Better Hardness Results for the Minimum Spanning Tree Congestion Problem

Huong Luu(✉) and Marek Chrobak

Department of Computer Science, University of California at Riverside, Riverside, USA
hluu008@ucr.edu

Abstract. In the spanning tree congestion problem, given a connected graph G, the objective is to compute a spanning tree T in G for which the maximum edge congestion is minimized, where the congestion of an edge e of T is the number of vertex pairs adjacent in G for which the path connecting them in T traverses e. The problem is known to be $\mathbb{NP}$-hard, but its approximability is still poorly understood, and it is not even known whether the optimum can be efficiently approximated with ratio $o(n)$. In the decision version of this problem, denoted $K-\mathsf{STC}$, we need to determine if G has a spanning tree with congestion at most K. It is known that $K-\mathsf{STC}$ is $\mathbb{NP}$-complete for $K \geq 8$, and this implies a lower bound of 1.125 on the approximation ratio of minimizing congestion. On the other hand, $3-\mathsf{STC}$ can be solved in polynomial time, with the complexity status of this problem for $K \in \{4, 5, 6, 7\}$ remaining an open problem. We substantially improve the earlier hardness result by proving that $K-\mathsf{STC}$ is $\mathbb{NP}$-complete for $K \geq 5$. This leaves only the case $K = 4$ open, and improves the lower bound on the approximation ratio to 1.2.

1 Introduction

Problems involving constructing a spanning tree that satisfies certain requirements are among the most fundamental tasks in graph theory and algorithmics. One such problem is the *spanning tree congestion problem*, STC for short, that has been studied extensively for many years. Roughly, in this problem we seek a spanning tree T of a given graph G that approximates the connectivity structure of G in the following sense: Embed G into T by replacing each edge (u, v) of G by the unique u-to-v path in T. Define the *congestion of an edge e of T* as the number of such paths that traverse e. The objective of STC is to find a spanning tree T that minimizes the maximum edge congestion.

The general concept of edge congestion was first introduced in 1986, under the name of *load factor*, as a measure of quality of an embedding of one graph into another [3] (see also the survey in [20]). The problem of computing trees with low congestion was studied by Khuller *et al.* [12] in the context of solving commodities network routing problems. The trees considered there were not required to be

M. Chrobak—Research partially supported by National Science Foundation grant CCF-2153723.

C.-C. Lin et al. (Eds.): WALCOM 2023, LNCS 13973, pp. 167–178, 2023.
https://doi.org/10.1007/978-3-031-27051-2_15

spanning subtrees, but the variant involving spanning trees was also mentioned. In 2003, Ostrovskii provided independently a formal definition of STC and established some fundamental properties of spanning trees with low congestion [17]. Since then, many combinatorial and algorithmic results about this problem have been reported in the literature — we refer the readers to the survey paper by Otachi [18] for complete information, most of which is still up-to-date.

As established by Löwenstein [15], STC is $\mathbb{NP}$-hard. As usual, this is proved by showing $\mathbb{NP}$-completeness of its decision version, where we are given a graph G and an integer K, and we need to determine if G has a spanning tree with congestion at most K. Otachi *et al.* [19] strengthened this by proving that the problem remains $\mathbb{NP}$-hard even for planar graphs. In [16], STC is proven to be $\mathbb{NP}$-hard for chain graphs and split graphs. On the other hand, computing optimal solutions for STC can be achieved in polynomial time for some special classes of graphs: complete k-partite graphs, two-dimensional tori [14], outerplanar graphs [5], and two-dimensional Hamming graphs [13].

In our paper, we focus on the decision version of STC where the bound K on congestion is a fixed constant. We denote this variant by $K-$STC. Several results on the complexity of $K-$STC were reported in [19]. For example, the authors show that $K-$STC is decidable in linear time for planar graphs, graphs of bounded treewidth, graphs of bounded degree, and for all graphs when $K = 1, 2, 3$. On the other hand, they show that the problem is $\mathbb{NP}$-complete for any fixed $K \geq 10$. In [4], Bodlaender *et al.* proved that $K-$STC is linear-time solvable for graphs in apex-minor-free families and chordal graphs. They also show an improved hardness result of $K-$STC, namely that it is $\mathbb{NP}$-complete for $K \geq 8$, even in the special case of apex graphs that only have one unbounded degree vertex. As stated in [18], the complexity status of $K-$STC for $K \in \{4, 5, 6, 7\}$ remains an open problem.

Little is known about the approximability of STC. The trivial upper bound for the approximation ratio is $n/2$ [18]. As a direct consequence of the $\mathbb{NP}$-completeness of $8-$STC, there is no polynomial-time algorithm to approximate the optimum spanning tree congestion with a ratio better than 1.125 (unless $\mathbb{P} = \mathbb{NP}$).

Our Contribution. Addressing an open question in [18], we provide an improved hardness result for $K-$STC:

Theorem 1. *For any fixed integer $K \geq 5$, $K-$STC is $\mathbb{NP}$-complete.*

The proof of this theorem is given in Sect. 3. Combined with the results in [19], Theorem 1 leaves only the status of $4-$STC open. Furthermore, it also immediately improves the lower bound on the approximation ratio for STC:

Corollary 1. *For $c < 1.2$ there is no polynomial-time c-approximation algorithm for STC, unless $\mathbb{P} = \mathbb{NP}$.*

We remark that this hardness result remains valid even if an additive constant is allowed in the approximation bound. This follows by an argument in [4]. (In

essence, the reason is that assigning a positive integer weight β to each edge increases its congestion by a factor β.)

Other Related Work. The spanning tree congestion problem is closely related to the tree spanner problem in which the objective is to find a spanning tree T of G that minimizes the stretch factor, defined as the maximum ratio, over all vertex pairs, between the length of the path in T and the length of the shortest path in G connecting these vertices. In fact, for any planar graph, its spanning tree congestion is equal to its dual's minimum stretch factor plus one [10,19]. This direction of research has been extensively explored, see [6,8,9]. As an aside, we remark that the complexity of the tree 3-spanner problem has been open since its first introduction in 1995 [6].

STC is also intimately related to problems involving cycle bases in graphs. As each spanning tree identifies a fundamental cycle basis of a given graph, a spanning tree with low congestion yields a cycle basis for which the edge-cycle incidence matrix is sparse. Sparsity of such matrices is desirable in linear-algebraic approaches to solving some graph optimization problems, for example analyses of distribution networks such as in pipe flow systems [1].

STC can be considered as an extreme case of the graph sparsification problem, where, given a graph G, the objective is to compute a sparse graph H that captures connectivity properties of G. Such H can be used instead of G for the purpose of various analyses, to improve efficiency. See [2,11,21] (and the references therein) for some approaches to graph sparsification.

2 Preliminaries

Let $G = (V, E)$ be a simple graph with vertex set V and edge set E. Consider a spanning tree $T \subseteq E$ of G. If $e = (u, v) \in T$, removing e from T splits T into two components. We denote by $T_{u,v}$ the component that contains u and by $T_{v,u}$ the component that contains v. Let the *cross-edge set* of e, denoted $\partial_{G,T}(e)$, be the set of edges in E that have one endpoint in $T_{u,v}$ and the other in $T_{v,u}$. In other words, $\partial_{G,T}(e)$ consists of the edges $(u', v') \in E$ for which the unique (simple) path in T from u' to v' goes through e. Note that $e \in \partial_{G,T}(e)$. The *congestion of* e, denoted by $\mathrm{cng}_{G,T}(e)$, is the cardinality of $\partial_{G,T}(e)$. The *congestion of tree* T is $\mathrm{cng}_G(T) = \max_{e \in T} \mathrm{cng}_{G,T}(e)$. Finally, the *spanning tree congestion of graph* G, denoted by $\mathrm{stc}(G)$, is defined as the minimum value of $\mathrm{cng}_G(T)$ over all spanning trees T of G.

The concept of the spanning tree congestion extends naturally to multigraphs. For multigraphs, only one edge between any two given vertices can be in a spanning tree, but all of them belong to the cross-edge set $\partial_{G,T}(e)$ of any edge $e \in T$ whose removal separates these vertices in T (and thus all contribute to $\mathrm{cng}_{G,T}(e)$). As observed in [19], edge subdivision does not affect the spanning tree congestion of a graph. Therefore any multigraph can be converted into a simple graph by subdividing all multiple edges, without changing its minimum congestion. We use positive integer weights to represent edge multiplicities: an edge (u, v) with weight ω represents a bundle of ω edges connecting u to v. While

we state our results in terms of simple graphs, we use weighted graphs in our proofs, with the understanding that they actually represent the corresponding simple graphs. As all weights used in the paper are constant, the computational complexity of $K-\mathsf{STC}$ is not affected.

In fact, it is convenient to generalize this further by introducing edges with *double weights*. A double weight of an edge e is denoted $\omega:\omega'$, where ω and ω' are positive integers such that $\omega \le \omega'$, and its interpretation in the context of $K-\mathsf{STC}$ is as follows: given a spanning tree T, if $e \in E\backslash T$ then e contributes ω to the congestion $\mathrm{cng}_{G,T}(f)$ of any edge f for which $e \in \partial_{G,T}(f)$, and if $e \in T$ then e contributes ω' to its own congestion, $\mathrm{cng}_{G,T}(e)$. The lemma below implies that including edges with double weights that add up to at most K does not affect the computational complexity of $K-\mathsf{STC}$, and therefore we can formulate our proofs in terms of graphs where some edges have double weights.

Lemma 1. *Let (u,v) be an edge in G with double weight $\omega:\omega'$, where $1 \le \omega \le \omega'$ and $\omega+\omega' \le K$ for some integer K. Consider another graph G' with vertex set $V' = V \cup \{w\}$ and edge set $E' = E \cup \{(u,w),(w,v)\}\backslash\{(u,v)\}$, in which the weight of (u,w) is ω and the weight of (w,v) is ω'. Then, $stc(G) \le K$ if and only if $stc(G') \le K$.*

Proof. ($\Rightarrow$) Suppose that G has a spanning tree T with $\mathrm{cng}_G(T) \le K$. We will show that there exists a spanning tree T' of G' with $\mathrm{cng}_{G'}(T') \le K$. We break the proof into two cases, in both cases showing that $\mathrm{cng}_{G',T'}(e) \le K$ for each edge $e \in T'$.

Case 1: $(u,v) \in T$. Let $T' = T \cup \{(u,w),(w,v)\}\backslash\{(u,v)\}$. T' is clearly a spanning tree of G'. If $(x,y) \in E'\backslash\{(u,w),(w,v)\}$, the x-to-y paths in T and T' are the same, except that if the x-to-y path in T traverses edge (u,v) then the x-to-y path in T' will traverse (u,w) and (w,v) instead. Therefore, if $e \in T'\backslash\{(u,w),(w,v)\}$, $\partial_{G',T'}(e) = \partial_{G,T}(e)$, so $\mathrm{cng}_{G',T'}(e) = \mathrm{cng}_{G,T}(e) \le K$. On the other hand, if $e \in \{(u,w),(w,v)\}$, $\partial_{G',T'}(e) = \partial_{G,T}(u,v)\backslash\{(u,v)\}\cup\{e\}$. Then, edge e contributes ω or ω' to $\mathrm{cng}_{G',T'}(e)$, while (u,v), by the definition of double weights, contributes $\omega' \ge \omega$ to $\mathrm{cng}_{G,T}(u,v)$. Hence, $\mathrm{cng}_{G',T'}(e) \le \mathrm{cng}_{G,T}(u,v) \le K$.

Case 2: $(u,v) \notin T$. Let $T' = T \cup \{(w,v)\}$, which is a spanning tree of G'. If $e \in T'\backslash\{(w,v)\}$, we have two subcases. If e is not on the u-to-v path in T', $\partial_{G',T'}(e) = \partial_{G,T}(e)$, so $\mathrm{cng}_{G',T'}(e) = \mathrm{cng}_{G,T}(e) \le K$. If e is on the u-to-v path in T', $\partial_{G',T'}(e) = \partial_{G,T}(e)\cup\{(u,w)\}\backslash\{(u,v)\}$. As (u,w) contributes ω to $\mathrm{cng}_{G',T'}(e)$ and, by the definition of double weights, (u,v) contributes ω to $\mathrm{cng}_{G,T}(e)$, we obtain that $\mathrm{cng}_{G',T'}(e) = \mathrm{cng}_{G,T}(e) \le K$. In the remaining case, for $e = (w,v)$, we have $\partial_{G',T'}(e) = \{(u,w),(w,v)\}$, so $\mathrm{cng}_{G',T'}(e) = \omega+\omega' \le K$.

($\Leftarrow$) Let T' be the spanning tree of G' with congestion $\mathrm{cng}_{G'}(T') \le K$. We will show that there exists a spanning tree T of G with $\mathrm{cng}_G(T) \le K$. Note that at least one of edges (u,w) and (v,w) has to be in T'. We now consider three cases, in each case showing that $\mathrm{cng}_{G,T}(e) \le K$ for each edge $e \in T$.

Case 1: $(u,w),(v,w) \in T'$. Let $T = T' \cup \{(u,v)\}\backslash\{(u,w),(w,v)\}$. T is clearly a spanning tree of G. The argument for this case is similar to Case 1 in the proof

for the ($\Rightarrow$) implication. For each edge $e \in T \backslash \{(u,v)\}$, its congestion in T is the same as in T'. The congestion of (u,v) in T is bounded by the congestion of (w,v) in T', which is at most K.

Case 2: $(v,w) \in T'$ and $(u,w) \notin T'$. Let $T = T' \backslash \{(w,v)\}$. T is a spanning tree of G. Here again, the argument is similar to the proof for Case 2 in the ($\Rightarrow$) implication. For each edge $e \in T$, if e is not on the u-to-v path in T, its congestion in T and T' is the same. If e is on the u-to-v path in T, the contributions of (u,v) and (u,w) to the congestion of e in T and T' are the same.

Case 3: $(u,w) \in T'$ and $(v,w) \notin T'$. Consider $T'' = T' \cup \{(v,w)\} \backslash \{(u,w)\}$, which is a different spanning tree of G'. It is sufficient to show that $\mathrm{cng}_{G'}(T'') \leq \mathrm{cng}_{G'}(T')$ because it will imply $\mathrm{cng}_{G'}(T'') \leq K$, and then we can apply Case 2 to T''. We examine the congestion values of each edge $e \in T''$. Suppose first that $e \neq (u,w)$. If e is not on the u-to-v path in T', $\partial_{G',T''}(e) = \partial_{G',T'}(e)$, so $\mathrm{cng}_{G',T''}(e) = \mathrm{cng}_{G',T'}(e)$. If e is on the u-to-v path in T', $\partial_{G',T''}(e) = \partial_{G',T'}(e) \cup \{(u,w)\} \backslash \{(v,w)\}$, so $\mathrm{cng}_{G',T''}(e) = \mathrm{cng}_{G',T'}(e) + \omega - \omega' \leq \mathrm{cng}_{G',T'}(e)$. In the last case when $e = (v,w)$, $\mathrm{cng}_{G',T''}(e) = \omega + \omega' \leq K$.

3 $\mathbb{NP}$-Completeness Proof of $K-$STC for $K \geq 5$

In this section we prove our main result, the $\mathbb{NP}$-completeness of $K-$STC. Our proof uses an $\mathbb{NP}$-complete variant of the satisfiability problem called (2P1N)-SAT [7,22]. An instance of (2P1N)-SAT is a boolean expression ϕ in conjunctive normal form, where each variable occurs exactly three times, twice positively and once negatively, and each clause contains exactly two or three literals of different variables. The objective is to decide if ϕ is satisfiable, that is if there is a satisfying assignment that makes ϕ true.

For each constant K, $K-$STC is clearly in $\mathbb{NP}$. We will present a polynomial-time reduction from (2P1N)-SAT. In this reduction, given an instance ϕ of (2P1N)-SAT, we construct in polynomial time a graph G with the following property:

($*$) ϕ has a satisfying truth assignment if and only if $\mathrm{stc}(G) \leq K$.

Throughout the proof, the three literals of x_i in ϕ will be denoted by x_i, x_i', and $\bar{x}_i$, where x_i, x_i' are the two positive occurrences of x_i and $\bar{x}_i$ is the negative occurrence of x_i. We will also use notation $\tilde{x}_i$ to refer to an unspecified literal of x_i, that is $\tilde{x}_i \in \{x_i, x_i', \bar{x}_i\}$.

We now describe the reduction. Set $k_i = K - i$ for $i = 1,2,3,4$. (In particular, for $K = 5$, we have $k_1 = 4$, $k_2 = 3$, $k_3 = 2$, $k_4 = 1$.) G will consist of gadgets corresponding to variables, with the gadget corresponding to x_i having three vertices x_i, x_i', and $\bar{x}_i$, that represent its three occurrences in the clauses. G will also have vertices representing clauses and edges connecting literals with the clauses where they occur (see Fig. 1b for an example). As explained in Sect. 2, without any loss of generality we can allow edges in G to have constant-valued weights, single or double. Specifically, starting with G empty, the construction of G proceeds as follows:

– Add a *root vertex* r.
– For each variable x_i, construct the x_i-gadget (see Fig. 1a). This gadget has three vertices corresponding to the literals: a *negative literal vertex* $\bar{x}_i$ and two *positive literal vertices* x_i, x_i', and two auxiliary vertices y_i and z_i. Its edges and their weights are given in the table below:

edge	$(\bar{x}_i, z_i)$	(z_i, x_i)	(x_i, x_i')	(r, x_i')	(r, y_i)	(y_i, z_i)	$(y_i, \bar{x}_i)$
weight	$1:k_3$	$1:k_3$	$1:k_2$	k_3	k_4	k_4	$1:k_2$

– For each clause c, create a *clause vertex* c. For each literal $\tilde{x}_i$ in c, add the corresponding *clause-to-literal edge* $(c, \tilde{x}_i)$ of weight $1:k_2$. Importantly, as all literals in c correspond to different variables, these edges will go to different variable gadgets.
– For each two-literal clause c, add a *root-to-clause* edge (r, c) of weight $1:k_1$.

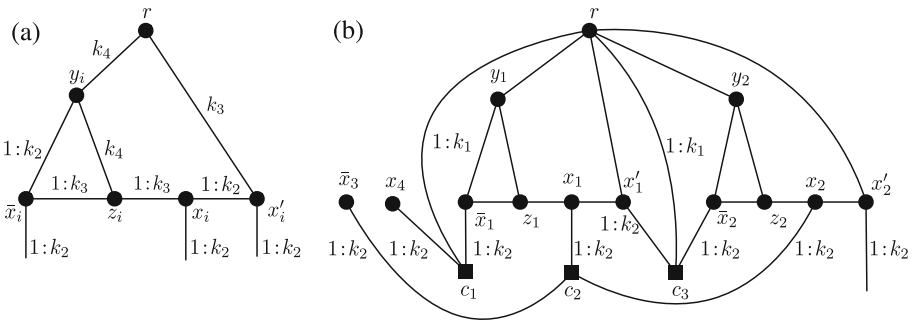

Fig. 1. (a)The x_i-gadget. (b) An example of a partial graph G for the boolean expression $\phi = (\bar{x}_1 \vee x_4) \wedge (x_1 \vee x_2 \vee \bar{x}_3) \wedge (x_1 \vee \bar{x}_2) \wedge \cdots$. Here, $c_1 = \bar{x}_1 \vee x_4$, $c_2 = x_1 \vee x_2 \vee \bar{x}_3$, and $c_3 = x_1 \vee \bar{x}_2$.

We now show that G has the required property $(*)$, proving the two implications separately.

$(\Rightarrow)$ Suppose that ϕ has a satisfying assignment. Using this assignment, we construct a spanning tree T of G as follows:

– For every x_i-gadget, include in T edges (r, x_i'), (r, y_i), and (y_i, z_i). If $x_i = 0$, include in T edges $(\bar{x}_i, z_i)$ and (x_i, x_i'), otherwise include in T edges $(y_i, \bar{x}_i)$ and (z_i, x_i).
– For each clause c, include in T one clause-to-literal edge that is incident to any literal vertex that satisfies c in our chosen truth assignment for ϕ.

By routine inspection, T is indeed a spanning tree: Each x_i-gadget is traversed from r without cycles, and all clause vertices are leaves of T. Figures 2 and 3 show how T traverses an x_i-gadget in different cases, depending on whether

$x_i = 0$ or $x_i = 1$ in the truth assignment for ϕ, and on which literals are chosen to satisfy each clause. Note that the edges with double weights satisfy the assumption of Lemma 1 in Sect. 2, that is each such weight $1:\omega'$ satisfies $1 \leq \omega'$ and $1 + \omega' \leq K$.

We need to verify that each edge in T has congestion at most K. All the clause vertices are leaves in T, thus the congestion of each clause-to-literal edge is $k_2 + 2 = K$; this holds for both three-literal and two-literal clauses. To analyze the congestion of the edges inside an x_i-gadget, we consider two cases, depending on the value of x_i in our truth assignment.

When $x_i = 0$, we have two sub-cases as shown in Fig. 2. The congestions of the edges in the x_i-gadget are as follows:

- In both cases, $\text{cng}_{G,T}(r, x'_i) = k_3 + 3$.
- In case (a), $\text{cng}_{G,T}(r, y_i) = k_4 + 3$. In case (b), it is $k_4 + 2$.
- In case (a), $\text{cng}_{G,T}(y_i, z_i) = k_4 + 4$. In case (b), it is $k_4 + 3$.
- In case (a), $\text{cng}_{G,T}(\bar{x}_i, z_i) = k_3 + 3$. In case (b), it is $k_3 + 2$.
- In both cases, $\text{cng}_{G,T}(x_i, x'_i) = k_2 + 2$.

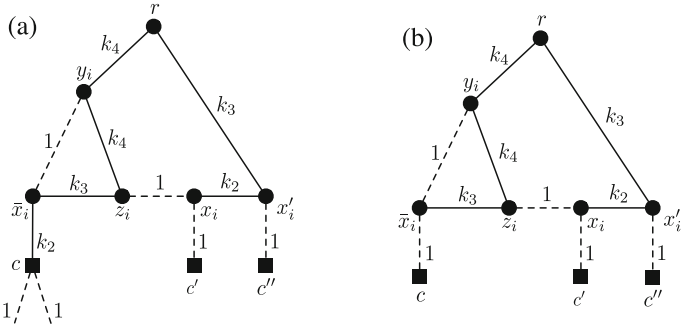

Fig. 2. The traversal of the x_i-gadget by T when $x_i = 0$. Solid lines are tree edges, dashed lines are non-tree edges. (a) $\bar{x}_i$ is chosen by clause c. (b) $\bar{x}_i$ is not chosen by clause c.

On the other hand, when $x_i = 1$, we have four sub-cases. Figure 2 illustrates cases (a)–(c). In case (d) (not shown in Fig. 2), none of the positive literal vertices x_i, x'_i is chosen to satisfy their corresponding clauses. The congestions of the edges in the x_i-gadget are as follows:

- In cases (a) and (b), $\text{cng}_{G,T}(r, x'_i) = k_3 + 3$. In cases (c) and (d), it is $k_3 + 2$.
- In cases (a) and (c), $\text{cng}_{G,T}(r, y_i) = k_4 + 4$. In cases (b) and (d), it is $k_4 + 3$.
- In cases (a) and (c), $\text{cng}_{G,T}(y_i, z_i) = k_4 + 4$. In cases (b) and (d), it is $k_4 + 3$.
- In cases (a) and (c), $\text{cng}_{G,T}(z_i, x_i) = k_3 + 3$. In cases (b) and (d), it is $k_3 + 2$.
- In all cases, $\text{cng}_{G,T}(y_i, \bar{x}_i) = k_2 + 2$.

In summary, the congestion of each edge of T is at most K. Thus $\text{cng}_G(T) \leq K$; in turn, $\text{stc}(G) \leq K$, as claimed.

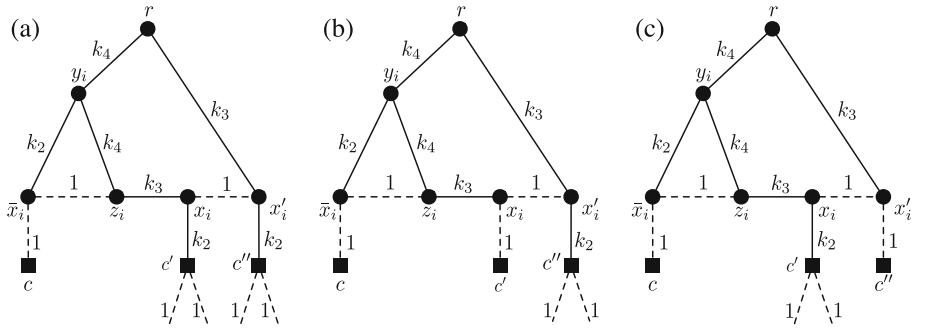

Fig. 3. The traversal of the x_i-gadget by T when $x_i = 1$. By c, c' and c'' we denote the clauses that contain literals $\bar{x}_i$, x_i and x'_i, respectively. (a) x_i and x'_i are chosen by clauses c' and c''. (b) x'_i is chosen by clause c''. (c) x_i is chosen by clause c'.

($\Leftarrow$) We now prove the other implication in ($*$). We assume that G has a spanning tree T with $\text{cng}_G(T) \leq K$. We will show how to convert T into a satisfying assignment for ϕ. The proof consists of a sequence of claims showing that T must have a special form that will allow us to define this truth assignment.

Claim 1. *Each x_i-gadget satisfies the following property: for each literal vertex $\tilde{x}_i$, if some edge e of T (not necessarily in the x_i-gadget) is on the r-to-$\tilde{x}_i$ path in T, then $\partial_{G,T}(e)$ contains at least two distinct edges from this gadget other than (y_i, z_i).*

This claim is straightforward: it follows directly from the fact that there are two edge-disjoint paths from r to any literal vertex $\tilde{x}_i \in \{\bar{x}_i, x_i, x'_i\}$ that do not use edge (y_i, z_i).

Claim 2. *For each two-literal clause c, edge (r, c) is not in T.*

For each literal $\tilde{x}_i$ of clause c, there is an r-to-c path via the x_i-gadget, so, together with edge (r, c), G has three disjoint r-to-c paths. Thus, if (r, c) were in T, its congestion would be at least $k_1 + 2 > K$, proving Claim 2.

Claim 3. *All clause vertices are leaves in T.*

To prove Claim 3, suppose there is a clause c that is not a leaf. Then, by Claim 2, c has at least two clause-to-literal edges in T, say $(c, \tilde{x}_i)$ and $(c, \tilde{x}_j)$. We can assume that the last edge on the r-to-c path in T is $e = (c, \tilde{x}_i)$. Clearly, $r \in T_{\tilde{x}_i, c}$ and $\tilde{x}_j \in T_{c, \tilde{x}_i}$. By Claim 1, at least two edges of the x_j-gadget are in $\partial_{G,T}(e)$, and they contribute at least 2 to $\text{cng}_{G,T}(e)$. We now have some cases to consider.

If c is a two-literal clause, its root-to-clause edge (r, c) is also in $\partial_{G,T}(e)$, by Claim 2. Thus, $\text{cng}_{G,T}(e) \geq k_2 + 3 > K$ (see Fig. 4a). So assume now that c is a three-literal clause, and let $\tilde{x}_l \neq \tilde{x}_i, \tilde{x}_j$ be the third literal of c. If T contains $(c, \tilde{x}_l)$, the x_l-gadget would also contribute at least 2 to $\text{cng}_{G,T}(e)$, so

$\mathrm{cng}_{G,T}(e) \geq k_2 + 4 > K$ (see Fig. 4b). Otherwise, $(c, \tilde{x}_l) \notin T$, and $(c, \tilde{x}_l)$ itself contributes 1 to $\mathrm{cng}_{G,T}(e)$, so $\mathrm{cng}_{G,T}(e) \geq k_2 + 3 > K$ (see Fig. 4c).

We have shown that if a clause vertex c is not a leaf in T, then in all cases the congestion of T would exceed K, completing the proof of Claim 3.

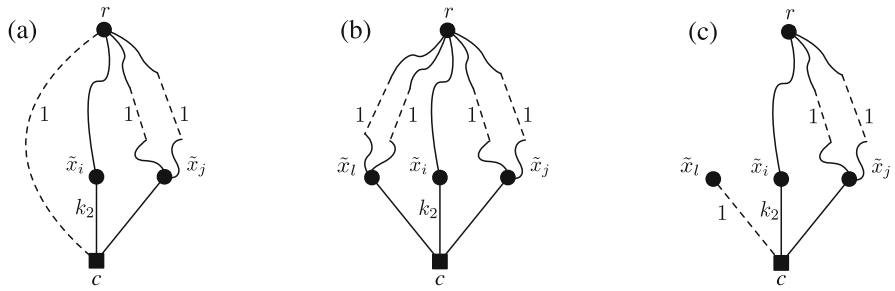

Fig. 4. Illustration of the proof of Claim 3. In (a) c is a two-literal clause; in (b) and (c), c is a three-literal clause.

Claim 4. *For each x_i-gadget, edge (r, x'_i) is in T.*

Towards contradiction, suppose that (r, x'_i) is not in T. Let (x'_i, c) be the clause-to-literal edge of x'_i. If only one of the two edges $(x'_i, x_i), (x'_i, c)$ is in T, making x'_i a leaf, then the congestion of that edge is $k_3 + k_2 + 1 > K$. Otherwise, both $(x'_i, x_i), (x'_i, c)$ are in T. Because c is a leaf in T by Claim 3, $e = (x_i, x'_i)$ is the last edge on the r-to-x'_i path in T. As shown in Fig. 5a, $\mathrm{cng}_{G,T}(e) \geq k_3 + k_2 + 2 > K$. This proves Claim 4.

Claim 5. *For each x_i-gadget, edge (r, y_i) is in T.*

To prove this claim, suppose (r, y_i) is not in T. We consider the congestion of the first edge e on the r-to-y_i path in T. By Claims 3 and 4, we have $e = (r, x'_i)$, all vertices of the x_i-gadget have to be in $T_{x'_i,r}$, and $T_{x'_i,r}$ does not contain literal vertices of another variable $x_j \neq x_i$. For each literal $\tilde{x}_i$ of x_i, if a clause-to-literal edge $(c, \tilde{x}_i)$ is in T, then the two other edges of c contribute 2 to $\mathrm{cng}_{G,T}(e)$, otherwise $(c, \tilde{x}_i)$ contributes 1 to $\mathrm{cng}_{G,T}(e)$. Then, $\mathrm{cng}_{G,T}(e) \geq k_4 + k_3 + 3 > K$ (see Fig. 5b), proving Claim 5.

Claim 6. *For each x_i-gadget, exactly one of edges (z_i, x_i) and (x_i, x'_i) is in T.*

By Claims 4 and 5, edges (r, y_i) and (r, x'_i) are in T. Since the clause neighbor c' of x_i is a leaf of T, by Claim 3, if none of (z_i, x_i), (x_i, x'_i) were in T, x_i would not be reachable from r in T. Thus, at least one of them is in T. Now, assume both (z_i, x_i) and (x_i, x'_i) are in T (see Fig. 6a). Then, edge (y_i, z_i) is not in T, as otherwise we would create a cycle. Let us consider the congestion of edge $e = (r, x'_i)$. Clearly, x_i and x'_i are in $T_{x'_i,r}$. The edges of the two clause neighbors

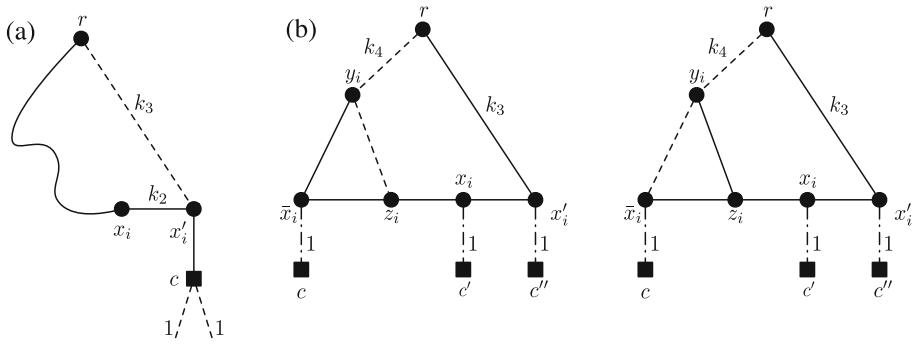

Fig. 5. (a) Illustration of the proof of Claim 4. (a) Illustration of the proof of Claim 5. Dot-dashed lines are edges that may or may not be in T.

c' and c'' of x_i and x'_i contribute at least 2 to $\text{cng}_{G,T}(e)$, by Claim 3. In addition, by Claim 1, besides e and (y_i, z_i), $\partial_{G,T}(e)$ contains another edge of the x_i-gadget which contributes at least another 1 to $\text{cng}_{G,T}(e)$. Thus, $\text{cng}_{G,T}(e) \geq k_4+k_3+3 > K$ — a contradiction. This proves Claim 6.

Claim 7. *For each x_i-gadget, edge (y_i, z_i) is in T.*

By Claims 4 and 5, the two edges (r, x'_i) and (r, y_i) are in T. Now assume, towards contradiction, that (y_i, z_i) is not in T (see Fig. 6b). By Claim 6, only one of (z_i, x_i) and (x_i, x'_i) is in T. Furthermore, the clause neighbor c' of x_i is a leaf of T, by Claim 3. As a result, (z_i, x_i) cannot be on the y_i-to-z_i path in T. To reach z_i from y_i, the two edges $(y_i, \bar{x}_i), (\bar{x}_i, z_i)$ have to be in T. Let us consider the congestion of $e = (y_i, \bar{x}_i)$. The edges of the clause neighbor c of $\bar{x}_i$ contribute at least 1 to the congestion of e, by Claim 3. Also, by Claim 1, besides e and (y_i, z_i), $\partial_{G,T}(e)$ contains another edge of the x_i-gadget which contributes at least 1 to $\text{cng}_{G,T}(e)$. In total, $\text{cng}_{G,T}(e) \geq k_4 + k_2 + 2 > K$, reaching a contradiction and completing the proof of Claim 7.

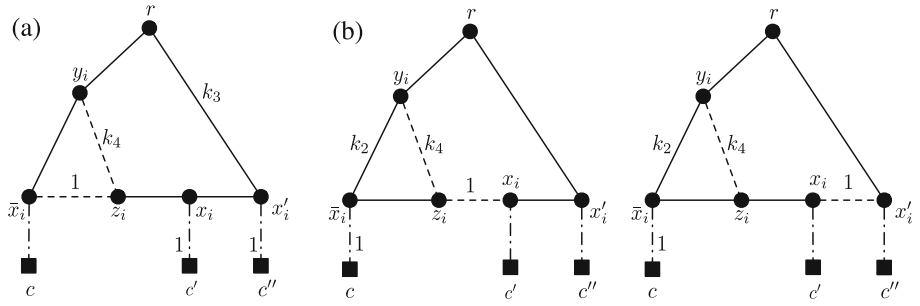

Fig. 6. (a) Illustration of the proof of Claim 6. (b) Illustration of the proof of Claim 7. Dot-dashed lines are edges that may or may not be in T.

Claim 8. *For each x_i-gadget, if its clause-to-literal edge $(\bar{x}_i, c)$ is in T, then its other two clause-to-literal edges (x_i, c') and (x'_i, c'') are not in T.*

Assume the clause-to-literal edge $(\bar{x}_i, c)$ of the x_i-gadget is in T. By Claim 7, edge (y_i, z_i) is in T. If $(y_i, \bar{x}_i)$ is also in T, edge $(\bar{x}_i, z_i)$ cannot be in T, and it contributes 1 to $\text{cng}_{G,T}(y_i, \bar{x}_i)$. As shown in Fig. 7a, $\text{cng}_{G,T}(y_i, \bar{x}_i) = k_2 + 3 > K$. Thus, $(y_i, \bar{x}_i)$ cannot be in T. Since c is a leaf of T, edge $(\bar{x}_i, z_i)$ has to be in T, for otherwise $\bar{x}_i$ would not be reachable from r. By Claim 6, one of edges (z_i, x_i) and (x_i, x'_i) is in T. If (z_i, x_i) is in T (see Fig. 7b), $\text{cng}_{G,T}(y_i, z_i) \geq k_4 + 5 > K$. Hence, (z_i, x_i) is not in T, which implies that (x_i, x'_i) is in T.

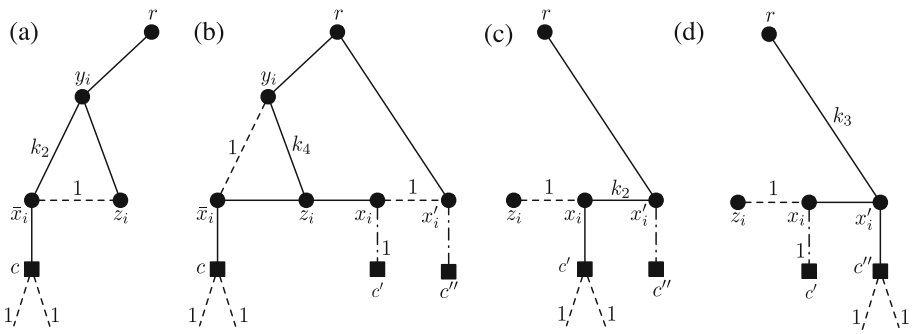

Fig. 7. Illustration of the proof of Claim 8. Dot-dashed lines are edges that may or may not be in T.

Now, we proceed by contradiction assuming that at least one other clause-to-literal edge of the x_i-gadget is in T. If edge (x_i, c') is in T, $\text{cng}_{G,T}(x_i, x'_i) \geq k_2 + 3 > K$, as shown in Fig. 7c. Similarly, if (x'_i, c'') is in T, $\text{cng}_{G,T}(r, x'_i) \geq k_3 + 4 > K$ (see Fig. 7d). So we reach a contradiction in both cases, thus proving Claim 8.

We are now ready to complete the proof of the ($\Leftarrow$) implication in the equivalence ($*$). We use our spanning tree T of congestion at most K to create a truth assignment for ϕ by setting $x_i = 0$ if the clause-to-literal edge of $\bar{x}_i$ is in T, otherwise $x_i = 1$. By Claim 8, this truth assignment is well-defined. Each clause has one clause-to-literal edge in T which ensures that all clauses are indeed satisfied.

References

1. Alvarruiz Bermejo, F., Martínez Alzamora, F., Vidal Maciá, A.M.: Improving the efficiency of the loop method for the simulation of water distribution networks. J. Water Resour. Plan. Manag. **141**(10), 1–10 (2015)
2. Benczúr, A.A., Karger, D.R.: Approximating $s - t$ minimum cuts in $\tilde{O}(n^2)$ time. In: Proceedings of the 28th Annual ACM Symposium on Theory of Computing, pp. 47–55 (1996)

3. Bhatt, S., Chung, F., Leighton, T., Rosenberg, A.: Optimal simulations of tree machines. In: Proceedings of the 27th Annual Symposium on Foundations of Computer Science, pp. 274–282 (1986)
4. Bodlaender, H., Fomin, F., Golovach, P., Otachi, Y., Leeuwen, E.: Parameterized complexity of the spanning tree congestion problem. Algorithmica **64**, 1–27 (2012)
5. Bodlaender, H.L., Kozawa, K., Matsushima, T., Otachi, Y.: Spanning tree congestion of k-outerplanar graphs. Discret. Math. **311**(12), 1040–1045 (2011)
6. Cai, L., Corneil, D.G.: Tree spanners. SIAM J. Discret. Math. **8**(3), 359–387 (1995)
7. Dahlhaus, E., Johnson, D.S., Papadimitriou, C.H., Seymour, P.D., Yannakakis, M.: The complexity of multiterminal cuts. SIAM J. Comput. **23**(4), 864–894 (1994)
8. Dragan, F.F., Fomin, F.V., Golovach, P.A.: Spanners in sparse graphs. J. Comput. Syst. Sci. **77**(6), 1108–1119 (2011)
9. Emek, Y., Peleg, D.: Approximating minimum max-stretch spanning trees on unweighted graphs. SIAM J. Comput. **38**(5), 1761–1781 (2009)
10. Fekete, S.P., Kremer, J.: Tree spanners in planar graphs. Discret. Appl. Math. **108**(1), 85–103 (2001)
11. Fung, W.S., Hariharan, R., Harvey, N.J., Panigrahi, D.: A general framework for graph sparsification. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, pp. 71–80 (2011)
12. Khuller, S., Raghavachari, B., Young, N.: Designing multi-commodity flow trees. Inf. Process. Lett. **50**(1), 49–55 (1994)
13. Kozawa, K., Otachi, Y.: Spanning tree congestion of rook's graphs. Discuss. Math. Graph Theory **31**(4), 753–761 (2011)
14. Kozawa, K., Otachi, Y., Yamazaki, K.: On spanning tree congestion of graphs. Discret. Math. **309**(13), 4215–4224 (2009)
15. Löwenstein, C.: In the Complement of a Dominating Set. Ph.D. thesis, Technische Universitat Ilmenau (2010)
16. Okamoto, Y., Otachi, Y., Uehara, R., Uno, T.: Hardness results and an exact exponential algorithm for the spanning tree congestion problem. In: Ogihara, M., Tarui, J. (eds.) TAMC 2011. LNCS, vol. 6648, pp. 452–462. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20877-5_44
17. Ostrovskii, M.: Minimal congestion trees. Discret. Math. **285**(1), 219–226 (2004)
18. Otachi, Y.: A survey on spanning tree congestion. In: Fomin, F.V., Kratsch, S., van Leeuwen, E.J. (eds.) Treewidth, Kernels, and Algorithms. LNCS, vol. 12160, pp. 165–172. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-42071-0_12
19. Otachi, Y., Bodlaender, H.L., van Leeuwen, E.J.: Complexity results for the spanning tree congestion problem. In: Thilikos, D.M. (ed.) WG 2010. LNCS, vol. 6410, pp. 3–14. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16926-7_3
20. Rosenberg, A.L.: Graph embeddings 1988: recent breakthroughs, new directions. In: Reif, J.H. (ed.) AWOC 1988. LNCS, vol. 319, pp. 160–169. Springer, New York (1988). https://doi.org/10.1007/BFb0040384
21. Spielman, D.A., Teng, S.H.: Spectral sparsification of graphs. SIAM J. Comput. **40**(4), 981–1025 (2011)
22. Yoshinaka, R.: Higher-order matching in the linear lambda calculus in the absence of constants is NP-complete. In: Giesl, J. (ed.) RTA 2005. LNCS, vol. 3467, pp. 235–249. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-32033-3_18