Beyond Metric Embedding: Approximating Group Steiner Trees on Bounded Treewidth Graphs

Parinya Chalermsook* Syamantak Das† Bundit Laekhanukit‡ Daniel Vaz§

November 7, 2016

Abstract

The *Group Steiner Tree* (GST) problem is a classical problem in combinatorial optimization and theoretical computer science. In the Edge-Weighted Group Steiner Tree (EW-GST) problem, we are given an undirected graph $G = (V, E)$ on n vertices with edge costs $c : E \to \mathbb{R}_{\geq 0}$, a source vertex s and a collection of subsets of vertices, called *groups*, $S_1, \ldots, S_k \subseteq V$. The goal is to find a minimum-cost tree $H \subseteq G$ that connects s to some vertex from each group S_i, for all $i = 1, 2, \ldots, k$. The Node-Weighted Group Steiner Tree (NW-GST) problem has the same setting, but the costs are associated with nodes. The goal is to find a minimum-cost node set $X \subseteq V$ such that $G[X]$ connects every group to the source.

When G is a tree, both EW-GST and NW-GST admit a polynomial-time $O(\log n \log k)$ approximation algorithm due to the seminal result of [Garg et al., SODA'98 and J. Algorithm]. The matching hardness of $\log^{2-\epsilon} n$ is known even for tree instances of EW-GST and NW-GST [Halperin and Krauthgamer STOC'03]. In general graphs, most of polynomial-time approximation algorithms for EW-GST reduce the problem to a tree instance using the metric-tree embedding, incurring a loss of $O(\log n)$ on the approximation factor [Bartal, FOCS'96; Fakcharoenphol et al., FOCS'03 and JCSS]. This yields an approximation ratio of $O(\log^2 n \log k)$ for EW-GST. Using metric-tree embedding, this factor cannot be improved: The loss of $\Omega(\log n)$ is necessary on some input graphs (e.g., grids and expanders). There are alternative approaches that avoid metric-tree embedding, e.g., the algorithm of [Chekuri and Pal, FOCS'05], which gives a tight approximation ratio, but none of which achieves polylogarithmic approximation in polynomial-time. This state of the art shows a clear lack of understanding of GST in general graphs beyond the metric-tree embedding technique. For NW-GST (for which the metric-tree embedding does not apply), not even a polynomial-time polylogarithmic approximation algorithm is known.

In this paper, we present $O(\log n \log k)$ approximation algorithms that run in time $n^{\tilde{O}(tw(G)^2)}$ for both NW-GST and EW-GST[1], where $tw(G)$ denotes the treewidth of graph G. The key to both results is a different type of "tree-embedding" that produces a tree of much bigger size, but *does not cause any loss on the approximation factor.* Our embedding is inspired by dynamic programming, a technique which is typically not applicable to Group Steiner problems.

1 Introduction

The *Group Steiner Tree* (GST) problem is a cornerstone problem in Combinatorial Optimization and Theoretical Computer Science that has received a lot of attention over the past two decades [15, 13, 11, 4, 21, 19, 20, 17, 10]. In this problem, we are given an undirected graph $G = (V, E)$ on n vertices and m edges with node or edge costs, a root vertex r and a collection of subset of vertices, called *groups*, $S_1, \ldots, S_k \subseteq V$. The goal is to find a minimum-cost tree that connects r to some vertex from each group S_i, for $i = 1, 2, \ldots, k$.

The *Edge-Weighted Group Steiner Tree* (EW-GST) problem, where costs are on edges, admits polynomial-time $O(\log n \log k)$-approximation algorithms on trees and $O(\log^2 n \log k)$ on general graphs due to the seminal result of Garg, Konjevod and Ravi [15] together with the Bartal's metric-tree embedding [2, 14] and admits quasi-polynomial-time $O(\log^2 k)$-approximation algorithm on general graphs by the result of Chekuri and Pal [11]. The algorithm of Garg et al. and that of Chekuri and Pal also works for the node-weighted case, i.e., the *Node-Weighted Group Steiner Tree* (NW-GST) problem. Thus, NW-GST admits polynomial-time $O(\log n \log k)$-approximation on trees and quasipolynomial-time $O(\log^2 k)$-approximation on general graphs. Nevertheless, due to the absence of metric-tree embedding that works for "node distances", there is a huge gap in the approximation ratios obtained by polynomial and quasi-polynomial time algorithms for NW-GST. The best known approximation ratio one could obtain in polynomial-time is $O(k^\epsilon)$, for any $\epsilon > 0$,

*Aalto University, Finland. Work partially done while at Max-Planck-Institut für Informatik, Germany, **email:** parinya.chalermsook@aalto.fi

†Universität Bremen, Germany. Work done while at Max-Planck-Institut für Informatik, Germany, **email:** syamanta@uni-bremen.de

‡The Weizmann Institute of Science, Israel, **email:** bundit.laekhanukit@weizmann.ac.il.

Partially supported by the ISF grant 621/12 and I-CORE grant No. 4/11.

§Max-Planck-Institut für Informatik, Germany and Graduate School of Computer Science, Saarland University, Germany, **email:** ramosvaz@mpi-inf.mpg.de

[1]The $\tilde{O}$ notation hides logarithmic factors, i.e., $\tilde{O}(x) = O(x \operatorname{polylog}(x))$

[7, 19, 11].

Hence, both EW-GST and NW-GST admit $O(\log^2 k)$-approximation algorithms in quasi-polynomial-time, which match the best-known approximation lower bounds of $\log^{2-\epsilon} k$, for any $\epsilon > 0$, by Halperin and Krauthgamer [18], which holds under the assumption NP $\not\subseteq$ DTIME($2^{\mathrm{polylog}(n)}$). However, the questions remain open for polynomial-time algorithms. Is there a polynomial-time algorithm that achieves $O(\log^2 k)$-approximation guarantee for EW-GST? Is it possible to get a polylogarithmic approximation ratio for NW-GST? These two problems have been long standing open problems in the area of Network Design.

Previous Techniques and Barriers. Over the decades, many techniques have been developed to approximate EW-GST and NW-GST. There are two types of approaches in approximating EW-GST. The first is the Recursive Greedy algorithm [7, 19, 11], which is a combinatorial algorithm. These algorithms do not require any kind of tree-embedding and eventually leads to a tight $O(\log^2 k)$-approximation algorithm for both EW-GST and NW-GST. Nevertheless, these algorithms could not accomplish polylogarithmic approximation in polynomial-time.

The second type is the Embedding-into-Tree technique. Garg et al. [15] apply the metric-tree embedding technique to reduce an instance of EW-GST on general graphs to a tree instance. Then they devise an LP-based algorithm for EW-GST on trees. This yields the best known polynomial-time $O(\log^2 n \log k)$-approximation algorithm for EW-GST. However, this technique is not applicable to NW-GST because there is no known metric-tree embedding for node-distances. Alternatively, there is a well-known reduction from GST on general graphs to a tree instance, which is given implicitly in the work of Zelikovsky [23]. This gives an $O(i^3 \cdot k^{1/i})$-approximation algorithm in time $O(n^i)$. This method is applicable to both EW-GST and NW-GST, but again, could not yield a polynomial-time polylogarithmic approximation algorithm (See, e.g., [20, 9]).

The barrier to obtain a better approximation algorithm for EW-GST is due to the stretch of the embedding and for NW-GST is due to the absence of node-distance metric-tree embedding.

Metric-Tree Embedding Barrier and Relation to Treewidth. Metric-tree embedding is a powerful tool in the design of approximation algorithms. It allows to transform any graph distance metric into a tree metric while approximately preserving the distance to within an $O(\log n)$ factor [14]. Thus, many difficult optimization problems on general graphs turn into amenable tree instances, which are easy to deal with, by paying a factor $O(\log n)$ in approximation ratio. This $O(\log n)$ loss here cannot be removed in general. Any metric-tree embedding of square-grid graphs and expanders incurs a factor of $\Omega(\log n)$ [2]. See [1] and references therein.

For GST, there is no known tool that could cope with general graphs beside the metric-tree embedding (unless we allow the algorithm to run in quasi-polynomial-time). A straightforward step in attacking EW-GST and NW-GST is clearly to develop an approximation algorithm for the case of square-grids and expanders. For the latter case, there is a known tool [3] available that yields a tight approximation factor for EW-GST (but again not for NW-GST). However, we still have no tools even for graphs that contain no large square-grids or square-grid minors.

At this point, readers who are familiar with Graph Minor Theorems may observe that the difficult instances of GST are indeed instances with "large treewidth". It is known as the *Grid Minor Theorem* [22] that any graph with treewidth w contains a square-grid minor of size $f(w)$, which is now known to be polynomial on w [8]. Keeping this observation in mind, a small step toward breaking the metric-tree embedding barrier is to develop an algorithm that works at least for graphs with small treewidth, which are more general than trees but still have no large square-grid minor.

1.1 Our Contributions.

The main purpose of this paper is to initiate the study of techniques that have a potential to get around the metric embedding barriers. Our technique still relies on (a slightly different kind of) "tree embedding", but is more problem-dependent, in that it works specifically for GST.

In particular, we define the following notion. Given a EW-GST instance $\mathcal{I} = (G, \{S_i\}_{i=1}^k, r)$, an EW-GST-*tree-sparsifier* [2] for $\mathcal{I}$ is a "generalized" GST instance [3] $\mathcal{J} = (T, \{S'_i\}_{i=1}^k, r')$ where T is a tree of height $O(\log n)$, $S'_i \subseteq V(T)$, and $r' \in V(T)$. Roughly, we say that $\mathcal{J}$ is an (α, β) GST-tree-sparsifier for $\mathcal{I}$ if $\mathrm{opt}(\mathcal{I}) \leq \mathrm{opt}(\mathcal{J}) \leq \alpha \mathrm{opt}(\mathcal{I})$ and $|V(T)| = O(\beta)$. The factor α and β are referred to as *distortion* and *size* respectively.

The following (simple) theorem follows almost directly.

Theorem 1.1. *If there is an efficient algorithm for constructing (α, β) EW-GST-tree-sparsifier, then there is an $O(\alpha \log n \log k)$ approximation algorithm for EW-GST that runs in time $\beta^{O(1)}$.*

We remark that the metric tree embedding result [14] gives $(O(\log n), \mathrm{poly}(n))$ EW-GST-tree-

[2] Analogously, one can define this for NW-GST.

[3] To be defined formally later. For now, the readers only need to know that the generalized GST problem admits $O(\log n \log k)$ approximation.

sparsifier trivially, and that in order to improve the long-standing ratio, it suffices to design a $(\log^{1-\epsilon} n, \mathrm{poly}(n))$ EW-GST-tree-sparsifier.

In this paper, we are interested in a tree sparsifier of potentially bigger size but smaller distortion. Our main result shows that this is indeed possible. We show a sparsifier with distortion 1, and with size depending on the treewidth of graph G. This result is summarized in the following theorem.

Theorem 1.2. *For any instance $\mathcal{I} = (G, \{S_i\}, r)$ of EW-GST, there is an efficient algorithm running in time $n^{O(tw(G))^2}$ that constructs a $(1, n^{O(tw(G))^2})$ EW-GST-tree-sparsifier for $\mathcal{I}$. The same holds for NW-GST.*

Corollary 1.1. *There are $O(\log n \log k)$ approximation algorithms for EW-GST and NW-GST, running in time $n^{O(tw(G)^2)}$. In particular, there exist polynomial-time $O(\log n \log k)$-approximation algorithms for EW-GST and NW-GST on bounded treewidth graphs.*

We remark that, for NW-GST, this is the first poly-logarithmic approximation algorithm that runs in polynomial time for a graph class more general than trees.

We believe that our new concept of GST-sparsifiers will open up some new directions to attack both EW-GST and NW-GST. The most interesting open question is whether there exists $(O(1), \mathrm{poly}(n))$ GST-sparsifier (which would settle the long-standing open problem.) We leave this as an open problem.

1.2 Overview of Our Techniques. Here we give an overview of our techniques and the intuition on how we construct the sparsifier.

To that end, it would be helpful to think of EW-GST on trees as a problem with local and global constraints: we have local constraints corresponding to the choice of edges, and global constraints that ensure the groups are covered. The local constraints would allow, in principle, a decomposition of the problem into subproblems, corresponding to distinct subtrees. On the other hand, the global constraints are not suitable to this kind of decomposition.

Consider an algorithm that does the following: for each child of the root, it decides whether the edge connecting it to the root is included in the solution; then, it recurses on the children of the root corresponding to the edges that were chosen. If the algorithm guesses the edges to add correctly, then it will solve the problem, but it is not easy to do this. The other possibility would be to enumerate all the possible subsets of edges incident to each node, which makes the algorithm find the correct solution, but the running time will be exponential. We can also look at the rounding algorithm proposed by Garg et al. [15] (from now on referred to as GKR rounding) in this perspective. By using the solution to the Linear Program, it guides the choice of edges at each node, in a way that is consistent (with some probability) with the global constraints.

Now let us get back to general graphs. The basic picture does not change much. Instead of considering one node at a time (every node in a tree is a separator), we consider a vertex cut S that splits the graph into multiple connected components. Now the algorithm must decide, first, which edges inside S to take, and then, which edges to take that connect S to each of the connected components of $G - S$. There is, however, one big difference between trees and general graphs: there are now multiple paths between two nodes. Therefore, it may happen that a path crosses S several times to use edges of different subproblems.

We solve this issue by using the concept of implied connections: for each subproblem, we additionally associate it with a set of connections Γ that its solution must implement. In other words, for each subproblem, there is a set Γ of pairs of nodes that it must connect in its solution.

Using these ideas, we can recursively decompose the problem into subproblems. However, it is still not clear how to choose, for each subproblem, both the set of edges that connect to it and the set of connections Γ that it must implement. The naive way would be to enumerate all such possibilities, which would run in time exponential in n. The other possibility, inspired by the problem on trees, is to use GKR rounding to guide the choices of the algorithm.

We can think of the subproblems as being organized in a tree, i.e., a problem is the parent of the subproblems it recurses into. If $tw(G) = w$, by standard facts, we are guaranteed to have a *balanced* vertex cut S of size $|S| = O(w)$ in every induced subgraph of G. Therefore, the process generates a tree $\widehat{T}$ of height $O(\log n)$ (by successively dividing each subgraph into two using a balanced vertex cut.)

Let P be a subproblem and Q and R be the subproblems obtained by removing the vertex cut S from P. As mentioned above, for every subproblem P, we shall also associate all possible connections that are potentially implemented by P. Let us denote this set of connections by $\Gamma \subseteq S \times S$ which is a set of vertex pairs $(u, v) \in S$. It is, however, important to realize that the actual path between a vertex pair $(u, v) \in \Gamma$ might be constructed as we recurse into the subproblems. In some sense, the inclusion of (u, v) in Γ gives a guarantee that either this path is already constructed by the edges that are bought in the parent subproblem or it would be constructed in a children or sibling subproblem. We encode all the above information by introducing a node in $\widehat{T}$, denoted by $x(P, \Gamma_p)$ where Γ_p is a connection set that P inherits from its parents. At P, corresponding to each possible choice of Γ, we now make one child of $x(P, \Gamma_p)$, denoted by $z(P, \Gamma_p, \Gamma)$. Note that there can be only $2^{|S|^2}$ possible choices of Γ and hence only those number of children for $x(P, \Gamma_p)$. The root of $\widehat{T}$ corresponds to a solution to the whole problem. Each $z(P, \Gamma_p, \Gamma)$ has two children $x(Q, \Gamma_p, \Gamma)$ and $x(R, \Gamma_p, \Gamma)$ corresponding to solving the subproblems at Q and R respectively and these solutions are allowed to use connections that are implied by $\Gamma_p \cup \Gamma$. When the algorithm solves the subproblem (Q, Γ_p, Γ), it goes through similar steps: find some vertex cut $S' \subseteq G[Q]$ and then enumerate over all the possible "connections" Γ' of pairs of vertices in S'.

This process then creates the children $\{z(Q, \Gamma_p, \Gamma, \Gamma')\}_{\Gamma' \subseteq S' \times S'}$ for subproblem (Q, Γ_p, Γ), and again each such $z(Q, \Gamma_p, \Gamma, \Gamma')$ has two children corresponding to vertices in the components of $G[Q - S']$.

A subtree H of $\widehat{T}$ is said to be *canonical* if each vertex of the form $x(V', \Gamma_1, \ldots, \Gamma_\ell)$ has exactly one child, and each vertex of the form $z(\Gamma_1, \ldots, \Gamma_\ell)$ has exactly two children. Roughly, the above conditions ensure that canonical subtree of $\widehat{T}$ corresponds to a solution to the original GST instance and vice versa. There are $2^{O(w^2)}$ potential choices of Γ at every subproblem, so the size of $\widehat{T}$ is $2^{\tilde{O}(w^2) \log n} = n^{\tilde{O}(w^2)}$. Finally, we show that the optimization problem of finding a canonical solution in $\widehat{T}$ can be done by GKR rounding, giving an approximation factor of $O(h(\widehat{T}) \log k)$.

The technical heart of the paper lies in constructing the sparsifier $\widehat{T}$ starting with a tree decomposition of the given graph G. We face several hurdles in the process and one of the more difficult ones is to handle dependency of subproblems on each other. To be more specific, consider a node $x(P)$, a connection set Γ and its descendants $x(Q, \Gamma)$ and $x(R, \Gamma)$. Now, when we solve the subproblem at $x(Q, \Gamma)$, some connection implied in $x(Q, \Gamma)$ might try to use some other connections implied in $x(P)$ or $x(R, \Gamma)$ and vice versa. This potentially creates circular dependencies leading to an infeasible solution if both the connections are selected. In order to avoid this scenario, we need to enforce a partial order on the connections and ensure that the partial orders are consistent between parent and children nodes in the sparsifier tree $\widehat{T}$. We show that the existence of a consistent partial order is guaranteed by the properties of tree decomposition.

1.3 Further related work. Group Steiner Trees on Special Graph Classes.

Special cases of group Steiner trees have also received attention. In particular, Demaine et al. [13] and Bateni et al. [4] studied the GST problem when each face can only contain one group, culminating in a recent PTAS for this special case.

Approximation Algorithms on Graphs of Bounded Treewidth. There has been several results on approximating graph problems for graphs that have bounded treewidth. In particular, there are several instances where restrictions to the class of graphs with low treewidth yields a better approximation factor in running time that is dependent on the treewidth. Bateni et al. [5] give approximation schemes for the classical Steiner Forest Problem on bounded treewidth graphs. Gupta et al. [16] show a constant factor approximation for the sparsest cut problem on graphs of constant treewidth that runs in time $n^{O(tw(G))}$, while Czumaj et al. [12] study the maximum independent set problem on graphs with low treewidth.

2 Preliminaries

Tree Decomposition. Let G be an input graph. A *tree decomposition* for G is given by a tree $\mathcal{T}$ and a collection of *bags* $\{X_t\}_{t \in V(\mathcal{T})}$, where $X_t \subseteq V(G)$, that satisfy the following properties:

- $V(G) = \bigcup_{t \in V(\mathcal{T})} X_t$
- For any edge $uv \in E(G)$, there is a bag X_t such that $u, v \in X_t$.
- For each vertex $v \in V(G)$, the nodes t for which X_t contains v form a connected subgraph of $\mathcal{T}$.

The treewidth of G, denoted by $tw(G)$, is the minimum integer k for which there exists a tree decomposition $(\mathcal{T}, \{X_t\}_{t \in V(\mathcal{T})})$, such that $\max |X_t| \leq k + 1$.

We will use the following result, which shows the existence of an $O(\log n)$-height binary tree decomposition of treewidth $O(tw(G))$.

THEOREM 2.1. ([6]) *Let G be any graph. There is a tree decomposition $(\mathcal{T}, \{X_t\}_{t \in V(\mathcal{T})})$ such that (i) The tree $\mathcal{T}$ has height at most $O(\log n)$ and degree at most three, and (ii) Each bag X_t satisfies $|X_t| \leq O(tw(G))$.*

Fix a tree decomposition $(\mathcal{T}, \{X_t\})$. For each node $t \in V(\mathcal{T})$, denote by $\mathcal{T}_t$ the subtree of $\mathcal{T}$ rooted at t. Also we can define an induced subgraph $G_t = G\left[\bigcup_{t' \in \mathcal{T}_t} X_{t'}\right]$.

For each $t \in V(\mathcal{T})$, we say that an edge $uv \in E(G)$ appears in the bag t if $u, v \in X_t$. We will assume w.l.o.g.

that each edge $uv \in E(G)$ appears *only* in the topmost bag, i.e., the topmost node t such that X_t contains both u and v. Denote by E_t the edges that appear in bag X_t.

3 Constructing the sparsifiers

First, we define the notion of *GST-tree-sparsifier* formally. Given an instance $\mathcal{I} = (G, \{S_i\}_{i=1}^k, r)$ of EW-GST, an (α, β) EW-GST-tree-sparsifier for $\mathcal{I}$ is a GST instance with degree constraints: $\mathcal{J} = (T, \{S'_i\}_{i=1}^k, r', \phi)$ where T is a tree of height $O(\log n)$, $S'_i \subseteq V(G)$, $r' \in V(T)$, and $\phi : V(T) \to \{1, \ldots, n\}$ is a degree constraint function, satisfying the following properties:

- (Completeness:) For any subgraph $H \subseteq G$, there is a canonical subgraph (defined below) $H' \subseteq T$ with $c(H') \leq \alpha c(H)$ and for any $i \in [k]$, r is connected to S_i in H if and only if r' is connected to S'_i in H'.
- (Soundness:) For any canonical subgraph $H' \subseteq T$, there is a subgraph $H \subseteq G$ with $c(H) \leq c(H')$ and for any $i \in [k]$, the root r is connected to S_i in H if and only if r' is connected to S'_i in H'.
- (Size:) $|V(T)| = \beta$.

Our construction presented above result in an instance of EW-GST with degree constraints, which we call *Degree-Constrained Group Steiner Tree* (DC-GST), which is an instance of EW-GST plus a degree constraint function $\phi : V(T) \to \{1, \ldots, n\}$. The goal of DC-GST is to find a subgraph $H' \subseteq T$ so that every vertex v appearing in H' must have degree "exactly" $\phi(v)$ (i.e., $\mathsf{deg}_{H'}(v) = \phi(v)$ for all $v \in V(H)$). From now on, we will call a subgraph H' of T that satisfies the degree constraints a *canonical subgraph*. Our task, thus, reduces to solving an instance of the DC-GST on a tree.

Now we proceed to describe the construction of our sparsifier. Let G be an input graph with $tw(G) = w$ and $(\mathcal{T}, \{X_t\}_{t \in V(\mathcal{T})})$ be a tree decomposition of G given by the following lemma. Also, let $S_1, \ldots, S_k \subseteq V(G)$ be the groups.

LEMMA 3.1. *Let G be an input graph with $tw(G) = w$. There is a tree decomposition $(\mathcal{T}, \{X_t\}_{t \in V(\mathcal{T})})$ with the following properties:*

1. *The height of $\mathcal{T}$ is at most $O(\log n)$*
2. *Each bag X_t satisfies $|X_t| \leq O(w)$*
3. *The root node r is in every bag*
4. *Every leaf bag has no edges, ($E_t = \emptyset$ for leaf $t \in \mathcal{T}$)*
5. *Every non-leaf has exactly 2 children*

Proof. Let $(\mathcal{T}, \{X_t\}_{t \in V(\mathcal{T})})$ be a tree decomposition of G given by Theorem 2.1. This satisfies properties 1 and 2. In order to satisfy Property 3, simply add the root node to every bag, which increases the sizes of the bags by 1. Property 4 can be satisfied by making a copy of the leaf node and adding it as its own child, which will cause the new leaf to have no edges.

Regarding Property 5, in the decomposition of Theorem 2.1, every node of $\mathcal{T}$ has degree at most 3. By picking the root node of the decomposition to be a node of degree at most 2, we make sure that every node has at most 2 children. Finally, for every node with only 1 child, we make a copy of the entire subtree rooted at the child, so that the node now has 2 children.

We remark that none of the proofs affects the previous properties, and so the proof is completed. ■

3.1 Configuration Gadgets. Let $t \in V(\mathcal{T})$. A *connection set* for t is a subset $\Gamma \subseteq X_t \times X_t$. Each element $(u, v) \in \Gamma$ is referred to as a Γ-connection or simply a connection when Γ is clear from context.

Let Σ be a connection set and $\preceq$ be a partial order on the elements of Σ. We use the notation $(a, b) \prec (u, v)$ to represent $(a, b) \preceq (u, v)$ and $(a, b) \neq (u, v)$. For $(u, v) \in \Sigma$, we write $\Sigma^{\prec(u,v)}$ ($\Sigma^{\preceq(u,v)}$ resp.) to denote the set of connections $(a, b) \in \Sigma$ such that $(a, b) \prec (u, v)$ ($(a, b) \preceq (u, v)$ resp.), that is, the connections in Σ that are ranked below (u, v) by the partial order $\preceq$. Two partial orderings $\preceq$ and $\preceq'$ defined respectively on the sets Σ and Σ' are *consistent* if and only if for any pair of connections $\{(u, v), (u'v')\} \in \Sigma \cap \Sigma'$, $(u, v) \preceq (u', v')$ if and only if $(u, v) \preceq' (u', v')$.

DEFINITION 1. *Given two connection sets $\Gamma, \Sigma \subseteq X_t \times X_t$, we say that a connection $(u, v) \in \Gamma$ is* implied by *Σ in X_t if there is a sequence $u = w_0, w_1, \ldots, w_\alpha = v$ such that $(w_\beta, w_{\beta+1}) \in \Sigma$ for all $\beta = 0, \ldots, \alpha - 1$.*

The key idea is that we will have a node corresponding to the subproblem of connecting some pairs of nodes in the subtree of $\mathcal{T}$ rooted at bag t (denoted by $\mathcal{T}_t$). We will maintain three sets of connections inside the bag X_t:

- Connection set $\Gamma \subseteq X_t \times X_t$ represents the pairs (u, v) that shall be connected in the subtree $\mathcal{T}_t$, that is, in the subtree, we buy some edges to make sure that u and v are connected.
- Connection set $\Pi \subseteq X_t \times X_t$ represents the pairs (u, v) that are implied by *parent*(t).
- Connection set $\Sigma \subseteq X_t \times X_t$ contains the pairs (u, v) that shall be connected by the "sibling" subproblem $\mathcal{T}_{t'}$ where t' is the sibling of t.

We intuitively think of the subproblem $\phi = (t, \Gamma, \Pi, \Sigma, \preceq)$ as follows: Find the minimum-cost subgraph that connects all pairs $(u, v) \in \Gamma$, subject to the fact that all pairs in $\Pi \cup \Sigma^{\prec(u,v)}$ have already been connected.

Definition 2. *Let $\Gamma_1, \Gamma_2 \subseteq X_t \times X_t$, and $\preceq$ be a partial order on Γ. We say that a triple $(\Gamma_1, \Gamma_2, \preceq)$ is a* feasible division of Γ via Σ *if every connection $(u, v) \in \Gamma$ is implied by $\Gamma_1^{\preceq(u,v)} \cup \Gamma_2^{\preceq(u,v)} \cup \Sigma$.*

Consider any node $t \in V(\mathcal{T})$, with children t' and t'' in $\mathcal{T}$, any subset $\Gamma, \Pi, \Sigma \subseteq X_t \times X_t$. The *configuration gadget* $H(t, \Gamma, \Pi, \Sigma, \preceq)$ is a tree constructed in the following manner:

- The root of $H(t, \Gamma, \Pi, \Sigma, \preceq)$ is denoted by $r(t, \Gamma, \Pi, \Sigma, \preceq)$.
- For each $Y \subseteq E_t$, we create a vertex $p(t, \Gamma, \Pi, \Sigma, \preceq, Y)$, which is a child of $r(t, \Gamma, \Pi, \Sigma, \preceq)$; the cost of such connecting edge is $c(Y)$. Denote by Γ_Y the set of Γ-connections (u, v) that are not implied by $\Pi \cup \Sigma^{\prec(u,v)} \cup Y$. Also, denote by Π_Y the set of pairs $(u, v) \in X_t \times X_t$ that are implied by $\Pi \cup \Sigma^{\prec(u,v)} \cup Y$.
- Now consider a vertex $p(t, \Gamma, \Pi, \Sigma, \preceq, Y)$. For each feasible division $\rho = (\Gamma_1, \Gamma_2, \preceq')$ of Γ via Π_Y such that $\Gamma_1 \subseteq (X_{t'} \cap X_t)^2$ and $\Gamma_2 \subseteq (X_{t''} \cap X_t)^2$, and partial order $\preceq'$ is consistent with $\preceq$ on $\Gamma_1 \cup \Gamma_2$, we create the following vertices: (i) a vertex $q(t, \Gamma, \Pi, \Sigma, Y, \preceq, \rho)$ which is a child of $p(t, \Gamma, \Pi, \Sigma, \preceq, Y)$ and two vertices $\{q_i(t, \Gamma, \Pi, \Sigma, \preceq, Y, \rho)\}_{i=1,2}$, which are the two children of $q(t, \Gamma, \Pi, \Sigma, \preceq, Y, \rho)$. The costs of all these edges are zero.

We remove all vertices of the form $p(t, \Gamma, \Pi, \Sigma, \preceq, Y)$ that do not have children. If the root does not have any child after such removal, we declare the gadget $H(t, \Gamma, \Pi, \Sigma, \preceq)$ *unusable.* Otherwise, the gadget is usable. The leaf gadgets $H(t, \Gamma, \Pi, \Sigma, \preceq)$, i.e., when t is a leaf of $\mathcal{T}$, only contain the root node $r(t, \Gamma, \Pi, \Sigma, \preceq)$, and must satisfy $\Gamma = \emptyset$ (otherwise they are removed).

Interpretation of our gadget: We will briefly explain the meaning of our gadget. Consider a set of edges Y. After we purchase the edges in Y, there are some connections that could not be realized by these edges. The pairs in the set Γ_Y are the connections that (possibly) remain unconnected, and we wish to connect them in descendant gadgets. The pairs in the set Π_Y are the connections that need to be realized, but we leave the task of connecting them to other subproblems on parent or sibling gadgets. The set Γ_Y and Π_Y provide information on the connections to children gadgets.

3.2 Gluing the gadgets.

We will be using the structure of $\mathcal{T}$ to connect the gadgets created in the last section to form a final tree $\widehat{\mathcal{T}}$. The gluing process starts by processing nodes in the tree $\mathcal{T}$, ordered by their distances to the root (the root is processed first). When processing the root of $\mathcal{T}$, we create a root $root(\widehat{\mathcal{T}})$ and connect it to all gadgets $H(root(\mathcal{T}), \Gamma, \Pi, \Sigma, \preceq)$ where $\Pi = \Sigma = \emptyset$.

Now, consider any node $t \in V(\mathcal{T})$ and its children $t', t'' \in V(\mathcal{T})$. We define how their gadgets are connected. Focus on gadgets $H(t, \Gamma, \Pi, \Sigma, \preceq)$ that are usable, i.e., there is $Y \subseteq E_t$ and there are $\rho = (\Gamma_1, \Gamma_2, \preceq^*)$ that form a feasible division of Γ via Π_Y. We say that a vertex $q_1(t, \Gamma, \Pi, \Sigma, \preceq, Y, \rho)$ is consistent with $r(t', \Gamma', \Pi', \Sigma', \preceq')$ if the following conditions are met:

- $\Pi' \subseteq \{(u, v) \in X_{t'}^2 : (u, v)$ is implied by $\Pi_Y\}$.
- $\Gamma_1 \subseteq \Gamma' \cup \Pi'$
- $\Sigma' \subseteq \Gamma_2$
- $\preceq'$ is consistent with $\preceq^*$

The conditions for $q_2(t, \Gamma, \Pi, \Sigma, \preceq, Y, \rho)$ being consistent with $r(t'', \Gamma'', \Pi'', \Sigma'', \preceq'')$ are analogous. We connect the consistent nodes together by making the vertex $r(t', \Gamma', \Pi', \Sigma', \preceq')$ a child of either q_1 or q_2. In case there is more than one vertex consistent with $q_1(t, \Gamma, \Sigma, Y, \Gamma_1, \Gamma_2, \preceq)$, we make the same number of copies of the gadget $H(t', \Gamma', \Pi', \Sigma', \preceq')$ so that each gadget's root is only connected to one such consistent vertex above it.

The leaf gadgets will not have any children. All these connecting edges have cost zero.

Size of the instance: The following proposition shows that the size of $\widehat{\mathcal{T}}$ is $2^{O(w^2 \log w \log n)}$

Proposition 3.1. *$\widehat{\mathcal{T}}$ uses at most $2^{O(w^2 \log w \log n)}$ gadgets, each of them containing at most $2^{O(w^2 \log w)}$ nodes. As a consequence, the size of $\widehat{\mathcal{T}}$ is $2^{O(w^2 \log w \log n)}$.*

Proof. Let us start by proving that each gadget $H(t, \Gamma, \Pi, \Sigma, \preceq)$ has at most $2^{O(w^2 \log w)}$ nodes.

Since $Y \subseteq X_t^2$, there are at most 2^{w^2} choices for Y. Therefore, the root node of the gadget has at most $2^{O(w^2)}$ children. Each of these nodes p_Y has a child for each feasible division of Γ via Π_Y.

A feasible division is a triple $(\Gamma_1, \Gamma_2, \preceq)$. The number of possibilities for Γ_1, Γ_2 is at most 2^{w^2}, as they are subsets of X_t^2, which has size at most w^2. We can see $\preceq$ as a subset of a total ordering of X_t^2, and there are at most $(w^2)! \leq 2^{2w^2 \log w}$ such total orderings. Since

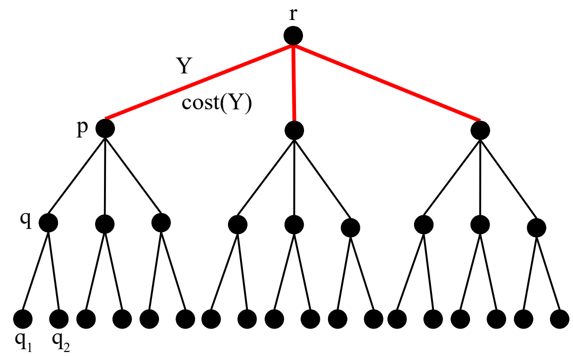

Figure 1: Structure of a gadget. Red edges have positive cost and represent subsets of edges in G

each total ordering has at most 2^{w^2} subsets, we have at most $2^{O(w^2 \log w)}$ possibilities for a partial order and, therefore, we have in total at most $2^{O(w^2 \log w)}$ choices for a feasible division. This, together with the two children of each node q_ρ makes for at most $2^{O(w^2 \log w)}$ nodes inside the gadget.

The proof follows by showing that each gadget has at most $2^{O(w^2 \log w)}$ children. $root(\widehat{\mathcal{T}})$ has at most $2^{O(w^2)}$ children, since $\Gamma \subseteq X_r^2$, and $X_r = O(w)$. Therefore, the number of subsets Γ is bounded by $2^{O(w^2)}$. Now, each gadget also has at most $2^{O(w^2 \log w)}$ children, since there are at most $2^{O(w^2)}$ ways to choose Γ', Π', Σ' and at most $2^{O(w^2 \log w)}$ ways to choose $\preceq'$ for the child t' of t. This also implies that, if $\widehat{\mathcal{T}}$ has height $O(\log n)$, then the number of gadgets is in total $2^{O(w^2 \log w \log n)}$. ■

Groups: For each leaf node $t \in V(\mathcal{T})$, we add the node $r(t, \Gamma, \Pi, \Sigma, \preceq)$ into the group $\widehat{\mathcal{S}}_i$ if and only if there is an ancestor $r(t', \Gamma', \Pi', \Sigma', \preceq')$ such that vertex $u \in S_i$ and $(s, u) \in \Pi' \cup \Gamma'^{\preceq(s,u)} \cup \Sigma'^{\prec(s,u)}$.

Feasible Solution for $\widehat{\mathcal{T}}$: We shall define a GST instance on the configuration tree $\widehat{\mathcal{T}}$. However, we also enforce degree bounds on certain nodes in a feasible solution. This is required to ensure that all feasible GST solutions to the original problem are embedded in the configuration tree. On the other hand, given a solution to the new instance, we can construct a feasible solution to the original problem with same cost.

A subtree $\widehat{\mathcal{Q}}$ is a feasible solution to DC-GST on $\widehat{\mathcal{T}}$ if the following hold:

- The root gadget is in $\widehat{\mathcal{Q}}$.
- Every group $\widehat{\mathcal{S}}_i$ is reachable from the root gadget.
- For a non-leaf gadget, a node of type $q(t, \Gamma, \Pi, \Sigma, \preceq, Y, \rho)$ has 2 children (degree 3).
- All other non-leaf node have exactly 1 child ($root(\widehat{\mathcal{T}})$ has degree 1, all others have degree 2).

In Section 4, we show how to modify the GKR rounding algorithm in order to accommodate these additional constraints.

Now we proceed to prove the correctness of our construction. Specifically, we show that given any feasible GST solution to the original problem, there exists a feasible solution $\widehat{\mathcal{Q}}$ to DC-GST on $\widehat{\mathcal{T}}$ of the same cost - we call this the *completeness* property of our construction. On the other hand, given a feasible solution to DC-GST on $\widehat{\mathcal{T}}$, we demonstrate a polynomial time construction of a solution to GST of the original instance with the same cost - we refer to this as the *soundness* property.

3.3 Completeness.

We devote this section to prove the following lemma, which shows the existence of a feasible solution to DC-GST on $\widehat{\mathcal{T}}$ that has the same cost as an optimal solution to the original GST instance.

LEMMA 3.2. *Any feasible solution $E' \subseteq E(G)$ for the original GST problem can be turned into a feasible solution $F \subset E(\widehat{\mathcal{T}})$ for the new problem on $\widehat{\mathcal{T}}$ such that $c(F) = c(E')$.*

To prove the lemma, we will show that we can choose, for each node $t \in \mathcal{T}$, a configuration gadget $H(t)$ for t such that all groups are covered. We also argue that the total cost incurred is at most $c(E')$.

Let us first introduce a structural lemma that we need for the proofs in this section.

LEMMA 3.3. (MONOCHROMATIC LEMMA) *Let G be any graph and $\hat{T}$ be a tree decomposition of G. Consider any bag $\hat{v} \in V(\hat{T})$ and a pair of vertices $x, y \in X_{\hat{v}}$. Suppose G has an x,y-path P of length at least 2 such that no vertex $z \in V(P) - \{x, y\}$ is in the bag $\hat{v}$. Then*

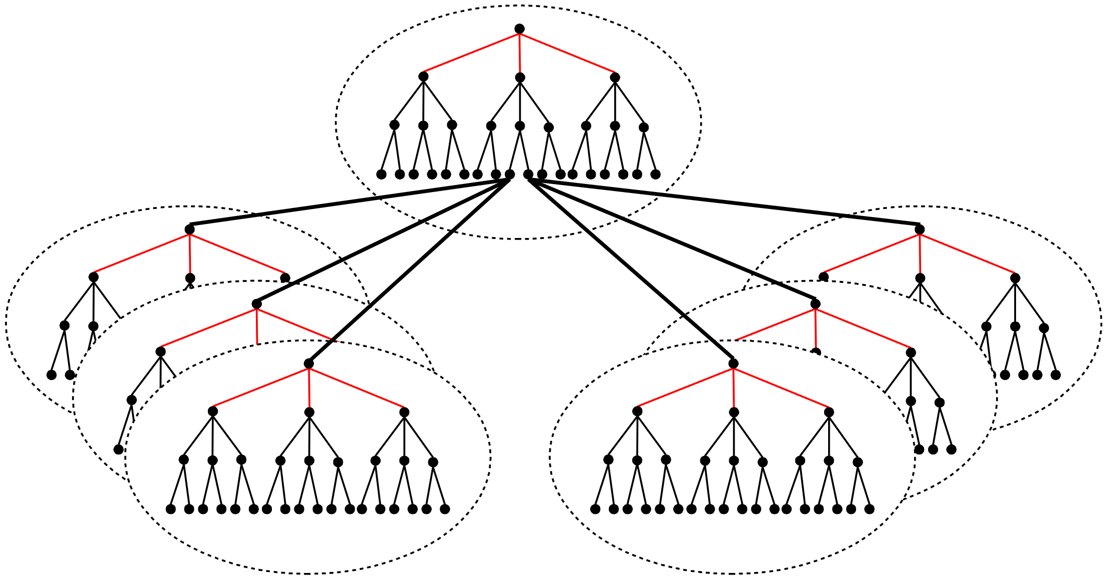

Figure 2: Gluing the gadgets: ovals represent individual gadgets, connected by bold lines. The leftmost three gadgets and the rightmost three gadgets correspond to the left and right children of the node in $\mathcal{T}$ associated with the center gadget.

there is a tree $\hat{T}'$ in $\hat{T} - \hat{v}$ such that, for any edge $ab \in E(P)$, $\hat{T}'$ has a bag $\hat{u}$ that contains both a and b. That is, every edge $ab \in E(P)$ appears in $\hat{T}'$.

Proof. We will prove a stronger statement: for any subpath (a, z, b) of P, any (maximal) connected component $\hat{T}'$ in $\hat{T} - \hat{v}$ that has a bag $\hat{u}$ containing both a and z must have a bag containing both z and b. (Note that P has such subpath (a, z, b) because it has length at least 2.)

We prove this statement by contradiction. Assume that $\hat{T}'$ is a component that contradicts the claim. Then $\hat{T}'$ has a bag $\hat{u}$ containing both a and z but has no bag containing both z and b. By the definition of tree decomposition, there must be a bag $\hat{w}$ that contains both z and b, and $\hat{w} \neq \hat{v}$ because $z \notin \mathrm{Bag}(\hat{v})$ by our assumption on P. So, there exists a (maximal) connected component $\hat{T}''$ in $\hat{T} - \hat{v}$ distinct from $\hat{T}'$ that contains $\hat{w}$. Observe that both bags $\hat{u}$ and $\hat{w}$ contain z. By the property of tree decomposition, the set of all bags containing z induces a connected component in $\hat{T}$, but this is not possible because $\hat{T}'$ and $\hat{T}''$ are distinct components in $\hat{T} - \hat{v}$ and $z \notin X_{\hat{v}}$. Thus, we have a contradiction.

Therefore, any connected component $\hat{T}'$ that has a bag $\hat{u}$ containing both a and z, for some edge $az \in E(P)$, must have a bag containing p and q for any edge $pq \in E(P)$, and the lemma follows. ∎

Choosing the gadgets:

We start by defining a partial ordering $\preceq^{E'}$ over $V(G)^2$ (more specifically, over E'). This way, we can easily define the required partial orderings on the gadgets as restrictions of $\preceq^{E'}$ to the relevant subsets.

Let $P_{uv} \subseteq E'$ be the simple path between u and v in E'. We define $(u_1, v_1) \preceq^{E'} (u_2, v_2)$ if $P_{u_1 v_1} \subseteq P_{u_2 v_2}$.

Let us now fix some $t \in \mathcal{T}$ and let $t' \in \mathcal{T}$ be the sibling of t. If t has no sibling (i. e. t is the root), then $\Pi = \Sigma = \emptyset$. Otherwise, consider the partition of $\mathcal{T}$ into $\mathcal{T}_t$, $\mathcal{T}_{t'}$ and $\mathcal{T} - (\mathcal{T}_t \cup \mathcal{T}_{t'})$. We will define Γ, Σ, and Π according to the connectivity in each of these partitions. More specifically:

$$\Pi = \{(u, v) \in X_t^2 : u, v \text{ are connected in } E' \cap G[\mathcal{T} - (\mathcal{T}_t \cup \mathcal{T}_{t'})]\}$$

$$\Gamma = \{(u, v) \in X_t^2 : u, v \text{ are connected in } E' \cap G[\mathcal{T}_t]\} - \Pi$$

$$\Sigma = \{(u, v) \in X_t^2 : u, v \text{ are connected in } E' \cap G[\mathcal{T}_{t'}]\} - \Pi$$

These sets along with the restriction $\preceq$ of $\preceq^{E'}$ to $\Gamma \cup \Sigma$, specify the gadget $H(t) = H(t, \Gamma, \Pi, \Sigma, \preceq)$ that we choose. We now specify the edges in the gadget that belong to our solution F.

First, let $Y = E_t \cap E'$. We add to F the edge that connects $r(t, \Gamma, \Pi, \Sigma, \preceq)$ to $p := p(t, \Gamma, \Pi, \Sigma, \preceq, Y)$. Such a node must exist since $Y \subseteq E_t$.

Next, we define $\rho = (\Gamma_1, \Gamma_2, \preceq')$, and prove it is a feasible division of Γ via Π_Y. This implies that the node $q = q(t, \Gamma, \Pi, \Sigma, \preceq, Y, \rho)$ exists, and thus we add to F edge (p, q), as well as the edges connecting q to its children. As before, we define $\preceq'$ as a restriction of $\preceq^{E'}$ to $\Gamma_1 \cup \Gamma_2$.

Let t_1, t_2 be the children of t in $\mathcal{T}$. We define Γ_1, Γ_2 as follows:

$$\Gamma_i = \{(u, v) \in (X_t \cap X_{t_i})^2 : u, v \text{ are connected in } E' \cap G[\mathcal{T}_{t_i}]\} - \Pi$$

The following claim implies that ρ is a feasible division of Γ via Π_Y.

Claim 1. *For any $(u, v) \in \Gamma$, (u, v) is implied by $\Gamma_1^{\preceq(u,v)} \cup \Gamma_2^{\preceq(u,v)} \cup \Pi_Y$*

Proof. Let P be the path between u and v in E'. Since $(u, v) \in \Gamma$, then this path must exist in $G[\mathcal{T}_t]$. Let $u = w_1, w_2, \ldots, w_k = v$ be all the vertices in $P \cap X_t$, in the order that they appear in P. Now, for each pair (w_i, w_{i+1}) there are two possibilities:

- (w_i, w_{i+1}) is an edge in $G[X_t]$. In this case, then either $(w_i, w_{i+1}) \in E_t$, which implies that $(w_i, w_{i+1}) \in Y$, or $(w_i, w_{i+1}) \notin E_t$, in which case the edge must be in some ancestor bag of t, which implies that $(w_i, w_{i+1}) \in \Pi$.

- (w_i, w_{i+1}) represents a path of length at least 2 in the graph $G[\mathcal{T}_t]$. In this case, since there is no vertex $z \in X_t - \{w_i, w_{i+1}\}$ in the path, then by Lemma 3.3, the path between w_i and w_{i+1} in E' is fully contained in either $G[\mathcal{T}_{t_1}]$ or $G[\mathcal{T}_{t_2}]$. In any case, $(w_i, w_{i+1}) \in \Gamma_1 \cup \Pi$ or $(w_i, w_{i+1}) \in \Gamma_2 \cup Pi$.

We conclude that, since $(w_i, w_{i+1}) \preceq (u, v)$ for all $i \in [k-1]$, then (u, v) is implied by $\Gamma_1^{\preceq(u,v)} \cup \Gamma_2^{\preceq(u,v)} \cup \Pi_Y$ (through path P). ■

Let Y_t be the chosen Y for gadget $H(t)$. Notice that, since the only edges in $\widehat{\mathcal{T}}$ with positive cost are those connecting p to the root of the gadget, the total cost of this solution is:

$$\begin{aligned} \sum_{t \in V(\mathcal{T})} c(Y_t) &= \sum_{t \in V(\mathcal{T})} c(E_t \cap E') \\ &= \sum_{e \in E'} c(e) = c(E') \end{aligned}$$

The second equality comes from the fact that each edge is in exactly one bag $t \in \mathcal{T}$. We conclude that $c(F) = c(E')$.

Connecting the gadgets:

Let $t, t', t'' \in V(\mathcal{T})$ such that t', t'' are the children of t. We show that the edges connecting $H(t)$ to $H(t')$ and $H(t'')$ exist, and add them to F. Let $H(t) = H(t, \Gamma, \Pi, \Sigma, \preceq)$, $H(t') = H(t', \Gamma', \Pi', \Sigma', \preceq'$) and $H(t'') = H(t'', \Gamma'', \Pi'', \Sigma'', \preceq'')$ be the chosen gadgets for t, t', and t''.

We show that, for the choices presented for Y and $\rho = (\Gamma_1, \Gamma_2, \preceq)$ in $H(t)$, we satisfy all the properties for the connection of the gadgets, and therefore the desired edges exist. We now prove the properties for t'. The properties for t'' are proved analogously.

It is clear that $\preceq'$ is consistent with $\preceq^*$, as both these partial orders are restrictions of the same partial order $\preceq^{E'}$.

Let us recall the definitions of Σ' and Γ_2.

$$\Sigma' = \{(u, v) \in X_{t'}^2 : u, v \text{ are connected in } E' \cap G[\mathcal{T}_{t''}]\} - \Pi'$$

$$\Gamma_2 = \{(u, v) \in (X_t \cap X_{t''})^2 : u, v \text{ are connected in } E' \cap G[\mathcal{T}_{t''}]\} - \Pi$$

From these definitions, we can see that $\Sigma' \subseteq \Gamma_2$, since $\Pi \cap X_{t'}^2 \subseteq \Pi'$, and for every $(u, v) \in \Sigma'$,

$$u, v \in X_{t'} \cap G[\mathcal{T}_{t''}] = X_{t'} \cap X_{t''} \subseteq X_t \cap X_{t''}$$

Similarly, we can deduce that $\Gamma_1 \subseteq \Gamma' \cup \Pi'$ from the definitions of Γ_1 and Γ'.

Finally, we need to prove that

$$\Pi' \subseteq \{(u, v) \in X_{t'}^2 : (u, v) \text{ is implied by } \Pi_Y\}$$

Recall that $\Pi_Y = \{(u, v) : (u, v) \text{ is implied by } \Pi \cup Y \cup \Sigma^{\prec(u,v)}\}$. The following claim implies the property.

Claim 2. *If $(u, v) \in \Pi'$, then (u, v) is implied by $\Pi \cup Y \cup \Sigma^{\prec(u,v)}$.*

Proof. Let P be the path between u and v in E', $\hat{t}$ be the sibling of t, and $p(t)$ the parent node of t. Since $(u, v) \in \Pi'$, then P must be contained in $G' := G[\mathcal{T} - (\mathcal{T}_{t'} \cup \mathcal{T}_{t''})]$. Let $w_1, \ldots, w_k$ be all the vertices in $P \cap X_{p(t)}$, in the order that they appear in P.

If $P \cap X_{p(t)} = \emptyset$, then u and v must be connected in $E_t - E_{p(t)}$, by the properties of the tree decomposition. Then, $P \subseteq Y$. Otherwise, the edges of P connecting u to w_1 and w_k to v are contained in $E_t - E_{p(t)}$, and therefore are contained in Y.

Now, for each pair (w_i, w_{i+1}) there are two possibilities:

- (w_i, w_{i+1}) is an edge in $G[X_{p(t)}]$. In this case, then (w_i, w_{i+1}) is either in $E_{p(t)}$ or in the bag of some ancestor of $p(t)$. In any case, this implies that $(w_i, w_{i+1}) \in \Pi$.

- (w_i, w_{i+1}) represents a path of length at least 2 in the graph G'. In this case, since there is no vertex $z \in X_{p(t)} - \{w_i, w_{i+1}\}$ in the path, then by Lemma 3.3, the path between w_i and w_{i+1} in E' is fully contained in either $G[\mathcal{T} - \mathcal{T}_{p(t)}]$, $G[\mathcal{T}_{\hat{t}}]$, or $G[X_t]$.

  In the first and second cases, this implies (w_i, w_{i+1}) is in Π or $\Sigma \cup \Pi$, respectively. If $(w_i, w_{i+1}) \in G[X_t]$, then either it is in E_t or in the bag of some ancestor, which implies it is either contained in Y or in Π.

In any of the previous cases, (w_i, w_{i+1}) is contained in either Y, Π, or Σ. We conclude that, since $(w_i, w_{i+1}) \prec (u, v)$ for all $i \in [k-1]$, as well as $(u, w_1) \prec (u, v)$ and $(w_k, v) \prec (u, v)$, then (u, v) is implied by $\Pi \cup Y \cup \Sigma^{\prec(u,v)}$ (through path P). ■

Covering the groups:

We will now show that the solution F covers all the groups of the instance. We first observe that each leaf of F corresponds to the gadget chosen for a leaf t of $\mathcal{T}$, with $\Gamma = \emptyset$.

Now, let us consider, for any group S_i, the vertex $u \in S_i$ connected by E' to s. Take any bag t such that $u \in X_t$ and $u \notin X_{p(t)}$ (or t is the root). Now, consider the path P between s and u in E'. Let t' be the sibling of t, and $p(t)$ the parent node of t. Let $s = w_1, \ldots, w_k$ be all the vertices in $P \cap X_{p(t)}$, in the order that they appear in P.

Notice that, by the properties of the tree decomposition, we get that $(w_k, u) \in \Gamma$, because the edges of P connecting w_k to u are contained in

$$G\left[\left(\bigcup_{\hat{t} \in \mathcal{T}_t} X_{\hat{t}} - X_{p(t)}\right) \cup w_k\right]$$

Now, for each pair (w_i, w_{i+1}) there are two possibilities:

- (w_i, w_{i+1}) is an edge in $G[X_{p(t)}]$. In this case, then $(w_i, w_{i+1}) \in \Pi$.
- (w_i, w_{i+1}) represents a path of length at least 2 in the graph G'. In this case, since there is no vertex $z \in X_{p(t)} - \{w_i, w_{i+1}\}$ in the path, then by Lemma 3.3, the path between w_i and w_{i+1} in E' is fully contained in either $G[\mathcal{T} - \mathcal{T}_{p(t)}]$, $G[\mathcal{T}_{t'}]$, or $G[X_t]$. This implies that (w_i, w_{i+1}) is in Π, $\Sigma \cup \Pi$, or $\Gamma \cup \Pi$, respectively.

In any of the previous cases, (w_i, w_{i+1}) is contained in either Π, Γ or Σ. We conclude that, since $(w_i, w_{i+1}) \prec (s, u)$ for all $i \in [k-1]$ and $(w_k, u) \preceq (s, u)$, then (u, v) is implied by $\Pi \cup \Gamma^{\preceq(s,u)} \cup \Sigma^{\prec(s,u)}$ (through path P).

3.4 Soundness.

We argue that, given any feasible solution $\widehat{\mathcal{Q}}$ to DC-GST on $\widehat{\mathcal{T}}$, we can construct a solution to the GST problem on graph G with groups $\mathcal{S}_i, i = 1, 2, \ldots, k$ and source node s.

We say that a gadget $H(t, \Gamma, \Pi, \Sigma, \preceq)$ is *active* if $r(t, \Gamma, \Pi, \Sigma, \preceq)$ is connected to the root gadget in $\widehat{\mathcal{Q}}$.

Now we define $E' \subseteq E(G)$ in the original input graph as follows: For each active gadget for $t \in V(\mathcal{T})$, let $Y \subseteq E(G[X_t])$ be the subset of edges bought inside this gadget. We add Y to E'. Notice that $c(E') \leq c(\widehat{\mathcal{Q}})$. For each $t \in V(\mathcal{T})$, define E'_t to include all edges in E' that appear in some bag node of the subtree $\mathcal{T}_t$.

The next lemma guarantees the connectivity between a pair of vertices u, v.

Lemma 3.4. *For any active gadget $H(t, \Gamma, \Pi, \Sigma, \preceq)$, if $(u, v) \in \Gamma^{\preceq(u,v)} \cup \Pi \cup \Sigma^{\prec(u,v)}$, then u and v are connected in E'.*

We defer the proof of Lemma 3.4 to Section 3.5. Given Lemma 3.4, we show that the chosen edges form a feasible solution to GST, i.e., there is a path from the source s to every group S_i.

Corollary 3.1. *Every group S_i is connected to source s in E' .*

Proof. Since $\widehat{\mathcal{Q}}$ is a feasible solution of DC-GST to $\widehat{\mathcal{T}}$, there exists an active leaf gadget $H(t, \Gamma, \Pi, \Sigma, \preceq)$ in $\widehat{\mathcal{Q}}$ such that $H \in \widehat{S}_i$. Moreover, by definition of a group node and the fact that source s is part of every bag t, H must have an ancestor $r(t', \Gamma', \Pi', \Sigma', \preceq')$ such that $\exists u \in S_i$ and $(s, u) \in \Pi' \cup \Gamma'^{\preceq(s,u)} \cup \Sigma'^{\prec(s,u)}$. Applying Lemma 3.4 at gadget root $r(t', \Gamma', \Pi', \Sigma', \preceq')$, (s, u) are connected in E'. ■

3.5 Proof of Lemma 3.4.

Lemma 3.5. (Γ-Lemma) *The following statement holds for all active gadgets $H(t, \Gamma, \Pi, \Sigma, \preceq)$:*

$$(\forall (u, v) \in \Gamma)(u, v) \text{ is implied by } E'_t \cup \Pi \cup \Sigma^{\prec(u,v)}$$

Proof. We will prove that the statement holds for all $(u, v) \in \Gamma$ by induction from leaf to root. At the leaf, we have $\Gamma = \emptyset$, so the statement trivially holds.

Now consider any node $r(t, \Gamma, \Pi, \Sigma, \preceq)$ that has $r(t', \Gamma', \Pi', \Sigma', \preceq')$ and $r(t'', \Gamma'', \Pi'', \Sigma'', \preceq'')$ as descendants for which the induction hypothesis holds. Note that due to enforcement of degree bounds in the solution $\widehat{\mathcal{Q}}$, the descendant nodes are part of active gadgets. Consider any $(u, v) \in \Gamma$. If $(u, v) \notin \Gamma_Y$, we are immediately done, since (u, v) is implied by $Y \cup \Sigma^{\prec(u,v)} \cup \Pi \subseteq E'_t \cup \Sigma^{\prec(u,v)} \cup \Pi$.

So, we assume that $(u,v) \in \Gamma_Y$. Then $u = w_0, \ldots, w_\ell = v$ such that, for all α, we have one of the following cases:

- $(w_\alpha, w_{\alpha+1}) \in \Pi \cup \Sigma^{\prec(u,v)} \cup Y$, in which case we are again done.
- $(w_\alpha, w_{\alpha+1}) \in \Gamma_1^{\preceq(u,v)}$. By the properties of connections in the configuration tree $\mathcal{T}$, $(w_\alpha, w_{\alpha+1}) \in \Gamma_1$ is implied by $\Gamma' \cup \Pi'$.

  We can then write $w_\alpha = v_0, v_1, \ldots, v_{\ell'} = w_{\alpha+1}$, such that one of the following happens:

  1. $(v_\beta, v_{\beta+1}) \in \Pi'$. Then we would be done, since Π' can be implied by Π_Y, which is implied by $\Pi \cup Y \cup \Sigma^{\prec(v_\beta, v_{\beta+1})}$; remark that $(v_\beta, v_{\beta+1}) \preceq (u,v)$.
  2. $(v_\beta, v_{\beta+1}) \in \Gamma'$.

     We prove this via Claim 3.
- $(w_\alpha, w_{\alpha+1}) \in \Gamma_2^{\preceq(u,v)}$ A similar proof follows for this case.

Claim 3. *Any $(x,y) \in \Gamma' \cup \Gamma''$ is implied by $E'_{t'} \cup E'_{t''} \cup \Pi' \cup \Pi''$.*

Proof. We need a definition in order to apply induction.

Definition 3. *The rank $\mathcal{R}$ of a connection $(x,y) \in \Gamma'$ is recursively defined as,*

$$\begin{aligned} \mathcal{R}(x,y) &= 0 \text{ if } \Sigma'^{\prec'(x,y)} = \emptyset \\ &= 1 + \max_{(u,v) \in \Sigma'^{\prec'(x,y)}} \mathcal{R}(u,v) \end{aligned}$$

We define $\mathcal{R}(x,y)$ for $(x,y) \in \Gamma''$ in a similar fashion replacing Σ' with Σ'' and $\preceq'$ with $\preceq''$.

Proposition 3.2. *The rank function $\mathcal{R}$ is well defined.*

The rank function is defined for all connections in $\Sigma' \cup \Sigma''$. Indeed this is the case since $\Sigma' \subseteq \Gamma''$ and $\Sigma'' \subseteq \Gamma'$. Moreover, the two partial orders $\preceq', \preceq''$ are consistent by definition.

Now we prove the claim by induction on rank of (x,y). Assume first that $(x,y) \in \Gamma'$. Applying the induction hypothesis of Lemma 3.5 on the node $r(t', \Gamma', \Pi', \Sigma', \preceq')$, we get that (x,y) is implied by $E'_{t'} \cup \Pi' \cup \Sigma'^{\prec'(x,y)}$. Now we use induction on $\mathcal{R}(x,y)$. The base case if $\mathcal{R}(x,y) = 0$, in which case $\Sigma'^{\prec'(x,y)}$ is empty and we are done. Assume the statement holds for all connections (x', y') with $\mathcal{R}(x', y') \le m$. Let $\mathcal{R}(x,y) = m+1$. (x,y) is implied by a sequence $x = u_0, u_1, \cdots, u_l = y$ such that $\mathcal{R}(u_i, u_{i+1}) \le m$ and $(u_i, u_{i+1} \in E'_{t'} \cup \Pi' \cup \Sigma'^{\prec'(x,y)}$. The only non-trivial case is $(u_i, u_{i+1} \in \Sigma'^{\prec'(x,y)}$. By definition of edges in the configuration tree $\widehat{\mathcal{T}}$, $(u_i, u_{i+1} \in \Gamma'' \cup \Pi''$. Again, we are done if $(u_i, u_{i+1}) \in \Pi''$. If $(u_i, u_{i+1}) \in \Gamma''$, then we can apply induction hypothesis on $\mathcal{R}(u_i, u_{i+1})$ and hence (u_i, u_{i+1}) is implied by $E'_{t'} \cup E'_{t''} \cup \Pi' \cup \Pi''$ which gives us that (x,y) is implied by $E'_{t'} \cup E'_{t''} \cup \Pi' \cup \Pi''$.

This gives us the proof for Γ-Lemma.

Now we proceed to prove a second lemma which will directly give us a proof to Lemma 3.4.

Lemma 3.6. (Π-Lemma) *The following statement holds for all active gadget roots $r(t, \Gamma, \Pi, \Sigma, \preceq)$:*

$(\forall (u,v) \in \Pi \cup \Sigma)$ (u,v) is connected by some path in E'

Proof. We show this by induction from root-to-leaf. At the root, $\Sigma \cup \Pi = \emptyset$, so the statement holds trivially.

Now consider a node $r(t, \Gamma, \Pi, \Sigma, \preceq)$ for which the statement holds. We will show that it also holds for both $r(t', \Gamma', \Pi', \Sigma', \preceq')$ and $r(t'', \Gamma'', \Pi'', \Sigma'', \preceq'')$ where t', t'' are the children of t in $\mathcal{T}$.

Consider $(u,v) \in \Pi'$, so (u,v) is implied by

$$\Pi_Y = \{(a,b) :\ (a,b) \text{ is implied by } \Pi \cup Y \cup \Sigma^{\prec(a,b)}\}$$

So $u = w_0, \ldots, w_\ell = v$ where (w_i, w_{i+1}) belongs to one of the following cases:

- $(w_i, w_{i+1}) \in \Pi$, we would be done by induction hypothesis.
- $(w_i, w_{i+1}) \in Y$, we would be done because $Y \subseteq E' - E'_t$.
- $(w_i, w_{i+1}) \in \Sigma$, we are also done by induction hypothesis.

Now consider $(u,v) \in \Sigma' \subseteq \Gamma'' \cup \Pi''$. We apply the Γ-Lemma to say that (u,v) is implied by $E'_t \cup \Pi'' \cup \Sigma''^{\prec''(u,v)}$. This means that $u = w_0, \ldots, w_\ell = v$ where (w_i, w_{i+1}) is in one of these cases:

- $(w_i, w_{i+1}) \in E'_{t''}$, and we would be done.
- $(w_i, w_{i+1}) \in \Pi''$. We do the same analysis as above and would be done.
- $(w_i, w_{i+1}) \in \Sigma''^{\prec''(u,v)}$. By definition, $(w_i, w_{i+1}) \in \Gamma' \cup \Pi'$. If $(u,v) \in \Pi'$, we are again done. If $(w_i, w_{i+1}) \in \Gamma'$, then by Claim 3, (u,v) is implied by $E'_{t'} \cup E'_{t''} \cup \Pi' \cup \Pi''$. This gives us the lemma since $\Pi' \cup \Pi'' \subseteq \Pi$ and any element in Π is connected by E' by induction hypothesis.

It is straightforward to see that the Γ-Lemma and Π-Lemma together gives a proof for Lemma 3.4. ∎

3.6 Node-weighted Group Steiner Tree. The sparsifier for NW-GST can be constructed using similar ideas and following similar lines of reasoning. We describe the ideas on how this sparsifier differs from the one in edge-weighted case.

For each node $t \in V(\mathcal{T})$, our gadget has an additional parameter $Z \subseteq X_t$, that is, we have $H(t, \Gamma, \Pi, \Sigma, \preceq, Z)$. Similarly to what is done with the edge-weighted case, we only allow nodes to be bought if X_t is the topmost node in $\mathcal{T}$ where they appear. The new parameter $Z \subseteq X_t$ represents the nodes $v \in X_t$ that also appear in $X_{t'}$ and have been bought. This set is needed because the nodes in Z may help connect some $(u, v) \in \Gamma$, with lower cost, as they have been bought already. Another difference is that, we only allow $\Pi, \Sigma \subseteq Z^2$ (we can only connect things whose endpoints are bought).

We slightly change the definition of connection $(u, v) \in \Gamma$ being implied by Σ. We say that (u, v) is implied by Σ in Z if there is a sequence $u = w_0, \dots, w_\alpha = v$ such that $(w_\beta, w_{\beta+1}) \in \Sigma \cup E(G)$ for every $\beta = 0, \dots, \alpha - 1$ and each $w_\beta \in Z$ for every $\beta = 0, \dots, \alpha$.

Now, we construct the gadget $H(t, \Gamma, \Pi, \Sigma, \preceq, Z)$ as follows:

- The root is denoted by r.
- For each $Y \subseteq (X_t - X_{p(t)})$, we create a vertex p_Y, which is a child of r; the cost of the connecting edge is $c(Y) = \sum_{v \in Y} c(v)$. Denote by Γ_Y the set of Γ-connections (u, v) that are not implied by $\Pi \cup \Sigma^{\prec(u,v)}$ in $Y \cup Z$. Also, denote by Π_Y the set of pairs $(u, v) \in X_t \times X_t$ that are implied by $\Pi \cup \Sigma^{\prec(u,v)}$ in $Y \cup Z$.
- For each vertex p_Y, consider a feasible partition $\rho = (\Gamma_1, \Gamma_2, \preceq')$ of Γ via Π_Y in $Z \cup Y$. We create the vertices $q(Y, \rho)$ which are children of p_Y and have two children each, $q_1(Y, \rho)$ and $q_2(Y, \rho)$.

The analysis closely follows the edge-weighted case.

We remark that, using this approach, the problem solved on the new instance is still edge-weighted.

4 Solving the DC-GST instance via GKR Rounding

In this section, we describe how to find the solution $\widehat{\mathcal{Q}}$ to the instance of DC-GST on $\widehat{\mathcal{T}}$ with groups $\widehat{S}_1, \dots \widehat{S}_k$.

In a feasible solution for DC-GST, every non-leaf node has either degree 1 (root), 3 (q-nodes), or 2. We will state this in a slightly different way by saying that every q-node is fully connected, that is, if it is in the solution, then all its children are as well, while for the other non-leaf nodes, if they are in the solution, exactly one of their children is in the solution as well.

Therefore, we can abstract this tree instance as a special case of DC-GST on trees with two sets of special nodes: V_{one}, the set of *pick-one* nodes, and V_{all}, the set of *pick-all nodes*.

Let C_v be the set of children of a node v. For every $v \in V_{all}$, if the edge $(p(v), v)$ is picked, then all the edges (v, u) are picked as well, for $u \in C_v$. For $v \in V_{one}$, if the edge $(p(v), v)$ is picked, exactly one of the edges (v, u) is picked, for $u \in C_v$.

This implies modifications both in the LP and the rounding used for this tree instance. Regarding the LP, it is sufficient to add constraints (1), (2). The LP for this problem is presented in Figure 3.

As to the rounding, the added LP constraints for vertices $v \in V_{all}$ ensure that all children edges will be picked if $(p(v), v)$ is picked. On the other hand, the rounding must be modified for edges of the form (v, u), with $v \in V_{one}, u \in C_v$, so as to pick exactly one of these edges.

We modify rounding so that, if $v \in V_{one}$ and $(p(v), v)$ is in the solution, it picks exactly one of the edges (v, u), $u \in C_v$, with probabilities given by $p_{(v,u)} = x_{(v,u)}/x_{(p(v),v)}$. If $(p(v), v)$ is not in the solution, then no children edge is picked, as before.

We now prove two lemmas necessary for the analysis of GKR rounding to apply: first, we prove that, in expectation, the cost of a solution rounded in this way is the optimal cost of the LP; and second, we prove that if we replace the vertices in V_{one} by normal vertices, then the probability of connecting a fixed group decreases. As a consequence, the lower bounds for such probabilities in the usual GKR rounding are still valid for our modified rounding.

One last consideration is needed to prove that this results in a $O(\log n \log k)$-approximation for the general problem. We remark that, since our instance has size $N = n^{O(w^2 \log w)}$, a naive approach would result in a $O(\log N \log k) = O(w^2 \log w \log n \log k)$ approximation. However, GKR rounding actually has an approximation ratio of $O(h \log k)$, where h is the height of the tree. Since, in general, $h = O(n)$, the analysis includes a transformation that reduces the height to $O(\log n)$. However, in our case, we know already that $h = O(\log n)$, and thus the approximation ratio is $O(\log n \log k)$.

Lemma 4.1. *The expected cost of the solution S obtained by the modified rounding is* opt*, the cost of the LP.*

Proof. It is sufficient to prove that each edge e is added

$$
\begin{aligned}
&\min && \sum_{e\in E} c_e x_e && \\
&\text{s. t.} && \sum_{e\in\delta(S)} x_e \geq 1 && \forall\, S\subseteq V : r\in S, S\cap S_i=\emptyset \text{ for some } i \\
&(1) && x_{(p(v),v)} = x_{(v,u)} && \forall\, v\in V_{all}, u\in C_v \\
&(2) && x_{(p(v),v)} = \sum_{u\in C_v} x_{(v,u)} && \forall\, v\in V_{one}
\end{aligned}
$$

Figure 3: Linear Program for DC-EW-GST

to the solution with probability x_e, which implies the lemma by linearity of expectation.

We remark that an edge e is only added to the solution if its parent is also in the solution. Using this fact, coupled with a simple induction argument, we get the desired result. First, remark that edges e incident to the root are added with probability x_e.

By induction on the depth of the edges, the probability of an edge e at depth h being picked is:

$$
\begin{aligned}
P[e\in S] &= P[e\in S \mid p(e)\in S]P[p(e)\in S] \\
&= \frac{x_e}{x_{p(e)}} x_{p(e)} = x_e
\end{aligned}
$$

■

Lemma 4.2. *Let S be a solution obtained by rounding with the modified procedure, and S' be a solution obtained by rounding according to GKR (with no consideration for pick-one nodes). Then, for any group g,*

$$
\begin{aligned}
&P[S \text{ does not connect } r \text{ to } g] \\
&\leq P[S' \text{ does not connect } r \text{ to } g]
\end{aligned}
$$

Proof. Let us first fix an arbitrary group g. It is sufficient to prove that the result holds if S and S' are rounded similarly, except on exactly one pick-one node $v\in V_{one}$, for which the edges (v,u) are rounded using GKR on S' and using the modified procedure in S. We can then apply this simpler result successively for each node in V_{one} to obtain the lemma.

Let $e=(p(v),v)$ and S_e, S'_e be the solutions S, S' restricted to the sub-tree consisting of e and descendant edges. We now prove that, given that $p(e)$ is picked, the result follows in S_e and S'_e. This implies the result for the entire tree, as probabilities for other parts of the solution are not changed.

We define $F(X)$ as the event that $X\cap g=\emptyset$, that is, group g is not connected in X. Then

$$
\begin{aligned}
&P[F(S_e) \mid p(e)\in S] \\
&= \left(1-\frac{x_e}{x_{p(e)}}\right) + \frac{x_e}{x_{p(e)}} P[F(S_e)\mid e\in S] \\
&= \left(1-\frac{x_e}{x_{p(e)}}\right) + \frac{x_e}{x_{p(e)}} \left(1-\sum_{c \text{ child of } e}\frac{x_c}{x_e} P[\bar{F}(S_c)\mid c\in S]\right) \\
&= 1-\frac{x_e}{x_{p(e)}} \sum_{c \text{ child of } e}\frac{x_c}{x_e}\left(1-P[F(S_c)\mid c\in S]\right)
\end{aligned}
$$

$$
\begin{aligned}
&P[F(S'_e) \mid p(e)\in S'] \\
&= \left(1-\frac{x_e}{x_{p(e)}}\right) + \frac{x_e}{x_{p(e)}} P[F(S'_e)\mid e\in S'] \\
&= \left(1-\frac{x_e}{x_{p(e)}}\right) + \frac{x_e}{x_{p(e)}} \prod_{c \text{ child of } e}\left(1-\frac{x_c}{x_e}+\frac{x_c}{x_e} P[\bar{F}(S'_c)\mid c\in S]\right) \\
&= 1-\frac{x_e}{x_{p(e)}}\left(1-\prod_{c \text{ child of } e}\left(1-\frac{x_c}{x_e}\left(1-P[\bar{F}(S'_c)\mid c\in S]\right)\right)\right)
\end{aligned}
$$

Let $f_c = 1-P[\bar{F}(S_c)\mid c\in S]$. As the only difference between S and S' is the sampling of the children edges of e, then we also have that $1-P[\bar{F}(S'_c)\mid c\in S]=f_c$.

The final result that we need in order to prove the lemma is the following generalization of Bernoulli's inequality.

Claim 4. *Let $a_i\in[0,1]$, $i\in[n]$. Then,*

$$
\prod_{i\in[n]}(1-a_i) \geq 1-\sum_{i\in[n]} a_i
$$

Proof. We prove the claim by simple induction. When $n=1$, the inequality holds trivially. Now, assume it holds for $n'<n$. Then,

$$\prod_{i\in[n]}(1-a_i) = (1-a_n)\prod_{i\in[n-1]}(1-a_i)$$

$$\text{(By IH)} \qquad \geq (1-a_n)\left(1-\sum_{i\in[n-1]} a_i\right)$$

$$= 1 - a_n - \sum_{i\in[n-1]} a_i + a_n\sum_{i\in[n-1]} a_i$$

$$\geq 1 - \sum_{i\in[n]} a_i$$

■

Since $x_c f_c/x_e \in [0,1]$, we can use this claim and get that:

$$P[F(S'_e) \mid p(e) \in S']$$

$$= 1 - \frac{x_e}{x_{p(e)}}\left(1 - \prod_{c \text{ child of } e}\left(1 - \frac{x_c}{x_e} f_c\right)\right)$$

$$\geq 1 - \frac{x_e}{x_{p(e)}}\left(1 - \left(1 - \sum_{c \text{ child of } e} \frac{x_c}{x_e} f_c\right)\right)$$

$$= 1 - \frac{x_e}{x_{p(e)}} \sum_{c\text{child of } e} \frac{x_c}{x_e} f_c$$

$$= P[F(S_e) \mid p(e) \in S]$$

■

References

[1] Ittai Abraham, Yair Bartal, and Ofer Neiman. Advances in metric embedding theory. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 271–286. ACM, 2006.

[2] Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193, 1996.

[3] Yair Bartal and Manor Mendel. Multiembedding of metric spaces. *SIAM J. Comput.*, 34(1):248–259, 2004. Preliminary version in SODA'03.

[4] MohammadHossein Bateni, Erik D. Demaine, MohammadTaghi Hajiaghayi, and Dániel Marx. A PTAS for planar group steiner tree via spanner bootstrapping and prize collecting. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 570–583. ACM, 2016.

[5] MohammadHossein Bateni, Mohammad Taghi Hajiaghayi, and Dániel Marx. Approximation schemes for steiner forest on planar graphs and graphs of bounded treewidth. *J. ACM*, 58(5):21, 2011.

[6] Hans L. Bodlaender. NC-algorithms for graphs with small treewidth. In Jan van Leeuwen, editor, *Graph-Theoretic Concepts in Computer Science, 14th International Workshop, WG '88, Amsterdam, The Netherlands, June 15-17, 1988, Proceedings*, volume 344 of *Lecture Notes in Computer Science*, pages 1–10. Springer, 1988.

[7] Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999. Preliminary version in SODA'98.

[8] Chandra Chekuri and Julia Chuzhoy. Polynomial bounds for the grid-minor theorem. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 60–69. ACM, 2014.

[9] Chandra Chekuri, Guy Even, Anupam Gupta, and Danny Segev. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Trans. Algorithms*, 7(2):18, 2011.

[10] Chandra Chekuri, Guy Even, and Guy Kortsarz. A greedy approximation algorithm for the group steiner problem. *Discrete Applied Mathematics*, 154(1):15–34, 2006.

[11] Chandra Chekuri and Martin Pál. A recursive greedy algorithm for walks in directed graphs. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 245–253, 2005.

[12] Artur Czumaj, Magnús M. Halldórsson, Andrzej Lingas, and Johan Nilsson. Approximation algorithms for optimization problems in graphs with superlogarithmic treewidth. *Inf. Process. Lett.*, 94(2):49–53, 2005.

[13] Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Philip N. Klein. Node-weighted steiner tree and group steiner tree in planar graphs. *ACM Trans. Algorithms*, 10(3):13:1–13:20, 2014.

[14] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004. Preliminary version in STOC'03.

[15] Naveen Garg, Goran Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000. Preliminary version in SODA'98.

[16] Anupam Gupta, Kunal Talwar, and David Witmer. Sparsest cut on bounded treewidth graphs: algorithms and hardness results. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 281–290. ACM, 2013.

[17] Eran Halperin, Guy Kortsarz, Robert Krauthgamer,

Aravind Srinivasan, and Nan Wang. Integrality ratio for group steiner trees and directed steiner trees. *SIAM J. Comput.*, 36(5):1494–1511, 2007.

[18] Eran Halperin and Robert Krauthgamer. Polylogarithmic inapproximability. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 585–594, 2003.

[19] Christopher S. Helvig, Gabriel Robins, and Alexander Zelikovsky. An improved approximation scheme for the group steiner problem. *Networks*, 37(1):8–20, 2001.

[20] Joseph Naor, Debmalya Panigrahi, and Mohit Singh. Online node-weighted steiner tree and related problems. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 210–219. IEEE Computer Society, 2011.

[21] Gabriele Reich and Peter Widmayer. Beyond steiner's problem: A VLSI oriented generalization. In Manfred Nagl, editor, *Graph-Theoretic Concepts in Computer Science, 15th International Workshop, WG '89, Castle Rolduc, The Netherlands, June 14-16, 1989, Proceedings*, volume 411 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 1989.

[22] Neil Robertson and Paul D. Seymour. Graph minors. v. excluding a planar graph. *J. Comb. Theory, Ser. B*, 41(1):92–114, 1986.

[23] Alexander Zelikovsky. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica*, 18(1):99–110, 1997.