A PTAS for subset TSP in minor-free graphs

Hung Le

Oregon State University, Oregon, USA
lehu@onid.oregonstate.edu

Abstract

We give the first polynomial time approximation scheme for the subset Traveling Salesperson Problem (subset TSP) in H-minor-free graphs. Our main technical contribution is a polynomial time algorithm that, given an edge-weighted H-minor-free graph G and a set of k terminals T, finds a subgraph of G with weight at most $O_H(\text{poly}(\frac{1}{\epsilon})\log k)$ times the weight of the minimum Steiner tree for T that preserves pairwise distances between terminals up to $(1+\epsilon)$ factor. This is the first such spanner for H-minor-free graphs. Given this spanner, we use the contraction decomposition of Demaine, Hajiaghayi and Kawarabayashi [20] to obtain a PTAS for the subset TSP problem. Our PTAS generalizes PTASes for the same problem by Klein [33] for planar graphs and by Borradaile, Demaine and Tazari [10] for bounded genus graphs.

1 Introduction

Given an edge-weighted graph G and a set of k terminals T in G, the subset TSP problem asks for a shortest tour that visit every terminal in T. Subset TSP is even considered more relevant than TSP in practice since most of the time, we would like to visit a small subset of vertices in a network rather than every vertex of the network.

In general graphs, one may reduce this problem to TSP by working with the shortest path metric on the specified subset. However, if the graph has a special structure, such as excluding a minor, taking the shortest path metric would destroy this structure that may otherwise be used to algorithmic advantage. Thus, exploiting graph structures to better approximate subset TSP is a significantly harder problem than approximating TSP. In planar graphs, Arora, Grigni, Karger, Klein and Woloszyn [3] designed the first polynomial time approximation scheme[1] (PTAS) for TSP and raised a PTAS for subset TSP as an open problem. In a seminal paper, Klein [33] gave a positive answer to this question. Borradaile, Demaine and Tazari [10] generalized Klein's PTAS to bounded genus graphs. Obtaining a PTAS for subset TSP in minor-free graphs is an important open problem that has been asked several times [20, 10, 14]. We note that minor-free graphs significantly generalize planar graphs and bounded genus graphs. For example, the complete bipartite graph $K_{3,n}$ has unbounded genus but is K_5-minor-free. The main result of this paper is:

Theorem 1. *For any fixed $\epsilon > 0$, there is a polynomial time algorithm that, given an edge-weighted H-minor-free graph G and a set of k terminals T in G, finds a tour that visits every terminal of T at least once whose length is at most $(1+\epsilon)$ times the length of the optimal tour.*

[1] A polynomial-time approximation scheme is an algorithm which, for a fixed error parameter ϵ, finds a solution whose value is within $1 \pm \epsilon$ of optimal in polynomial time.

The precise running time of our algorithm in Theorem 1 is $n^{O_H(\mathrm{poly}(\frac{1}{\epsilon}))}$ where O_H hides the dependency of the constant on the size of H.

In designing PTAS for connectivity problems in planar graphs and minor-free graphs, there are two main ingredients: *light spanners* and a *contraction decomposition framework*.

Light spanners A light spanner for subset TSP, called a *light subset spanner*, is a subgraph S that satisfies two conditions: (i) $d_G(x,y) \leq d_S(x,y) \leq (1+\epsilon)d_S(x,y)$ for every $x, y \in T$ and (ii) $w(S) \leq f(\epsilon)w(\mathrm{ST})$ where ST is an optimal Steiner tree that spans T and $f(\epsilon)$ is a constant, called *lightness*, that depends on ϵ only.

Klein [33] was the first to give a polynomial time construction of light subset spanners for planar graphs. Klein's construction can be generalized to bounded genus graphs via the cutting technique, as noted by Borradaile, Demaine and Tazari [10]. Borradaile, Demaine and Tazari [10] conjectured that by using Robertson and Seymour's decomposition [38], it is possible to extend Klein's construction to minor-free graphs. However, this direction has not been fruitful. The main difficulty is that known subset spanner constructions rely on a charging argument based on non-crossing embeddings of input graphs. It is unclear how the charging argument can be modified at the presence of crossings. Thus, even in bounded treewidth graphs, which are normally regarded as easy instances of minor-free graphs, it is unknown whether light subset spanners exist.

In this work, we take a different path toward constructing a subset spanner with small weight. First, we introduce the *ℓ-close spanner* problem that captures the difficulty of constructing light subset spanners. An ℓ-close spanner for a terminal set T is a subgraph that almost preserves distances between terminal pairs whose shortest paths have weight at most ℓ. We show how to construct ℓ-close spanners of small weight based on two ingredients: (1) single-source spanners for general graphs that generalize Klein's planar single-source spanners [33] and (2) shortest path separators [1] for H-minor-free graphs. Our ℓ-close spanner construction is inspired by the terminal path cover construction for planar graphs of Cheung, Goranci and Henzinger [16]. An $O_H(\log k)$ factor is introduced into the lightness of ℓ-close spanners. Second, we show a lightness-preserving reduction from constructing light subset spanners to constructing light ℓ-close spanners. Our reduction is inspired by ideas in our recent joint work with Borradaile and Wulff-Nilsen [14] on analyzing greedy spanners for minor-free graphs. Since an $O(\log k)$ factor is introduced in the first step, the overall lightness of our spanner is $O_H(\log k\, \mathrm{poly}(\frac{1}{\epsilon}))$. For bounded treewidth graphs, we are able to remove $O(\log k)$ factor based on the recent work of Krauthgamer, Nguyễn, and Zondiner [35] in constructing terminal distance preserving minors.

Theorem 2. *Let T be a subset of k vertices of an H-minor-free graph G. Let* ST *be a minimum Steiner tree of G for T. There is a polynomial time algorithm that can find a subgraph S of G such that:*

1. $d_G(x,y) \leq d_H(x,y) \leq (1+\epsilon)d_G(x,y)$ *for every two distinct terminals* $x, y \in T$.
2. $w(S) = O_H(\mathrm{poly}(\frac{1}{\epsilon}) \log k)w(\mathrm{ST})$.

where $O_H(.)$ hides the dependency of the constant on $|H|$. Furthermore, if G has treewidth tw*, then* $w(S) = O(\mathrm{poly}(\frac{1}{\epsilon})\mathrm{tw}^5)w(\mathrm{ST})$.

Contraction decomposition framework A contraction decomposition framework is the second ingredient to our PTAS. Such a decomposition for minor-free graphs was found by Demaine, Hajiaghayi and Kawarabayashi [20].

Theorem 3 (Theorem 2 [20])**.** *Given an optimization problem $\mathscr{P}$ in H-minor-free graphs, if we can find (i) a spanner of lightness at most α for some $\alpha > 0$ in $n^{O(1)}$ time and (ii) an optimal solution for $\mathscr{P}$ in H-minor-free graphs of treewidth at most $O_H(\alpha)$ in time $h(\alpha)$, then we can obtain a PTAS for $\mathscr{P}$ in H-minor-free graph with running time $O_H(h(\alpha)n^{O(1)})$.*

Using the standard dynamic program for bounded treewidth graphs [6], we can find an optimal solution for subset TSP in $2^{O(\alpha \log \alpha)}n^{O(1)}$ time. Since our spanner has a $O(\log k)$ factor in the lightness, the resulting running time is $k^{O(\log \log k)}n^{O(1)}$ which is not a PTAS when $k = \Omega(n^\delta)$ for any constant $\delta > 0$. However, there are several advanced techniques for general graphs [19, 7, 25] and H-minor-free graphs [22] that can be employed to speed up the dynamic program to $2^{O(\alpha)}n^{O(1)}$:

Theorem 4. *There is a $2^{O(\mathrm{tw})}n^{O(1)}$-time algorithm that can solve subset TSP optimally in graphs of treewidth at most* tw*.*

We prove Theorem 4 using the rank based method [7] in the full version of this paper (appended). Theorem 4, Theorem 2 and Theorem 3 immediately imply Theorem 1. The running time of our PTAS is $O_H(k^{O_H(\mathrm{poly}(\frac{1}{\epsilon}))}n^{O(1)})$.

1.1 Implication for other connectivity problems

Many connectivity problems have been shown to have PTASes in planar graphs. Notable problems among them are TSP [32], subset TSP [33], Steiner tree [12], prize-collecting Steiner tree [4], Steiner forest [5] and survivable-network design [11]. Borradaile, Demaine and Tazari [10] generalized the PTASes to bounded genus graphs via the cutting technique. However, in H-minor-free graphs, prior to our work, only TSP was known to have a PTAS [20, 14]. Almost all known PTASes for connectivity problem in planar and bounded genus graphs rely on two components: light spanners and the contraction decomposition framework. The contraction decomposition framework for H-minor-free graphs was found by Demaine, Hajiaghayi and Kawarabayashi [20]. Thus, the main difficulty in obtaining PTASes for connectivity problems in H-minor-free graphs is the spanner construction step. Spanners for TSP problem are special in the sense that a simple greedy algorithm [2] gives light spanners for TSP in most interesting classes of graphs [14, 13]. There is no such a "universal" algorithm for other connectivity problems. Our spanner for subset TSP can be seen as a first step toward constructing light spanners for other problems in H-minor-free graphs.

1.2 Related works

In this section, we will review related work on spanners and their applications to approximating TSP, subset spanners and their applications to approximating subset TSP, and parameterized complexity of susbet TSP.

Spanners are subgraphs that preserve distances up to a certain factor for *all pairs* of vertices. Such spanners have been studied extensively in literature since the 90s [37, 2]. It has long been known that a simple greedy algorithm [2] gives an $(1 + \epsilon)$-spanner. Interestingly, Filtser and Solomon [24] showed, via an existential argument, that greedy spanners have the same asymptotic weight as optimal spanners in most settings. However, giving an explicit, tight bound on the weight of greedy spanners is a very difficult problem.

Althöfer, Das, Dobkin, Joseph and Soares [2] showed that greedy spanners in planar graphs have weight at most $(1 + \frac{2}{\epsilon})w(\mathrm{MST})$. Klein [32] used this spanner and his contraction decomposition framework for planar graphs to give a PTAS for planar TSP with running time $2^{O(\frac{1}{\epsilon^2})}n$.

Grigni [26] showed that greedy spanners in genus-g graphs have weight at most $O(\frac{g}{\epsilon})w(\text{MST})$. Demaine, Hajiaghayi and Mohar [21] used Grigni's spanner and their contraction decomposition framework to give a PTAS for TSP in bounded genus graphs with running time $2^{O(\text{poly}(\frac{1}{\epsilon}))}n^{O(1)}$. In H-minor-free graphs, Grigni and Sissokho [27] showed that greedy spanners have weight at most $O_H(\log n)w(\text{MST})$. Demaine, Hajiaghayi and Kawarabayashi [20] used Grigni and Sissokho's result, in combination with their contraction decomposition framework, to give a PTAS for TSP with running time $n^{O_H(\text{poly}(\frac{1}{\epsilon}))}$. It had been long open whether greedy spanners have weight at most $O_H(\text{poly}(\frac{1}{\epsilon}))w(\text{MST})$, until our recent joint work with Borradaile and Wulff-Nilsen [14], thus implying a PTAS with running time $2^{O_H(\text{poly}(\frac{1}{\epsilon}))}n^{O(1)}$.

In contrast, there are only two related results on constructing subset spanners of small weight: planar subset spanners by Klein [33] and subset spanners in bounded genus graphs by Borradaile, Demaine and Tazari [10]. Even in unweighted graphs, the subset spanner problem is highly nontrivial while the spanner problem becomes trivial; the sparsity of H-minor-free graphs implies that the whole graph is a light spanner. Indeed, sparsity is repeatedly used in the analysis of light spanners for H-minor-free graphs [14]. However, sparsity does not seem to help in the subset spanner problem.

Other closely related spanners are *pairwise spanners* and *pairwise preservers* where we want to approximately or exactly preserve the distances between a prescribed set of vertex pairs. There is a rich literature on pairwise spanners and preservers. However, most works focus on minimizing the number of edges in the spanners [17, 18, 30, 29, 9, 8].

The subset TSP problem is also studied in the parameterized complexity community. The classical dynamic programming algorithm of Held and Karp [28] can solve subset TSP in $O(2^k)n^{O(1)}$ time. Klein and Marx [34] design the first sub-exponential $(2^{O(\sqrt{k}\log k)}+W)n^{O(1)}$-time algorithm for subset TSP in planar graphs with maximum integer weight W. Marx, Pilipczuk and Pilipczuk [36] generalize Klein and Marx's algorithm to directed planar graphs and improve the running time to $2^{O(\sqrt{k}\log k)}n^{O(1)}$.

2 Subset spanner construction overview

We say an edge-weighted graph H is a *strict minor* of G if (i) H is a minor of G, (ii) $V(H) \subseteq V(G)$ and (iii) for every edge $e \in H$ with two endpoints x, y, $w_H(e) = d_G(x, y)$. Given a terminal set T of G, Krauthgamer, Nguyễn, and Zondiner [35] showed that G can be compressed by applying minor transformations such that the distances between every pair of terminals are preserved.

Lemma 1 (Theorem 2.1 [35])**.** *Let T be a set of k terminals in a graph G. There is a strict minor G' of G such that (i) $T \subseteq V(G')$, (ii) $V(G') = O(k^4)$ and $E(G') = O(k^4)$ and (iii) $d_{G'}(x, y) = d_G(x, y)$ for every two distinct terminals $x, y \in T$. Furthermore, G' can be found in polynomial time.*

By Lemma 1, we can assume w.l.o.g that G only has $O(k^4)$ vertices since we can find a subset spanner for terminals in the compressed graph of G and then decompress the subset spanner by replacing each edge by a shortest path between the edge's endpoints in G. Thus, the $\log n$ factor incurred in the weight of our subset spanner construction below can be reduced to $\log k$.

2.1 Spanners for close terminal pairs

We say two terminals $x, y \in T$ are *ℓ-close* if $d_G(x, y) \leq \ell$.

Definition 1 (ℓ-close spanners). *Given a graph G and a set of terminals T, a subgraph S of G is an* ℓ-close spanner *for T if for every two distinct ℓ-close terminals $x, y \in T$, $d_G(x, y) \le d_S(x, y) \le (1+\epsilon)d_G(x, y)$.*

Our first major contribution is to show that one can obtain an ℓ-close spanner of small weight in H-minor-free graphs. Since there are at most $O(k^2)$ terminal pairs, one can trivially obtain a spanner of weight at most $O(k^2\ell)$ by adding in a shortest path for each ℓ-close terminal pair. However, in our problem, we need an ℓ-close spanner of smaller weight. By exploiting H-minor-freeness, we can replace a factor k by a factor $\log n$. We also show a stronger result for graphs of treewidth at most tw.

Theorem 5. *Given an H-minor-free graph G of n vertices, a terminal set T of size k and a positive parameter ℓ, there is a polynomial time algorithm that can find an ℓ-close spanner S for T with weight at most $O_H(\ell k \log n \operatorname{poly}(\frac{1}{\epsilon}))$. Furthermore, if G has treewidth at most* tw*, then $w(S) = O(\mathrm{tw}^5 \ell k)$.*

We first overview the proof of Theorem 5 when G has treewidth at most tw. We borrow some ideas from recent developments on terminal distance preserving minors by Krauthgamer, Nguyễn, and Zondiner [35] who showed that:

Lemma 2 (Theorem 1.5 [35]). *Let T be a set of k terminals in a graph G of treewidth at most* tw*. There is a strict minor G' of G such that (i) $T \subseteq V(G')$, (ii) $V(G') = O(\mathrm{tw}^3 k)$ and (iii) $d_{G'}(x, y) = d_G(x, y)$ for every two distinct terminals $x, y \in T$. Furthermore, we can find G' in polynomial time.*

Intuitively, Lemma 2 tells us that shortest paths between terminals in bounded treewidth graphs share many edges. Thus, by carefully choosing a set of shortest paths between terminal pairs, we can obtain an ℓ-close spanner of weight at most $O(k\ell)$ from such paths. One may ask whether we can apply Lemma 1 to obtain an ℓ-close spanner with small weight for minor-free graphs. In our construction, to obtain an ℓ-close spanner with (nearly) constant lightness, we need a strict minor of (nearly) linear size. Lemma 1 only gives us an ℓ-close spanner of lightness $O(k^3)$, which is worst than the trivial spanner that includes all pairwise shortest paths.

A natural idea to deal with H-minor-free graphs is extending Lemma 2 to H-minor-free graphs. However, a negative result by Krauthgamer, Nguyễn, and Zondiner [35] showed that it is impossible to do so, even in planar graphs. Formally, they showed that any minor must have at least $\Omega(k^2)$ Steiner vertices[2] to preserve pairwise distances of k terminals exactly. Even in the approximate setting where one seeks to approximately preserve terminal distances up to $(1+\epsilon)$ factor, the best known approximate terminal distance preserving minors for planar graphs have $\Omega(k^2 \operatorname{poly}(\log k)/\epsilon^2)$ Steiner vertices [16].

Inspired by the construction of the terminal path cover for planar graphs by Cheung, Goranci and Henzinger [16] that was in turn inspired by the construction of distance oracles for H-minor-free graphs by Kawarabayashi, Klein and Sommer [31], we propose an ℓ-close spanner construction based on single-source spanners. Instead of bounding the number of Steiner vertices as in previous papers [31, 16], we bound the weight of the spanner. To achieve that goal, we need several new technical ingredients. We first show that Klein's planar single-source spanners [33] are light even without planarity.

[2]Vertices in $V(G) \setminus T$ are called *Steiner vertices*.

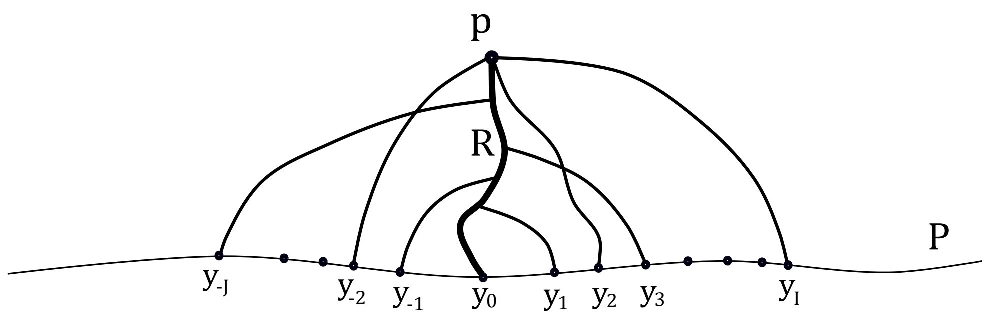

Figure 1: A single-source spanner constructed by Klein's algorithm. The thick path is R.

Lemma 3. *Let p be a vertex and P be a shortest path in a graph G. Let $y_0 \in P$ be such that $d_G(p, y_0) = d_G(p, P)$. Let $R = d_G(p, P)$. Fix an endpoint of P to be its left-most vertex. Let $\{y_1, \ldots, y_I\} \subseteq V(P)$ be a maximal set of vertices such that y_i is the closest point to the right of y_{i-1} such that:*

$$(1+\epsilon)d_G(p, y_i) < d_G(p, y_{i-1}) + d_P(y_{i-1}, y_i) \qquad 1 \leq i \leq I \tag{1}$$

We symmetrically define a maximal set of points $(y_{-1}, y_{-2}, \ldots, y_{-J})$ to the left of y_0 on P. Let $\mathcal{Q} = \{Q_{-J}, Q_{-J+1}, \ldots, Q_{-1}, Q_0, Q_1, \ldots, Q_I\}$ be a set of shortest paths where Q_i is a shortest p-to-y_i path in G, $-J \leq i \leq I$. Then, we have:

(1) $d_{\mathcal{Q}\cup P}(p, q) \leq (1+\epsilon)d_G(p, q)$ for every $q \in P$.
(2) $w(\mathcal{Q}) \leq 8\epsilon^{-2}R$.
(3) $I, J \leq 8\epsilon^{-2}$.
(4) $d_P(y_0, y_I) \leq 4\epsilon^{-1}R$ and $d_P(y_{-J}, y_0) \leq 4\epsilon^{-1}R$.

See Figure 1 for an illustration of a single-source spanner defined in Lemma 3.

For any two paths P and Q, we say P *crosses* Q if $P \cap Q \neq \emptyset$. We say P crosses a set of paths $\mathcal{Q}$ if there exists a path $Q \in \mathcal{Q}$ such that P crosses Q. We use Lemma 3 to show that:

Lemma 4. *Let $\mathcal{P}$ be a set of shortest paths in an edge-weighted graph G. Let $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_r\}$ be another set of shortest paths in G such that Q_i crosses $\mathcal{P}$ and $w(Q_i) \leq \ell$, for every $1 \leq i \leq r$. We denote the endpoints of each Q_i by s_i and t_i. Let k be the number of distinct endpoints of $\mathcal{Q}$. There is a subgraph H of G with weight at most $O(k\epsilon^{-2}\ell|\mathcal{P}|)$ such that $d_H(s_i, t_i) \leq (1+\epsilon)d_G(s_i, t_i)$ for every $1 \leq i \leq r$. Furthermore, H can be found in polynomial time.*

Let PTPSpanner$(G, \mathcal{P}, \mathcal{Q}, \ell, \epsilon)$ (PTP means path-to-path.) be the subgraph of Lemma 4. We use this to construct an ℓ-close spanner S as stated in Theorem 5. (See Figure 2.) The input to EllCloseSpanner$(G, T, \mathcal{Q}, \ell, \epsilon)$ consists of an edge-weighted H-minor-free graph G, a set of terminals T, a set of shortest paths $\mathcal{Q} = \{Q_1, \ldots, Q_h\}$ between ℓ-close terminals in T and the stretch parameter ϵ. The algorithm makes use of the following shortest path separator for H-minor-free graphs by Abraham and Gavoille [1].

Lemma 5 (Theorem 1 [1])**.** *For every connected H-minor-free graph G of n vertices, there is a family of γ sets of paths $\Omega = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_\gamma\}$ of G such that:*

1. *$\sum_{i=1}^{\gamma} |\mathcal{P}_i| = O_H(1)$.*
2. *$\mathcal{P}_1$ is a set of shortest paths of G and $\mathcal{P}_i$ is a set of shortest paths of $G \setminus V(\cup_{j<i}\mathcal{P}_i)$ for $i \geq 2$.*
3. *Connected components of $G \setminus V(\Omega)$ have size at most $n/2$.*

```
EllCloseSpanner(G, T, Q, ℓ, ε)
    if |T| ≤ 1 return ∅
    S ← ∅
    P_0 ← ∅; Ω ← {P_1, ..., P_γ} as in Lemma 5
    for i ← 1 to γ
        G_i ← G \ (∪_{j=0}^{i-1} P_j)
        Q_i ← the set of paths in Q that cross P_i
        S ← S∪ PTPSpanner(G_i, P_i, Q_i, ℓ, ε)
        Q ← Q \ Q_i
    for each component G' of G \ V(Ω)
        T' ← T ∩ V(G')
        Q' ← remaining paths in Q with both endpoints in T'
        S ← S∪ EllCloseSpanner(G', T', Q', ℓ, ε)
    return S
```

Figure 2: An ℓ-spanner construction algorithm.

We represent the execution of EllCloseSpanner$(G, T, \mathcal{Q}, \ell, \epsilon)$ by a recursion tree $\mathcal{T}$ where each node represents a recursive call on a subgraph, say K of G, and its child nodes are recursive calls on connected components of $K \setminus \Omega_K$. Here Ω_K is a shortest-path separator of K as in Lemma 5. The root node of $\mathcal{T}$ is a call on G. Since the size of child graphs in recursive calls is at most half the size of the parent graph, $\mathcal{T}$ has depth $O(\log n)$.

We now bound the total weight of S that is the output of EllCloseSpanner$(G, T, \mathcal{Q}, \ell, \epsilon)$. Consider i-th iteration in the first **for** loop in Figure 2. Since $\mathcal{Q}_i$ is a set of shortest paths in G, it is also a set of shortest paths in G_i. By Lemma 4 and (1) of Lemma 5, the total weight of S after the first for loop is at most $O(k\epsilon^{-2}\ell \sum_{i=1}^{\gamma} |\mathcal{P}_i|) = O_H(k\epsilon^{-2}\ell)$.

That implies at each level of $\mathcal{T}$, the weight of the returned subgraph of each node is $O_H(k\epsilon^{-2}\ell)$ plus the weight of the subgraphs returned from recursive calls. Since the depth of $\mathcal{T}$ is $O(\log n)$, $w(S) \leq O_H(k\epsilon^{-2}\ell \log n)$. We leave the details of the proof that $d_S(x, y) \leq (1+\epsilon)d_G(x, y)$ for every distinct ℓ-close terminals $x, y \in \mathcal{T}$ to the full version, but the argument is similar to other spanner constructions, such as for planar or bounded genus graphs.

2.2 A lightness-preserving reduction to constructing ℓ-close spanners

Our second major contribution is a reduction from the problem of constructing a subset spanner to that of constructing an ℓ-close spanner.

Theorem 6. *Given an H-minor-free graph G of n vertices and a terminal set T of size k. If for any given ℓ and any subset $T' \subseteq T$, there is an ℓ-close spanner for T' with weight at most $O(\tau(\epsilon, k, n)|T|'\ell)$, then G has a subset spanner with weight at most $O(\mathrm{poly}(\frac{1}{\epsilon})\tau(\epsilon, k, n))w(\mathrm{ST})$ where $\tau(\epsilon, k, n)$ is a function of ϵ, k, n.*

Theorem 2 follows from Theorem 6 since $\tau(\epsilon, k, n) = O(\log n \,\mathrm{poly}(\frac{1}{\epsilon}))$ when G is H-minor-free and $\tau(\epsilon, k, n) = O(\mathrm{tw}^5)$ when G has treewidth at most tw (Theorem 5). By Lemma 1, we can further improve the $\log n$ factor to $\log k$.

Our reduction is based on the iterative super-clustering technique, that was used to analyze greedy spanners for H-minor graphs and graphs of doubling dimension in our recent joint works with Borradaile and Wulff-Nilsen [14, 13]. The technique was also used before to construct sparse and light spanners for general graphs [15, 23].

We first find a constant approximation (in linear time [39]) of the optimal Steiner tree (ST) of G for terminal set T. Let $n_0 = \max(n, \frac{k(k-1)}{2})$ and $w_0 = \frac{w(\mathrm{ST})}{n_0}$. We subdivide every ST edge, say e, of length more than w_0 into $\lceil \frac{w(e)}{w_0} \rceil$ edges of length at most w_0 each. We then allocate $c(\epsilon)w_0$ credits to each new ST edge, where $c(\epsilon)$ is a constant that will be specified later. That is, the amount of credits allocated to each ST edge is at least $c(\epsilon)$ times its length. We can show that the total allocated credits is $O(c(\epsilon)w(\mathrm{ST}))$. We will only use credits of ST edges to pay for edges that will be added to the subset spanner. Thus, we can think of $c(\epsilon)$ as the asymptotic lightness of the subset spanner that we will construct.

Let $\mathcal{Q}$ be a maximal set of shortest paths between terminals in T such that no terminal is an internal vertex of a path in $\mathcal{Q}$. By the triangle inequality, it suffices to construct a spanner for paths in $\mathcal{Q}$. We build a subset spanner S in multiple steps. First, we add to S every path in $\mathcal{Q}$ of length at most w_0. We show in the full version that the total weight of paths of length at most w_0 is $O(\frac{w(\mathrm{ST})}{\epsilon})$. Let $J = \lceil \log(1/\epsilon) \rceil$ and $I = \lceil \log_{1/\epsilon} n_0 \rceil$. For a fixed i, j where $1 \leq j \leq J, 0 \leq i \leq I$, we define:

$$\Pi_i^j = \left\{ Q \in \mathcal{Q} : \frac{2^{j-1}}{\epsilon^i} w_0 < w(Q) \leq \frac{2^j}{\epsilon^i} w_0 \right\}$$

For a fixed j, $1 \leq j \leq J$, we define a hierarchy of paths $\mathcal{H}_j = \cup_{i=0}^{I} \Pi_i^j$. We refer to paths in Π_j^i as *level-i paths* of hierarchy $\mathcal{H}_j$. We will find a low weight spanner for shortest paths in each hierarchy separately. Since there are at most $O(\log \frac{1}{\epsilon})$ hierarchies, an $O(\log \frac{1}{\epsilon})$ factor will be introduced to the final lightness of the spanner.

For each level, say i, of hierarchy $\mathcal{H}_j$, we construct a set of clusters $\mathcal{C}_i$, where each cluster $C \in \mathcal{C}_i$ is a connected subgraph of S. Unlike prior works [14, 13], clusters in our setting are not necessarily vertex-disjoint. That introduces various technical complications in our cluster construction. We call clusters in $\mathcal{C}_i$ *level-i clusters*. We construct clusters in all levels iteratively. Level-i clusters will be constructed from level-$(i-1)$ clusters and level-0 clusters will be constructed from the Steiner tree ST. Let $\ell_i = \frac{2^j}{\epsilon^i} w_0$. We will inductively maintain the following invariants:

(I1) Each level-i cluster has diameter at most $g\ell_i$ where $g = 125$.
(I2) Each level-i cluster of diameter d has at least $c(\epsilon) \cdot \max(d, \ell_i/2)$ credits.

Intuitively, invariant (I1) guarantees that the diameter of level i-clusters is roughly the same as the length of level-i paths. Let T' be the subset of terminals in T that are endpoints of paths in Π_i^j. For each level-$(i-1)$ cluster that contains at least one terminal in T', we designate one terminal to be its center. Let T'' be the set of centers. We construct an $O(\ell_i)$-close spanner, say K, for T'' and add all edges of K to S. Since paths in Π_i^j have length at most ℓ_i, we can show that K is an $(1+\epsilon)$-spanner for paths in Π_i^j.

Suppose that we are allowed to use all credits of level-$(i-1)$ clusters containing terminals in T'' to pay for edges of K. By invariant (I1) and (I2), each level-$(i-1)$ cluster has at least $c(\epsilon)(\ell_{i-1}/2) = c(\epsilon)\epsilon\ell_i/2$ credits, for a total of $\Omega(c(\epsilon)\epsilon\ell_i|T''|)$ credits. By the assumption of Theorem 6, $w(K) \leq O(\tau(\epsilon, k, n)\ell_i|T''|)$. By choosing $c(\epsilon) = \Theta(\frac{\tau(\epsilon,k,n)}{\epsilon})$, the total credit of level-$(i-1)$ clusters containing terminals in T'' can pay for all edges of K.

However, we cannot use all credits of level-$(i-1)$ clusters since we need to maintain invariant (I2) for level-i clusters. Instead, our cluster construction algorithm will guarantee that after spending credits to maintain (I2), on average, each level-$(i-1)$ cluster still has at least $c(\epsilon)\,\mathrm{poly}(\epsilon)\ell_{i-1} = c(\epsilon)\,\mathrm{poly}(\epsilon)\ell_i$ credits left. By choosing $c(\epsilon) = \Theta(\mathrm{poly}(\frac{1}{\epsilon})\tau(\epsilon,k,n))$ we are still able to pay for all edges of K.

We now give intuition for how to maintain invariant (I2). Recall level-0 clusters are constructed from the Steiner tree ST and that each ST edge (of length at most w_0) has at least $c(\epsilon)w_0$ credits. Indeed, our construction guarantees that level-0 clusters are vertex-disjoint. Thus, each level-0 cluster can take credits directly from ST edges inside it. Credits of ST edges outside level-0 clusters are unused and hence can be used to guarantee invariant (I2) for higher level clusters.

To maintain invariant (I2) for level-i clusters, we will use partial credits of level-$(i-1)$ clusters and ST edges connecting level-$(i-1)$ clusters. Since clusters are non-disjoint, we need to to guarantee that credits of each ST edge are used at most once during the cluster construction. To that end, after constructing level-i clusters, we maintain a *cluster tree* $\mathcal{ST}_i(\mathcal{V}_i,\mathcal{E}_i)$ whose vertices are level-i clusters and whose edges are ST edges connecting two vertices in the two corresponding clusters. $\mathcal{ST}_i(\mathcal{V}_i,\mathcal{E}_i)$ satisfies the following invariant:

(I3) Credits of edges of $\mathcal{ST}_i(\mathcal{V}_i,\mathcal{E}_i)$ have not been used in the construction of level-i or lower level clusters.

Below we describe the cluster construction in more detail. To construct clusters for level 0, we greedily break ST into subtrees of diameter at least ℓ_0 and at most $6\ell_0$. Each subtree then serves as a level-0 cluster. Note that level-0 clusters are vertex-disjoint subgraphs of S (we add every edge of ST to S).

Suppose that we already have constructed level-$(i-1)$ clusters and the corresponding level-$(i-1)$ cluster tree $\mathcal{ST}_{i-1}(\mathcal{V}_{i-1},\mathcal{E}_{i-1})$. To simplify the presentation, we drop the index i. That is, we use Π, ℓ, $\mathcal{ST}(\mathcal{V},\mathcal{E})$ to denote Π_i^j, ℓ_i and $\mathcal{ST}_{i-1}(\mathcal{V}_{i-1},\mathcal{E}_{i-1})$, respectively. We refer to clusters in level $(i-1)$ as *ϵ-clusters* since their diameter is an ϵ-fraction of the diameter of level-i clusters. Let Q be a path in Π, that we call a Π*-path*. Since an ϵ-cluster has diameter at most $g\epsilon\ell$, when ϵ is sufficiently small, there is no Π-path that has both endpoints in the same ϵ-cluster.

Π-path removal We say two Π-paths are *parallel* if their endpoints belong to the same two ϵ-clusters. For each maximal set of parallel Π-paths, we only keep one Π-path of minimum length and remove other paths from Π. We apply this removal process to all maximal subsets of parallel paths of Π. We then remove every Π-path Q from Π such that the distance between two endpoints of Q in S (constructed so far) is at most $(1+s\cdot\epsilon)w(Q)$ where $s = 16g+1 = 2001$, since there is already an $(1+\epsilon)$-stretch path between Q's endpoints in S.

Constructing spanners for paths in Π Since ϵ-clusters are non-disjoint, a terminal can be contained in many different ϵ-clusters. For each terminal $t \in T$, we designate an (arbitrary) ϵ-cluster containing t to be its *primary ϵ-cluster*. We say that an ϵ-cluster C is *incident* to a Π-path Q if C is a primary ϵ-cluster of at least one of Q's endpoints.

We call an ϵ-cluster X a Π*-neighbor* of an ϵ-cluster Y if X and Y are incident to the same Π-path. We say an ϵ-cluster has *high-degree* if it has at least $\frac{3g}{\epsilon}$ Π-neighbors and *low-degree* otherwise. For each low-degree ϵ-cluster X, we add to spanner S all Π-paths incident to X. Let $\mathcal{C}_\epsilon$ be the set of all high-degree ϵ-clusters. For each $X \in \mathcal{C}_\epsilon$, we designate a terminal to be its center. Note that X must have a terminal since it is incident to a Π-path. Let T' be the set of centers of all ϵ-clusters in $\mathcal{C}_\epsilon$. Since each terminal has exactly one primary ϵ-cluster, $T' = |\mathcal{C}_\epsilon|$. Let $K =$

EllCloseSpanner$(G, T', \mathcal{Q}', 3\ell, \epsilon)$ where $\mathcal{Q}'$ is the maximal set of shortest paths of length at most 3ℓ between terminals in T'. K is a (3ℓ)-close spanner for T'. By the assumption of Theorem 6, we have:

$$w(K) = O(\tau(\epsilon, k, n)\ell|T'|) = O(\tau(\epsilon, k, n)\ell|\mathcal{C}_\epsilon|) \quad (2)$$

We then add every edge of K to S. This completes the spanner construction for Π-paths. In the full version, we show to guarantee the stretch for paths in Π.

Constructing level-i clusters Recall that in the spanner construction step, every Π-path incident to a low-degree ϵ-cluster is added to S and every Π-path incident to two high-degree ϵ-clusters has an $(1+\epsilon)$-approximate shortest path in S. Let $\mathcal{E}'$ be the set of edges between vertices in $\mathcal{V}$ where each edge in $\mathcal{E}'$ corresponds to a Π-path Q connecting its incident ϵ-clusters or Q's approximate shortest path in S if both endpoint ϵ-clusters of Q have high degree. We call edges of $\mathcal{E}'$ Π-*edges*. We denote the graph, called *cluster graph*, with vertex set $\mathcal{V}$ and edge set $\mathcal{E} \cup \mathcal{E}'$ by $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$. Observe that $\mathcal{ST}(\mathcal{V}, \mathcal{E})$ is a spanning tree of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$. We use bold lowercase letters to denote vertices and edges of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$.

Let $\kappa(.)$ be the function that maps each vertex $\mathbf{v} \in \mathcal{V}$ to the corresponding ϵ-cluster and each edge $\mathbf{e} \in \mathcal{E} \cup \mathcal{E}'$ to the corresponding ST edge or paths. We first observe that $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$ is a simple graph when $\epsilon < \frac{1}{4g+2}$. We define a weight function $\boldsymbol{\omega} : \mathcal{V} \cup \mathcal{E} \cup \mathcal{E}' \to \mathbb{R}$ where $\boldsymbol{\omega}(\mathbf{v}) = \mathrm{diam}(\kappa(\mathbf{v}))$ for each vertex $\mathbf{v} \in \mathcal{V}$ and $\boldsymbol{\omega}(\mathbf{e}) = w(\kappa(\mathbf{e}))$ for each edge $\mathbf{e} \in \mathcal{E} \cup \mathcal{E}'$. Let $\mathcal{P}$ be a path of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$. We define $\mathcal{P}$'s weight, denoted by $\boldsymbol{\omega}(\mathcal{P})$, to be its total vertex and edge weights.

Recall that high-degree ϵ-cluster is incident to at least $\frac{3g}{\epsilon}$ Π-paths. We call the corresponding vertex $\kappa^{-1}(X) \in \mathcal{V}$ of a high-degree ϵ-cluster X a *high-degree vertex*. Instead of constructing level-i clusters directly, we will construct a set of connected subgraphs Γ of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$. Each subgraph $\mathcal{S} \in \Gamma$ will then define a level-i cluster $\kappa(\mathcal{S}) = (\cup_{\mathbf{v} \in \mathcal{S}} \kappa(\mathbf{v})) \bigcup (\cup_{\mathbf{e} \in \mathcal{S}} \kappa(\mathbf{e}))$. The construction proceeds in four phases. We only sketch the intuition of each phase and defer the details to the full version.

Phase 1: High-degree vertices We greedily construct subgraphs in three steps. The main purpose of this phase is to guarantee that every high-degree vertex and its Π-neighbors are grouped into subgraphs.

Phase 2: Low-degree, branching vertices Let $\mathcal{F}$ be the forest of $\mathcal{ST}(\mathcal{V}, \mathcal{E})$ obtained by removing vertices involved in Phase 1. We say a vertex $\mathbf{v}$ $\mathcal{F}$-*branching* if it has degree at least 3 in $\mathcal{F}$. Let $\mathcal{P}$ be a path of $\mathcal{F}$. We define *effective diameter* of $\mathcal{P}$ to be the total vertex weight of $\mathcal{P}$. We then define effective diameter of a subtree of $\mathcal{F}$ to be the maximum effective diameter over all paths of the tree. This phase has two steps. The purpose is to group every $\mathcal{F}$-branching vertices of high effective diameter trees into subgraphs.

Phase 3: High-diameter paths of $\mathcal{F}$ We say a vertex $\mathbf{v}$ in a high-diameter path $\mathcal{P}$ *deep* if it is not an endpoint of $\mathcal{P}$ and the two subpaths of $\mathcal{P} - \{\mathbf{v}\}$ each has effective diameter at least 2ℓ. Let $\mathbf{e}$ be a Π-edge with two endpoints, say $\mathbf{x}, \mathbf{y}$, that are deep vertices. We group $\mathbf{e}$ and four subpaths of $\mathcal{F}$ that share $\mathbf{x}, \mathbf{y}$ as endpoints into a new subgraph of Γ.

Phase 4: Remaining high-diameter paths of $\mathcal{F}$ Let $\mathcal{P}$ be a high-diameter path of $\mathcal{F}$ after Phase 3. We break $\mathcal{P}$ into segments of effective diameter at least 2ℓ and at most 4ℓ. Let $\mathcal{X}$ be a segment of $\mathcal{P}$. If $\mathcal{X}$ has an ST edge to an existing subgraph in Γ (formed in previous phases), we defer the processing of $\mathcal{X}$ to Phase 5. Otherwise, we form a new subgraph of Γ from $\mathcal{X}$.

Phase 5: Remaining low-diameter trees of $\mathcal{F}$ Remaining components of $\mathcal{F}$ are trees (and paths) of effective diameter at most 4ℓ. Since $\mathcal{ST}(\mathcal{V}, \mathcal{E})$ is a spanning tree of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$, each tree in $\mathcal{F}$, say $\mathcal{T}$, must has at least one ST edge, say $\mathbf{e}$, to an existing subgraph in Γ, say $\mathcal{S}$, that is originated in the first three phases. We augment $\mathcal{F}$ with $\mathcal{T}$ and $\mathbf{e}$. We apply the augmentation to every tree of $\mathcal{F}$.

This completes the construction of Γ. We now show how to maintain cluster invariants and paying for spanner edges. By construction, subgraphs in Γ are vertex-disjoint. By bounding the diameter of subgraphs in Γ, we can show that:

Lemma 6. *Level-i clusters have diameter at most 125ℓ when ϵ is sufficiently smaller than $1/g$.*

Since $g = 125$, invariant (I1) is satisfied. To maintain invariant (I2), we would argue that credits of vertices and ST edges inside subgraphs of Γ are sufficient to both maintain invariant (I2) and pay for spanner edges. By construction, each ϵ-cluster after Phase 1 is incident to at most $\frac{3g}{\epsilon}$ Π-edges.

Subgraphs originating in Phase 1 By construction, each Phase-1 subgraph has at least $\frac{3g}{\epsilon}$ vertices. Let $\mathcal{Z}_1$ and $\mathcal{Z}_2$ be any two disjoint subsets of vertices of $\mathcal{S}$ such that $|\mathcal{Z}_1| = \frac{2g}{\epsilon}, |\mathcal{Z}_2| = \frac{g}{\epsilon}$. We can show that vertex credit of $\mathcal{Z}_1$ is sufficient to maintain invariant (I2) of $\mathcal{S}$. We then redistribute vertex credit of $\mathcal{Z}_2$ to every vertex in $\mathcal{Z}_1 \cup \mathcal{Z}_2$. On average, each vertex has at least $(\frac{g}{\epsilon}c(\epsilon)\epsilon\ell/2)/(\frac{3g}{\epsilon}) = c(\epsilon)\epsilon\ell/6$ credits.

Recall the set of ϵ-clusters that correspond to high-degree vertices of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$ is $\mathcal{C}_\epsilon$ defined in the spanner construction step. The total remaining vertex credits of $\mathcal{C}_\epsilon$ is at least $|\mathcal{C}_\epsilon| c(\epsilon)\ell/6$. By Equation 2, credits of $\mathcal{C}_\epsilon$ are sufficient to pay for K when $c(\epsilon) = \Omega(\frac{\tau(\epsilon,k,n)}{\epsilon})$.

Subgraphs originating in Phase 2+3 We argue that after maintaining invariant (I2) for Phase 2 or 3 subgraphs in Γ, each vertex has at least $\frac{c(\epsilon)\epsilon^2\ell}{g}$ credit. (Recall that in spanner construction step, we add all Π-paths incident to ϵ-clusters of degree at most $\frac{3g}{\epsilon}$). Thus, remaining credits of each vertex is sufficient to pay for its incident Π-edges when $c(\epsilon) = \Omega(\frac{g^2}{\epsilon^3})$.

Subgraphs originating in Phase 4 By construction, each subgraph originating in Phase 4, say $\mathcal{S}$, is a path whose edges are ST edges. We can show that Phase 4 subgraphs can maintain invariant (I2) using credits of its vertices and ST edges. However, to pay for incident Π-edges, we need to distinguish between two types of paths. We say a path $\mathcal{S}$ *internal* if it is not an affix of a high-diameter path $\mathcal{P}$ in Phase 4. We can show that Π-edges incident to internal subpaths are already paid for by subgraphs originating in first three phases. If $\mathcal{S}$ is an affix of a long path $\mathcal{P}$, we would use credits of vertices in another affix of $\mathcal{P}$, say $\mathcal{X}$, to pay for Π-edges incident to $\mathcal{S}$. This is possible because $\mathcal{X}$ would be merged to other subgraphs in Γ during Phase 5.

To maintain invariant (I3), we note that Γ is a collection of connected, vertex-disjoint subgraphs of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$. Thus, by contracting each subgraph in Γ into a vertex, we obtain a multigraph $\mathcal{G}'$ from $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$. Since $\mathcal{ST}(\mathcal{V}, \mathcal{E})$ is a spanning tree of $\mathcal{G}(\mathcal{V}, \mathcal{E} \cup \mathcal{E}')$, there is a spanning tree, say $\mathcal{ST}_i(\mathcal{V}_i, \mathcal{E}_i)$, of $\mathcal{G}'$ that contains only ST edges. Since we never use credits of ST edges outside subgraphs in Γ to maintain invariant (I2), $\mathcal{ST}_i(\mathcal{V}_i, \mathcal{E}_i)$ satisfies invariant (I3).

Acknowledgement: We thank Cora Borradaile for constructive comments. This material is based upon work supported by the National Science Foundation under Grant No. CCF-1252833.

References

[1] I. Abraham and C. Gavoille. Object location using path separators. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Principles of Distributed Computing*, PODC '06, pages 188–197, 2006.

[2] I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete Computational Geometry*, 9(1):81–100, 1993.

[3] S. Arora, M. Grigni, D. R. Karger, P. N. Klein, and A. Woloszyn. A polynomial-time approximation scheme for weighted planar graph TSP. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '98, pages 33–41, 1998.

[4] M. Bateni, C. Chekuri, A. Ene, M. Hajiaghayi, N. Korula, and D. Marx. Prize-collecting steiner problems on planar graphs. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 1028–1049, 2011.

[5] M. Bateni, M. Hajiaghayi, and D. Marx. Approximation schemes for steiner forest on planar graphs and graphs of bounded treewidth. *Journal of the ACM (JACM)*, 58(5):21:1–21:37, 2011.

[6] H. L Bodlaender. Dynamic programming on graphs with bounded treewidth. In *The 15th International Colloquium on Automata, Languages and Programming*, ICALP '88, pages 105–118, 1988.

[7] H. L. Bodlaender, M. Cygan, S. Kratsch, and J. Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Information and Computation*, 243(Supplement C):86–111, 2015.

[8] G. Bodwin. Linear size distance preservers. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 600–615, 2017.

[9] G. Bodwin and V. V. Williams. Better distance preservers and additive spanners. In *Proceedings of the Twenty-seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 855–872, 2016.

[10] G. Borradaile, E. D. Demaine, and S. Tazari. Polynomial-time approximation schemes for subset-connectivity problems in bounded-genus graphs. *Algorithmica*, 68(2):287–311, 2014.

[11] G. Borradaile and P. Klein. The two-edge connectivity survivable-network design problem in planar graphs. *ACM Transactions on Algorithms (TALG)*, 12(3):30:1–30:29, 2016.

[12] G. Borradaile, P. Klein, and C. Mathieu. An $o(n \log n)$ approximation scheme for steiner tree in planar graphs. *ACM Transactions on Algorithms (TALG)*, 5(3):31:1–31:31, 2009.

[13] G. Borradaile, H. Le, and C. Wulff-Nilsen. Greedy spanners are optimal in doubling metrics. *arXiv preprint arXiv:1712.05007*, 2017.

[14] G. Borradaile, H. Le, and C. Wulff-Nilsen. Minor-free graphs have light spanners. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science*, FOCS '17, pages 767–778, 2017.

[15] S. Chechik and C. Wulff-Nilsen. Near-optimal light spanners. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'16, pages 883–892, 2016.

[16] Y. K. Cheung, G. Goranci, and M. Henzinger. Graph minors for preserving terminal distances approximately - lower and upper bounds. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 131:1–131:14, 2016.

[17] D. Coppersmith and M. Elkin. Sparse sourcewise and pairwise distance preservers. *SIAM Journal on Discrete Mathematics*, 20(2):463–501, 2006.

[18] M. Cygan, F. Grandoni, and T. Kavitha. On pairwise spanners. In *30th International Symposium on Theoretical Aspects of Computer Science (STACS 2013)*, pages 209–220, 2013.

[19] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 150–159. IEEE, 2011.

[20] E. D. Demaine, M. Hajiaghayi, and K. Kawarabayashi. Contraction decomposition in H-minor-free graphs and algorithmic applications. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, STOC ' 11, pages 441–450, 2011.

[21] E. D. Demaine, M. Hajiaghayi, and B. Mohar. Approximation algorithms via contraction decomposition. *Combinatorica*, 30(5):533–552, 2010.

[22] Frederic Dorn, Fedor V. Fomin, and Dimitrios M. Thilikos. Catalan structures and dynamic programming in H-minor-free graphs. *Journal of Computer and System Sciences*, 78(5):1606–1622, 2012.

[23] M. Elkin and D. Peleg. $(1+\epsilon, \beta)$-spanner constructions for general graphs. *SIAM Journal on Computing*, 33(3):608–631, 2004.

[24] A. Filtser and S. Solomon. The greedy spanner is existentially optimal. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, PODC '16, pages 9–17, 2016.

[25] F. V. Fomin, D. Lokshtanov, F. Panolan, and S. Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *Journal of the ACM (JACM)*, 63(4):29:1–29:60, 2016.

[26] M. Grigni. Approximate TSP in graphs with forbidden minors. In *Proceedings of the 27th International Colloquium on Automata, Languages and Programming*, ICALP '00, pages 869–877, 2000.

[27] M. Grigni and P. Sissokho. Light spanners and approximate TSP in weighted graphs with forbidden minors. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '02, pages 852–857, 2002.

[28] M. Held and R. M. Karp. The traveling-salesman problem and minimum spanning trees: Part II. *Mathematical Programming*, 1(1):6–25, 1971.

[29] T. Kavitha. New pairwise spanners. *Theory of Computing Systems*, 61(4):1011–1036, 2017.

[30] T. Kavitha and N. M. Varma. Small stretch pairwise spanners. In *Proceedings of the 40th International Conference on Automata, Languages, and Programming*, ICALP'13, pages 601–612, 2013.

[31] K. Kawarabayashi, P. N. Klein, and C. Sommer. Linear-space approximate distance oracles for planar, bounded-genus and minor-free graphs. In *Proceedings of the 38th International Colloquim Conference on Automata, Languages and Programming*, ICALP '11, pages 135–146, 2011.

[32] P. N. Klein. A linear-time approximation scheme for planar weighted TSP. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '05, pages 647–657, 2005.

[33] P. N. Klein. Subset spanner for planar graphs, with application to subset TSP. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, STOC '06, pages 749–756, 2006.

[34] P. N. Klein and D. Marx. A subexponential parameterized algorithm for subset tsp. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 1812–1830, 2014.

[35] R. Krauthgamer, H. L. Nguyễn, and T. Zondiner. Preserving terminal distances using minors. *SIAM Journal on Discrete Mathematics*, 28(1):127–141, 2014.

[36] D. Marx, M. Pilipczuk, and M. Pilipczuk. On subexponential parameterized algorithms for steiner tree and directed subset tsp on planar graphs. *arXiv preprint arXiv:1707.02190*, 2017.

[37] D. Peleg and A. A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

[38] N. Robertson and P. D. Seymour. Graph minors. XVI. Excluding a non-planar graph. *Journal of Combinatoral Theory Series B*, 89(1):43–76, 2003.

[39] S. Tazari and M. Müller-Hannemann. Shortest paths in linear time on minor-closed graph classes, with an application to steiner tree approximation. *Discrete Applied Mathematics*, 157(4):673–684, 2009.