAL CONNECTION problem (S-TCP), which has the same input of TCP but further requires that the sought connection tree T

TABLE 1. Comparison between the computational complexity of TCP with the computational complexity of STEINER TREE.

	Problem			
Graph class/Parameter	TCP	TCP fixed ℓ	TCP fixed r	STEINER TREE
Split	NP-c Thm. 2.2	NP-c Thm. 2.2	Poly, for $r \geq 1$ Thm. 2.1	NP-c [33]
Strongly chordal	NP-c Thm. 2.6	Open	NP-c Thm. 2.6	Poly [33]
Clique-width	W[1]-h Thm. 3.1	Open	W[1]-h Thm. 3.1	FPT [1]
Cographs	Poly Thm. 3.9	Poly Thm. 3.9	Poly Thm. 3.9	Poly [4]
Maximum degree 3	NP-c Thms. 4.1 and 4.3	NP-c Thm. 4.1	NP-c Thm. 4.3	NP-c [22]

satisfies leaves$(T) = W \subseteq V(T)$. It is worth mentioning that, just as TCP can be seen as a generalization of STEINER TREE, S-TCP can be seen as a generalization of FULL STEINER TREE, which is a widely studied variant of STEINER TREE [18, 21, 23]. Similarly to TCP, it was proved that S-TCP is polynomial-time solvable when the parameters $\ell \geq 0$ and $r \geq 0$ are both fixed [10], and that the problem is still NP-complete if $\ell \geq 0$ is fixed [10]. Nevertheless, except for the case $r \in \{0, 1\}$, which was shown to be polynomial-time solvable [24], the complexity of S-TCP for fixed $r \geq 2$ has remained open. Motivated by this question, S-TCP was also investigated in [25, 27]. In particular, in [25], S-TCP was proved to be NP-complete (and W[2]-hard when parameterized by r), even if $\ell \geq 0$ is constant and the input graph is restricted to split graphs. An interesting fact of this proof is that it can be easily adapted to TCP. Consequently, we obtain that TCP is also NP-complete (and W[2]-hard when parameterized by r) on split graphs even if $\ell \geq 0$ is constant. Besides this result, it was analysed in [25] the complexity of S-TCP when restricted to graphs of bounded maximum degree, and it was also proved that S-TCP is polynomial-time solvable on cographs.

A previous version of this work appeared as an extended abstract at SOFSEM 2021 conference [26]. Besides the full proofs omitted in [26], the present paper contains further contributions, such as the tractability of TCP on split graphs and the W[1]-hardness of TCP parameterized by clique-width.

1.2. Graph notation

Now, we present some basic notation and terminologies of graph theory that are used throughout this paper. For any missing definition or terminology, we refer to [3].

In this work, all graphs are finite, simple and undirected. Let G be a graph. We let $V(G)$ and $E(G)$ denote the vertex set and the edge set of G, respectively. For every vertex $u \in V(G)$, we let $N_G(u)$ and $N_G[u] = N_G(u) \cup \{u\}$ denote the *(open) neighbourhood* and the *closed neighbourhood* of u in G, respectively; and we let $d_G(u) = |N_G(u)|$ denote the *degree* of u in G. Two distinct vertices $u, v \in V(G)$ are said to be *false twins* (resp. *true twins*) in G if in G if $N_G(u) = N_G(v)$ (resp. $N_G[u] = N_G[v]$). The *length* of a path P is defined as the number of edges of P. The *distance* between two vertices $u, v \in V(G)$ is the length of a path of G between u and v of minimum length. For every non-empty subset $S \subseteq V(G)$, we let $G[S]$ denote the *subgraph of G induced by S*.

Let $G_1, \ldots, G_k$ be $k \geq 2$ graphs. The *disjoint union* of $G_1, \ldots, G_k$ is the graph H, denoted by $G_1 \cup \cdots \cup G_k$, with vertex set $V(H) = V(G_1) \uplus \cdots \uplus V(G_k)$ and edge set $E(H) = E(G_1) \uplus \cdots \uplus E(G_k)$. The *join* of $G_1, \ldots, G_k$

is the graph H, denoted by $G_1 \wedge \cdots \wedge G_k$, with vertex set $V(H) = V(G_1 \cup \cdots \cup G_k)$ and edge set

$$E(H) = E(G_1 \cup \cdots \cup G_k) \uplus \{uv \mid u \in V(G_i),\, v \in V(G_j),\, i, j \in \{1, \ldots, k\},\, i \neq j\}.$$

2. Separating classes: Split and strongly chordal

In this section, we present the main results of this work. First, we prove that, when restricted to split graphs, TCP is polynomial-time solvable if $r \geq 1$ is fixed. Second, we prove that, when restricted to strongly chordal graphs, TCP is NP-complete even if $r \geq 0$ is fixed. Such results separate the complexity of TCP from the complexity of Steiner tree, since Steiner tree is known to be NP-complete on split graphs [33] and polynomial-time solvable on strongly chordal graphs [33].

2.1. Split graphs

A *split* graph is a graph whose vertex set can be partitioned into a clique and a stable set. In what follows, we prove the following theorem.

Theorem 2.1. *For $r \geq 1$, TCP can solved in time $n^{\mathcal{O}(r)}$ on split graphs.*

To prove Theorem 2.1, we propose a polynomial-time reduction from TCP, with $r \geq 1$, to its strict variant S-TCP, in which the terminal vertices are further required to coincide with the leaf set of the sought tree. S-TCP was shown to admit an $n^{\mathcal{O}(r)}$-time algorithm on split graphs [25]. In a nutshell, the algorithm presented in [25] enumerates each possible candidate router set $R \subseteq V(G) \setminus W$, with $|R| \leq r$, and then decides through matching techniques whether the input graph G admits a connection tree T for the terminal set W, such that $|\mathsf{L}(T)| \leq \ell$, $\mathsf{R}(T) = R$ and $\text{leaves}(T) = W$. Thus, combining our polynomial-time reduction with this algorithm, we obtain that TCP can be solved in time $n^{\mathcal{O}(r)}$ on split graphs for $r \geq 1$.

It is worth mentioning that this result is optimum, *i.e.* the $n^{\mathcal{O}(r)}$-time complexity cannot be considerably improved. Indeed, the following theorem immediately comes from a trivial adaptation of a parameterized polynomial-time reduction from the Set cover problem to S-TCP presented in [25] (see Theorem 7 of [25]).

Theorem 2.2 ([25])**.** *For any computable functions f and h, TCP cannot be solved in time $f(r) \cdot n^{h(\ell)}$, unless $\mathsf{FPT} = \mathsf{W}[2]$, and cannot be solved in time $f(r) \cdot n^{o(r)}$, unless ETH fails.*

In what follows, we write $G\langle K, S\rangle$ to refer a split graph G and explicitly denote that $K \cup S$ is a partition of the vertex set of G into a clique K and a stable set S.

Lemma 2.3. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$. If $|W| \geq 3$, $W \cap K = \emptyset$ and there exists a connection tree T of G for W such that $\mathsf{R}(T) = \emptyset$, then there exists a connection tree T' of G for W such that $\mathsf{L}(T') \subseteq |\mathsf{L}(T)|$ and $|\mathsf{R}(T')| = 1$.*

Proof. Since $|W| \geq 3$ and $\mathsf{R}(T) = \emptyset$, there exists a terminal vertex $w \in W$ whose degree in T is at least 2. Then, let u and u' be two distinct neighbours of w in T. Since $W \cap K = \emptyset$, $u, u' \in \mathsf{L}(T) \cap K$. Let T' be the graph obtained from T by removing the edge wu' and adding the edge uu'. Clearly, T' is a connection tree of G for W such that $\mathsf{L}(T') = \mathsf{L}(T) \setminus \{u\}$ and $\mathsf{R}(T') = \mathsf{R}(T) \cup \{u\}$. □

Lemma 2.4. *Let $G\langle K, S\rangle$ be a split graph and $W \subseteq V(G)$ be a non-empty set. Suppose that G admits a connection tree T for W. There exists a connection tree T' of G for W, with $\mathsf{L}(T') \subseteq \mathsf{L}(T)$ and $|\mathsf{R}(T')| \leq |\mathsf{R}(T)|$, that simultaneously satisfies the following conditions:*

1. *$\mathsf{L}(T') \subseteq K$ and $\mathsf{R}(T') \subseteq K$;*
2. *If $\mathsf{R}(T) \cap K \neq \emptyset$ or $W \cap K \neq \emptyset$, then every vertex in $W \cap S$ is a leaf of T'.*

Proof. (1) Suppose that $(\mathsf{L}(T) \cup \mathsf{R}(T)) \cap S \neq \emptyset$. Then, there exists a vertex $u \in V(T) \cap K$. Let T' be the graph obtained from T as follows:

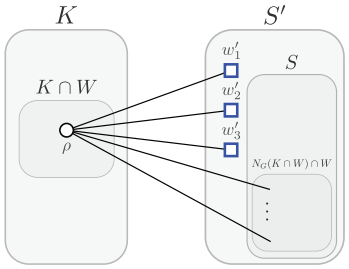


FIGURE 1. Split graph $G'\langle K, S'\rangle$ of the instance $g(I)$ of S-TCP described in Construction 1, obtained from a split graph $G\langle K, S\rangle$ of an instance I of TCP, with $K \cap W \neq \emptyset$.

- Remove all vertices belonging to $(\mathsf{L}(T) \cup \mathsf{R}(T)) \cap S$ and their incident edges;
- For each $u' \in \mathsf{L}(T) \cap S$, add the edge vv', where $N_T(u') = \{v, v'\}$;
- For each $u' \in N_T(\mathsf{R}(T) \cap S)$, add the edge uu'.

Clearly, T' is a connection tree of G for W such that $\mathsf{L}(T') \subseteq K$ and $\mathsf{R}(T') \subseteq K$. Moreover, note that $\mathsf{L}(T') \subseteq \mathsf{L}(T) \setminus S$, $\mathsf{R}(T') = \mathsf{R}(T)$ if $\mathsf{R}(T) \cap S = \emptyset$, and $\mathsf{R}(T') \subseteq (\mathsf{R}(T) \cup \{u\}) \setminus S$ otherwise.

(2) Suppose that $W \cap S \neq \emptyset$ and that there exists a vertex $u \in (\mathsf{R}(T) \cup W) \cap K$. Note that, in this case, every vertex $w \in W \cap S$ has at least one neighbour, say $\alpha(w)$, in T. Then, let T' be the graph obtained from T as follows:

- For each $w \in W \cap S$, remove all edges of T that are incident to w except for $w\alpha(w)$; additionally, for each $v \in N_T(w)$, add the edge uv.

Clearly, T' is a connection tree of G for W such that every vertex in $W \cap S$ is a leaf of T'. Furthermore, one can verify that $\mathsf{L}(T') = \mathsf{L}(T)$ and $\mathsf{R}(T') = \mathsf{R}(T)$. □

Next, we present our polynomial-time reduction to S-TCP.

Construction 1 (Reduction from TCP to S-TCP on split graphs)**.** Let $G\langle K, S\rangle$ be a split graph and $I = (G, W, \ell, r)$ be an instance of TCP. If $W \cap K = \emptyset$, then we define our reduction instance of S-TCP as simply $g(I) = I$. Otherwise, let $\rho \in W \cap K$ and consider the graph G' obtained from G as follows (see Fig. 1):

- Add all vertices and all edges of G;
- For each $u \in W \cap S \cap N_G(W \cap K) \setminus N_G(\rho)$, add the edge ρu;
- Add three new vertices w'_1, w'_2 and w'_3, and make them adjacent to ρ.

Note that G' is a split graph, and that $K \cup S'$ is a partition of $V(G')$ into a clique and a stable set, where $S' = S \cup \{w'_1, w'_2, w'_3\}$. We then define our reduction instance of S-TCP as $g(I) = (G', W', \ell, r + 1)$, where $W' = (W \setminus \{\rho\}) \cup \{w'_1, w'_2, w'_3\}$.

The following lemma concludes the proof of Theorem 2.1.

Lemma 2.5. *Let $G\langle K, S\rangle$ be a split graph and $I = (G, W, \ell, r)$ be an instance of TCP such that $|W| \geq 3$. Also, let $g(I)$ be the instance of S-TCP obtained from I, as described in Construction 1. If $r \geq 1$ or $W \cap K \neq \emptyset$, then I is a yes-instance of TCP if and only if $g(I)$ is a yes-instance of S-TCP.*

Proof. First, suppose that I is a yes-instance of TCP. Then, there exists a connection tree T of G for W such that $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq r$. By Lemma 2.3, we can assume that $\mathsf{R}(T) \neq \emptyset$ or $W \cap K \neq \emptyset$. Furthermore, by Lemma 2.4, we can assume that every vertex in $W \cap S$ is a leaf of T. This implies $W \setminus \text{leaves}(T) \subseteq K$. If $W \cap K = \emptyset$, then $\text{leaves}(T) = W$ and, therefore, we immediately obtain that $g(I) = I$ is a yes-instance of S-TCP. Thus, suppose that $W \cap K \neq \emptyset$. Additionally, by Lemma 2.4, assume that $\mathsf{L}(T) \subseteq K$ and $\mathsf{R}(T) \subseteq K$. Note that every vertex in $V(T) \cap S$ is a leaf of T. Since T is a tree and $\rho \in V(T)$, for each vertex $w \in W \cap K \setminus \{\rho\}$, there

exists a single path between ρ and w in T and a single vertex in this path, say $\alpha(w)$, that belongs to $N_T(w) \cap K$. Thus, let T' be the graph obtained from T as follows:

- For each $w \in W \cap K \setminus \{\rho\}$ and each $w' \in N_T(w) \setminus \{\alpha(w)\}$, remove the edge ww' and add the edge $\rho w'$;
- For each $i \in \{1, 2, 3\}$, add the vertex w'_i and the edge $\rho w'_i$.

One can verify that T' is a connection tree of G' for W', such that $\mathsf{leaves}(T') = W'$, $\mathsf{L}(T') = \mathsf{L}(T)$ and $\mathsf{R}(T) = \mathsf{R}(T') \cup \{\rho\}$.

Conversely, suppose that $g(I)$ is a yes-instance of S-TCP. If $W \cap K = \emptyset$, then $g(I) = I$ and, therefore, I is a yes-instance of TCP. Thus, suppose that $W \cap K \neq \emptyset$, and let T' be a connection tree of G' for W', such that $\mathsf{leaves}(T') = W'$, $|\mathsf{L}(T')| \leq \ell$ and $|\mathsf{R}(T')| \leq r + 1$. Since the only neighbour of the terminal vertices w'_1, w'_2 and w'_3 in G' is the vertex ρ, we have that ρ necessarily belongs to T' and, besides that, is a router of T'. Moreover, by construction of G', if a vertex w is a neighbour of ρ in T' but is not a neighbour of ρ in G, then $w \in W \cap S$ and there exists a vertex in $W \cap K$, say $\beta(w)$, which is a neighbour of w in G. Then, let T be the graph obtained from T' as follows:

- Remove the vertices w'_1, w'_2 and w'_3 and their incident edges;
- For each $w \in N_{T'}(\rho) \setminus N_G(\rho)$, remove the edge ρw and add the edge $\beta(w)w$.

One can verify that T is a connection tree of G for W, such that $\mathsf{L}(T) = \mathsf{L}(T')$ and $\mathsf{R}(T) = \mathsf{R}(T') \setminus \{\rho\}$. □

2.2. Strongly chordal graphs

A *chord* of a cycle C is an edge between any two non-consecutive vertices of C. A graph G is called *chordal* if every cycle of G of length at least 4 has a chord. In other words, a graph G is chordal if every induced cycle of G has length 3. An *even cycle* is a cycle of even length. A chord uv of an even cycle C is called an *odd chord* if the distance between u and v in C is odd. A graph G is called *strongly chordal* if it is chordal and every even cycle of G of length at least 6 has an odd chord.

A vertex u of a graph G is called a *simple vertex* if, for any two vertices $v, v' \in N_G(u)$, $N_G[v] \subseteq N_G[v']$ or $N_G[v'] \subseteq N_G[v]$. In other words, a vertex u of a graph G is simple if the collection $\{N_G[v] \mid v \in N_G(u)\}$ can be linearly ordered by set inclusion. Farber [13] proved that a graph G is strongly chordal if and only if there exists a linear ordering $(u_1, \ldots, u_n)$ of the vertices of G, called *simple elimination ordering*, such that u_i is a simple vertex of $G[\{u_i, \ldots, u_n\}]$ for each $i \in \{1, \ldots, n\}$.

We prove that TCP remains NP-complete on strongly chordal graphs:

Theorem 2.6. *For each $r \geq 0$, TCP remains NP-complete when restricted to strongly chordal graphs.*

In order to prove Theorem 2.6, we provide a polynomial-time reduction from the Hamiltonian path problem, which was shown to be NP-complete on strongly chordal graphs by Müller [28]. The Hamiltonian path problem has as input a graph G and asks whether G admits a *Hamiltonian path*, *i.e.* a path that contains all vertices of G.

The next lemma presents some important properties of the class of strongly chordal graphs, which are used in our reduction.

Lemma 2.7. *The class of strongly chordal graphs is closed under the following operations:*

1. *Adding true twin vertices;*
2. *For any pair of true twin vertices v and v', adding a new vertex w and adding the edges vw and wv'.*

Proof. Let G be a strongly chordal graph and $(u_1, \ldots, u_n)$ be a simple elimination ordering of G. For each $i \in \{1, \ldots, n\}$, let G_i denote $G[\{u_i, \ldots, u_n\}]$.

(1) Let H be the graph obtained from G by adding a true twin v of u_i, for some $i \in \{1, \ldots, n\}$. We claim that

$$(u_1, \ldots, u_i, v, u_{i+1} \ldots, u_n)$$

is a simple elimination ordering of H. First, we show that v is a simple vertex of H_v, where H_v denotes $H[\{v, u_{i+1}, \ldots, u_n\}]$. Since u_i and v are true twins in H, $N_{H_v}[x] = (N_{G_i}[x] \setminus \{u_i\}) \cup \{v\}$ and $N_{H_v}[y] = (N_{G_i}[y] \setminus \{u_i\}) \cup \{v\}$ for every pair $x, y \in N_{H_v}(v)$. Moreover, since u_i is a simple vertex of G_i, we have that $N_{G_i}[x] \subseteq N_{G_i}[y]$ or $N_{G_i}[y] \subseteq N_{G_i}[x]$ for every pair $x, y \in N_{G_i}(v)$. Finally, we remark that $N_{H_v}(v) = N_{G_i}(u_i)$. Then, let $x, y \in N_{H_v}(v)$ and assume without loss of generality that $N_{G_i}[x] \subseteq N_{G_i}[y]$. One can verify that

$$N_{H_v}[x] = (N_{G_i}[x] \setminus \{u_i\}) \cup \{v\} \subseteq (N_{G_i}[y] \setminus \{u_i\}) \cup \{v\} = N_{H_v}[y].$$

Therefore, v is indeed a simple vertex of H_v.

Now, let $j \in \{1, \ldots, n\}$. We prove that v_j is a simple vertex of H_j, where H_j denotes $H[\{u_j, \ldots, u_i, v, u_{i+1}, \ldots, u_n\}]$ if $j \leq i$, and $H[\{u_j, \ldots, u_n\}]$ otherwise. Note that, if $j \geq i + 1$, then u_j is trivially a simple vertex of H_j, since in this case $H_j = G_j$. Thus, assume that $j \leq i$. One can verify that, for every $x \in V(H_j) \setminus \{v\}$,

$$N_{H_j}[x] = \begin{cases} N_{G_j}[x] \cup \{v\} & \text{if } u_i \in N_G(x) \\ N_{G_j}[x] & \text{otherwise.} \end{cases}$$

Let $x, y \in N_{H_j}(u_j)$. We prove that $N_{H_j}[x] \subseteq N_{H_j}[y]$ or $N_{H_j}[y] \subseteq N_{H_j}[x]$, implying that u_j is indeed a simple vertex of H_j. First, suppose that $y = v$. Note that, if $N_{G_j}[x] \subseteq N_{G_j}[u_i]$, then

$$N_{H_j}[x] = N_{G_j}[x] \cup \{v\} \subseteq N_{G_j}[u_i] \cup \{v\} = N_{H_j}[u_i] = N_{H_j}[v].$$

On the other hand, if $N_{G_j}[u_i] \subseteq N_{G_j}[x]$, then

$$N_{H_j}[v] = N_{H_j}[u_i] = N_{G_j}[u_i] \cup \{v\} \subseteq N_{G_j}[x] \cup \{v\} = N_{H_j}[x].$$

Now, suppose that $x \neq v$ and $y \neq v$. Assume without loss of generality that $N_{G_j}[x] \subseteq N_{G_j}[y]$. Note that, if $u_i \in N_{G_j}(x)$, then $u_i \in N_{G_j}(y)$ and, thus,

$$N_{H_j}[x] = N_{G_j}[x] \cup \{v\} \subseteq N_{G_j}[y] \cup \{v\} = N_{H_j}[y].$$

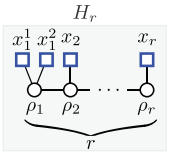
On the other hand, if $u_i \notin N_G(x)$, then

$$N_{H_j}[x] = N_{G_j}[x] \subseteq N_{G_j}[y] \subseteq N_{H_j}[y].$$

(2) Let H be the graph obtained from G by adding a new vertex w and adding the edges vw and $v'w$, where v and v' are true twins of G. Since $N_H(w) = \{v, v'\}$ and $N_H[v] = N_H[v']$, it is immediate that w is a simple vertex of $H[\{w, u_1, \ldots, u_n\}]$. Moreover, for every $i \in \{1, \ldots, n\}$, u_i is a simple vertex of $G[\{u_i, \ldots, u_n\}]$. Thus,

$$(w, u_1, \ldots, u_i, \ldots, u_n)$$

is a simple elimination ordering of H, and therefore, H is strongly chordal. □



[th]

FIGURE 2. Gadget H_r for $r \geq 1$, described in Construction 2.

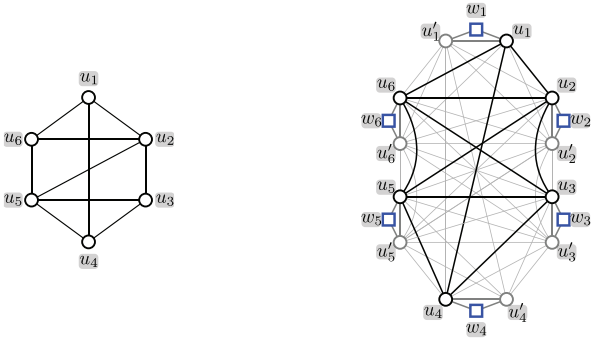


FIGURE 3. A graph G and the graph G' obtained from G (and $r = 0$) as described in Construction 3.

Construction 2 (Gadget H_r and Terminal Set W_r)**.** Let r be a positive integer. We define the gadget H_r as the graph such that (see Fig. 2)

$$V(H_r) = \{\rho_1, \ldots, \rho_r\} \cup \{x_1^1, x_1^2\} \cup \{x_i \mid i \in \{2, \ldots, r\}\} \text{ and}$$
$$E(H_r) = \{\rho_i\rho_{i+1} \mid i \in \{1, \ldots, r-1\}\} \cup \{x_1^1\rho_1, x_1^2\rho_1\} \cup \{x_i\rho_i \mid i \in \{2, \ldots, r\}\}.$$

Moreover, we let $W_r = \{x_1^1, x_1^2\} \cup \{x_2, \ldots, x_r\}$ be the terminal set of H_r.

Construction 3 (Reduction from HAMILTONIAN PATH to TCP)**.** Let G be a graph, with vertex set $V(G) = \{u_1, \ldots, u_n\}$, and r be a non-negative integer. We let G' be the graph obtained from G and r as follows (see Fig. 3):

- Add all vertices and all edges of G to G';
- For each vertex $u_i \in V(G)$, add a true twin u'_i of u_i, in such a way that $N_{G'}[u'_i] = N_{G'}[u_i]$;
- For each vertex $u_i \in V(G)$, add a new vertex w_i and add the edges u_iw_i and u'_iw_i, where u'_i denotes the true twin of u_i added in the last step;
- If $r \geq 1$, create the gadget H_r and define the terminal set W_r as described in Construction 2, besides adding the edge $\rho_r w_1$; otherwise, if $r = 0$, define $W_r = \emptyset$.

We then define our reduction instance of TCP as $g(G, r) = (G', W, \ell, r)$, where $W = \{w_1, \ldots, w_n\} \cup W_r$ and $\ell = 2n - 2$.

We remark that, the graph G' described in Construction 3 is similar to the one constructed in [11] to prove the NP-completeness of TCP on general graphs for fixed $r \geq 0$. The main difference is the fact that, in the graph constructed in [11], for each $u_i \in V(G)$, it is added a false twin, instead of a true twin, of u_i. However, this makes the original graph not be strongly chordal, even if the input graph is strongly chordal; for instance, a cycle C_3 of length 3 is strongly chordal, but the graph resulting from adding a false twin for each vertex of C_3 is not strongly chordal, since it contains an induced cycle of length 4. The next lemma, which states that,

whenever the input graph is strongly chordal, our constructed graph is strongly chordal as well, immediately follows from Lemma 2.7 and from the fact that the vertices of H_r are not contained in any cycle of G'.

Lemma 2.8. *Let G be a graph and r be a non-negative integer. Also, let G' be the graph of the instance $g(G,r)$ of TCP obtained from G and r, as described in Construction 3. If G is strongly chordal, then so is G'.*

Lemma 2.9. *Let G be a graph and r be a non-negative integer. Also, let $g(G,r)$ be the instance of TCP obtained from G and r, as described in Construction 3. Then, G admits a Hamiltonian path if and only if $g(G,r)$ is a* yes*-instance of TCP.*

Proof. Assume that $V(G) = \{u_1, \ldots, u_n\}$ and that $g(G,r) = (G', W, \ell, r)$. Additionally, for simplicity, consider $W_r = V(H_r) = E(H_r) = \emptyset$ if $r = 0$.

First, suppose that there exists in G a Hamiltonian path $(u_{j_1}, \ldots, u_{j_n})$. Then, let T be the graph with vertex set

$$V(T) = V(H_r) \cup V(P) \cup \{w_{j_1}, u'_{j_1}, u_{j_n}, w_{j_n}\} \cup \{u_{j_i}, w_{j_i}, u'_{j_i} \mid i \in \{2, \ldots, n-1\}\}$$

and edge set

$$\begin{aligned} E(T) = {} & E(H_r) \cup \{\rho_r w_1, w_1 u'_1\} \\ & \cup \{u'_{j_{i-1}} u_{j_i}, u_{j_i} w_{j_i}, w_{j_i} u'_{j_i} \mid i \in \{2, \ldots, n-1\}\} \cup \{u'_{j_{n-1}} u_{j_n}, u_{j_n} w_{j_n}\}, \end{aligned}$$

where u'_{j_i} denotes the true twin of u_{j_i} added in the construction of G'. Note that T is a connection tree of G' for W with $\mathsf{L}(T) = \{u'_{j_1}, u_{j_n}\} \cup \{u_{j_2}, u'_{j_2}, \ldots, u_{j_{n-1}}, u'_{j_{n-1}}\}$ and $\mathsf{R}(T) = \{\rho_1, \ldots, \rho_r\}$. Therefore, $g(G,r)$ is a yes-instance of TCP.

Conversely, suppose that $g(G,r)$ is a yes-instance of TCP. Let T be a connection tree of G' for W such that $|\mathsf{L}(T)| \leq 2n-2$ and $|\mathsf{R}(T)| \leq r$. We remark that ρ_1 is the only neighbour of the terminal vertices $x_1^1, x_1^2 \in W_r$ and, for each $i \in \{2, \ldots, r\}$, ρ_i is the only neighbour of the terminal vertex $x_i \in W_r$. As a result, T must contain all the vertices $\rho_1, \ldots, \rho_r$. More specifically, such vertices must be routers of T. This implies that $T' = T - H_r$ cannot contain any router, and all non-terminal vertices of T' must be linkers. Hence, T' is a path, since, by construction of G', w_i has degree at most 2 in T' for every $i \in \{1, \ldots, n\}$. Then, let $P' = (w_{j_1}, \ldots, w_{j_n})$ be a sequence of distinct vertices such that, for each $i \in \{1, \ldots, n-1\}$, the path in T' between w_{j_i} and $w_{j_{i+1}}$ does not contain any other terminal vertex. Note that, since $|\mathsf{L}(T)| \leq \ell = 2n-2$, every path in T' between any two consecutive vertices w_{j_i} and $w_{j_{i+1}}$ in P' must be of one of the forms: $(w_{j_i}, u'_{j_i}, u'_{j_{i+1}}, w_{j_{i+1}})$, $(w_{j_i}, u'_{j_i}, u_{j_{i+1}}, w_{j_{i+1}})$, $(w_{j_i}, u_{j_i}, u_{j_{i+1}}, w_{j_{i+1}})$, or $(w_{j_i}, u_{j_i}, u'_{j_{i+1}}, w_{j_{i+1}})$. As a result, it follows from the construction of G' that, for each $i \in \{1, \ldots, n-1\}$, u_{j_i} and $u_{j_{i+1}}$ are adjacent in G. Therefore, $(u_{j_1}, \ldots, u_{j_n})$ is a Hamiltonian path of G. □

3. Graphs of bounded clique-width

In this section, we prove that TCP parameterized by the clique-width of the input graph is W[1]-hard. Similarly to the results presented in Section 2, this contrasts with the complexity Steiner tree, since Steiner tree is known to be in FPTwhen parameterized by clique-width [1]. On the other hand, agreeing with the complexity of Steiner tree, we prove that TCP is linear-time solvable on cographs, which are precisely the graphs of clique-width 2.

The notion of clique-width was introduced by Courcelle *et al.* [7], and it is one of the most studied graph parameters. Next, we present the definition of this notion *cf.* [14, 15].

Let k be a positive integer. A graph is called a *k-graph* if its vertices are labelled with integers in $\{1, \ldots, k\}$. An *initial k-graph* is a k-labelled graph on a single vertex. The *clique-width* of a graph G, denoted by $\mathsf{cwd}(G)$, is the smallest positive integer k such that G can be constructed by repeated application of the following four operations:

1. *introducing* (denoted by $\mathsf{int}(u,i)$): construction of an initial k-graph, whose single vertex u is labelled by an integer $i \in \{1,\dots,k\}$ and has not been introduced yet;
2. *disjoint union* (here, denoted by $\oplus$);
3. *relabelling* (denoted by $\mathsf{rel}_{i,j}$): changing all labels i to j, for $i,j \in \{1,\dots,k\}$;
4. *join* (denoted by $\eta_{i,j}$): connecting all vertices labelled by i with all vertices labelled by j, for $i,j \in \{1,\dots,k\}$, $i \neq j$.

A construction of a graph G using the operations (1)-(4) described above can be represented by an algebraic term, called *cwd-expression* defining G, composed of int, $\oplus$, $\mathsf{rel}_{i,j}$, and $\eta_{i,j}$ *cf.* [14], where i and j are distinct positive integers. Note that cwd-expressions define a tree language, where each expression can be represented by a rooted tree T *cf.* [15], where each $\mathsf{int}(u,i)$ of the expression is associated with a leaf of T, and each vertex of G is introduced exactly once. A *k-expression* is a cwd-expression that contains at most k distinct labels *cf.* [14]. Thus, one can verify that, a graph G has clique-width at most k if and only if there exists a k-expression defining G.

3.1. Parameterization by clique-width

Now, we prove the following theorem.

Theorem 3.1. *For each $r \geq 0$, TCP parameterized by clique-width is* W[1]*-hard.*

More specifically, we show that, if a graph G has clique-width at most k for some $k \geq 2$, then the graph G' obtained from G as described in Construction 3 has clique-width at most $k+1$. This, along with Lemma 2.9 and the fact that Hamiltonian path is W[1]-hard parameterized by clique-width [15], implies the W[1]-hardness of TCP.

The following lemma is a well-known fact, and it can be immediately verified by an inductive argument on the number of vertices of the tree.

Lemma 3.2. *Every tree has clique-width at most 3. Moreover, if T is a tree and u is a leaf of T, then there exists a 3-expression defining a construction of T in which at the root all vertices but u have the same label.*

Lemma 3.3. *Let G be a graph. For each $r \geq 0$, if $\mathsf{cwd}(G) = k$ for some $k \geq 2$, then $\mathsf{cwd}(G') \leq k+1$, where G' denotes the graph obtained from G and r as described in Construction 3.*

Proof. Assume that $V(G) = \{u_1,\dots,u_n\}$ and $\mathsf{cwd}(G) \leq k$. Then, let γ_G be a k-expression defining G. Also, let H' be the subgraph of G' induced by $V(H_r) \cup \{w_1\}$. Note that H' is a tree. Thus, by Lemma 3.2, there exists a construction (3-expression) of a vertex-labelled copy of H' (for short γ'_H) in which all vertices but w_1 have the same label. Assume, without loss of generality, that w_1 is labelled by 1 at the root of $\gamma_{H'}$, and that all the other vertices of $\gamma_{H'}$ are labelled by 2. In what follows, we show that we can obtain from γ_G and $\gamma_{H'}$ a $(k+1)$-expression $\gamma_{G'}$ defining our constructed graph G'. We recall that each vertex $u_i \in V(G)$ has a true twin u'_i in G', and that $N_{G'-H_r}(w_i) = \{u_i, u'_i\}$. Consider $b = k+1$. We define $\gamma_{G'}$ as the cwd-expression obtained from γ_G as follows:

- Let $\mathsf{int}(u_1,i)$ be the leaf term of u_1 in γ_G, for $i \in \{2,\dots,k\}$. Replace the occurrence of $\mathsf{int}(u_1,i)$ in γ_G with
$$\mathsf{rel}_{1,b}\Big(\eta_{i,1}\Big(\mathsf{rel}_{b,i}\big(\eta_{i,b}(\mathsf{int}(u_1,i),\mathsf{int}(u'_1,b))\big),\mathsf{rel}_{2,b}(\gamma_{H'})\Big)\Big).$$
For $i = 1$, it is similar (just replace the occurrences of 1 by 2 and vice versa).
- For each $u_j \in V(G) \setminus \{u_1\}$, if $\mathsf{int}(u_j,i)$ is the leaf term of u_j in γ_G, replace the occurrence of $\mathsf{int}(u_j,i)$ with
$$\eta_{i,b}\Big(\mathsf{rel}_{b,i}\big(\eta_{i,b}(\mathsf{int}(u_j,i),\mathsf{int}(u'_j,b))\big),\mathsf{int}(w_j,b)\Big).$$

We recall that, besides being represented by leaves, each vertex is introduced exactly once in a expression tree. Moreover, we note that the operations described above consists in local replacements in the corresponding leaves of the expression tree associated to γ_G. Thus, one can verify that $\gamma_{G'}$ defines G'. In addition, it is straightforward that $\gamma_{G'}$ is a $(k+1)$-expression, whenever $k \geq 2$. Therefore, $\mathsf{cwd}(G') \leq k+1$. □

3.2. Cographs

A *cograph* is a graph that does not contain a path of length 3 as an induced subgraph. Alternatively, cographs are characterized by the following recursive definition, given by Corneil *et al.* [5]:

- A graph on a single vertex is a cograph;
- If $G_1, \ldots, G_k$ are cographs, then so is their *disjoint union* $G_1 \cup \cdots \cup G_k$;
- If G is a cograph, then so is its complement $\overline{G}$.

We note that, if G is a connected cograph on more than one vertex, then there exist $k \geq 2$ cographs $G_1, \ldots, G_k$ such that G is their *join* $G_1 \wedge \cdots \wedge G_k$. Moreover, it is straightforward that a graph is a cograph if and only if its clique-width is exactly 2.

A key algorithmic property of cographs is the fact that, up to isomorphism, each cograph G can be uniquely represented by a rooted tree $\mathcal{T}_G$, called *cotree* [5], which can be seen as a specialization of a 2-expression defining G. The leaves of $\mathcal{T}_G$ correspond to the vertices of G, and each internal node u of $\mathcal{T}_G$ represents either the disjoint union or the join operation of the respective cographs induced by the leaves of the subtrees of $\mathcal{T}_G$ rooted at each child of u. Another important property is that, given a graph G, recognising G as a cograph, as well as obtaining its respective cotree (if any), can be performed in time linear in the number of vertices and the number of edges of G [6].

Let $I = (G, W, \ell, r)$ be an instance of TCP, where G is a cograph. Since TCP can be easily solved in linear-time if $|W| < 3$ or $G[W]$ is connected, we assume throughout this section that $|W| \geq 3$ and $G[W]$ is not connected. Moreover, we assume that G is connected and, therefore, is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$.

Lemma 3.4. *Let G be a cograph that is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$, and let $W \subseteq V(G)$ be a terminal set such that $|W| \geq 3$ and $G[W]$ is not connected. There exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$. Moreover, G admits a connection tree for W that contains exactly one router and no linker.*

Proof. For the sake of contradiction, suppose that, for some $i, j \in \{1, \ldots, k\}$ with $i \neq j$, $V(G_i) \cap W \neq \emptyset$ and $V(G_j) \cap W \neq \emptyset$. Then, let $u \in V(G_i) \cap W$, $v \in V(G_j) \cap W$, and let T be the graph with vertex set $V(T) = W$ and edge set $E(T) = \{uw \mid w \in W \setminus V(G_i)\} \cup \{vw \mid w \in V(G_i) \cap W\}$. Clearly, T is a connected subgraph of $G[W]$. Therefore, there exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$. This implies that $V(G_j) \cap W = \emptyset$ for some $j \in \{1, \ldots, k\} \setminus \{i\}$. Then, let $u' \in V(G_j)$ and T' be the graph with vertex set $V(T') = \{u'\} \cup W$ and edge set $E(T') = \{u'w \mid w \in W\}$. One can verify that T' is a connection tree of G for W such that $\mathsf{L}(T') = \emptyset$ and $\mathsf{R}(T') = \{u'\}$. □

Considering the input graph G as the join of $k \geq 2$ cographs $G_1, \ldots, G_k$, it follows from Lemma 3.4 that TCP can be trivially solved if $r \geq 1$, or $V(G_i) \cap W \neq \emptyset$ and $V(G_j) \cap W \neq \emptyset$ for some $i, j \in \{1, \ldots, k\}$, with $i \neq j$. Thus, we dedicate the remainder of this section to resolve the case in which $r = 0$ and there exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$.

Lemma 3.5. *Let G be a cograph and $W \subseteq V(G)$ be a non-empty terminal set. If T is a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$ and $|\mathsf{L}(T)|$ is minimum, then $N_T(u) \subseteq W$ for each $u \in \mathsf{L}(T)$.*

Proof. For the sake of contradiction, suppose that $N_T(u) \not\subseteq W$ for some linker $u \in \mathsf{L}(T)$. Since $\mathsf{R}(T) = \emptyset$ and $\mathrm{leaves}(T) \subseteq W$, u belongs to a path P of T between two terminal vertices $w, w' \in W$, such that $(V(P) \setminus \{w, w'\}) \cap W = \emptyset$. Thus, it follows from the assumption $N_T(u) \not\subseteq W$ that $|V(P)| \geq 4$. Since cographs do not contain paths of length 3 as induced subgraphs, there exists a path P' of G between w and w' such that $|V(P')| \leq 3$ and $V(P') \subseteq V(P)$. Then, let T' be the graph with vertex set $V(T') = (V(T) \setminus V(P)) \cup V(P')$ and

edge set $E(T') = (E(T) \setminus E(P)) \cup E(P')$. One can easily verify that T' is a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$ and $\mathsf{L}(T') \subsetneq \mathsf{L}(T)$, which contradicts the minimality of $|\mathsf{L}(T)|$. □

For each graph G, we let $\mathsf{cc}(G)$ denote the set of connected components of G, and we let $o(G) = |\mathsf{cc}(G)|$ denote the number of connected components of G.

Corollary 3.6. *Let G be a cograph, $W \subseteq V(G)$ be a non-empty terminal set, and let T be a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$. If $|\mathsf{L}(T)|$ is minimum, then $|\mathsf{L}(T)| = o(G[W]) - 1$.*

Proof. Since $\mathsf{R}(T) = \emptyset$, it is straightforward that $|\mathsf{L}(T)| \geq o(G[W]) - 1$. On the other hand, it follows from Lemma 3.5 that, for each $u \in \mathsf{L}(T)$, $N_T(u) \subseteq W$. In addition, we note that, if $u \in \mathsf{L}(T)$ and $N_T(u) = \{w, w'\}$, then w and w' belong to distinct connected components of $G[W]$, otherwise the path (w, u, w') of T could be replaced by a shortest path of $G[W]$ between w and w', yielding a connection tree T' of G for W such that $\mathsf{L}(T') \subsetneq \mathsf{L}(T)$. Therefore, $|\mathsf{L}(T)| \leq o(G[W]) - 1$. □

Corollary 3.6 establishes that, whenever a cograph G admits a connection tree for a non-empty terminal set $W \subseteq V(G)$ that does not contain routers, G admits a connection tree T for W such that $\mathsf{R}(T) = \emptyset$ and $\mathsf{L}(T) = o(G[W]) - 1$. More importantly, it establishes that $o(G[W]) - 1$ is the minimum possible number of linkers that such a tree T can have. Therefore, if $I = (G, W, \ell, r)$ is an instance of TCP such that G is a cograph and $r = 0$, then ℓ must be at least $o(G[W]) - 1$, otherwise I is certainly a no-instance of the problem.

A *connection forest* of a graph G for a non-empty terminal set W is a subgraph F of G such that F is a forest and $\bigcup_{T \in \mathsf{cc}(F)} \mathrm{leaves}(T) \subseteq W \subseteq V(F)$. A connection forest F is said to be *routerless* if $\bigcup_{T \in \mathsf{cc}(F)} \mathsf{R}(T) = \emptyset$. For each graph G and each non-empty terminal $W \subseteq V(G)$, we let

$$\lambda[G, W] = \min\{o(F) \mid F \text{ is a routerless connection forest of } G \text{ for } W\}.$$

As a degenerate case, we define $\lambda[G, \emptyset] = 0$.

We note that $\lambda[G, W] = 1$ if and only if G admits a connection tree of G for W such that $\mathsf{R}(T) = \emptyset$.

Lemma 3.7. *Let G be a cograph and $W \subseteq V(G)$ be a terminal set. If G is the disjoint union of $k \geq 2$ cographs $G_1, \ldots, G_k$, then*

$$\lambda[G, W] = \sum_{i \in \{1, \ldots, k\}} \lambda[G_i, V(G_i) \cap W].$$

Proof. Since G is the disjoint union of $G_1, \ldots, G_k$, there is no edge between the vertices of G_i and the vertices of G_j for any $i, j \in \{1, \ldots, k\}$, with $i \neq j$. Thus, $\lambda[G, W] \geq \sum_{i \in \{1, \ldots, k\}} \lambda[G_i, V(G_i) \cap W]$. On the other hand, for each $i \in \{1, \ldots, k\}$ with $V(G_i) \cap W \neq \emptyset$, let F_i be a routerless connection forests of G_i for $V(G_i) \cap W$ with the minimum number of connected components. One can readily verify that $F = F_1 \cup \cdots \cup F_k$ is a routerless connection forests of G for W. Therefore, $\lambda[G, W] \leq \sum_{i \in \{1, \ldots, k\}} \lambda[G_i, V(G_i) \cap W]$. □

Lemma 3.8. *Let G be a cograph and $W \subseteq V(G)$ be a terminal set. If G is the join of $k \geq 2$ cographs $G_1, \ldots, G_k$ and there exists a unique $i \in \{1, \ldots, k\}$ such that $V(G_i) \cap W \neq \emptyset$, then*

$$\lambda[G, W] = \max\{1, \lambda[G_i, W] - n + n_i\},$$

where $n = |V(G)|$ and $n_i = |V(G_i)|$.

Proof. Let F be a routerless connection forest of G for W. Since $W \subseteq V(G_i)$, $d_F(u) = 2$ for each $u \in V(F) \setminus V(G_i)$. This implies that, for each $u \in V(G) \setminus V(G_i)$, there at most two distinct connected components of G_i that

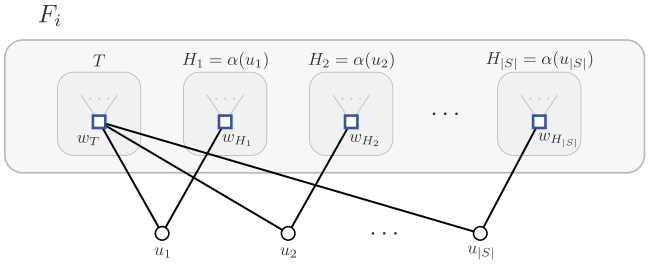


FIGURE 4. Graph F, with $S = \{u_1, \ldots, u_{|S|}\}$ and $\alpha(u_l) = H_l$ for each $l \in \{1, \ldots, |S|\}$.

are connected in F by u. In other words, if T is the connected component of F that contains $u \in V(G) \setminus V(G_i)$, then $o(T - u) \leq 2$. Thus,

$$\lambda[G, W] \geq \max\{1, \lambda[G_i, W] - |V(G) \setminus V(G_i)|\}.$$

On the other hand, let F_i be a routerless connection forest of G_i for W with the minimum number of connected components, *i.e.* $o(F_i) = \lambda[G_i, W]$, and let $S \subseteq V(G) \setminus V(G_i)$ such that $|S| = \min\{|V(G) \setminus V(G_i)|, o(F_i) - 1\}$. Also, let $T \in \mathsf{cc}(F_i)$, $w_T \in V(T) \cap W$ and $\alpha \colon S \to \mathsf{cc}(F_i) \setminus \{T\}$ be an injective map. Additionally, let $w_H \in V(H) \cap W$ for each $H \in \mathsf{cc}(F_i) \setminus \{T\}$. We define F as the graph (see Fig. 4) with vertex set $V(F) = V(F_i) \cup S$ and edge set

$$E(F) = E(F_i) \cup \{w_T u, u w_H \mid u \in S, H \in \mathsf{cc}(F_i) \setminus \{T\}, \alpha(u) = H\}.$$

One can verify that F is as routerless connection forest of G for W such that $o(F) = \lambda[G_i, W] - |S| = \max\{1, \lambda[G_i, W] - |V(G) \setminus V(G_i)|\}$. This implies that

$$\lambda[G, W] \leq \max\{1, \lambda[G_i, W] - n + n_i\},$$

concluding the proof. □

Theorem 3.9. *TCP is linear-time solvable on cographs.*

Proof. Let $I = (G, W, \ell, r)$ be an instance of TCP, where G is a cograph on n vertices and m edges. Assume without loss of generality that $|W| \geq 3$, G is connected but $G[W]$ is not connected. Moreover, based on Lemma 3.4 and on Corollary 3.6, assume that $r = 0$ and $\ell \geq o(G[W])$, respectively. Then, compute $\lambda[G, W]$ following the rules described below:

$$\lambda[G, W] = \begin{cases} \left[\begin{array}{l} \textbf{case 1.}\ |V(G)| = 1: \\ \quad 0 \quad \text{if } V(G) \cap W = \emptyset, \\ \quad 1 \quad \text{otherwise;} \end{array}\right. \\ \left[\begin{array}{l} \textbf{case 2.}\ G = G_1 \cup \cdots \cup G_k, \text{ for some } k \geq 2: \\ \quad \sum_{i \in \{1, \ldots, k\}} \lambda[G_i, V(G_i) \cap W]; \end{array}\right. \\ \left[\begin{array}{l} \textbf{case 3.}\ G = G_1 \wedge \cdots \wedge G_k, \text{ for some } k \geq 2: \\ \quad 0 \quad \text{if } \forall\, i \in \{1, \ldots, k\},\ V(G_i) \cap W = \emptyset, \\ \quad 1 \quad \text{if } \exists\, i, j \in \{1, \ldots, k\},\ i \neq j,\ V(G_i) \cap W \neq \emptyset \text{ and } V(G_j) \cap W \neq \emptyset, \\ \quad \max\{1, \lambda[G_i, W] - n + n_i\} \quad \text{if } \exists!\, i \in \{1, \ldots, k\},\ V(G_i) \cap W \neq \emptyset, \\ \text{where } n = |V(G)| \text{ and } n_i = |V(G_i)|. \end{array}\right. \end{cases}$$

The correctness of the rules follows from Lemmas 3.7 and 3.8. Since G admits a routerless connection tree if and only if $\lambda[G, W] = 1$, we have that I is a yes-instance of TCP if and only if $\lambda[G, W] = 1$.

Now, we analyse the time complexity of this algorithm. First, we note that $\lambda[G, W]$ can be computed in a bottom-up manner, according to the post-order traversal of the cotree $\mathcal{T}_G$ associated with G, using a dynamic programming matrix indexed by the nodes of $\mathcal{T}_G$. Moreover, we recall that $\mathcal{T}_G$ can be obtained in time $\mathcal{O}(n + m)$ *cf.* [6], and that, by definition, the number of nodes of $\mathcal{T}_G$ is $\mathcal{O}(n)$. Additionally, we note that, before computing $\lambda[G, W]$, $\mathcal{T}_G$ can be preprocessed in time $\mathcal{O}(n)$ so that each node u of $\mathcal{T}_G$ is associated with a flag which informs whether or not $V(G_u) \cap W \neq \emptyset$, where G_u denotes the subgraph of G corresponding to the subtree $\mathcal{T}_G^u$ of $\mathcal{T}_G$ rooted at u, *i.e.* G_u is the subgraph of G induced by the leaves of $\mathcal{T}_G^u$. Thus, one can verify that, for each node u of $\mathcal{T}_G$, the cell related to u of our dynamic programming matrix, which corresponds to $\lambda[G_u, V(G_u) \cap W]$, can be computed in time $\mathcal{O}(d_{\mathcal{T}_G}(u))$. Since $\mathcal{T}_G$ is a tree on $\mathcal{O}(n)$ nodes, we have that $\sum_{u \in V(\mathcal{T}_G)} d_{\mathcal{T}_G}(u) = \mathcal{O}(n)$. Therefore, $\lambda[G, W]$ can be computed in linear time. □

4. Graphs of bounded maximum degree

In this section, we analyse the complexity of TCP when restricted to graphs of bounded maximum degree. More specifically, we prove that TCP remains NP-complete on graphs of maximum degree 3 even if either the parameter $\ell \geq 0$ or the parameter $r \geq 0$ is fixed. In particular, for fixed $r \geq 0$, we show that TCP is NP-complete on graphs of maximum degree 3 that are *planar*.

It is worth mentioning that, if the input graph G is connected and has maximum degree at most 2, then G is either a path or a cycle, and consequently TCP can be trivially solved in polynomial-time, regardless of ℓ or r. Thus, we obtain that our results establish an NP-*complete versus polynomial-time solvable dichotomy* for TCP with respect to the maximum degree of the input graph.

Another interesting fact about our results is that they separate the complexity of TCP from the complexity of its strict variant, S-TCP. Indeed, while we prove that, for each fixed $\ell \geq 0$, TCP is NP-complete on graphs of maximum degree 3, S-TCP was proved to be polynomial-time solvable on graphs of maximum degree 3 even if $\ell \geq 0$ is fixed [25].

4.1. Fixed number of linkers

First, we consider the case in which the parameter $\ell \geq 0$ is fixed:

Theorem 4.1. *For each $\ell \geq 0$, TCP remains* NP*-complete when restricted to graphs of maximum degree* 3*.*

To prove Theorem 4.1, we present a polynomial-time reduction from an NP-complete variant of 3-SAT called 3-SAT(3) *cf.* [28]. The 3-SAT(3) problem has as input a set X of boolean variables and a set $\mathcal{C}$ of clauses over X that satisfy the following conditions:

- Each clause in $\mathcal{C}$ has two or three distinct literals;
- Each variable in X appears exactly twice positive and once negative in the clauses belonging to $\mathcal{C}$.

The problem then asks whether there exists a truth assignment $\alpha \colon X \to \{\mathit{false}, \mathit{true}\}$ such that every clause in $\mathcal{C}$ has at least one true literal under α.

Construction 4 (Reduction from 3-SAT(3) to TCP on Graphs of Maximum Degree 3)**.** Let $I = (X, \mathcal{C})$ be an instance of 3-SAT(3), with variable set $X = \{x_1, x_2, \ldots, x_p\}$ and clause set $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$, and let ℓ be a non-negative integer. We let G be the graph obtained from I and ℓ as follows (see Fig. 5a):

- Create the vertices $u_1, u_2, \ldots, u_\ell$ and, for each $i \in \{1, 2, \ldots, \ell - 1\}$, add the edges $u_i u_{i+1}$; moreover, create the vertices w_I and v_I and add the edges $w_I u_1$ and $u_\ell v_I$, originating the path $P_I = (w_I, u_1, \ldots, u_\ell, v_I)$;
- For each variable $x_i \in X$, create the gadget G_i such that

$$V(G_i) = \{w_i^1, w_i^2, t_i^1, t_i^2, f_i\} \text{ and } E(G_i) = \{w_i^1 t_i^1, t_i^1 t_i^2, t_i^2 w_i^2, w_i^2 f_i, f_i w_i^1\};$$

- Create a complete binary tree T_I, rooted at v_I, whose leaves are the vertices $w_1^1, \ldots, w_p^1$;
- For each clause $C_j \in \mathcal{C}$, create the vertices v_j^1, v_j^2 and v_j^3, and add the edges $v_j^1v_j^2$, $v_j^2v_j^3$ and $v_j^3v_j^1$;
- For each clause $C_j \in \mathcal{C}$, add the edge $t_i^a v_j^b$ if the b-th literal belonging to C_j corresponds to the a-th occurrence in I of the positive literal x_i, for $x_i \in X$, $a \in \{1,2\}$ and $b \in \{1, \ldots, |C_i|\}$; on the other hand, add the edge $f_i v_j^b$ if the b-th literal belonging to C_j corresponds to the (single) occurrence in I of the negative literal $\overline{x}_i$, for $x_i \in X$ and $b \in \{1, \ldots, |C_j|\}$.

Clearly, G is a graph of maximum degree 3. Then, we let $g(I, \ell) = (G, W, \ell, r)$ be the instance of TCP such that $W = \{w_I\} \cup V(T_I) \cup \{w_i^1, w_i^2 \mid x_i \in X\} \cup \{v_j^1, v_j^2, v_j^3 \mid C_j \in \mathcal{C}\}$ and $r = 2p$.

Lemma 4.2. *Let $I = (X, \mathcal{C})$ be an instance of 3-SAT(3). For each $\ell \geq 0$, I is a yes-instance of 3-SAT(3) if and only if the instance $g(I, \ell)$ described in Construction 4 is a yes instance of TCP.*

Proof. Assume that $X = \{x_1, x_2, \ldots, x_p\}$ and $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$. Additionally, assume that $g(I, \ell) = (G, W, \ell, r)$.

First, suppose that there exists a truth assignment $\alpha \colon X \to \{\mathit{false}, \mathit{true}\}$ such that every clause belonging to $\mathcal{C}$ has at least one true literal under α. Then, let S be the vertex set defined as follows

$$S = \{t_i^1, t_i^2 \mid x_i \in X, \alpha(x_i) = \mathit{true}\} \cup \{f_i \mid x_i \in X, \alpha(x_i) = \mathit{false}\} \\ \cup \{w_i^1, w_i^2 \mid x_i \in X\} \cup \{v_j^1, v_j^2, v_j^3 \mid C_j \in \mathcal{C}\} \cup V(P_I) \cup V(T_I),$$

and let $G[S]$ be the subgraph of G induced by S. We note that $G[S]$ is connected but may contain cycles. Thus, let T be a spanning tree subgraph of $G[S]$ that contains all edges of $G[S]$ except for possibly not containing some edges between the vertices v_j^1, v_j^2 and v_j^3, for $C_j \in \mathcal{C}$. In other words, T is a spanning tree subgraph of $G[S]$ such that $E(T) \supseteq E(G[S]) \setminus \{v_j^a v_j^b \mid a, b \in \{1,2,3\}, C_j \in \mathcal{C}\}$. It is not hard to check that T is a connection tree of G for W with linker set $\mathsf{L}(T) = \{u_1, \ldots, u_\ell\}$ and router set

$$\mathsf{R}(T) = \{t_i^1, t_i^2 \mid x_i \in X, \alpha(x_i) = \mathit{true}\} \cup \{f_i \mid x_i \in X, \alpha(x_i) = \mathit{false}\}.$$

Therefore, $g(I, \ell)$ is a yes-instance of TCP.

Figure 5 depicts the instance $g(I, \ell) = (G, W, \ell, r)$ of TCP, obtained from an instance $I = (X, \mathcal{C})$ of 3-SAT(3) and a non-negative integer ℓ. It also depicts a connection tree T of G for W, obtained from a truth assignment $\alpha \colon X \to \{\mathit{false}, \mathit{true}\}$.

Conversely, suppose that $g(I, \ell)$ is a yes-instance of TCP, and let T be a connection tree of G for W such $|\mathsf{L}(T)| \leq \ell$ and $|\mathsf{R}(T)| \leq 2p$. We note that the path P_I must be in T, since every path of G between the terminal vertex w_I and any other terminal vertex $w \in W \setminus \{w_I\}$ contains all the vertices of P_I. Consequently, the graph $T' = T - P_I$ cannot contain any linker, and all non-terminal vertices of T' must be routers. This, along with the fact that $\Delta(G) = 3$, implies that $N_T(v) = N_G(v)$ for each $v \in V(T') \setminus W$. Hence, if $t_i^1 \in V(T)$ or $t_i^2 \in V(T)$, then $w_i^1, t_i^2 \in N_T(t_i^1)$ and $w_i^2, t_i^1 \in N_T(t_i^2)$. Analogously, if $f_i \in V(T)$, then $w_i^1, w_i^2 \in N_T(f_i)$. Thus, since T is acyclic, we have that, for each $x_i \in X$, either $t_i^1, t_i^2 \in V(T)$ and $f_i \notin V(T)$, or $t_i^1, t_i^2 \notin V(T)$ and $f_i \in V(T)$. Then, we define a truth assignment $\alpha \colon X \to \{\mathit{false}, \mathit{true}\}$ as follows: for each $x_i \in X$, $\alpha(x_i) = \mathit{false}$ if and only if $f_i \in V(T)$. We note that, for each $C_j \in \mathcal{C}$, every path of G between the terminal vertices v_j^1, v_j^2, v_j^3 and any other terminal vertex $w \in W \setminus \{v_j^1, v_j^2, v_j^3\}$ must contain one of the vertices t_i^1, t_i^2, f_i for some $x_i \in X$. Moreover, by supposition, $V(T) \supseteq W \supseteq \{v_j^1, v_j^2, v_j^3 \mid C_j \in \mathcal{C}\}$. Consequently, every clause in $\mathcal{C}$ has at least one true literal under α. Therefore, I is a yes-instance of 3-SAT(3). □

4.2. Fixed number of routers

Now, we consider the case in which the parameter $r \geq 0$ is fixed:

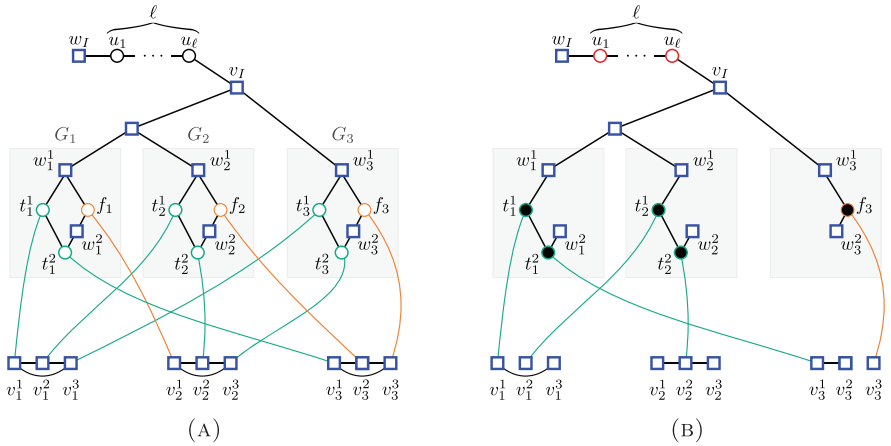


FIGURE 5. (a) Graph G and terminal set W (blue square vertices) of the instance $g(I,\ell)$ of TCP obtained from the instance $I=(X,\mathcal{C})$ of 3-SAT(3), where $X=\{x_1,x_2,x_3\}$ and $\mathcal{C}=\{C_1=\{x_1,x_2,x_3\},C_2=\{\overline{x}_1,x_2,x_3\},C_3=\{x_1,\overline{x}_2,\overline{x}_3\}\}$, and from a non-negative integer ℓ. (b) Connection tree T of G for W, obtained from the truth assignment $\alpha\colon X\to\{\mathit{false},\mathit{true}\}$ such that $\alpha(x_1)=\mathit{true}$, $\alpha(x_2)=\mathit{true}$ and $\alpha(x_3)=\mathit{false}$.

Theorem 4.3. *For each $r\geq 0$, TCP remains* NP*-complete when restricted to planar graphs of maximum degree* 3*.*

To prove Theorem 4.3, we first show through Propositions 4.4 and 4.5 that the *st*-HAMILTONIAN PATH problem is NP-complete on planar graphs of maximum degree 3. Then, we present a polynomial-time reduction from this particular case of *st*-HAMILTONIAN PATH to TCP. The *st*-HAMILTONIAN PATH problem is the variant of the HAMILTONIAN PATH problem that asks whether the input graph has a Hamiltonian path between two given vertices s and t.

Next proposition is an intermediate step in order to show the NP-completeness of *st*-HAMILTONIAN PATH on planar graphs of maximum degree 3.

Proposition 4.4. HAMILTONIAN CYCLE *remains* NP*-complete when restricted to planar graphs of maximum degree* 3 *that have at least two adjacent vertices of degree* 2 *each.*

Proof. Itai *et al.* [19] proved that HAMILTONIAN CYCLE is NP-complete on planar graphs of maximum degree 3. Based on their proof (see Lemma 2.1 [19]), we can suppose without loss of generality that the input graph G has at least one vertex of degree 2. Thus, let $u\in V(G)$ be such a vertex, and let $e=uv$ be an edge that has u and v as endpoints, for some $v\in V(G)\setminus\{u\}$. Then, we define H as the graph obtained from G by *subdividing* e, *i.e.* by removing e, adding a new vertex u_e and adding the edges uu_e and u_ev. We note that H is a graph of maximum degree 3 that has at least two adjacent vertices of degree 2 each, namely u and u_e. Furthermore, it is immediate that G has a Hamiltonian cycle if and only if H has a Hamiltonian cycle. □

Proposition 4.5. *st*-HAMILTONIAN PATH *remains* NP*-complete when restricted to planar graphs of maximum degree* 3 *in which s and t have degree* 1 *each.*

Proof. Let G be a planar graph of maximum degree 3. Based on Proposition 4.4, assume without loss of generality that G contains two vertices $u,v\in V(G)$ such that $uv\in E(G)$ and $d_G(u)=d_G(v)=2$. Then, let H be the graph obtained from G by adding two new vertices s and t, and by adding the edges su and vt. We note that H is a graph of maximum degree 3 and that s and t have degree 1 in H each. Furthermore, it is straightforward that G has a Hamiltonian cycle if and only if H has a st-Hamiltonian path. □

(A) $d_G(u_i) = 2$ (B) $d_G(u_i) = 3$

FIGURE 6. (a) Case in which $d_G(u_i) = 2$: vertices $v_i^1, v_i^2, u_i^1, u_i^2$. (b) Case in which $d_G(u_i) = 3$: vertices $v_i^1, v_i^2, u_i^1, u_i^2, u_i^3$.

Below, we finally describe our polynomial-time reduction from *st*-HAMILTONIAN PATH to TCP. We note that this reduction is slightly similar to the one described in Construction 3 to prove the NP-completeness of TCP on strongly chordal graphs.

Construction 5 (Reduction from *st*-HAMILTONIAN PATH to TCP on Planar Graphs of Maximum Degree 3)**.** Let G be a planar graph of maximum degree 3 and $s, t \in V(G)$ be distinct vertices of G. Based on Proposition 4.5, assume without loss of generality that $d_G(s) = d_G(t) = 1$. Moreover, assume that every vertex of G different from s and t has degree at least 2, otherwise G would certainly not admit a *st-Hamiltonian path*, *i.e.* a Hamiltonian path between s and t. Also, assume that $V(G) = \{u_1, \ldots, u_n\}$, for some positive integer n, where $s = u_1$ and $t = u_n$. Let r be a non-negative integer. For each $u_i \in V(G) \setminus \{s, t\}$, let $\alpha_i \colon N_G(u_i) \to |N_G(u_i)|$ be the bijection such that, for each two distinct vertices $u_{j_1}, u_{j_2} \in N_G(u_i)$, we have that $\alpha_i(u_{j_1}) < \alpha_i(u_{j_2})$ if and only if $j_1 < j_2$. We let G' be the graph obtained from G, s, t and r as follows (see Fig. 7):

- Add all vertices of G to G';
- For each vertex $u_i \in V(G)$ of degree 2 in G, add new vertices $v_i^1, v_i^2, u_i^1, u_i^2$ and add the edges $u_i v_i^1$, $u_i v_i^2$, $v_i^1 u_i^1$ and $v_i^2 u_i^2$ (see Fig. 6a);
- For each vertex $u_i \in V(G)$ of degree 3 in G, add new vertices $v_i^1, v_i^2, u_i^1, u_i^2, u_i^3$ and add the edges $u_i v_i^1$, $u_i v_i^2$, $v_i^1 u_i^2$, $v_i^2 u_i^2$, $v_i^1 u_i^1$ and $v_i^2 u_i^3$; (see Fig. 6b)
- For each vertex $u_i \in V(G)$ and each vertex $u_j \in N_G(u_i)$, add the edges $u_i^a u_j^b$, where $a = \alpha_i(u_j)$ and $b = \alpha_j(u_i)$;
- If $r \geq 1$, create the gadget H_r and the terminal set W_r described in Construction 2, and add the edge $\rho_r s$; otherwise, define $W_r = \emptyset$.

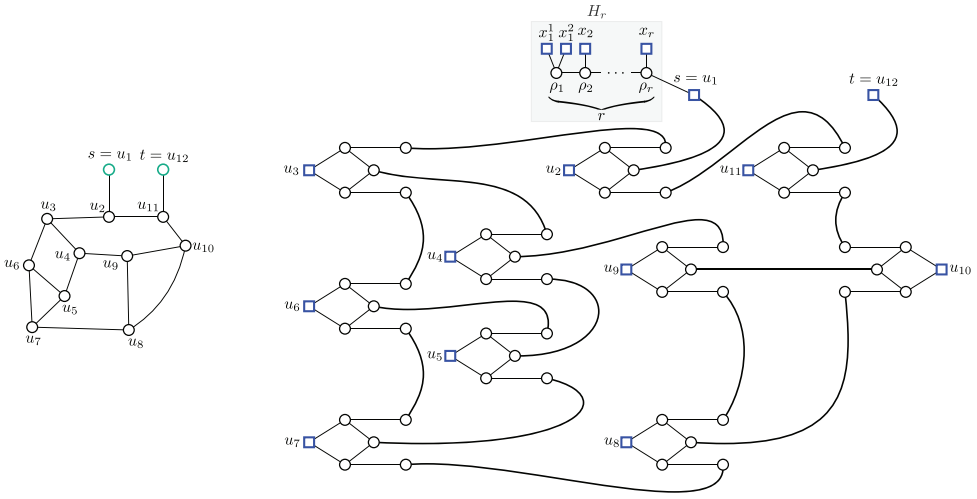
For each $u_i \in V(G)$, let G'_i be the subgraph of G' illustrated in Figure 6, *i.e.* the subgraph of G' induced by $\{u_i, v_i^1, v_i^2\} \cup \{u_i^j \mid j \in \{1, \ldots, d_G(u_i)\}\}$. Note that, H_r and, for each $u_i \in V(G)$, G'_i are planar. Additionally, it is not hard to verify that the input graph G is isomorphic to the graph resulting from $G' - H_r$ by identifying every subgraph G'_i into the vertex u_i. Therefore, since G is planar, we have that G' is planar as well. Furthermore, it straightforward that G' is a graph of maximum degree 3. Then, we let $g(G, s, t, r) = (G', W, \ell, r)$ be the instance of TCP such that $W = V(G) \cup W_r$ and $\ell = 4n - 4$.

Lemma 4.6. *Let G be a graph of maximum degree 3 and $s, t \in V(G)$ be two distinct vertices of G. Assume that s and t have degree 1 in G each. For each $r \geq 0$, G admits a st-Hamiltonian path if and only if the instance $g(G, s, t, r)$ described in Construction 5 is a yes-instance of TCP.*

Proof. Assume that $V(G) = \{u_1, \ldots, u_n\}$, where $s = u_1$ and $t = u_n$, and that $g(G, s, t, r) = (G', W, \ell, r)$. Additionally, for simplicity, consider $W_r = V(H_r) = E(H_r) = \emptyset$ if $r = 0$.

First, suppose that there exists in G a Hamiltonian path

$$P = (u_{j_1}, u_{j_2}, \ldots, u_{j_{n-1}}, u_{j_n})$$



(A) Graph G (B) Graph G'

FIGURE 7. (a) Graph G of maximum degree 3, with two distinct vertices $s, t \in V(G)$ such that $d_G(s) = d_G(t) = 1$. (b) Graph G' with terminal set W (blue square vertices) of the instance $g(G, s, t, r)$ of TCP described in Construction 5, obtained from G, the vertices s and t, and a non-negative integer r.

such that $s = u_{j_1}$ and $t = u_{j_n}$. Then, let S be the vertex set defined as follows:

$$S = V(H_r) \cup V(P) \cup \{v_i^1, v_i^2 \mid i \in \{2, \ldots, n-1\}\} \cup \{u_{j_2}^{\alpha_{j_2}(s)}, u_{j_{n-1}}^{\alpha_{j_{n-1}}(t)}\}$$
$$\cup \{u_{j_i}^a, u_{j_{i+1}}^b \mid a = \alpha_{j_i}(u_{j_{i+1}}), b = \alpha_{j_{i+1}}(u_{j_i}), i \in \{2, \ldots, n-2\}\},$$

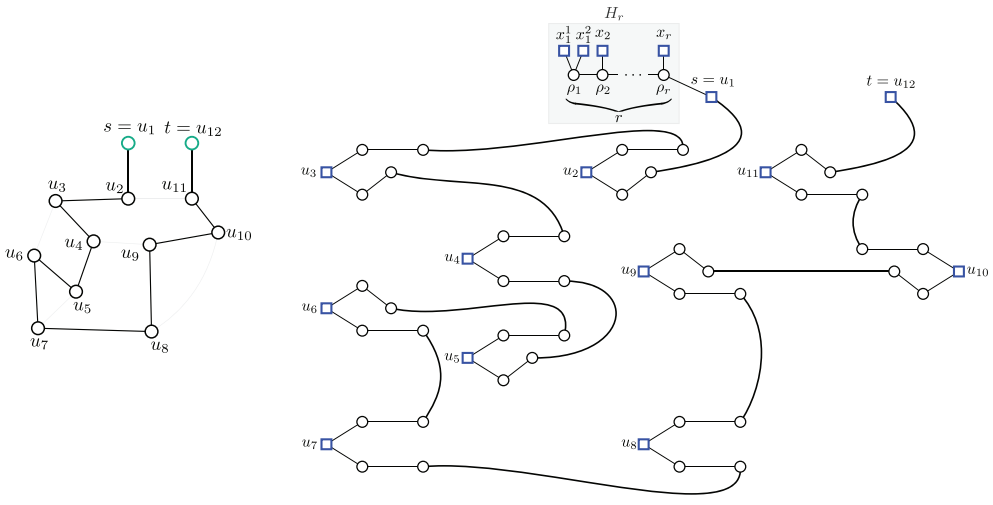
where α_i denotes the bijection from $N_G(u_i)$ to $|N_G(u_i)|$ described in Construction 5. We note that $G'[S]$ is connected but may contain cycles. More precisely, every cycle of $G'[S]$ is of the form $(u_i, v_i^1, u_i^2, v_i^2, u_i)$, and it exists if and only if $d_G(u_i) = 3$ and either $S \supseteq \{u_i^1, u_i^2\}$ or $S \supseteq \{u_i^2, u_i^3\}$, for $u_i \in V(G) \setminus \{s, t\}$. Thus, we let T be the graph obtained from $G'[S]$ by removing, for each vertex $u_i \in V(G) \setminus \{s, t\}$ with $d_G(u_i) = 3$, the edge $v_i^1 u_i^2$ if $S \supseteq \{u_i^1, u_i^2\}$, or the edge $v_i^2 u_i^2$ if $S \supseteq \{u_i^2, u_i^3\}$. One can verify that T is a connection tree of G' for W such that $\mathsf{L}(T) = S \setminus (V(H_r) \cup V(G))$ and $\mathsf{R}(T) = \{\rho_1, \ldots, \rho_r\}$. Therefore, $g(G, s, t, r)$ is a yes-instance of TCP.

Consider the graph G depicted in Figure 7a and the graph G' and the terminal set W depicted in Figure 7b, obtained from G and a non-negative integer r. Figure 8 illustrates a connection tree T of G' for W obtained from the st-Hamiltonian path of G depicted in Figure 8a.

Conversely, suppose that $g(G, s, t, r)$ is a yes-instance of TCP, and let T be a connection tree of G' for W such that $|\mathsf{L}(T)| \leq 4n - 4$ and $|\mathsf{R}(T)| \leq r$. We note that $\mathsf{R}(T) = \{\rho_1, \ldots, \rho_r\}$. Consequently, $T' = T - H_r$ cannot contain any router, and all non-terminal vertices of T' must be linkers. Moreover, by construction, s and t have degree 1 in T' each. This implies that the vertices $u_2, \ldots, u_{n-1}$ have degree exactly 2 in T' each, otherwise T would not be connected or $W \not\subseteq V(T)$. Hence, T' consists in a path P' between s and t of the form

$$P' = (s, u_{j_2}^{a_2}, v_{j_2}^{c_2}, u_{j_2}, v_{j_2}^{c'_2}, u_{j_2}^{b_2}, \ldots, u_{j_{n-1}}^{a_{n-1}}, v_{j_{n-1}}^{c_{n-1}}, u_{j_{n-1}}, v_{j_{n-1}}^{c'_{n-1}}, u_{j_{n-1}}^{b_{n-1}}, t),$$

where, for each $i \in \{2, \ldots, n-2\}$, $a_i = \alpha_{j_i}(u_{j_{i-1}})$, $b_i = \alpha_{j_i}(u_{j_{i+1}})$, and $c_i, c'_i \in \{1, 2\}$ with $c_i \neq c'_i$. Therefore, one can verify that $(s, u_{j_2}, \ldots, u_{j_{n-1}}, t)$ is a st-Hamiltonian path of G. □



(A) *st*-Hamiltonian path P

(B) Connection tree T

FIGURE 8. (a) *st*-Hamiltonian path P of the graph G depicted in Figure 7a. (b) Connection tree T of the graph G' for the terminal set W depicted in Figure 7b, obtained from P.

5. Concluding remarks

We conclude this work by posing some open questions. First, we ask about the existence of a graph class $\mathcal{G}$ on which Steiner tree is polynomial-time solvable while TCP remains NP-complete for fixed ℓ. We have shown that, on strongly chordal graphs, TCP is NP-complete for each $r \geq 0$, whereas Steiner tree is known to be polynomial-time solvable. However, the complexity of TCP on strongly chordal graphs for fixed ℓ has not been settled yet. Analogously, we ask whether there exists a graph class $\mathcal{G}$ on which TCP is polynomial-time solvable for fixed ℓ while Steiner tree remains NP-complete. We have shown that, on split graphs, TCP is polynomial-time solvable for fixed $r \geq 1$, whereas Steiner tree is known to be NP-complete. However, up to our knowledge, for fixed ℓ, there is no known example of such a separating class.

In addition, it is worth mentioning that, in our tractability proof of TCP on split graphs, only the cases in which $r \geq 1$ or $W \cap K \neq \emptyset$ are considered. Such hypotheses are imperative in our argumentation so as to ensure the connectivity of the sought connection tree. Thus, we leave as an open question whether TCP can be solved in polynomial-time on split graphs when $r = 0$ and $W \cap K = \emptyset$.

We also leave as an open question whether TCP parameterized by clique-width is in XP. Through a parameterized-reduction from Hamiltonian path, we have shown that TCP parameterized by clique-width is W[1]-hard. Nevertheless, the question whether TCP is in XP remains unsettled.

Finally, we ask about the complexity of TCP parameterized by the number of terminal vertices. Even though it is well-known that Steiner tree parameterized by the number of terminal vertices is in FPT [12], the complexity of the corresponding parameterization of TCP is widely open.

References

[1] B. Bergougnoux and M. Kanté, Fast exact algorithms for some connectivity problems parameterized by clique-width. *Theor. Comput. Sci.* **782** (2019) 30–53.

[2] A. Björklund, T. Husfeldt, P. Kaski and M. Koivisto, Fourier meets Möbius: fast subset convolution, in Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing, STOC '07, Association for Computing Machinery, New York, NY, USA (2007), pp. 67–74.

[3] A. Bondy and U. Murty, Graph Theory, Graduate Texts in Mathematics. Springer London (2008).

[4] C.J. Colbourn and L.K. Stewart, Permutation graphs: connected domination and Steiner trees. *Discrete Math.* **86** (1990) 179–189.

[5] D.G. Corneil, H. Lerchs and S.L. Burlingham, Complement reducible graphs. *Discrete Appl. Math.* **3** (1981) 163–174.
[6] D.G. Corneil, Y. Perl and L.K. Stewart, A linear recognition algorithm for cographs. *SIAM J. Comput.* **14** (1985) 926–934.
[7] B. Courcelle, J. Engelfriet and G. Rozenberg, Handle-rewriting hypergraph grammars. *J. Comput. Syst. Sci.* **46** (1993) 218–270.
[8] M. Cygan, M. Pilipczuk, M. Pilipczuk and J.O. Wojtaszczyk, Kernelization hardness of connectivity problems in d-degenerate graphs. *Discrete Appl. Math.* **160** (2012) 2131–2141.
[9] A. D'Atri and M. Moscarini, Distance-hereditary graphs, Steiner trees, and connected domination. *SIAM J. Comput.* **17** (1988) 521–538.
[10] M.C. Dourado, R.A. Oliveira, F. Protti and U.S. Souza, Conexão de Terminais com Número Restrito de Roteadores e Elos, in Proceedings of XLVI Simpósio Brasileiro de Pesquisa Operacional (2014) 2965–2976.
[11] M.C. Dourado, R.A. Oliveira, F. Protti and U.S. Souza, Design of connection networks with bounded number of non-terminal vertices, in Proceedings of V Latin-American Workshop on Cliques in Graphs. *Matemática Contemporânea*, vol. **42**, SBM, Buenos Aires (2014) 39–47.
[12] S.E. Dreyfus and R.A. Wagner, The Steiner problem in graphs. *Networks* **1** (1971) 195–207.
[13] M. Farber, Characterizations of strongly chordal graphs. *Discrete Math.* **43** (1983) 173–189.
[14] M.R. Fellows, F.A. Rosamond, U. Rotics and S. Szeider, Clique-width is NP-complete. *SIAM J. Discrete Math.* **23** (2009) 909–939.
[15] F.V. Fomin, P.A. Golovach, D. Lokshtanov and S. Saurabh, Clique-width: on the price of generality, in Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms, *SIAM* (2009) 825–834.
[16] M.R. Garey and D.S. Johnson, The rectilinear Steiner tree problem is NP-complete. *SIAM J. Appl. Math.* **32** (1977) 826–834.
[17] L. Gargano, M. Hammar, P. Hell, L. Stacho and U. Vaccaro, Spanning spiders and light-splitting switches. *Discrete Math.* **285** (2004) 83 –95.
[18] F.K. Hwang, D.S. Richards and P. Winter, The Steiner tree problem. *Ann. Discrete Math.* **53** (1992).
[19] A. Itai, C.H. Papadimitriou and J.L. Szwarcfiter, Hamilton paths in grid graphs. *SIAM J. Comput.* **11** (1982) 676–686.
[20] R.M. Karp, Reducibility among Combinatorial Problems. Springer US, Boston, MA (1972), pp. 85–103.
[21] G. Lin and G. Xue, On the terminal Steiner tree problem. *Inf. Process. Lett.* **84** (2002) 103–107.
[22] G.D. Lozzo and I. Rutter, Strengthening Hardness Results to 3-Connected Planar Graphs. Preprint arXiv:1607.02346 (2016).
[23] C.L. Lu, C.Y. Tang and R.C.-T. Lee, The full Steiner tree problem. *Theor. Comput. Sci.* **306** (2003) 55–67.
[24] A.A. Melo, C.M.H. Figueiredo and U.S. Souza, Connecting terminals using at most one router, in Proceedings of VII Latin-American Workshop on Cliques in Graphs. Vol. 45 of *Matemática Contemporânea*. SBM (2017) 49–57.
[25] A.A. Melo, C.M.H. Figueiredo and U.S. Souza, A multivariate analysis of the strict terminal connection problem. *J. Comput. Syst. Sci.* **111** (2020) 22–41.
[26] A.A. Melo, C.M.H. Figueiredo and U.S. Souza, On the terminal connection problem, in Proceedings of 47th International Conference on Current Trends in Theory and Practice of Computer Science. Vol. 12607 of *Lecture Notes in Computer Science*. Springer-Verlag New York, Inc. (2021) 278–292.
[27] A.A. Melo, C.M.H. Figueiredo and U.S. Souza, On undirected two-commodity integral flow, disjoint paths and strict terminal connection problems. *Networks* **77** (2021) 559–571.
[28] H. Müller, Hamiltonian circuits in chordal bipartite graphs. *Discrete Math.* **156** (1996) 291–298.
[29] H. Müller and A. Brandstädt, The NP-completeness of Steiner tree and dominating set for chordal bipartite graphs. *Theor. Comput. Sci.* **53** (1987) 257–265.
[30] J. Nederlof, Fast polynomial-space algorithms using inclusion-exclusion. *Algorithmica* **65** (2013) 868–884.
[31] D. Watel, M.-A. Weisser, C. Bentz and D. Barth, Steiner problems with limited number of branching nodes, in Proceedings of 20th International Colloquium on Structural Information and Communication Complexity. Vol. 8179 of *Lecture Notes in Computer Science*. Springer-Verlag New York, Inc. (2013) 310–321.
[32] D. Watel, M.-A. Weisser, C. Bentz and D. Barth, Directed Steiner trees with diffusion costs. *J. Combinat. Optim.* **32** (2016) 1089–1106.
[33] K. White, M. Farber and W. Pulleyblank, Steiner trees, connected domination and strongly chordal graphs. *Networks* **15** (1985) 109–124.