Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 15, no. 6, pp. 727–751 (2011)

Hardness Results and an Exact Exponential Algorithm for the Spanning Tree Congestion Problem

Yoshio Okamoto[1] *Yota Otachi*[2] *Ryuhei Uehara*[3] *Takeaki Uno*[4]

[1]Center for Graduate Education Initiative,
JAIST, Asahidai 1–1, Nomi, Ishikawa 923–1292, Japan.
[2]Graduate School of Information Sciences,
Tohoku University, Sendai 980–8579, Japan.
[3]School of Information Science,
JAIST, Asahidai 1–1, Nomi, Ishikawa 923–1292, Japan.
[4]National Institute of Informatics,
2–1–2 Hitotsubashi, Chiyoda-ku, Tokyo, 101–8430, Japan.

Abstract

Spanning tree congestion is a relatively new graph parameter, which has been studied intensively. This paper studies the complexity of the problem to determine the spanning tree congestion for non-sparse graph classes, while it was investigated for some sparse graph classes before. We prove that the problem is NP-hard even for chain graphs and split graphs. To cope with the hardness of the problem, we present a fast (exponential-time) exact algorithm that runs in $O^*(2^n)$ time, where n denotes the number of vertices. Additionally, we present simple combinatorial lemmas, which yield a constant-factor approximation algorithm for cographs, and a linear-time algorithm for chordal cographs.

Submitted:	Reviewed:	Revised:	Accepted:
May 2011	September 2011	October 2011	October 2011
	Final:	Published:	
	October 2011	October 2011	

Article type:	Communicated by:
Regular paper	S.-H. Hong

Extended abstract of this paper appeared in the proceedings of TAMC 2011 [22].

E-mail addresses: okamotoy@jaist.ac.jp (Yoshio Okamoto) otachi@dais.is.tohoku.ac.jp (Yota Otachi) uehara@jaist.ac.jp (Ryuhei Uehara) uno@nii.ac.jp (Takeaki Uno)

1 Introduction

Spanning tree congestion is a graph parameter defined by Ostrovskii [23] in 2004. Simonson [27] also studied the same parameter under a different name as a variant of cutwidth. After Ostrovskii [23], several graph-theoretic results have been presented [5, 8, 14, 16, 17, 18, 19, 20, 21, 24, 25], and very recently the complexity of the problem for determining the parameter has been studied [4, 26]. The parameter is defined as follows. Let G be a connected graph and T be a spanning tree of G. The *detour* for an edge $\{u, v\} \in E(G)$ is a unique u–v path in T. We define the *congestion* of $e \in E(T)$, denoted by $\mathrm{cng}_{G,T}(e)$, as the number of edges in G whose detours contain e. The *congestion of G in T*, denoted by $\mathrm{cng}_G(T)$, is the maximum congestion over all edges in T. The *spanning tree congestion* of G, denoted by $\mathrm{stc}(G)$, is the minimum congestion over all spanning trees of G. We denote by STC the problem of determining whether a given graph has spanning tree congestion at most given k. If k is fixed, then we denote the problem by k-STC.

Bodlaender, Fomin, Golovach, Otachi, and van Leeuwen [4, 26] studied the complexity of STC and k-STC. They showed that k-STC is linear-time solvable for apex-minor-free graphs and bounded-degree graphs, while k-STC is NP-complete even for K_6-minor-free graphs with only one vertex of unbounded degree if $k \geq 8$. They also showed that STC is NP-complete for planar graphs. Bodlaender, Kozawa, Matsushima, and Otachi [5] showed that the spanning tree congestion can be determined in linear time for outerplanar graphs. Although several complexity results are known as mentioned above, they are restricted to sparse graphs. The complexity for non-sparse graphs such as chordal graphs and chordal bipartite graphs were unknown.

In this paper, we show that STC is NP-complete for these important non-sparse graph classes. More precisely, we show that STC is NP-complete even for chain graphs and split graphs. It is known that every chain graph is chordal bipartite, and every split graph is chordal. The hardness for chain graphs is quite unexpected, since there is no other natural graph parameter that is known to be NP-hard for chain graphs, to the best of our knowledge. The hardness for chain graphs also implies the hardness for graphs of clique-width at most three. To cope with the hardness of the problem, we present a fast exponential-time exact algorithm. Our algorithm runs in $O^*(2^n)$ time, while a naive algorithm that examines all spanning trees runs in $O^*(2^m)$ or $O^*(n^n)$ time, where n and m denote the number of vertices and the number of edges. Note that $O^*(f(n)) = O(f(n) \cdot \mathrm{poly}(n))$. The idea, which allows us to achieve this running time, is to enumerate all possible combinations of cuts instead of all spanning trees. Using this idea, we can design a dynamic-programming-based algorithm that runs in $O^*(3^n)$ time. Then, by carefully applying the fast subset convolution method developed by Björklund, Husfeldt, Kaski, and Koivisto [1], we finally get the running time $O^*(2^n)$. We also study the problem on cographs. It is known that cographs are precisely the graphs of clique-width at most two. For some cographs such as complete graphs and complete p-partite graphs, the closed formulas for the spanning tree congestion are known [14, 17, 19, 23]. Although the complexity of STC for cographs remains unsettled, we provide a constant-factor approximation algorithm for them. Furthermore, we present a linear-time algorithm for chordal cographs.

Graphs in this paper are finite, simple, and connected, if not explicitly stated otherwise. We deal with edge-weighted graphs in Subsections 1.2 and 2.1. Our exponential-time exact algorithm runs in $O^*(2^n)$ time for edge-weighted graphs, too.

1.1 Graphs

Let G be a connected graph. For $S \subseteq V(G)$, we denote by $G[S]$ the subgraph induced by S. For an edge $e \in E(G)$, we denote by $G - e$ the graph obtained from G by the deletion of e. Similarly, for a vertex $v \in V(G)$, we denote by $G - v$ the graph obtained from G by the deletion of v and its incident edges. By $N_G(v)$, we denote the (*open*) *neighborhood* of v in G; that is, $N_G(v)$ is the set of vertices adjacent to v in G. For $S \subseteq V(G)$, we denote $\bigcup_{v \in S} N_G(v)$ by $N_G(S)$. We define the *degree* of v in G as $\deg_G(v) = |N_G(v)|$. If $\deg_G(v) = |V(G)| - 1$, then v is a *universal vertex* of G.

Let G and H be graphs. We say that G and H are *isomorphic*, and denote it by $G \simeq H$, if there is a bijection $f : V(G) \to V(H)$ such that $\{u, v\} \in E(G)$ if and only if $\{f(u), f(v)\} \in E(H)$. Now assume $V(G) \cap V(H) = \emptyset$. Then the *disjoint union* of G and H, denoted by $G \cup H$, is the graph with the vertex set $V(G) \cup V(H)$ and the edge set $E(G) \cup E(H)$. The *join* of G and H, denoted by $G \oplus H$, is the graph with the vertex set $V(G) \cup V(H)$ and the edge set $E(G) \cup E(H) \cup \{\{u, v\} \mid u \in V(G), v \in V(H)\}$.

For $A, B \subseteq V(G)$, we define $E_G(A, B) = \{\{u, v\} \in E(G) \mid u \in A,\ v \in B\}$. For $S \subseteq V(G)$, we define the *boundary edges* of S, denoted by $\theta_G(S)$, as $\theta_G(S) = E_G(S, V(G) \setminus S)$. Note that $\theta_G(\emptyset) = \theta_G(V(G)) = \emptyset$. The congestion $\mathrm{cng}_{G,T}(e)$ of an edge $e \in E(T)$ in a spanning tree T of G satisfies $\mathrm{cng}_{G,T}(e) = |\theta_G(A_e)|$, where A_e is the vertex set of one of the two components of $T - e$. For an edge e in a tree T, we say that e *separates* A and B if $A \subseteq A_e$ and $B \subseteq B_e$, where A_e and B_e are the vertex sets of the two components of $T - e$. Clearly, if T is a spanning tree of G and $e \in E(T)$ separates A and B, then $\mathrm{cng}_{G,T}(e) \geq |E(A, B)|$. If e separates A and B, we also say that e *divides* $A \cup B$ into A and B.

Let T be a tree rooted at $r \in V(T)$. Then we denote by $\mathrm{prt}_T(v)$ the parent of $v \in V(T)$ in T. The parent of the root r is not defined. We denote by $\mathrm{Ch}_T(v)$ the children of $v \in V(T)$ in T. Clearly, $N_T(v) = \{\mathrm{prt}_T(v)\} \cup \mathrm{Ch}_T(v)$ for every non-root vertex v.

1.2 Spanning tree congestion of weighted graphs

A graph G may be associated with an edge-weight function $\mathrm{wei} : E(G) \to \mathbb{Z}^+$. If a graph has such a function, then we call it an *edge-weighted graph* or just a *weighted graph*. Note that unweighted graphs can be considered as weighted graphs by setting $\mathrm{wei}(e) = 1$ for each edge e. For an edge-weighted graph G and $F \subseteq E(G)$, we define $\mathrm{wei}(F) = \sum_{f \in F} \mathrm{wei}(f)$ for $F \subseteq E(G)$. We extend

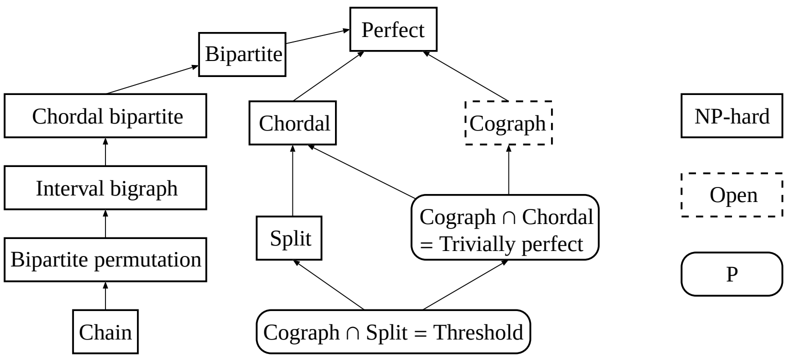

Figure 1: Relations among graph classes.

the notion of spanning tree congestion to edge-weighted graphs by defining the congestion of an edge e as the sum of the weights of edges whose detours pass through the edge e. If $e \in E(T)$ separates vertex sets A and B, then $\mathrm{cng}_{G,T}(e) \geq \mathrm{wei}(E(A,B))$.

For a weighted graph G, we define the *weighted degree* of v in G as $\mathrm{wdeg}_G(v) = \mathrm{wei}(\theta_G(\{v\}))$. It is not difficult to see that the following fact holds.

Proposition 1.1 *Let G be a weighted graph, and let $S \subseteq V(G)$. Then*

$$\mathrm{wei}(\theta_G(S)) = \sum_{v \in S} \mathrm{wdeg}_G(v) - 2\mathrm{wei}(E(G[S])).$$

It is known that STC for weighted graphs is equivalent to STC for unweighted graphs in the following sense.

Lemma 1.2 ([4, 26]) *Let G be a weighted graph and let $e \in E(G)$. Let G' be the graph obtained from G by removing the edge e and adding* $\mathrm{wei}(e)$ *internally disjoint paths of arbitrary lengths between the endpoints of e, where each edge in the added paths is of unit weight. Then,* $\mathrm{stc}(G) = \mathrm{stc}(G')$.

1.3 Graph classes

Here we introduce some graph classes. Figure 1 depicts relations among graph classes. For graph classes not defined in this subsection see textbooks on graph classes [6, 12].

A graph is *chordal* if it has no induced cycle of length greater than three. A graph G is a *split graph* if its vertex set $V(G)$ can be partitioned into two sets C and I so that C is a clique of G and I is an independent set of G. Clearly, every split graph is a chordal graph (see [12]). A *cograph* (or *complement-reducible graph*) is a graph that can be constructed recursively by the following rules:

1. K_1 is a cograph;

2. if G and H are cographs, then so is $G \cup H$;

3. if G and H are cographs, then so is $G \oplus H$.

Note that if G is a connected cograph with at least two vertices, then G can be expressed as $G_1 \oplus G_2$ for some nonempty cographs G_1 and G_2. A cograph is a *chordal cograph* if it is also a chordal graph. Chordal cographs are also known as *trivially perfect graphs* [6, 12] and *quasi-threshold graphs* [28]. It is known that in the construction of a chordal cograph by the above rules, we can assume one of two operands of $\oplus$ is K_1 [28].

Analogously to chordal graphs, *chordal bipartite graphs* are defined as the bipartite graphs without an induced cycle of length greater than four. A bipartite graph $G = (X, Y; E)$ is a *chain graph* if there is an ordering $<$ on X such that $u < v$ implies $N_G(u) \subseteq N_G(v)$. It is known that every chain graph is $2K_2$-free [29], and thus chordal bipartite.

Clique-width is a graph parameter which generalizes treewidth in some sense. Many hard problems can be solved efficiently for graphs of bounded clique-width. It is known that every chain graph has clique-width at most three [7], and that a graph has clique-width at most two if and only if it is a cograph [9]. For the definition and further information of clique-width, see a recent survey by Hliněný, Oum, Seese, and Gottlob [13].

2 Hardness for split graphs and chain graphs

This section presents our hardness results for split graphs and chain graphs. Namely, we prove the following theorems.

Theorem 2.1 *STC is NP-complete for split graphs.*

Theorem 2.2 *STC is NP-complete for chain graphs.*

Since every chain graph has clique-width at most three, we have the following corollary.

Corollary 2.3 *STC is NP-complete for graphs of clique-width at most three.*

The weighted edge argument [4, 26] allows us to present a simple proof for split graphs. However, we are unable to present a simple proof based on the weighted edge argument for chain graphs. This is because, in the process of modifying a weighted graph to an unweighted graph, we may introduce many independent edges (see Lemma 1.2). Although we need somewhat involved arguments for chain graphs, the proofs are based on essentially the same idea.

Clearly, STC is in NP. The proofs of NP-hardness are done by reducing the following well-known NP-complete problem to STC for both graph classes.

Problem: 3-Partition [11, SP15]

Instance: A multi-set $A = \{a_1, a_2, \dots, a_{3m}\}$ of $3m$ positive integers and a bound $B \in \mathbb{Z}^+$ such that $\sum_{a_i \in A} a_i = mB$, $a_1 \leq a_2 \leq \cdots \leq a_{3m}$, and $B/4 < a_i < B/2$ for each $a_i \in A$.

Question: Can A be partitioned into m disjoint sets $A_1, A_2, \dots, A_m$ such that, for $1 \leq i \leq m$, $\sum_{a \in A_i} a = B$? (Thus each A_i must contain exactly three elements from A.)

It is known that 3-PARTITION is NP-complete in the strong sense [11]. Thus we assume $a_{3m} \leq \mathrm{poly}(m)$, where $\mathrm{poly}(m)$ is some polynomial on m. By scaling each $a \in A$, we can also assume that $a_1 \geq 3m + 2$, $m \geq 3$, $B \geq 8$, and $B/4 + 1 \leq a_i \leq B/2 - 1$.

2.1 Hardness for split graphs

In this subsection, we prove that STC is NP-hard for split graphs. We first show that STC is NP-hard for edge-weighted split graphs with weighted edges only in the maximum clique, by reducing an instance A of 3-PARTITION to an edge-weighted split graph G_A such that A is a yes instance if and only if $\mathrm{stc}(G_A) \leq k$ for some k. We then show that G_A can be modified to an unweighted split graph G'_A in polynomial time so that $\mathrm{stc}(G_A) = \mathrm{stc}(G'_A)$. This proves Theorem 2.1.

Let A be an instance of 3-PARTITION. We now construct G_A from A in polynomial time (see Figure 2). Let $I = \{u_i \mid 1 \leq i \leq 3m\}$ and $C = \{x\} \cup V \cup W$, where $V = \{v_i \mid 1 \leq i \leq m\}$ and $W = \{w_i \mid m + 1 \leq i \leq a_{3m}\}$. The graph G_A has vertex set $I \cup C$. The sets I and C are independent set and a clique of G_A, respectively. Each $u_i \in I$ is adjacent to all vertices in V and vertices $w_{m+1}, w_{m+2}, \dots, w_{a_i}$. More formally, $E(G_A)$ is defined as follows:

$$\begin{aligned} E(G_A) = {} & \{\{c, c'\} \mid c, c' \in C\} \cup \{\{u, v\} \mid u \in I, v \in V\} \\ & \cup \{\{u_i, w_j\} \mid u_i \in I, m + 1 \leq j \leq a_i\}. \end{aligned}$$

Recall that $a_i > m$ for any $i \geq 1$. The degrees of vertices in G_A can be determined as follows: $\deg_{G_A}(u_i) = a_i$, $\deg_{G_A}(v_i) = |C| + |I| - 1$, and $\deg_{G_A}(w_i) = |C| + |\{j \mid a_j \geq i\}| - 1$. Some edges of G_A have heavy weights. Let $k = 2B + 2|C| + 2|I| - 15$. Then

$$\mathrm{wei}(e) = \begin{cases} \alpha := (k + 1)/2 & \text{if } e = \{x, v_i\}, \\ \beta_i := k - \deg_{G_A}(w_i) + 1 & \text{if } e = \{x, w_i\}, \\ 1 & \text{otherwise.} \end{cases}$$

Clearly, G_A is a split graph with weighted edges only in the clique C. The weighted degrees of vertices in G_A is as follows: $\mathrm{wdeg}_{G_A}(u_i) = a_i$, $\mathrm{wdeg}_{G_A}(v_i) = \alpha + |C| + |I| - 2 = k - B + 6$, and $\mathrm{wdeg}_{G_A}(w_i) = k$.

Lemma 2.4 *Let $k = 2B + 2|C| + 2|I| - 15$. Then A is a yes instance if and only if* $\mathrm{stc}(G_A) \leq k$.

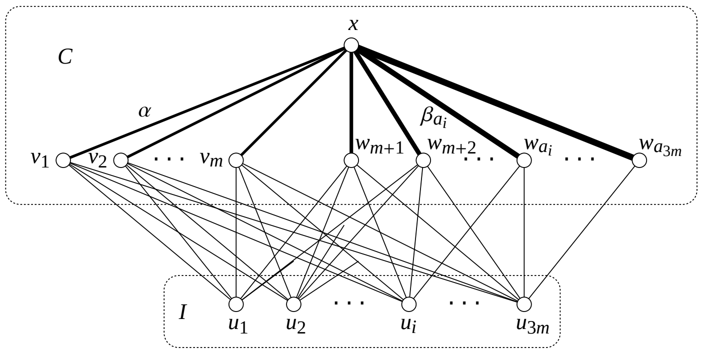

Figure 2: Reduction. (Unweighted edges in C are not depicted.)

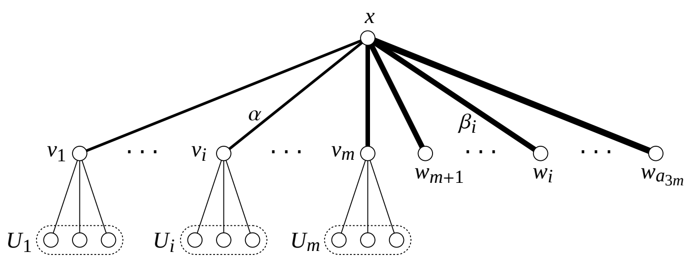

Figure 3: Spanning tree T.

Proof: ($\Longrightarrow$) Let $A_1, \dots, A_m$ be a partition of A such that $\sum_{a \in A_i} a = B$ for $1 \le i \le m$. Let U_i denote the set $\{u_j \mid a_j \in A_i\}$. The desired spanning tree T of G_A can be obtained from the partition $A_1, \dots, A_m$ as follows: $E(T) = \{\{x, c\} \mid c \in C \setminus \{x\}\} \cup \bigcup_{1 \le i \le m} \{\{v_i, u_j\} \mid u_j \in U_i\}$. Clearly, T is a spanning tree of G_A (see Figure 3).

The congestion of a leaf edge in T is at most $\Delta(G_A) = \max\{a_{3m}, k, k-B+6\}$. Clearly, $k \ge k - B + 6$ since $B \ge 8$. Since $|C| = a_{3m} + 1$ and $B \ge 8$, we have $k = 2B + 2|C| + 2|I| - 15 > 2a_{3m} \ge a_{3m}$. Thus $\Delta(G_A) = k$ and the congestion of any leaf edge is at most k.

Every inner edge of T is of the form $\{x, v_i\}$. From the definition of T, $\mathrm{cng}_{G_A,T}(\{x, v_i\}) = \mathrm{wei}(\theta_{G_A}(\{v_i\} \cup U_i))$. Since $G_A[\{v_i\} \cup U_i]$ is a star with center v_i,

$$\begin{aligned}\mathrm{wei}(\theta_{G_A}(\{v_i\} \cup U_i)) &= \mathrm{wdeg}_{G_A}(v_i) + \sum_{u_j \in U_i} \mathrm{wdeg}_{G_A}(u_j) - 2|U_i| \\ &= (k - B + 6) + B - 6 = k.\end{aligned}$$

($\Longleftarrow$) Let T be a spanning tree of G_A such that $\mathrm{cng}_{G_A}(T) \le k$. First we prove the following two properties of T.

Property 2.5 *All edges incident to x are included in T.*

Proof: Suppose there is a vertex $y \in C \setminus \{x\}$ such that $\{x, y\} \notin E(T)$. Then, a unique path from x to y in T first visits a vertex $z \in C \setminus \{x, y\}$ immediately after x. Hence, the edge $\{x, z\} \in E(T)$ has congestion at least

$$\mathrm{wei}(E_{G_A}(\{x\}, \{y, z\})) = \mathrm{wei}(\{x, y\}) + \mathrm{wei}(\{x, z\}) \geq 2 \min\{\alpha, \beta_i\}.$$

If $\beta_i < \alpha$, then $k < 2 \deg_{G_A}(w_i) - 1 \leq 2|C| + 2|I| - 3$. This contradicts $B \geq 8$. Therefore, $\mathrm{wei}(E_{G_A}(\{x\}, \{y, z\})) \geq 2\alpha = k + 1$. □

Property 2.6 *All u_i and w_i are leaves of T.*

Proof: By Property 2.5, it is easy to see that every u_i is a leaf of T, since otherwise T has a cycle. Let N_i denote $\{u_j \mid \{w_i, u_j\} \in E(T)\}$. Suppose $N_i \neq \emptyset$. Since each $u_j \in N_i$ is a leaf, $\mathrm{cng}_{G_A,T}(\{x, w_i\}) = \mathrm{wei}(\theta_{G_A}(\{w_i\} \cup N_i))$. Since $G_A[\{w_i\} \cup N_i]$ is a star with center w_i,

$$\begin{aligned}\mathrm{wei}(\theta_{G_A}(\{w_i\} \cup N_i)) &= \mathrm{wdeg}_{G_A}(\{w_i\}) + \sum_{u_j \in N_i} \mathrm{wdeg}_{G_A}(u_j) - 2|N_i| \\ &= k + \sum_{u_j \in N_i} (a_j - 2).\end{aligned}$$

Since $a_j > m \geq 2$, we have $\mathrm{cng}_{G_A,T}(\{x, w_i\}) > k$, which is a contradiction. □

From the above observations, each u_i is a leaf and its unique neighbor in T is some v_j. Therefore, T gives a partition $A_1, \ldots, A_m$ of A such that $a_i \in A_j$ if and only if u_i is adjacent to v_j in T. Suppose there is some j such that $\sum_{a_i \in A_j} a_i > B$. Then clearly $|A_j| \geq 3$ since $a_i < B/2$ for any $a_i \in A$. From the properties of T discussed above, $\mathrm{cng}_{G_A,T}(\{x, v_j\}) = \mathrm{wei}(\theta_{G_A}(\{v_j\} \cup U_j))$. By the same argument in the proof of Property 2.6,

$$\begin{aligned}\mathrm{wei}(\theta_{G_A}(\{v_j\} \cup U_j)) &= \mathrm{wdeg}_{G_A}(\{v_j\}) + \sum_{u_i \in U_j} \mathrm{wdeg}_{G_A}(u_i) - 2|U_j| \\ &= k - B + 6 + \sum_{u_i \in U_j} a_i - 2|U_j| > k + 6 - 2|U_j|.\end{aligned}$$

If $|U_j| = 3$, then $\mathrm{wei}(\theta_{G_A}(\{v_j\} \cup U_j)) > k + 6 - 2|U_j| = k$. Hence, $|U_j| \geq 4$. Since $a_i \geq B/4 + 1$,

$$\begin{aligned}\mathrm{wei}(\theta_{G_A}(\{v_j\} \cup U_j)) &= k - B + 6 + \sum_{u_i \in U_j} a_i - 2|U_j| \\ &\geq k - B + 6 + 4(B/4 + 1 - 2) = k + 2\end{aligned}$$

This contradicts $\mathrm{cng}_{G_A}(T) \leq k$. □

Now we prove the NP-hardness of STC for unweighted split graphs. To this end, we first reduce an instance A of 3-PARTITION to a weighted split graph G_A as stated above. Recall that all weighted edges of G_A are in $G_A[C]$. We need the following lemma.

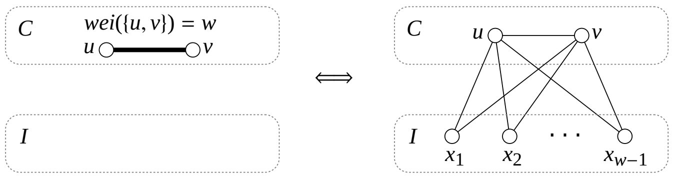

Figure 4: Weighted edge in the clique C.

Lemma 2.7 *Let G be an edge-weighted split graph with a partition (C, I) of $V(G)$, where C and I are a clique and an independent set of G, respectively. If the weighted edges are only in $G[C]$ and the maximum edge weight is $w_{\max}$, then an edge-unweighted split graph G' satisfying $\mathrm{stc}(G) = \mathrm{stc}(G')$ can be obtained from G in $O(w_{\max} \cdot |E(G)|)$ time.*

Proof: Let $\{u, v\} \in E(G)$ be an edge of weight $w = \mathrm{wei}(\{u, v\}) > 1$. We replace the weighted edge between u and v by an unweighted edge, and add $w - 1$ unweighted u–v paths of length two, where the inner point of ith path is a new vertex $x_i \in I$ (see Figure 4). Let us call the obtained graph H. Obviously, this can be done in $O(w)$ time, H has less weighted edges than G, and $\mathrm{stc}(H) = \mathrm{stc}(G)$ by Lemma 1.2. Also it is easy to see that H is a split graph with weighted edges only in $H[C]$. Therefore, repeatedly applying this local modification, we eventually obtain the desired graph G' in $O(w_{\max} \cdot |E(G)|)$ time. $\square$

Observe that the maximum edge-weight in G_A is bounded by a polynomial function on B and m. Thus the above lemma implies that from an instance A of 3-Partition, we can construct in polynomial time an unweighted split graph G'_A and $k \in \mathbb{Z}^+$ such that A is a yes instance if and only if $\mathrm{stc}(G'_A) \le k$. This proves Theorem 2.1.

2.2 Hardness for chain graphs

Next we prove the NP-hardness for chain graphs. Given an instance A of 3-Partition, we construct the graph $G_A = (P, Q; E)$. For convenience, let $M = B + 3m - 4$ and $\gamma_i = |\{a_j \in A \mid a_j \ge i\}|$. Note that $0 < \gamma_i \le 3m$ for $m + 1 \le i \le a_{3m}$. In particular, $\gamma_{m+1} = 3m$ and $\gamma_{a_{3m}} > 0$. First we define the vertex sets $P = U \cup V \cup W$ and $Q = X \cup Y \cup Z$ as follows:

$$\begin{aligned} U &= \{u_i \mid 1 \le i \le m\}, & X &= \{x_i \mid 1 \le i \le 3m\}, \\ V &= \{v_i \mid m+1 \le i \le a_{3m}\}, & Y &= \{y_i \mid m+1 \le i \le a_{3m}\}, \\ W &= \{w_i \mid 1 \le i \le M - a_{3m}\}, & Z &= \{z_i \mid 1 \le i \le M - a_{3m}\}. \end{aligned}$$

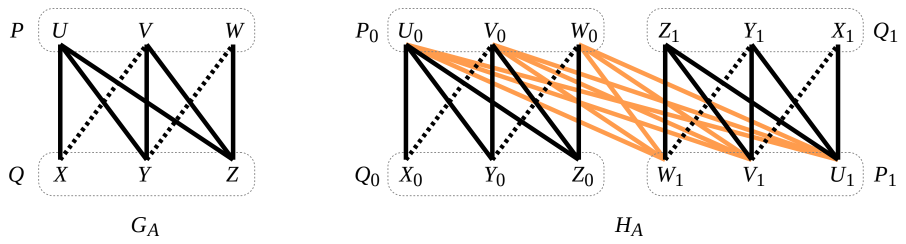

Figure 5: Graphs G_A and H_A. A solid line between two sets implies that the two sets induce a complete bipartite graph, and a dotted line between two sets implies that there are some (but not all) edges between the two sets. Two color classes of H_A are $P_0 \cup Q_1$ and $Q_0 \cup P_1$.

Next we define the edge set as follows:[1]

$$\begin{aligned} E = (X \times U) &\cup \big(Y \times (U \cup V)\big) \cup \big(Z \times (U \cup V \cup W)\big) \\ &\cup \{\{x_i, v_j\} \mid x_i \in X, m+1 \le j \le a_i\} \\ &\cup \{\{y_i, w_j\} \mid y_i \in Y, 1 \le j \le M - a_{3m} - \gamma_i\}. \end{aligned}$$

See Figure 5 for a simplified illustration of G_A.

Let G_0 and G_1 be two disjoint copies of G_A. That is, $G_A \simeq G_0 \simeq G_1$ and $V(G_0) \cap V(G_1) = \emptyset$. By P_i, Q_i, U_i, V_i, W_i, X_i, Y_i, and Z_i, we denote the vertex sets of G_i, $i \in \{0,1\}$, that correspond to the vertex sets P, Q, U, V, W, X, Y, and Z of G_A, respectively. Similarly, we denote the vertices of G_i, $i \in \{0,1\}$, that correspond to vertices u_j, v_j, w_j, x_j, y_j, and z_j of G_A by u^i_j, v^i_j, w^i_j, x^i_j, y^i_j, and z^i_j, respectively. We define the graph H_A as follows (see Figure 5): $V(H_A) = V(G_0) \cup V(G_1)$ and $E(H_A) = E(G_0) \cup E(G_1) \cup (P_0 \times P_1)$.

Lemma 2.8 *The graph H_A is a chain graph.*

Proof: Observe that in H_A the following relations hold:

$$\begin{aligned} N_{H_A}(x^0_1) \subseteq \cdots \subseteq N_{H_A}(x^0_{3m}) \subseteq N_{H_A}(y^0_1) \subseteq \cdots \subseteq N_{H_A}(y^0_{a_{3m}}) \\ \subseteq N_{H_A}(z^0_1) = \cdots = N_{H_A}(z^0_{M-a_{3m}}) \subseteq N_{H_A}(w^1_{M-a_{3m}}) \subseteq \cdots \subseteq N_{H_A}(w^1_1) \\ \subseteq N_{H_A}(v^1_{a_{3m}}) \subseteq \cdots \subseteq N_{H_A}(v^1_1) \subseteq N_{H_A}(u^1_m) = \cdots = N_{H_A}(u^1_1). \end{aligned}$$

This ordering shows that H_A is a chain graph. □

Lemma 2.9 $\deg_{H_A}(u^i_j) = 2M + 2m$, $\deg_{H_A}(v^i_j) = 2M - m + \gamma_j > 2M - m$, $2M - a_{3m} \le \deg_{H_A}(w^i_j) \le 2M - m$, $\deg_{H_A}(x^i_j) = a_i$, $\deg_{H_A}(y^i_j) = M - \gamma_j < M$, *and* $\deg_{H_A}(z^i_j) = M$. *Moreover,* $\Delta(H_A) = 2M + 2m$ *and* $\delta(H_A) = a_1$. □

[1] For simplicity, we denote by $S \times T$ the set of unordered pairs $\{\{s,t\} \mid s \in S, t \in T\}$.

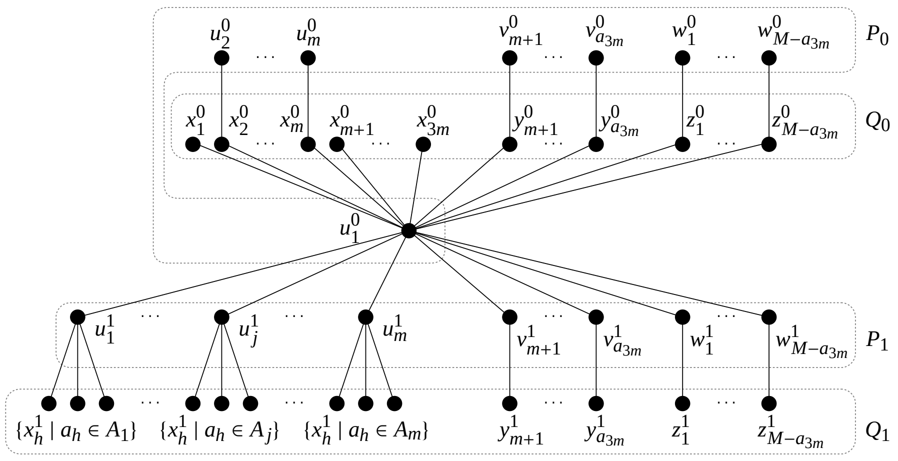

Figure 6: Spanning tree T in the proof of Lemma 2.10.

Now we prove that A is a yes instance of 3-Partition if and only if $\text{stc}(H_A) \leq k$. We divide the proof into the only-if-part (Lemma 2.10) and the if-part (Lemma 2.11).

Lemma 2.10 *Let $k = 3M - m - 2$. If A is a yes instance, then* $\text{stc}(H_A) \leq k$.

Proof: Let $A_1, \ldots, A_m$ be the partition of A such that $|A_i| = 3$ and $\sum_{a_j \in A_i} a_j = B$ for $1 \leq i \leq m$. We shall show that there is a spanning tree T of H_A such that $\text{cng}_{H_A}(T) \leq k$. Roughly speaking, T is constructed as follows (see Figure 6).

- Take all edges incident to u_1^0 in H_A.
- Take $\{u_j^1, x_h^1\}$ if and only if $a_h \in A_j$.
- Take a perfect matching between $\{u_2^0, \ldots, u_m^0\}$ and $\{x_2^0, \ldots, x_m^0\}$.
- Take a perfect matching between V_i and Y_i for each $i \in \{0, 1\}$.
- Take a perfect matching between W_i and Z_i for each $i \in \{0, 1\}$.

More precisely, the edge set of T is defined as follows:

$$\begin{aligned} E(T) = {} & \{\{u_1^0, v\} \mid v \in Q_0 \cup P_1\} \cup \{\{u_j^1, x_h^1\} \mid a_h \in A_j,\ 1 \leq j \leq m\} \\ & \cup \{\{u_j^0, x_j^0\} \mid 2 \leq j \leq m\} \cup \{\{v_j^i, y_j^i\} \mid i \in \{0,1\},\ m+1 \leq j \leq a_{3m}\} \\ & \cup \{\{w_j^i, z_j^i\} \mid i \in \{0,1\},\ 1 \leq j \leq M - a_{3m}\}. \end{aligned}$$

Clearly, T is a spanning tree of H_A. It is easy to see that edges not incident to u_1^0 are leaf edges. The congestions of these edges are at most $\Delta(H_A) = 2M + 2m$. Since $B \geq 8$, it follows that $2M + 2m = k - B + 6 < k$. There are four types of inner edges, and they divide $V(H_A)$ as follows.

1. $\{u_1^0, u_j^1\}$ divides $V(H_A)$ into $\{u_j^1\} \cup \{x_h^1 \mid a_h \in A_j\}$ and its complement.
2. $\{u_1^0, x_j^0\}$, $2 \leq j \leq m$, divides $V(H_A)$ into $\{u_j^0, x_j^0\}$ and its complement.
3. $\{u_1^0, v_j^1\}$ or $\{u_1^0, y_j^0\}$ divides $V(H_A)$ into $\{v_j^i, y_j^i\}$ and its complement.
4. $\{u_1^0, w_j^1\}$ or $\{u_1^0, z_j^0\}$ divides $V(H_A)$ into $\{w_j^i, z_j^i\}$ and its complement.

Hence, it suffices to show that all $|\theta_{H_A}(\{u_j^1\} \cup \{x_h^1 \mid a_h \in A_j\})|$, $|\theta_{H_A}(\{u_j^0, x_j^0\})|$, $|\theta_{H_A}(\{v_j^i, y_j^i\})|$, and $|\theta_{H_A}(\{w_j^i, z_j^i\})|$ are at most k. Note that $\{u_j^1\} \cup \{x_h^1 \mid a_h \in A_j\}$ induces a star $K_{1,3}$, and all $\{u_j^0, x_j^0\}$, $\{v_j^i, y_j^i\}$, and $\{w_j^i, z_j^i\}$ induce edges in H_A. Recall that $M = B+3m-4$ and $k = 3M-m-2$. Thus $2M+2m = k-B+6$. *(1)* Since $\deg_{H_A}(u_j^1) = 2M + 2m = k - B + 6$ and $\sum_{a_h \in A_j} \deg_{H_A}(x_h^1) = B$, we have $|\theta_{H_A}(\{u_j^1\} \cup \{x_h^1 \mid a_h \in A_j\})| = (k - B + 6) + B - 6 = k$. *(2)* Since $\deg_{H_A}(u_j^0) = 2M + 2m = k - B + 6$ and $\deg_{H_A}(x_j^0) = a_j$, we have $|\theta_{H_A}(\{u_j^0, x_j^0\})| = (k - B + 6) + a_j - 2 = k - B + a_j + 4$. Assumptions $a_j < B/2$ and $B \geq 8$ imply that $|\theta_{H_A}(\{u_j^0, x_j^0\})| \leq k$. *(3)* Since $\deg_{H_A}(v_j^i) = 2M - m + \gamma_j$ and $\deg_{H_A}(y_j^i) = M - \gamma_j$, we have $|\theta_{H_A}(\{v_j^i, y_j^i\})| = 3M - m - 2 = k$. *(4)* Since $\deg_{H_A}(w_j^i) \leq 2M - m$ and $\deg_{H_A}(z_j^i) = M$, we have $|\theta_{H_A}(\{w_j^i, z_j^i\})| \leq 3M - m - 2 = k$. □

Lemma 2.11 *Let $k = 3M - m - 2$. If $\mathrm{stc}(H_A) \leq k$, then A is a yes instance.*

Proof: Let T be a spanning tree of H_A such that $\mathrm{cng}_{H_A}(T) \leq k$. We first prove the following properties of T.

- Property 2.12. No edge e in T divides $P_0 \cup P_1$ into R and S such that $\min\{|R|, |S|\} \geq 2$.
- Property 2.13. There exist $i \in \{0, 1\}$ and $p_i \in P_i$ such that $N_T(p_i) \supseteq P_{1-i}$.

We may assume without loss of generality that $i = 0$ in Property 2.13 and T is rooted at p_0. Assuming Properties 2.12 and 2.13 with these assumptions, we have the following properties.

- Property 2.14. All vertices in Q_1 are leaves of T, and for any $p_1 \in P_1$, $\mathrm{Ch}_T(p_1) \subseteq Q_1$.
- Property 2.15. For any $z_j^1 \in Z_1$, $\mathrm{prt}_T(z_j^1) \in W_1$ and $\mathrm{Ch}_T(\mathrm{prt}_T(z_j^1)) = \{z_j^1\}$.
- Property 2.16. T can be modified to a spanning tree T^* of H_A so that $\mathrm{cng}_{H_A}(T^*) = \mathrm{cng}_{H_A}(T)$, T^* has Properties 2.12–2.15, and additionally $\mathrm{prt}_{T^*}(y_j^1) \in V_1'$ and $\mathrm{Ch}(\mathrm{prt}_T(y_j^1)) = \{y_j^1\}$ for any $y_j^1 \in Y_1'$.

We postpone the proofs of the above properties, and first prove the lemma with these properties at hand.

We assume T satisfies the last condition in Property 2.16; that is, T is already modified. By Properties 2.15 and 2.16, $\mathrm{Ch}_T(V_1 \cup W_1) = Y_1 \cup Z_1$. Thus

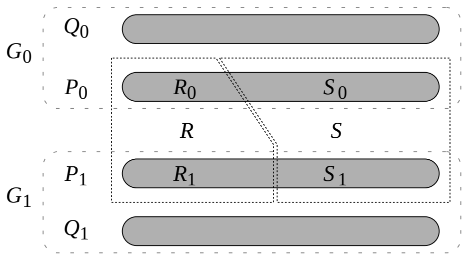

Figure 7: Illustration of R and S.

by Property 2.14, $\mathrm{prt}_T(x_j^1) \in U_1$ for any $x_j^1 \in X_1$. This implies that T gives a partition of A into m (possibly empty) sets $A_1, \ldots, A_m$ such that $a_j \in A_h$ if and only if $\{x_j^1, u_h^1\} \in E(T)$. Suppose A is not a yes instance. Then there exists some h such that $\sum_{a_j \in A_h} a_j > B$. Since $a_j < B/2$, we have $|A_h| \geq 3$. It is easy to see that

$$\mathrm{cng}_{H_A,T}(\{p_0, u_h^1\}) = \deg_{H_A}(u_h^1) + \sum_{a_j \in A_h} (\deg_{H_A}(x_j^1) - 2).$$

If $|A_h| = 3$, then $\deg_{H_A}(u_h^1) + \sum_{a_j \in A_h} (\deg_{H_A}(x_j^1) - 2) > (2M + 2m) + B - 6 = k$. Thus $|A_h| \geq 4$. Since $a_j \geq B/4 + 1$, we have

$$\deg_{H_A}(u_h^1) + \sum_{a_j \in A_h} (\deg_{H_A}(x_j^1) - 2) \geq (2M + 2m) + 4(B/4 - 1) = k + 2.$$

This contradicts $\mathrm{cng}_{H_A}(T) \leq k$, and thus the lemma holds.

Now we shall prove the properties.

Property 2.12 *No edge e in T divides $P_0 \cup P_1$ into R and S with* $\min\{|R|, |S|\} \geq 2$.

Proof: Suppose that $e \in E(T)$ divides $P_0 \cup P_1$ into R and S such that $\min\{|R|, |S|\} \geq 2$. Clearly, $\mathrm{cng}_{H_A,T}(e) \geq |E_{H_A}(R, S)|$. By symmetry, we may assume $|R| \leq |S|$. Let $R_i = R \cap P_i$ and $S_i = S \cap P_i$ (see Figure 7). Observe that $H_A[R \cup S] = H_A[P_0 \cup P_1] \simeq K_{M,M}$. Hence, $|E_{H_A}(R, S)| = |E_{K_{M,M}}(R, S)| = |R| \cdot M - 2|R_0| \cdot |R_1|$. Obviously, $|R_0| \cdot |R_1| \leq |R|^2/4$ since $|R_0| + |R_1| = |R|$. Thus we have

$$|E_{H_A}(R, S)| \geq |R| \cdot M - |R|^2/2.$$

Now we divide the rest of the proof into three cases: $|R| = 2$, $|R| \in \{3, 4, 5\}$, and $|R| \geq 6$.

[Case 1] $|R| \geq 6$. Since $\mathrm{cng}_{H_A}(T) \leq k$, we have $|R| \cdot M - |R|^2/2 \leq k < 3M$. Thus

$$M < \frac{|R|^2}{2(|R| - 3)} = |R| \cdot \frac{|R|}{2(|R| - 3)} \leq |R|.$$

This contradicts the assumptions $|R| \leq |S|$ and $|R| + |S| = 2M$.

[Case 2] $|R| \in \{3, 4, 5\}$. For this case, we can easily verify that

$$|R| \cdot M - |R|^2/2 > 3M - 5 \geq 3M - m - 2 = k,$$

since $M = B + 3m - 4$, $B \geq 8$, and $m \geq 3$. This contradicts $\mathrm{cng}_{H_A,T}(e) \leq k$.

[Case 3] $|R| = 2$. Let T_R and T_S be the components of $T - e$ containing R and S, respectively. From the definition, $\mathrm{cng}_{H_A,T}(e) = |E(T_R, T_S)|$. Recall that $|Z| = M - a_{3m}$ and $N_{G_A}(z) = P = U \cup V \cup W$ for any $z \in Z$. Here we have two subcases.

[Case 3-1] $R \not\subseteq P_i$ for any i. Let $R = \{r_0, r_1\}$, where $r_i \in R_i$. Clearly, $|E_{H_A}(R, S)| = |S_0| + |S_1| = 2(M - 1)$. If $z_i \in Z_i$ is included in T_R, then the $M - 1$ detours from z_i to $P_{1-i} \setminus \{r_{1-i}\}$ pass through the edge e. If $z_i \in Z_i$ is included in T_S, then the detour from z_i to r_{1-i} passes through e. In both cases, at least one detour for $z_i \in Z_i$ and its neighbor passes through e. Hence,

$$\begin{aligned}\mathrm{cng}_{H_A,T}(e) &\geq 2(M - 1) + 2(M - a_{3m}) \\ &= k + 4m - 4 + B - 2a_{3m} \\ &> k + 4m - 4 \geq k.\end{aligned}$$

[Case 3-2] $R \subseteq P_i$ for some i. Clearly, $|E_{H_A}(R, S)| = |R| \cdot |S_{1-i}| = 2M$. If $z_i \in Z_i$ is included in T_R, then the $M - 2$ detours from z_i to $P_i \setminus R$ pass through the edge e. If $z_i \in Z_i$ is included in T_S, then the two detours from z_i to R pass through e. In both cases, at least two detours for $z_i \in Z_i$ and its neighbor pass through e. Hence, $\mathrm{cng}_{H_A,T}(e) \geq 2M + 2(M - a_{3m}) > k$. □

Property 2.13 *There exist $i \in \{0, 1\}$ and $p_i \in P_i$ such that $N_T(p_i) \supseteq P_{1-i}$.*

Proof: Assume T has Property 2.12. Observe that there is an edge $e \in E(T)$ that lies between P_0 and P_1, since otherwise T is disconnected. Let $e = \{p_0, p_1\}$, where $p_i \in P_i$. By Property 2.12, we may assume, without loss of generality, that e divides $P_0 \cup P_1$ into $\{p_1\}$ and $(P_0 \cup P_1) \setminus \{p_1\}$. We shall show that p_0 is the desired vertex; that is, $N_T(p_0) \supseteq P_1$.

Let T_0 and T_1 be the connected components of $T - e$, where T_i includes p_i (see Figure 8). Let $p_1' \in P_1 \setminus \{p_1\}$. Since $p_0, p_1' \in V(T_0)$, a unique p_0–p_1' path in T is contained in T_0. It suffices to show that the path is a single edge $\{p_0, p_1'\} \in E(T)$. Suppose this is not the case. Let the unique p_0–p_1' path be $(p_0, s_1, \ldots, s_\ell, p_1')$ with $\ell \geq 1$. Clearly, $(p_1, p_0, s_1, \ldots, s_\ell, p_1')$ is a unique p_1–p_1' path in T. By Property 2.12, no inner vertex s_j can be included in $P_0 \cup P_1$, since otherwise the edge $\{p_0, s_1\}$ divides $P_0 \cup P_1$ into two non-singleton sets. Hence, $\{s_1, \ldots, s_\ell\} \subseteq Q_0 \cup Q_1$. Observe that $Q_0 \cup Q_1$ is an independent set of H_A. Thus we have $\ell = 1$ and the p_0–p_1' path is (p_0, s_1, p_1'). However, this cannot be happen, since no vertex in $Q_0 \cup Q_1$ has neighbors both in P_0 and P_1. □

In the rest of the proof, we assume T has Properties 2.12 and 2.13. We also assume without loss of generality that $i = 0$ in Property 2.13 and T is rooted at p_0.

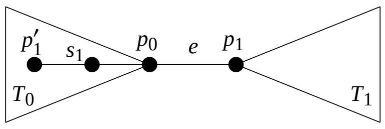

Figure 8: The p_0–p_1' path in T has to be a single edge.

Property 2.14 *All vertices in Q_1 are leaves of T, and $\mathrm{Ch}_T(p_1) \subseteq Q_1$ for any $p_1 \in P_1$.*

Proof: Observe that $N_{H_A}(Q_1) = P_1$. Thus if $q \in Q_1$ has two neighbors p and p' in T, then $p, p' \in P_1$. By Property 2.13, p and p' have p_0 as their common neighbor. Hence, T has a cycle induced by q, p, p', and p_0. This is a contradiction. Therefore, all vertices in Q_1 are leaves of T. If $p_1 \in P_1$ has a neighbor p_0' in $P_0 \setminus \{p_0\}$, then $\{p_0, p_1\} \in E(T)$ separates $\{p_1, p_0'\}$ and $\{p_0\} \cup P_1 \setminus \{p_1\}$. This contradicts Property 2.12. Thus for any $p_1 \in P_1$, $\mathrm{Ch}_T(p_1) \subseteq Q_1$. $\square$

Property 2.15 *For any $z_j^1 \in Z_1$, $\mathrm{prt}_T(z_j^1) \in W_1$ and $\mathrm{Ch}_T(\mathrm{prt}_T(z_j^1)) = \{z_j^1\}$.*

Proof: Clearly, $\mathrm{prt}_T(z_j^1) \in P_1$ and $\mathrm{Ch}_T(\mathrm{prt}_T(z_j^1)) \subseteq Q_1$ by Property 2.14. We first prove that $\mathrm{prt}_T(z_j^1) \in W_1$ for any $z_j^1 \in Z_1$. Since $\mathrm{Ch}_T(\mathrm{prt}_T(z_j^1)) \subseteq Q_1$ and vertices in Q_1 are leaves of T,

$$\begin{aligned}\mathrm{cng}_{H_A,T}(\{p_0, \mathrm{prt}_T(z_j^1)\}) &= |\theta_{H_A}(\{\mathrm{prt}_T(z_j^1)\} \cup \mathrm{Ch}_T(\mathrm{prt}_T(z_j^1)))| \\ &= \deg_{H_A}(\mathrm{prt}_T(z_j^1)) + \sum_{q \in \mathrm{Ch}_T(\mathrm{prt}_T(z_j^1))} (\deg_{H_A}(q) - 2) \\ &\geq \deg_{H_A}(\mathrm{prt}_T(z_j^1)) + \deg_{H_A}(z_j^1) - 2.\end{aligned}$$

If $\mathrm{prt}_T(z_j^1) \in U_1 \cup V_1$, then $\deg_{H_A}(\mathrm{prt}_T(z_j^1)) > 2M - m$. Since $\deg_{H_A}(z_j^1) = M$,

$$\deg_{H_A}(\mathrm{prt}_T(z_j^1)) + \deg_{H_A}(z_j^1) - 2 > 3M - m - 2 = k.$$

Thus $\mathrm{prt}_T(z_j^1) \in W_1$ and $\mathrm{Ch}_T(\mathrm{prt}_T(z_j^1)) \subseteq Y_1 \cup Z_1$. Let $\mathrm{prt}_T(z_j^1) = w_h^1 \in W_1$. Next we prove that z_j^1 is a unique child of w_h^1. Suppose $\mathrm{Ch}_T(w_h^1) \setminus \{z_j^1\} \neq \emptyset$. Then

$$\sum_{q \in \mathrm{Ch}_T(w_h^1) \setminus \{z_j^1\}} (\deg_{H_A}(q) - 2) \geq \min\{\deg_{H_A}(q) \mid q \in Y_1 \cup Z_1\} - 2 = \deg_{H_A}(y_1^1) - 2,$$

and thus

$$\begin{aligned}\mathrm{cng}_{H_A,T}(\{p_0, w_h^1\}) &\geq \deg_{H_A}(w_h^1) + \deg_{H_A}(z_j^1) - 2 + \deg_{H_A}(y_1^1) - 2 \\ &\geq (2M - a_{3m}) + M + (M - 3m) - 4 \\ &= k + M - a_{3m} - 2m - 2 \\ &= k + m + B - a_{3m} - 6 \\ &> k + m + B/2 - 6.\end{aligned}$$

Since $m \geq 3$ and $B \geq 8$, we have $\mathrm{cng}_{H_A,T}(\{p_0, w_h^1\}) > k$. □

Property 2.16 *T can be modified to T^* so that $\mathrm{cng}_{H_A}(T^*) = \mathrm{cng}_{H_A}(T)$, T^* has Properties 2.12–2.15, and additionally $\mathrm{prt}_{T^*}(y_j^1) \in V_1'$ and $\mathrm{Ch}(\mathrm{prt}_T(y_j^1)) = \{y_j^1\}$ for any $y_j^1 \in Y_1'$.*

Proof: By Property 2.14, $\mathrm{Ch}_T(\mathrm{prt}_T(y_j^1)) \subseteq Q_1$. By Property 2.15, $\mathrm{prt}_T(y_j^1) \in U_1 \cup V_1$. We first prove that $\mathrm{Ch}(\mathrm{prt}_T(y_j^1)) = \{y_j^1\}$. Suppose $\mathrm{Ch}(\mathrm{prt}_T(y_j^1)) \setminus \{y_j^1\} \neq \emptyset$. Since $\mathrm{Ch}_T(\mathrm{prt}_T(y_j^1)) \subseteq Q_1$ and vertices in Q_1 are leaves of T, we have

$$\begin{aligned}
\mathrm{cng}_{H_A,T}(\{p_0, \mathrm{prt}_T(y_j^1)\}) &= |\theta_{H_A}(\{\mathrm{prt}_T(y_j^1)\} \cup \mathrm{Ch}_T(\mathrm{prt}_T(y_j^1)))| \\
&\geq \min_{p \in U_1 \cup V_1} \deg_{H_A}(p) + \deg_{H_A}(y_j^1) - 2 \\
&\qquad + \min_{q \in Q_1} \left(\deg_{H_A}(q) - 2\right) \\
&> (2M - m) + (M - \gamma_j) - 2 + a_1 - 2 \\
&= k + a_1 - \gamma_j - 2.
\end{aligned}$$

Since $a_1 \geq 3m + 2$ and $\gamma_j \leq 3m$, we have $\mathrm{cng}_{H_A,T}(\{p_0, \mathrm{prt}_T(y_j^1)\}) > k$. This is a contradiction, and thus $\mathrm{Ch}(\mathrm{prt}_T(y_j^1)) = \{y_j^1\}$. Hence,

$$\mathrm{cng}_{H_A,T}(\{p_0, \mathrm{prt}_T(y_j^1)\}) = \deg_{H_A}(\mathrm{prt}_T(y_j^1)) + \deg_{H_A}(y_j^1) - 2.$$

Next we prove that $\mathrm{prt}_T(y_j^1) \in V_1 \setminus V_1'$ for any $y_j^1 \in Y_1 \setminus Y_1'$, where V_1' and Y_1' denote $\{v_h^1 \in V_1 \mid \gamma_h = 3m\}$ and $\{y_h^1 \in Y_1 \mid \gamma_h = 3m\}$, respectively. It is easy to see that $\deg_{H_A}(v) = 2M + 2m$ for any $v \in U_1 \cup V_1'$. Suppose that $\mathrm{prt}_T(y_j^1) \in U_1 \cup V_1'$ for some $y_j^1 \in Y_1'$. Then $\mathrm{cng}_{H_A,T}(\{p_0, \mathrm{prt}_T(y_j^1)\}) = (2M + 2m) + (M - \gamma_j) - 2 = k + 3m - \gamma_j > k$. This is a contradiction, and thus $\mathrm{prt}_T(y_j^1) \in V_1 \setminus V_1'$.

We now show that T can be modified to T^*, without loosing other properties, so that $\mathrm{prt}_{T^*}(y_j^1) \in V_1'$ for any $y_j^1 \in Y_1'$. Observe that $N_{H_A}(v_j^1) = N_{H_A}(\mathrm{prt}_T(y_j^1))$ for any $y_j^1 \in Y_1'$. Let h be the maximum index such that $\mathrm{prt}_T(y_h^1) \neq v_h^1$. (If there is no such h, then we are done.) It is easy to see that $\mathrm{Ch}_T(v_h^1) \subseteq X_1 \cup \{y_j \mid m + 1 \leq j \leq h - 1\}$. We swap the children of v_h^1 for the children of $\mathrm{prt}_T(y_h^1)$ (see Figure 9). That is, we remove E^- from T and then add E^+ to T, where E^- and E^+ denote the following edge sets:

$$\begin{aligned}
E^- &= \left\{\{y_h^1, \mathrm{prt}_T(y_h^1)\}\right\} \cup \left\{\{v_h^1, q_1\} \mid q_1 \in \mathrm{Ch}_T(v_h^1)\right\}, \\
E^+ &= \left\{\{y_h^1, v_h^1\}\right\} \cup \left\{\{\mathrm{prt}_T(y_h^1), q_1\} \mid q_1 \in \mathrm{Ch}_T(v_h^1)\right\}.
\end{aligned}$$

We call the new tree T'. It is not difficult to see that T' is a spanning tree of H_A, and $\mathrm{prt}_{T'}(y_h^1) = v_h^1 \in V_1$. Since $N_{H_A}(v_h^1) = N_{H_A}(\mathrm{prt}_T(y_h^1))$, it follows that $\mathrm{cng}_{H_A}(T') = \mathrm{cng}_{H_A}(T)$. Observe that if we denote by h' the maximum index such that $prt_{T'}(y_{h'}^1) \neq v_{h'}^1$, then $h' < h$ since $\mathrm{Ch}_T(v_h^1) \subseteq X_1 \cup \{y_j \mid m + 1 \leq j \leq h - 1\}$. Therefore, by repeatedly applying this swapping, we eventually obtain a desired tree T^*. □

As we mentioned earlier, Properties 2.12–2.16 imply the lemma. □

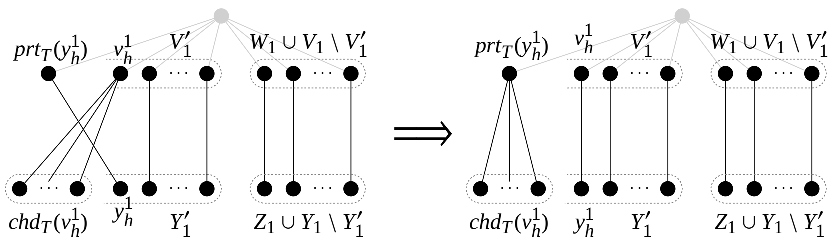

Figure 9: Swapping $\mathrm{Ch}_T(v_h^1)$ for $\mathrm{Ch}_T(\mathrm{prt}_T(y_h^1))$.

3 Exponential-time exact algorithm

We have shown that STC is NP-complete even for very simple graphs. It is widely believed that NP-hard problems cannot be solved in polynomial time. Thus we need *fast* exponential-time (or sub-exponential-time) algorithms for these problems. Nowadays, designing fast exponential-time exact algorithms becomes an important topic in theoretical computer science. See a recent textbook of exponential-time exact algorithms by Fomin and Kratsch [10]. For STC, we can easily design an $O^*(2^m)$- or $O^*(n^n)$-time algorithm that examine all spanning trees of input graphs, where n and m denote the number of vertices and the number of edges, respectively. In this section, we describe an algorithm for STC that runs in $O^*(2^n)$ time. Although it is still an exponential-time algorithm, it is significantly faster than a naive algorithm.

Let $G = (V, E)$ be a given undirected graph. For convenience, we denote $|\theta_G(X)|$ by $c(X)$. Note that $c(\emptyset) = c(V) = 0$. Consider a spanning tree T with congestion at most k. We regard T as a rooted tree with root $r \in V$. We denote this rooted tree by (T, r). Let $e = \{u, v\} \in E(T)$ be an edge of T, and without loss of generality, let u be the parent of v. Then, the congestion of e in T is equal to $c(D_{T,r}(v))$, where $D_{T,r}(v)$ denotes the set of descendants of v in (T, r). Since the congestion of T is at most k, we see that $c(D_{T,r}(v)) \leq k$. See Figure 10. Conversely, if $c(D_{T,r}(v)) \leq k$ for all $v \in V \setminus \{r\}$, then the congestion of T is at most k. This is because there exists a one-to-one correspondence between the edges e of T and the vertices v in $V \setminus \{r\}$ so that v is a deeper endpoint of e. We summarize this observation in the following lemma.

Lemma 3.1 *The congestion of a rooted tree (T, r) is at most k if and only if $c(D_{T,r}(v)) \leq k$ for every vertex $v \in V \setminus \{r\}$.* □

The lemma above suggests the following dynamic-programming approach. We call a pair (X, v) of a subset $X \subseteq V$ and a vertex $v \notin X$ a *rooted subset* of V. By definition, $X \neq V$ for a rooted subset (X, v) of V. A rooted subset (X, v) of V is *good* if there exists a rooted spanning tree (T, v) of $G[X \cup \{v\}]$ such that $c(D_{T,v}(u)) \leq k$ for all $u \in X$. Here, c is a cut function of G, not of $G[X \cup \{v\}]$. By definition (X, v) is good when $X = \emptyset$. Note that there exists a rooted spanning tree (T, r) of G with congestion at most k if and only if the

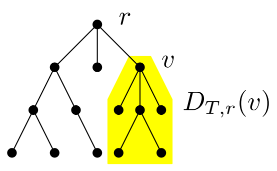

Figure 10: The definition of $D_{T,r}(v)$.

rooted set $(V \setminus \{r\}, r)$ is good.

The following lemma provides a recursive formula that forms a basis of our algorithm (see Figure 11).

Lemma 3.2 *Let (X, v) be a rooted subset of V with $|X| \geq 1$. Then, (X, v) is good if and only if at least one of the following holds.*

1. *$c(X) \leq k$ and there exists a vertex $u \in X \cap N_G(v)$ such that $(X \setminus \{u\}, u)$ is good.*
2. *There exists a non-empty proper subset $Y \subseteq X$ such that both of (Y, v), $(X \setminus Y, v)$ are good.*

Proof: For the only-if-part, assume that (X, v) is good. Then, there exists a rooted spanning tree (T, v) of $G[X \cup \{v\}]$ such that $c(D_{T,v}(u)) \leq k$ for all $u \in X$.

Now, we have two cases. In the first case, v has only one child u in T. Then, $u \in X \cap N_G(v)$, and $T - v$ is a spanning tree of $G[X]$ rooted at u. Hence, for all $u' \in X$ we have $c(D_{T-v,u}(u')) = c(D_{T,v}(u')) \leq k$, which means that $c(X) = c(D_{T,v}(u)) \leq k$ and $(X \setminus \{u\}, u)$ is good. Thus, Condition 1 is satisfied.

In the second case, v has at least two children in T. Let $u_1, \ldots, u_k$ be the children of v in T, and $T_1, \ldots, T_k$ be the subtrees of T rooted at $u_1, \ldots, u_k$, respectively. Then, we set Y to be the vertices of T_1. Then, $X \setminus Y$ is composed of the vertices of $T_2, \ldots, T_k$. Furthermore, as in the first case, we see that (Y, v) and $(X \setminus Y, v)$ are good. Thus, Condition 2 is satisfied.

We prove the if-part by induction on $|X|$. When $|X| = 1$, let $X = \{u\}$. First note that Condition 2 cannot be satisfied since $\{u\}$ cannot have a non-empty proper subset. Therefore, we assume that $c(\{u\}) \leq k$ and $u \in N_G(v)$. Then, there exists a unique spanning tree T of $G[\{u, v\}]$ rooted at v such that $D_{T,v}(v) = \{u, v\}$ and $D_{T,v}(u) = \{u\}$. Since $c(D_{T,v}(u)) = c(\{u\}) \leq k$, we see that $(\{u\}, v)$ is good.

We now proceed to the induction step, and let $|X| \geq 2$. If Condition 1 is satisfied, by the induction hypothesis, there exists a spanning tree T' of $G[X]$ rooted at u such that $c(D_{T',u}(u')) \leq k$ for all $u' \in X \setminus \{u\}$. Now we form a spanning tree T of $G[X \cup \{v\}]$ rooted at v, by attaching v to T' through u. This is possible since $u \in N_G(v)$ (from Condition 1). Then, $c(D_{T,v}(u')) = c(D_{T',u}(u')) \leq k$ for all $u' \in X$ by the induction hypothesis, and $c(D_{T,v}(u)) = c(X) \leq k$ by Condition 1. Thus, (X, v) is good.

If Condition 2 is satisfied, by the induction hypothesis, there exist a spanning tree T_1 of $G[Y \cup \{v\}]$ rooted at v and a spanning tree T_2 of $G[(X \setminus Y) \cup \{v\}]$

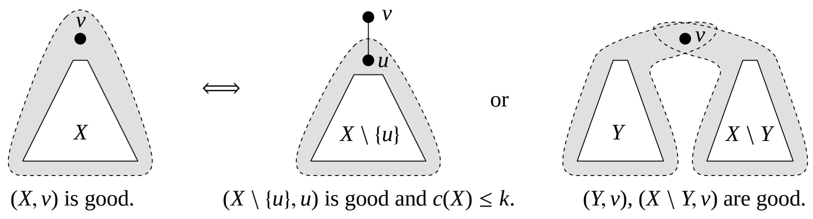
v
X
(X, v) is good.
⟺
v
u
X \ {u}
(X \ {u}, u) is good and c(X) ≤ k.
or
v
Y
X \ Y
(Y, v), (X \ Y, v) are good.

Figure 11: An illustration of Lemma 3.2.

rooted at v such that $c(D_{T_1,v}(u_1)) \le k$ for all $u_1 \in Y$ and $c(D_{T_2,v}(u_2)) \le k$ for all $u_2 \in X \setminus Y$. Now we form a spanning tree T of $G[X]$ from T_1 and T_2 by identifying their roots v. If $u \in Y$, then $c(D_{T,v}(u)) = c(D_{T_1,v}(u)) \le k$; If $u \in X \setminus Y$, then $c(D_{T,v}(u)) = c(D_{T_2,v}(u)) \le k$. Furthermore, $c(D_{T,v}(v)) = c(X) \le k$. Therefore, (X, v) is good. $\square$

Lemmas 3.1 and 3.2 above readily give an $O^*(3^n)$-time dynamic programming algorithm. However, the fast subset convolution method enables us to solve the problem in $O^*(2^n)$ time. We give more details below.

Let S be a finite set. For two functions $f, g\colon 2^S \to \mathbb{R}$, their *subset convolution* is a function $f * g\colon 2^S \to \mathbb{R}$ defined as

$$(f * g)(X) = \sum_{Y \subseteq X} f(Y) g(X \setminus Y)$$

for every $X \subseteq S$. Given $f(X), g(X)$ for all $X \subseteq S$, we can compute $(f * g)(X)$ for all $X \subseteq S$ in $O^*(2^n)$ total time, where $n = |S|$ [1] (see also Husfeldt's survey paper [15]).

Back to the spanning tree congestion problem, let $v \in V$ be an arbitrary vertex. We define the function $f_v\colon 2^{V \setminus \{v\}} \to \mathbb{R}$ by the following recursion: $f_v(X) = 1$ if $X = \emptyset$; otherwise,

$$f_v(X) = \sum_{u \in X \cap N_G(v)} f_u(X \setminus \{u\}) \max\{0, k - c(X) + 1\} + \sum_{\emptyset \ne Y \subsetneq X} f_v(Y) f_v(X \setminus Y),$$

where the empty sum is defined to be 0. It is easy to verify that $f_v(X)$ is non-negative for every $v \in V$ and every $X \subseteq V \setminus \{v\}$.

The following lemma connects the functions f_v, $v \in V$ and good rooted sets.

Lemma 3.3 *Let (X, v) be a pair of a subset $X \subseteq V \setminus \{v\}$ and a vertex $v \in V$. Then, $f_v(X) > 0$ if and only if (X, v) is a good rooted subset of V.*

Proof: For the if-part, let (X, v) be a good rooted subset of V. We prove that $f_v(X) > 0$ by the induction on $|X|$. The base case when $|X| = 0$ is straightforward from the definition of f_v.

If $|X| \ge 1$, then one of the two conditions in Lemma 3.2 holds. If Condition 1 is satisfied, then $c(X) \le k$ and, by the induction hypothesis, $f_u(X \setminus \{u\}) > 0$

for some $u \in X \cap N_G(v)$. Therefore, $f_v(X) > 0$ by definition. If Condition 2 is satisfied, then $f_v(Y)$ and $f_v(X \setminus Y)$ are positive for some non-empty proper subset $Y \subseteq X$. Thus, by definition, $f_v(X) > 0$.

For the only-if-part, assume that $f_v(X) > 0$. We now show that (X, v) is good by the induction on $|X|$. The base case when $|X| = 0$ is immediate. If $|X| \geq 1$, then one of the following two holds.

- $f_u(X \setminus \{u\}) \max\{0, k - c(X) + 1\} > 0$ for some $u \in X \cap N_G(v)$.
- $f_v(Y) f_v(X \setminus Y) > 0$ for some $\emptyset \neq Y \subsetneq X$.

When the former condition is satisfied, $f_u(X \setminus \{u\}) > 0$ and $c(X) \leq k$. By the induction hypothesis, $(X \setminus \{u\}, u)$ is good. Hence, Condition 1 in Lemma 3.2 holds, and (X, v) is good. When the latter condition is satisfied, by the induction hypothesis, (Y, v) and $(X \setminus Y, v)$ are good. Therefore, Condition 2 in Lemma 3.2 holds, and (X, v) is good. □

To apply the subset convolution method, we use the following functions. For each $i \in \{0, 1, \ldots, n-1\}$, where $n = |V|$, and $v \in V$, let $f_v^i \colon 2^{V \setminus \{v\}} \to \mathbb{R}$ be defined by

$$f_v^i(X) = \begin{cases} f_v(X) & \text{if } |X| \leq i, \\ 0 & \text{if } |X| > i, \end{cases}$$

for all $X \subseteq V \setminus \{v\}$. Then, it is not difficult to see the following.

1. For all $v \in X$ and $X \subseteq V \setminus \{v\}$, $f_v^{n-1}(X) = f_v(X)$.

$$f_v^{n-1}(X) = f_v(X).$$

2. For all $v \in V$ and $X \subseteq V \setminus \{v\}$,

$$f_v^0(X) = \begin{cases} 1 & \text{if } X = \emptyset, \\ 0 & \text{otherwise.} \end{cases}$$

3. For all $i \in \{1, \ldots, n-1\}$, $v \in V$, and $X \subseteq V \setminus \{v\}$

$$\begin{aligned} f_v^i(X) = & \sum_{u \in X \cap N_G(v)} f_u^{i-1}(X \setminus \{u\}) \max\{0, k - c(X) + 1\} \\ & + \sum_{\emptyset \neq Y \subsetneq X} f_v^{i-1}(Y) f_v^{i-1}(X \setminus Y) \\ = & \sum_{u \in X \cap N_G(v)} f_u^{i-1}(X \setminus \{u\}) \max\{0, k - c(X) + 1\} \\ & + \sum_{Y \subseteq X} f_v^{i-1}(Y) f_v^{i-1}(X \setminus Y) - 2 f_v^{i-1}(\emptyset) f_v^{i-1}(X) \\ = & \sum_{u \in X \cap N_G(v)} f_u^{i-1}(X \setminus \{u\}) \max\{0, k - c(X) + 1\} \\ & + (f_v^{i-1} * f_v^{i-1})(X) - 2 f_v^{i-1}(\emptyset) f_v^{i-1}(X). \end{aligned}$$

Our algorithm is based on these formulas.

Step 1. For all $v \in V$ and $X \subseteq V \setminus \{v\}$, compute $f_v^0(X)$ based on the formulas above.

Step 2. For each $i = 1, \ldots, n-1$ in the ascending order, do the following.

> **Step 2-1.** For all $v \in V$, compute the subset convolution $f_v^{i-1} * f_v^{i-1}$.
>
> **Step 2-2.** For all $v \in V$ and all $X \subseteq V \setminus \{v\}$, compute $f_v^i(X)$ based on the formula above.

Step 3. If $f_v^{n-1}(V) > 0$, then output Yes. Otherwise, output No.

The correctness is immediate from the discussion so far. The running time is $O^*(2^n)$ since the running time of each step is bounded by $O^*(2^n)$. This is an algorithm for solving the decision problem, but a simple binary search on $k \in \{1, \ldots, |E|\}$ can provide the spanning tree congestion. Thus, we obtain the following theorem.

Theorem 3.4 *The spanning tree congestion of a given undirected graph can be computed in $O^*(2^n)$ time.*

Note that the algorithm also works for the weighted case with the $O(n)$-factor increase of the running time, since the number of distinct cut values $c(X)$ is bounded by 2^n and so the binary search over the all possible values of $c(X)$ takes at most $O(\log(2^n)) = O(n)$ iterations. This is possible if we compute $c(X)$ for all $X \subseteq V$ beforehand, which only takes $O^*(2^n)$ time.

4 Remarks on cographs

We have shown the NP-completeness of STC for graphs of clique-width at most three. Here we study the spanning tree congestion of the graphs of clique-width at most two; that is, cographs [9]. To deal with STC on cographs, we first present two combinatorial lemmas.

Let $\mu_G(u, v)$ be the maximum number of edge-disjoint u–v paths in G, and let $\mu(G) = \max_{u,v \in V(G)} \mu_G(u, v)$. Ostrovskii [23] showed that $\mathrm{stc}(G) \geq \mu(G)$ for any graph G. Recall that a vertex $v \in V(G)$ is *universal* if v is adjacent to all other vertices in G.

Lemma 4.1 *The spanning tree congestion of a graph G with a universal vertex u is $\Delta(G - u) + 1$.*

Proof: Let G be a graph and u its universal vertex. The star spanning tree centered at u has congestion $\Delta(G-u)+1$. Let v be a vertex of $G-u$ with degree $\Delta(G - u)$. For each $w \in N_{G-u}(v)$, there is a u–v path (u, w, v) in G. Also, the edge $\{u, v\}$ itself is a u–v in G. Since these paths are pairwise edge-disjoint, we have $\mathrm{stc}(G) \geq \Delta(G - u) + 1$. □

Lemma 4.2 *The spanning tree congestion of $G = G_1 \oplus G_2$ can be approximated within a factor three in polynomial time.*

Proof: For the sake of simplicity, let n_i and Δ_i denote $|V(G_i)|$ and $\Delta(G_i)$, respectively. We assume $n_1, n_2 \geq 2$ since otherwise we can determine $\mathrm{stc}(G)$ by Lemma 4.1.

Let $u, v \in V(G_1)$. Since $G = G_1 \oplus G_2$, both $N_G(u)$ and $N_G(v)$ contain $V(G_2)$. Therefore, $\mu(G) \geq \mu_G(u,v) \geq n_2$. By symmetry, we have $\mu(G) \geq n_1$ as well. For $i \in \{1,2\}$, let u_i be a maximum degree vertex of G_i. For each $w \in N_{G_1}(u_1) \cup N_{G_2}(u_2)$, we have a u_1–u_2 path (u_1, w, u_2) in G. Thus $\mu(G) \geq \Delta_1 + \Delta_2$. Combining these bounds, we have

$$\mathrm{stc}(G) \geq \mu(G) \geq \max\{n_1, n_2, \Delta_1 + \Delta_2\} \geq (n_1 + n_2 + \Delta_1 + \Delta_2)/3.$$

Let $V(G_i) = \{v_1^i, v_2^i, \ldots, v_{n_i}^i\}$ for $i \in \{1,2\}$. Assume $n_1 \leq n_2$ without loss of generality. We construct a spanning tree T of G as follows (in polynomial time):

$$E(T) = \{\{v_1^1, v_j^2\} \mid 1 \leq j \leq n_2\} \cup \{\{v_j^1, v_j^2\} \mid 2 \leq j \leq n_1\}.$$

Every leaf edge in T has congestion at most $\Delta(G) = \max\{\Delta_1 + n_2, \Delta_2 + n_1\}$. It is easy to see that every inner edge e of T divides $V(G)$ into $\{v_1, v_2\}$ and its complement, where $v_i \in V(G_i)$. Therefore, $\mathrm{cng}_{G,T}(e) \leq \deg_G(v_1) + \deg_G(v_2) \leq \Delta_1 + n_2 + \Delta_2 + n_1$. Thus $\mathrm{cng}_G(T) \leq \Delta_1 + n_2 + \Delta_2 + n_1 \leq 3\mathrm{stc}(G)$. □

Let G be a connected cograph with at least two vertices. Then G can be expressed as $G_1 \oplus G_2$ for some nonempty cographs G_1 and G_2. Furthermore, if $G = G_1 \oplus G_2$ is a chordal cograph, then at least one of G_1 and G_2 is K_1 [28]. Therefore, every connected chordal cograph with at least two vertices has a universal vertex. By Lemmas 4.1 and 4.2, we have the following corollary.

Corollary 4.3 *The spanning tree congestion of cographs can be approximated within a factor three in polynomial time. Furthermore, the spanning tree congestion of chordal cographs can be determined in linear time.*

5 Conclusion

We showed that the problem of determining the spanning tree congestion is NP-hard even for very simple graphs such as split graphs and chain graphs. To cope with the hardness, we presented an exact $O^*(2^n)$-time algorithm for the spanning tree congestion problem, where n is the number of vertices.

The complexity of STC for some important graph classes such as cographs and interval graphs remains unsettled. Note that k-STC is solvable in linear time even for chordal graphs (and thus for interval graphs). To see this, recall that a chordal graph has small treewidth if and only if it contains no large clique [3]. Since a large clique contains many edge disjoint paths, we can conclude that a chordal graph has small spanning tree congestion only if its treewidth is small. Thus we have a linear-time algorithm for k-STC on chordal graphs, since small treewidth can be checked in linear time [2] and k-STC can be solved in linear time for small treewidth graphs [4].

Acknowledgments

Part of this research is supported by the Funding Program for World-Leading Innovative R&D on Science and Technology, Japan, and Grants-in-Aid for Scientific Research from Ministry of Education, Science and Culture, Japan, and Japan Society for the Promotion of Science.

References

[1] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Fourier meets Möbius: Fast subset convolution. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC '07)*, pages 67–74, 2007.

[2] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25:1305–1317, 1996.

[3] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoret. Comput. Sci.*, 209:1–45, 1998.

[4] H. L. Bodlaender, F. V. Fomin, P. A. Golovach, Y. Otachi, and E. J. van Leeuwen. Parameterized complexity of the spanning tree congestion problem. *Algorithmica*, to appear.

[5] H. L. Bodlaender, K. Kozawa, T. Matsushima, and Y. Otachi. Spanning tree congestion of k-outerplanar graphs. *Discrete Math.*, 311:1040–1045, 2011.

[6] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph Classes: A Survey.* SIAM, 1999.

[7] A. Brandstädt and V. V. Lozin. On the linear structure and clique-width of bipartite permutation graphs. *Ars Combin.*, 67:273–281, 2003.

[8] A. Castejón and M. I. Ostrovskii. Minimum congestion spanning trees of grids and discrete toruses. *Discuss. Math. Graph Theory*, 29:511–519, 2009.

[9] B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Appl. Math.*, 101:77–114, 2000.

[10] F. V. Fomin and D. Kratsch. *Exact Exponential Algorithms.* Springer, 2010.

[11] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, 1979.

[12] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*, volume 57 of *Annals of Discrete Mathematics.* North Holland, second edition, 2004.

[13] P. Hliněný, S. Oum, D. Seese, and G. Gottlob. Width parameters beyond tree-width and their applications. *Comput. J.*, 51:326–362, 2008.

[14] S. W. Hruska. On tree congestion of graphs. *Discrete Math.*, 308:1801–1809, 2008.

[15] T. Husfeldt. Invitation to algorithmic uses of inclusion–exclusion. In *Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP 2011), Part II*, volume 6756 of *Lecture Notes in Comput. Sci.*, pages 42–59. Springer-Verlag, 2011.

[16] K. Kozawa and Y. Otachi. Spanning tree congestion of rook's graphs. *Discuss. Math. Graph Theory*, 31:753–761, 2011.

[17] K. Kozawa, Y. Otachi, and K. Yamazaki. On spanning tree congestion of graphs. *Discrete Math.*, 309:4215–4224, 2009.

[18] H.-F. Law. Spanning tree congestion of the hypercube. *Discrete Math.*, 309:6644–6648, 2009.

[19] H.-F. Law and M. I. Ostrovskii. Spanning tree congestion: duality and isoperimetry; with an application to multipartite graphs. *Graph Theory Notes N. Y.*, 58:18–26, 2010.

[20] H.-F. Law and M. I. Ostrovskii. Spanning tree congestion of some product graphs. *Indian J. Math.*, 52:103–111, 2011.

[21] C. Löwenstein, D. Rautenbach, and F. Regen. On spanning tree congestion. *Discrete Math.*, 309:4653–4655, 2009.

[22] Y. Okamoto, Y. Otachi, R. Uehara, and T. Uno. Hardness results and an exact exponential algorithm for the spanning tree congestion problem. In *Proceedings of the 8th Annual Conference on Theory and Applications of Models of Computation (TAMC 2011)*, volume 6648 of *Lecture Notes in Comput. Sci.*, pages 452–462. Springer-Verlag, 2011.

[23] M. I. Ostrovskii. Minimal congestion trees. *Discrete Math.*, 285:219–226, 2004.

[24] M. I. Ostrovskii. Minimum congestion spanning trees in planar graphs. *Discrete Math.*, 310:1204–1209, 2010.

[25] M. I. Ostrovskii. Minimum congestion spanning trees in bipartite and random graphs. *ActaMath. Sci.*, 31B:634–640, 2011.

[26] Y. Otachi, H. L. Bodlaender, and E. J. van Leeuwen. Complexity results for the spanning tree congestion problem. In *Proceedings of the 36th International Workshop on Graph Theoretic Concepts in Computer Science (WG 2010)*, volume 6410 of *Lecture Notes in Comput. Sci.*, pages 3–14. Springer-Verlag, 2010.

[27] S. Simonson. A variation on the min cut linear arrangement problem. *Math. Syst. Theory*, 20:235–252, 1987.

[28] J.-H. Yan, J.-J. Chen, and G. J. Chang. Quasi-threshold graphs. *Discrete Appl. Math.*, 69:247–255, 1996.

[29] M. Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM J. Alg. Disc. Meth.*, 2:77–79, 1981.