Polynomial Time Approximation Schemes for Dense and Geometric k-Restricted Forest Problems

Mathias Hauptmann* Sören Kühl‡
Richard Schmied† Claus Viehmann§

Abstract

The k-Steiner Forest Problem asks for a minimum cost forest F for a given terminal set S such that F consists of at most k connected components. We construct polynomial time approximation schemes for dense and geometric versions of the k-Steiner Forest Problem and the k-Tree Cover Problem.

1 Introduction

The k-Steiner Forest Problem asks for a minimum-cost forest F that connects (covers) a given set of terminals S in a metric space (V, c) such that F consists of at most k connected components. This problem was defined by R. Ravi who observed that the primal-dual approach yields a 2-approximation algorithm for this problem ([R94]). In 2003, Even et al. considered the k-Tree Cover Problem where a set of terminals is to be covered by at most k not necessarily disjoint trees such as to minimize the maximum tree-cost ([EGKRS04]). They obtain a $(4 + \epsilon)$-approximation scheme for the general metric case of this problem.

In this paper we construct polynomial time approximation schemes for two special cases of these problems, namely ϵ-dense instances and geometric instances in constant dimension. Furthermore we extend these results to the variant of the k-Tree Cover Problem where the trees have to be pairwise node-disjoint (Disjoint k-Tree Cover Problem).

*Dept. of Computer Science, University of Bonn. Email:hauptman@cs.uni-bonn.de

†Dept. of Computer Science, University of Bonn. Work supported by Hausdorff PhD grant. Email:schmied@cs.uni-bonn.de

‡Dept. of Computer Science, University of Bonn. Email:kuehl@cs.uni-bonn.de

§Dept. of Computer Science, University of Bonn. Work partially supported by Bonn International Graduate School in Mathematics. Email:viehmann@cs.uni-bonn.de

Our approximation schemes for the geometric instances are based on an adaption of techniques from [A98] combined with a sophisticated choice of subproblems which are then solved in a dynamic programming approach. First, the instance is decomposed into disjoint subproblems to which the perturbation techniques from [A98] and [ARR98] become applicable.

The ϵ-Dense Steiner Tree Problem was defined by Karpinski and Zelikovsky ([KZ98]) who showed that for every fixed ϵ, there exists a polynomial time approximation scheme for this problem with density parameter ϵ. An instance of the Steiner Tree Problem is called ϵ-Dense if the underlying metric space is induced by a graph $G = (V, E)$ (with all edges of weight 1), such that each terminal has at least $\epsilon \cdot |V \setminus S|$ neighbours in $V \setminus S$.

In [H07], the results of Karpinski and Zelikovsky were extended to give an efficient polynomial time approximation scheme (i.e. the running time is $f(1/\epsilon) \cdot p(n)$ where $p(n)$ is a polynomial of fixed degree not depending on ϵ). Furthermore, the density notion was relaxed, and polytime approximation schemes were also constructed for several other Steiner problems including Group Steiner Tree and Price Collecting Steiner Tree (cf. [H04a], [H04b], [H07]). Our approximation schemes for the ϵ-dense version of the k-Steiner Forest, k-Tree Cover and Disjoint k-Tree Cover Problem combine the methods from [KZ98] and [H07] with a careful case analysis depending on the parameter k. Our results are listed in the table below.

Problem Definitions	General Case	ϵ-Dense Case	Geometric Case
k-Steiner Forest	A.R. 2 [R94]	PTAS	PTAS
k-Tree Cover	A.R. 4 [EGKRS04]	PTAS for $k \in \omega(\log \lvert S \rvert) \cap o(\lvert S \rvert)$	PTAS
disjoint k-Tree Cover	?	PTAS for $k \in \omega(\sqrt{\lvert S \rvert}) \cap o(\lvert S \rvert)$	PTAS

Table 1: Overview of our results.

The rest of the paper is organized as follows: In Section 2 we construct a ptas for the ϵ-Dense k-Steiner Forest Problem. Section 3 deals with the ϵ-Dense k-Tree Cover Problem. In section 4 a polynomial-time approximation scheme for geometric instances of the k-Steiner Forest Problem is given. Finally, Section 5 describes the polynomial time approximation schemes for the k-Tree Cover and the disjoint k-Tree Cover Problem.

2 A PTAS for the ϵ-Dense Case

Consider an instance of the ϵ-Dense k-Steiner Forest Problem consisting of graph $G = (V, E)$, terminal set $S \subseteq V$ and an integer number $k \in \{1, \ldots, |S|\}$. For a given node $v \in V$ and a subset of vertices $U \subseteq V$, let $N_U(v)$ denote the set of neighbours of v in U. By definition of ϵ-density, for each terminal $s \in S$ we have $|N_{V \setminus S}(s)| \geq \epsilon \cdot |V \setminus S|$. Hence there exists at least one node $v \in V \setminus S$ with $|N_S(v)| \geq \epsilon \cdot |S|$.

The approximation schemes for the ϵ-Dense Steiner Problems constructed in [KZ98] and [H07] are based on repeatedly picking and contracting such stars that contain at least a constant fraction of the current terminal set. This greedy phase ends up with a residual instance of sufficiently small size such that it can be solved to optimality using an exact algorithm for the Steiner Tree Problem.

It turns out [H07] that in order to preserve density in the greedy phase, these greedy picks have to be made pairwise disjoint. This gives a residual instance of logarithmic size (number of terminals) which can be solved to optimality using the Dreyfus-Wagner algorithm ([DW71]).

In this section we will construct a ptas for the ϵ-Dense k-Steiner Forest Problem. This is based on a careful case analysis depending on k: If $k = n - O(1)$ then we can solve the problem to optimality. If $k = O(\log(n))$ then we can justify sufficiently many greedy picks such that the residual instance is of logarithmic size and can be solved to optimality. For this purpose we construct an exact algorithm k-St-DW for the k-Steiner Forest Problem which runs in time polynomial in the number of non-terminals and linearly-exponential in the number of terminals. In the remaining case we will argue that a set of greedily picked stars will already give a good approximation.

The section is organized as follows. In subsection 2.1 we describe the exact algorithm for the k-Steiner Forest Problem which we denote as k-St-DW. In subsection 2.2 we describe the greedy phase of our algorithm where we consider various distinct cases concerning the value of k.

2.1 An Exact Algorithm

Let St-DW denote the Dreyfus-Wagner algorithm [DW71]. Given an instance $G = (V, E), S \subseteq V$ of the Steiner Tree Problem in graphs, Dreyfus-Wagner algorithm computes a minimum-length Steiner tree $T = \text{St-DW}(S, G)$ for terminal set S in G. Based on this algorithm we will now construct an exact algorithm k-St-DW for the k-Steiner Forest Problem. Algorithm k-

St-DW is best described in terms of the following recursion formula:

$$k\text{-St-DW}(S,k) = \min_{\emptyset \subset S' \subset S} (k\text{-St-DW}(S',1) + k\text{-St-DW}(S \setminus S', k-1))$$

Note that k-St-DW$(S',1) =$ St-DW(S). However, if one would implement the algorithm in this way, the running time would be of order $O(2^{|S|} \cdot (3^{|S|}|V| + 2^{|S|}|V|^2 + |V|^3) + (k-1)3^{|S|})$.

Instead we implement the algorithm k-St-DW by combining the recursion formulas of k-St-DW and St-DW in order to build up just one single dynamic programming table. The algorithm is given below.

Algorithm k-St-DW
Input: $G = (V, E)$, $c: E \to \mathbb{R}_+$, $S \subseteq V$, $k \in \{1, \ldots, |S|\}$
Output: minimal cost $c(T)$ for a forest $T = (V', E')$
with $S \subseteq V'$, $E' \subseteq E$ consisting of at most k trees.

Compute $dist_{(G,c)}(x,y)$ for all $x, y \in V$
Set $p(\{x,y\},1) := dist_{(G,c)}$ for all $x, y \in V$
For $i := 2$ **to** $|S| - k + 1$
  For all $U \subseteq S$ with $|U| = i$ and all $x \in V \setminus U$
    set $q(U \cup \{x\}, x) := \min_{\emptyset \neq U' \subset U}(p(U' \cup \{x\}, 1) + p((U \setminus U') \cup \{x\}, 1))$
  For all $U \subseteq S$ with $|U| = i$ and all $x \in V \setminus U$
    set $p(U \cup \{x\}, 1) := \min\{\min_{y \in U}(p(U,1) + dist_{(G,c)}(x,y)),$
      $\min_{y \in V \setminus U}(q(U \cup \{y\}, y) + dist_{(G,c)}(x,y))\}$
For $i = 2$ **to** k
  For all $U \subseteq S$ with $U \neq \emptyset$
    set $p(U,i) = \min_{\emptyset \subset U' \subset U,\, |U'| \leq |S|-k+1}(p(U',1) + p(U \setminus U', i-1))$
Return $p(S,k)$

Lemma 2.1. *The running time of algorithm k-St-DW is* $O(3^{|S|}|V| + 2^{|S|}|V|^2 + |V|^3 + (k-1)3^{|S|})$.

2.2 The Greedy Contraction Phase

Assume that there are no edges connecting two terminals, since otherwise, if there are g such edges with $g > 0$, then we collect $\min\{g, |S| - k\}$ of these edges and consider the remaining instance with $|S| - \min\{g, |S| - k\}$ terminals and the same value k.

Case 1: $k \geq |S| - c$ for a constant c (the choice of c will depend on the approximation parameter δ). In this case we solve the problem to optimality by a brute-force approach: We enumerate all subsets of terminals of size up to $2 \cdot c$ and for each of them compute an optimum Steiner forest with at most c components. The best of all these solutions together with the remaining terminals as one-element components is an optimum solution for the k-Steiner Forest Problem. This brute-force algorithm can be implemented to have time complexity $O(|S|^{2c} \cdot |V|^{2c+3})$.

Case 2: k is sufficiently small, namely $k \leq c_2 \cdot \log(|S|)$. Then we can iteratively pick stars ST_i, each consisting of a non-terminal and all its terminal neighbours. The first such star ST_1 contains at least $\epsilon \cdot |S|$ terminals. If we remove ST_1, the remaining instance is still ϵ-dense. If we iterate this process, after i greedy picks we have collected stars $ST_1, \ldots, ST_i$, and size of the remaining terminal set S_{i+1} is $|S_{i+1}| \leq (1-\epsilon)^i \cdot |S|$. The greedy phase ends after the minimum number j of greedy picks such that $|S_j| \leq c_3$, where c_3 is a constant depending on δ and yet to be specified. Note that in this case, $j = \left\lceil \log\left(\frac{|S|}{c_2}\right) / \log\left(\frac{1}{1-\epsilon}\right) \right\rceil$. Hence the remaining instance contains only $O(\log(|S|))$ terminals and can be solved to optimality by the exact algorithm DW-kSteiner. In order to estimate the resulting approximation ratio, we use the same kind of argument as in [KZ98]. Let F^* be an optimum solution consisting of connected components $F_1^*, \ldots, F_k^*$. Let S_{ST_i} denote the terminal set of star ST_i. If we add edges of a tree M_i spanning S_{ST_i} to the graph G, we obtain a graph G' such that the optimum solution in G' is not more expensive than in G. Furthermore, by adding all edges of trees M_i and possibly removing edges of $F \setminus \bigcup_i M_i$, we may assume that F^* contains the trees M_i as subtrees. Let F denote the forest constructed by the algorithm, let F_0 denote an optimum forest connecting the stars ST_i. Hence the approximation ratio in this case is bounded as follows:

$$\frac{c(F)}{c(F^*)} \leq \frac{c(F_0) + \sum_i c(ST_i)}{c(F_0) + \sum_i (c(ST_i) - 1)} \leq \frac{\sum_i |S_i|}{\sum_i (|S_i| - 1)} \leq \max_i \frac{|S_i}{|S_i| - 1}$$

Hence if we choose c_3 such that $\frac{\epsilon \cdot c_3}{\epsilon \cdot c_3 - 1} \leq 1 + \delta$ then due to the density condition we obtain approximation ratio $1+\delta$. Hence we choose $c_3 = \lceil \frac{1+\delta}{\epsilon \cdot \delta} \rceil$.
Case 3: $c_2 \cdot \log(|S|) < k < |S| - c_1$. In this case, ϵ-density implies existence of a star ST_1 with center $v \in V \setminus S$ containing at least $\epsilon \cdot |S|$ terminals. We take a subgraph of this star collecting $|S|-k+1$ terminals. Since a minimum spanning tree for $|S| - k + 1$ terminals gives a lower bound of $|S| - k$, the resulting approximation ratio in this case is $\frac{|S|-k+1}{|S|-k} = 1 + \frac{1}{|S|-k} \leq 1 + c_1^{-1}$.

Hence if for a given instance $G = (V, E), S, k$ and given $\delta > 0$ we want to guarantee approximation ratio at most $1 + \delta$, it suffices to choose $c_1 = \frac{1}{\delta}$, $c_2 = 1$ and $c_3 = \lceil \frac{1+\delta}{\epsilon \cdot \delta} \rceil$.

We are now ready to formulate our algorithm.

Algorithm Dense-kSTF
Input: $G = (V, E)$, $S \subseteq V$, $k \in \{1, \ldots, |S|\}, \epsilon > 0$
Output: $(1 + \epsilon)$-approx. k-Steiner Forest for S in G
Choose $c_1 = \frac{1}{\delta}$, $c_2 = 1$ and $c_3 = \lceil \frac{1+\delta}{\epsilon \cdot \delta} \rceil$.
Case: $k \geq |S| - c_1$
  solve to optimality by brute force
Case: $k \leq c_2 \cdot \log |S|$
  Phase 1: Greedy Picks
  while $|S| > c_\delta$
    $v :=$argmax$\{|N_S(u)|; u \in V \setminus S\}$
    $S := S \setminus N_S(v)$
    $ST_v :=$ the star consisting of v and $N_S(v)$
    if $|S| < c_\delta$ then
      $N' :=$ the subset of $N_S(v)$ with $|S \cup N'| = c_\delta$
      $S := S \cup N'$
      $ST_v :=$ the star consisting of v and $N_S(v) \setminus N'$
  Phase 2: DW-kSteiner
  for each star ST_v collected in Phase 1
    contract ST_v into s_v
    $S := S \cup \{s_v\}$
  solve the remaining instance using DW-kSteiner, obtain F_0
  return $F_0 \cup \bigcup_{\text{Phase 1}} ST_v$
Case: $c_2 \cdot \log |S| < k < |S| - c_1$
  Take Satisfying Star:
  choose $v \in V \setminus S$ with $|S_v| \geq \epsilon |S|$
  remove edges from ST_v to let ST_v having $|S| - k + 1$ terminals
  return ST_v

This yields the following theorem.

Theorem 2.1. *For each $\epsilon > 0$, the algorithm Dense-kSTF is a PTAS for the ϵ-Dense k-Steiner Forest Problem.*

3 The ϵ-Dense k-Tree Cover Problem

We consider the following two versions of the ϵ-Dense k-Tree Cover Problem.

- Version 1: *(Non-Disjoint Version)*
  Given an instance $G = (V, E), S \subseteq V, k \in \{1, \ldots, |S|\}$, construct a set of at most k trees T_i in G (not necessarily vertex- or edge-disjoint) covering the terminal set S such as to minimize $\max_i c(T_i)$.

- Version 2: *(Disjoint Version)*
  Given an instance $G = (V, E), S \subseteq V, k \in \{1, \ldots, |S|\}$, construct a forest F in G of at most k connected components F_i covering S such as to minimize $\max_i c(F_i)$.

3.1 Non-Disjoint Version

Consider an instance of the ϵ-dense k-Tree Cover Problem (version 1) consisting of a graph $G = (V, E)$, a terminal set $S \subseteq V$ and an integer $k \in [1, |S|]$. First observe that a lower bound for the optimum solution cost is given by $U_{\min} := \frac{|S|}{k} - 1$. Let $\delta > 0$. In order to obtain approximation ratio $1 + \delta$ we will accept trees of size ($\sharp$ edges) at most $U_\delta := (1 + \delta) \cdot U_{\min}$.

We proceed as follows: First we run a greedy phase in which we collect a logarithmic number of stars ST_i (each consisting of a non-terminal center node v_i and a set of terminal leaves $S_i \subseteq S$). The number of remaining terminals will be bounded by some constant $c = c_{\epsilon,\delta}$. Afterwards we split each star ST_i which was constructed in the greedy phase into $\lceil \frac{|S_i|}{U_\delta} \rceil$ stars of cost at most U_δ. The remaining terminals will be declared as singleton trees of cost 0.

Thus it remains to investigate for which values of k this procedure ends up with a total number of at most k trees. Let us first give a precise formulation of our algorithm.

Algorithm Dense-k-Tree-Cover
Input: $G = (V, E)$, $S \subseteq V$, $k \in [1, |S|]$, $\delta > 0$

Initialization:
  $S_{act} := S$, $STARS := \emptyset$
Phase 1: Greedy Picks
  While $|S_{act}| \geq c$ **do**
    $v := \text{argmax}_{u \in V} |N_{S_{act}}(v)|$
    $ST_v :=$ star centered at v with leaves $S_v := N_{S_{act}}(v)$
    $STARS := STARS \cup \{ST_v\}$
    $S_{act} := S_{act} \setminus N_{S_{act}}(v)$
Phase 2: Splitting Phase
  For each $ST_v \in STARS$ **do**
    Split ST_v into $n_v := \lceil \frac{|S_v|}{U_\delta} \rceil$ trees $T_{v,1}, \ldots, T_{v,n_v}$

/$\star$ *Return list of trees covering S, T_s denotes singleton tree with terminal s* $\star$/
Return $\bigcup_{ST_v \in STARS} \{T_{v,1}, \ldots, T_{v,n_v}\} \;\cup\; \bigcup_{s \in S_{act}} \{T_s\}$

Analysis. It remains to give an estimate (upper bound) for the number of trees constructed by algorithm **Dense-k-Tree-Cover** and then compute for which values of k this is always bounded by k.

Let S^i_{act} denote the set S_{act} after i iterations of the while-loop in the greedy phase. Since the instance is ϵ-dense, $|S^i_{act}| \leq (1-\epsilon)^i|S|$. The greedy phase stops when $|S_{act}| \leq c$. Hence the number of iterations of the while loop is at most $\log(\frac{|S|}{c})/\log(\frac{1}{1-\epsilon})$. Let $ST_1, \ldots, ST_r$ be the stars constructed in the greedy phase of algorithm Dense-k-Tree-Cover. Let $ST_i = ST_{v_i}$ with set of terminal leaves S_i. Then the total number of trees constructed by **Dense-k-Tree-Cover** is

$$\begin{aligned} \textstyle\sum_{i=1}^r \lceil \frac{|S_i|}{U_\delta} \rceil + c &\leq \textstyle\sum_{i=1}^r \frac{|S_i|}{U_\delta} + \log_{\frac{1}{1-\epsilon}}(|S|) + c \\ &\leq \left(\frac{|S|-c_\epsilon}{(1+\delta)(|S|-k)} + \frac{\log_{\frac{1}{1-\epsilon}}(|S|)+c_\epsilon}{k} \right) \cdot k \end{aligned}$$

This term is $\leq k$ if $k \in \omega(\log(|S|)) \cap o(|S|)$.
Hence the algorithm Dense-k-Tree-Cover constructs a $(1+\delta)$-approximate solution if k is sub-linear and super-logarithmic in the number of terminals. Altogether we obtain the following result.

Theorem 3.1. *Algorithm* **Dense-k-Tree-Cover** *is a PTAS for the ϵ-Dense Non-Disjoint k-Tree Cover Problem provided $k \in \omega(\log(|S|)) \cap o(|S|)$.*

3.2 Disjoint Version

In the disjoint case (version 2) we can not simply perform greedy picks as before and split stars into several different trees. We proceed as follows. Again we take the lower bound $U_{min} = \frac{|S|}{k} - 1$ for the cost of an optimum solution. Then we iteratively collect stars ST_v centered at some non-terminal v with a set S_v of terminal leaves. However, in each greedy step we collect only $(1+\delta)U_{min}$ terminals from the current star. Since for each of the remaining terminals of the current star, number of non-terminal neighbors is reduced by 1, these *partial greedy-picks* destroy the ϵ-density property.

We perform such greedy steps until the size of the current star ST_v drops below $(1+\delta)U_{min}$ or the number of collected trees together with the number of remainig terminals is $\leq k$.

Greedy-Phase
$S_{act} := S,\ VS_{act} := V, \quad STARS := \emptyset$
For $(i := 1;\ i + |S_{act}| > k;\ i++)$
  $v := \text{argmax}\{|N_{S_{act}(u)}|,\ u \in V \setminus V\}$
  $ST_i :=$ star centered at v with terminal neighbors of v as leaves
  Add ST_i to $STARS$
  $S_{act} := S_{act} \setminus N_{S_{act}(u)}, \quad VS_{act} := VS_{act} \setminus \{v\}$

In order to analyze this procedure, we introduce the following notations.

- $t_i := |S^i_{act}|$ denotes the number of remaining terminals after i iterations in the greedy phase, where $t_0 = |S|$,
- $\epsilon_i > 0$ is such that after i-th iteration the terminal set S^i_{act} satisfies ϵ_i-density condition: $\forall s \in S^i_{act}\ |N_{V\setminus S}(s)| \geq \epsilon_i \cdot |VS^i_{act}|$

Thus in order to obtain an $(1+\delta)$-approximate solution, it suffices to guarantee that for each greedy iteration i conditions (1) and (2) hold.

(1) $i + t_i \leq k$ (2) $\epsilon_i \cdot t_i \geq (1+\delta)U_{min}$
If $n_i := |VS^i_{act}|$ denotes the number of remaining non-terminals after the i-th iteration, then we obtain the following recursions.

$$\begin{array}{ll} (a) & t_i = t_{i-1} - (1+\delta)U_{min} = t_0 - i \cdot (1+\delta)U_{min} \text{ with } t_0 = |S| \\ (b) & n_i = n_{i-1} - 1 = n_0 - i \text{ with } n_0 = |V \setminus S| \\ (c) & \epsilon_i = \dfrac{\epsilon_{i-1} n_{i-1} - 1}{n_{i-1} - 1} > \epsilon_{i-1} - \dfrac{1}{n_{i-1} - 1} > \epsilon_0 - \dfrac{i}{n_i} \end{array}$$

The first condition gives us the following lower bound i_0 on the number of iterations: $i + t_0 - i \cdot (1+\delta)U_{min} \leq k \Leftrightarrow i \geq \frac{t_0 - k}{U_\delta - 1} \Rightarrow i = i_0 = \left\lceil \frac{t_0-k}{U_\delta - 1} \right\rceil$. This lower bound i_0 has to satisfy the second condition:

$$\begin{aligned} U_\delta &\leq \left(\epsilon_0 - \tfrac{i}{n_0 - i}\right)(t_0 - i \cdot U_\delta) \\ &\leq \left(\epsilon_0 - \frac{\frac{t_0-k}{U_\delta-1}}{n_0 - \frac{t_0-k}{U_\delta-1}}\right) \cdot \left(t_0 - \left(\tfrac{t_0-k}{U_\delta-1}\right) \cdot U_\delta\right) \\ &= \left(\epsilon_0 - \frac{\frac{t_0-k}{U_\delta-1}}{\frac{n_0(U_\delta-1)-t_0+k}{U_\delta-1}}\right)\left(\frac{t_0U_\delta - t_0 - t_0U_\delta + k(1+\delta)(\frac{t_0}{k}-1)}{U_\delta-1}\right) \\ &= \left(\epsilon_0 - \frac{t_0-k}{n_0(U_\delta-1)-t_0+k}\right)\left(\frac{(1+\delta)t_0 - t_0 - k(1+\delta)}{U_\delta-1}\right) \\ &= \left(\epsilon_0 - \frac{t_0-k}{n_0(U_\delta-1)-t_0+k}\right)\left(\frac{\delta t_0 - k(1+\delta)}{U_\delta(U_\delta-1)}\right) U_\delta \end{aligned}$$

This term is of order $(\epsilon_0 - o(1))(\omega(1) - 1)U_\delta$ if we restrict k to be in $\omega(\sqrt{t_0}) \cap o(t_0)$ and $n_0 \in \Omega(t_0)$. Hence we obtain:

Theorem 3.2. *There is a PTAS for the special case of the disjoint version of the ϵ-Dense k-Tree Cover Problem when $k \in \omega(\sqrt{|S|}) \cap o(|S|)$ and $|V \setminus S| \in \Omega(|S|)$.*

4 Geometric k-Steiner Forest Problem

In order to obtain a PTAS for the Geometric k-Steiner Forest Problem we combine methods from [A98] and [ARR98] with a sophisticated definition of subproblems in the dynamic-programming scheme. For the sake of completeness, we first give a brief description of the rounding and decomposition procedure from [ARR98]. We will only describe the two-dimensional case with L_p-metric for a fixed p - all the methods described in this and the next section can be generalized to the d-dimensional case where d is constant.

4.1 Decomposition and Perturbation

(a) Compute a 2-approximate solution to the k-Steiner Forest Problem, let its length be D. Let OPT denote the optimum value, then $\frac{D}{2} \leq OPT \leq D$.
(b) Decompose the problem into problems with disjoint bounding boxes the length of wich is at most polynomially longer than the length of a solution.

For this purpose, we take a grid of granularity $\frac{D}{c \cdot p(|S|)}$ for a polynomial $p(\cdot)$ yet to be specified. Move each point to its nearest grid point which changes the optimum by at most $|S| \cdot \frac{D}{c \cdot p(|S|)}$. Choose $p(\cdot)$ such that this length increase becomes small. Now the minimum nonzero distance between terminals is at least $\frac{D}{c \cdot p(|S|)}$.
Rescale such that each point coordinate is integer and the minimum internode distance is 8. Let L denote the new integral bounding box length and D the rescaled 2-approximate upper bound.

Case 1: $D \geq L/n^2$. Then after perturbation and rescaling, the size of quadtrees associated to the bounding box of the instance will be polynomial in n such that the instance of the k-Steiner Forest Problem can be solved using shifted quadtrees and the dynamic programming approach which is described in the next subsection.

Case 2: $D < L/n^2$. In this case we use the methods of [ARR98] in order to decompose the problem into pairwise disjoint subproblems that have a sufficiently small bounding box length. It is shown in [ARR98] that if a shift $(a, b) \in [0, L]^2$ is taken uniformly at random, with probability at least $1 - \frac{1}{\Omega(\log n)}$ no edge of the optimum solution crosses the boundary of any square in the shifted quadtree of size $D \cdot \log(n)$. We build a binary tree whose leaves are the quadtree nodes of size $D \cdot \log(n)$ in the shifted quadtree $Q_{a,b}$. For each such node, using the algorithm described in the next subsection, we solve each instance of the k-Steiner Tree Problem consisting of the set of terminals that are inside q and value $k' \in \{1, \dots, k\}$. Then we use a dynamic programming approach to combine these solutions bottom-up along the binary tree (cf. [ARR98]).

4.2 Structure Theorem and Dynamic Programming

Hence we may now assume that the instance consists of a terminal set $S \subset \mathbb{R}^2$ of cardinality $|S| = n$ within a bounding box of length L and a number $k \in \{1, \dots, n\}$, that all coordinates of terminals are integral, the minimum inter-terminal distance is at least 8 and such that $L \leq n^c$ for some constant $c > 1$. We may furthermore assume that $S \subset B := [0, L] \times [0, L]$ Let Q denote a quadtree for B, and let $Q_{a,b}$ the shifted quadtree with shift (a, b)

On the boundary of each square q in the quadtree we place m portals equally-spaced. A Steiner forest is called (m, r)-light if it crosses each quadtree node at most r times, and each crossing happens at a portal [A98].

Theorem 4.1. *(Structure Theorem)[A98]*
Let L denote the bounding box length of the instance. If shifts a, b are taken uniformly at random, then for $m = c \cdot \log(L)$ and $r = O(c)$, with probability at least … an optimum (m, r)-light solution with respect to the shifted quadtree $Q = Q_{a,b}$ is an $\left(1 + \frac{1}{c}\right)$-approximate solution to the k-Steiner Forest Problem.

This theorem indeed holds for the *k-Steiner Forest Problem*: Given an optimum solution F^* one can transform it into a (m, r)-light forest F' (cf. [A98] , using the *Patching Lemma*). We note that the number of components does not increase by applying the Patching Lemma.

Subproblems. A subproblem consists of a node q in the quadtree, a subset of portals P of size at most r, a partition of P into nonempty subsets $P_1, \dots P_l$ and a number $k' \in \{1, \dots, k\}$. A solution is a (m, r)-light forest F inside q that contains all the terminals in q and such that each P_j is connected by a different tree in F and for each connected component F_i of F, either F_i does not touch the boundary of q or its intersection with the boundary is one of the sets P_i. Furthermore, the number of connected components of F that are completely inside q is equal to k'.

Dynamic Programming. Obviously, the k-Steiner Forest Problem is a special case of this subproblem with $q =$ the root of the quadtree and $P = \emptyset, k' = k$. We are now ready to describe how subproblems can be solved by dynamic programming. Given an instance $q, P = P_1 \cup \ldots \cup P_l, k'$ of the subproblem as defined above, if q is a leaf then it contains only a constant number of terminals and the subproblem can be solved to optimality by brute-force. Otherwise, let $q_1, \ldots q_4$ denote the children of q in the quadtree Q. In order to solve the instance $q, P = P_1 \cup \ldots \cup P_l, k'$, the dynamic programming scheme tries out all combinations of solutions of subproblem instances $q_i, P^{(i)} = P_1^{(i)} \cup \ldots \cup P_{j_i}^{(i)}, k_i'$ associated to the children q_i for which the sets and paritions of portals are consistent and furthermore, k' is equal to $k_1' + \ldots + k_4' +$ the number of connected components that cross the boundary of some of the q_i but not the boundary of q. Note that this is completely determined and can be checked based only on the subproblem instances (and not their solutions).

The number of subproblem instances per quadtree node is at most $O(m^r \cdot r! \cdot k)$. Hence the total dynamic programming scheme can be implemented to run in time $O(n^{c_\epsilon} \cdot m^r \cdot r! \cdot k) = O(n^{c_\epsilon} \cdot (\log n)^{1/\epsilon} \cdot (1/\epsilon)!)$.

Altogether, we have shown:

Theorem 4.2. *There is a PTAS for the Geometric k-Steiner Forest Problem, more precisely: For each $d \in \mathbb{N}$ and $p \in \mathbb{N}_0$, the k-Steiner Forest Problem for terminal sets in $\mathbb{R}^d$ with the L_p-metric provides a polynomial-time approximation scheme.*

5 The Geometric k-Tree Cover Problem

In the geometric case, the k-Tree Cover Problem and the Disjoint k-Tree Cover Problem can be handled in the same way. Given a collection of trees that are not pairwise vertex-disjoint, one can introduce a sufficiently fine grid and move overlapping trees slightly in order to make them disjoint. Hence in this section we restrict ourselves to consider only the Geometric k-Tree Cover Problem.

Most of the methods described in the preceeding section can also be applied to the Geometric k-Tree Cover Problem. We use the $(4+\epsilon)$-appoximation algorithm from [EGKRS04] in order to obtain a lower bound D such that $D \leq OPT \leq (4+\epsilon) \cdot D$, where OPT denotes the cost of an optimum solution. Then the instance is decomposed into instances of bounding box length $L \leq D \cdot n^{O(1)}$. (cf. section 4). Well-Rounding and Structure Theorem carry over to the Geometric k-Tree Cover Problem as well.

The remaining task is to define appropriate subproblems associated to nodes q of a shifted quadtree and to construct a dynamic programming algorithm in order to solve these subproblems.

Concerning the construction of *Subproblems*, the following difficulty occurs. When combining solutions to subproblems associated to boxes that are neighbours in the quadtree Q, we have to control the length increase, i.e. it possibly makes a crucial difference if we combine two rather long trees or we combine each of them to shorter trees.

We use methods from [ARR98] and [KR07] in order to handle this problem. We assign *guesses* to the portals of a quadtree node. A *guess* is a predetermined upper bound (budget) for the length of the tree connected to a portal p of a node q "on the other side of the boundary", namely the remaining tree length which we can assign to p within neighbouring quadtree nodes. By using geometric expansions we guarantee that in oder to get a good approximation, the number of different guesses that have to be assigned to a portal is polynomially bounded.

We are now ready to give the precise definition of subproblems.

Subproblem:
Given: a node q in the shifted quadtree with $O(m)$ portals on its boundary, a subset P of at most r portals, a partition of P into nonempty subsets $P_1, \dots, P_{r'}$, a partition of the set $\{1, \dots, r'\}$ into nonempty subsets J_l and for each J_l of size ≥ 2, for each $P_i, i \in J_l$ a portal $p_{l,i} \in P_i$ such that alll these portals $p_{l,i}$ have the same guess which we denote as g_l (interpretation: sets P_i, P_j with $i, j \in J_l$ for some l, then these sets of portals belong to the same tree of the global solution, connected via portals $p_{l,i} \in P_i, p_{l,j} \in P_j$) for each of those portals $p \in P$ a guess g_p of an approximate remaining length of a "Steiner Tree" behind that portal,

$$g_p \in \left\{ (1 + f(\epsilon))^i \;\middle|\; 0 \leq i \leq \frac{2 \cdot \log(L)}{\log(1 + f(\epsilon))} \right\}$$

hence the number of different values to be considered is $O(\log L)$, and a number $k' \in \{0, \dots k\}$ (guess for the number of connected components of F that are completely in the interior of q)

Solution: A forest F in q collecting all terminals in q such that each component of F is either in the interior of q or there is some $1 \leq j \leq r'$ such that the intersection of the component with the set of portals is equal to P_j, and each P_j is the set of portals of some component F_j of F

Objective:

$$\text{minimize} \ \max_l \left\{ \sum_{j \in J_l} (c(F_j) + \sum_{p \in P_j} g_p) \ - \ [|J_l| \geq 2] \cdot (|J_l| - 1) \cdot g_l \right\}$$

The number of subproblems per quadtree node with a set of $O(m)$ portals on its boundary is

$$O\left(m^{O(r)} \cdot r! \cdot \left(\frac{2 \cdot \log(L)}{\log(1 + f(\epsilon))} \right)^{O(r)} \right)$$

which is $O((\log n)^{O(1/\epsilon)} \cdot O(\frac{1}{\epsilon}) \cdot (\log(L))^{O(1/\epsilon)})$ provided $m = O(\log(n))$ and $r = O(1/\epsilon)$.

Hence we obtain the following result.

Theorem 5.1. *For each $d \in \mathbb{N}$ and $p \in \mathbb{N}_0$, the k-Tree Cover Problem restricted to terminal sets in $\mathbb{R}^d$ with the L_p-norm $d_p(x,y) = \left(\sum_{i=1}^d |x_i - y_i|^p \right)^{1/p}$ provides a polynomial-time approximation scheme.*

6 Remarks

We think that the running time of our polynomial time approximation schemes for the geometric instances (cf. Section 4 and 5)can be improved by the techniques of Kolliopoulos and Rao ([KR07]).

References

[A98] S. Arora, *Polynomial Time Approximation Schemes for Euclidean TSP and Other Geometric Problems*, J. of the ACM 45, pp. 753-782, 1998.

[ARR98] S. Arora, P. Raghavan and S. Rao, *Approximation Schemes for Euclidean k-Medians and Related Problems*, Proc. 30th STOC 1998, pp. 106-113.

[DW71] S. Dreyfus and R. Wagner, *The Steiner Problem in Graphs*, Networks 1, pp. 195-207, 1971.

[EGKRS04] G. Even, N. Garg, J. Könemann, R. Ravi and A. Sinha, *Min-Max Tree Covers of Graphs*, Oper. Res. Lett. 32, pp. 309-315, 2004; preliminary version in Proc. APPROX 2003 with the title *Covering Graphs using Trees and Stars*, pp. 24-35.

[FKW07] B. Fuchs, W. Kern and X. Wang, *Speeding up the Dreyfus-Wagner Algorithm for Minimum Steiner Trees*, Math. Meth. of Operations Research 66, pp. 117-125, 2007.

[GW95] M. Goemans and D. Williamson, *A General Approximation Technique for Constrained Forest Problems*, SIAM J. Comput. 24, pp. 296-317, 1995.

[H04a] M. Hauptmann, *Approximation Complexity of Optimization Problems: Structural Foundations and Steiner Tree Problems*, PhD-Thesis, University of Bonn, 2004, URL: `http://hss.ulb.uni-bonn.de/diss_online/math_nat_fak/2004/hauptmann_mathias/0380.pdf`.

[H04b] M. Hauptmann, *PTAS for Dense Steiner Tree Problems*, Oberwolfach Reports, No.28/2004, pp. 42-45, 2004.

[H07] M. Hauptmann, *On the Approximability of Dense Steiner Tree Problems*, in Proc. 10th ICTCS, p. 15-26, 2007.

[KZ98] M. Karpinski and A. Zelikovsky, *Approximating Dense Cases of Covering Problems*, DIMACS series in Disc. Math. and Theor. Comp. Sci. 40, pp. 169-178, 1998.

[KR95] P. Klein and R. Ravi, *A Nearly Best-Possible Approximation Algorithm for Node-Weighted Steiner Trees*, J. Algorithms 19, pp. 104-115, 1995.

[KR07] S. Kolliopoulos and S. Rao, *A Nearly Linear-time Approximation Scheme for the Euclidean k-Median Problem*, SIAM J. Computing 37, pp. 757-782, 2007.

[R94] R. Ravi, em A Primal-Dual Approximation Algorithm for the Steiner Forest Problem, Inf. Process. Lett. 50, pp. 185-190, 1994.

[RZ00] G. Robins and A. Zelikovsky, *Improved Steiner Tree Approximation in Graphs*, Proc. 11th SODA 2000, pp. 770-779.

Appendix

Exact Algorithm for the k-Tree Cover Problem
In section 2.1 we presented an exact algorithm for the k-Steiner Forest Problem with time complexity polynomial in the number of vertices and exponential in the number of terminals. A similar algorithm can also be obtained for the k-Tree Cover Problem.

It suffices to give an associated recursive formula:

$$p(U,k) = \min_{\emptyset \subset U' \subset U} \max\{p(U',1), p(U \setminus U', k-1)\}.$$

So by replacing the body of the last for-loop in algorithm k-St-DW by

(3) set $p(U,i) = \min_{\emptyset \subset U' \subset U \wedge |U'| \leq |S|-k+1} \max\{p(U',1), p(U \setminus U', k-1)\}$

we obtain an exact algorithm DW-kTCP for the k-Tree Cover Problem.

Lemma 6.1. *The time complexity of algorithm DW-kNFP is* $O(3^{|S|}|V| + 2^{|S|}|V|^2 + |V|^3 + (k-1)3^{|S|})$.