Shallow-Light Steiner Arborescences with Vertex Delays

Stephan Held and Daniel Rotter

Research Institute for Discrete Mathematics, University of Bonn
held@or.uni-bonn.de,rotter@or.uni-bonn.de

Abstract. We consider the problem of constructing a Steiner arborescence broadcasting a signal from a root r to a set T of sinks in a metric space, with out-degrees of Steiner vertices restricted to 2. The arborescence must obey delay bounds for each r-t-path ($t \in T$), where the path delay is imposed by its total edge length and its inner vertices.
We want to minimize the total length. Computing such arborescences is a central step in timing optimization of VLSI design where the problem is known as the repeater tree problem [1,5]. We prove that there is no constant factor approximation algorithm unless $P = NP$ and develop a bicriteria approximation algorithm trading off signal speed (shallowness) and total length (lightness). The latter generalizes results of [8,3], which do not consider vertex delays. Finally, we demonstrate that the new algorithm improves existing algorithms on real world VLSI instances.

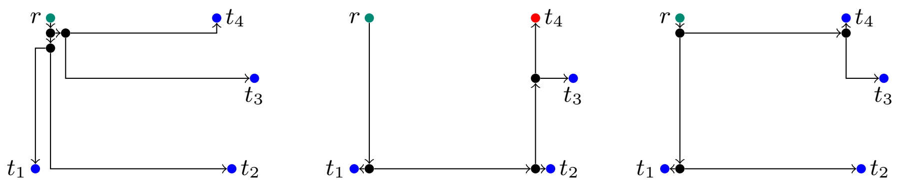
1 Introduction

The input to our problem is a set T of sink vertices and a root r, which are embedded into a metric space (M, dist) by a function $p : T \cup \{r\} \to M$. We are mostly interested in the cases where (M, dist) is the metric closure of a weighted graph or $\mathbb{R}^2$ with the l_1-norm, but our main algorithm works in general.

A solution, which we also call *topology for* $T + r$, consists of an arborescence A rooted at r such that the set of leaves of A is exactly the set T, together with an extension of p to the internal vertices (*Steiner vertices*) in $V(A) \setminus (T \cup \{r\})$. We require that the root r has exactly one outgoing edge and each Steiner vertex has exactly two outgoing edges. This structural restriction is tributed to the delay model we use. By splitting and contracting vertices and orienting edges, a Steiner tree for $T \cup \{r\}$ in (M, dist) can be transformed in linear time into a topology for $T + r$ with the same total edge length and vice versa.

To shorten the notation, we use $A = (A, p)$ to denote an arborescence A together with a placement function $p : V(A) \to M$.
Also, we define $\mathit{dist}(v, w) := \mathit{dist}(p(v), p(w))$ for all vertices $v, w \in V(A)$ associated with placements $p(v), p(w) \in M$.

The topology might be considered as a broadcast network that delivers a signal originating in r to each sink $t \in T$. The cost of a topology A is given by its total edge length $\mathit{cost}(A) := \sum_{e=(v,w)\in E(A)} \mathit{dist}(v, w)$. We assume that there is a constant $b \in \mathbb{R}_+$ that specifies a time penalty for traversing a Steiner vertex,



(a) A shallow/fast topology with high cost.

(b) A light/cheap topology with $delay_A(t_4)$ large.

(c) A light and shallow topology.

Fig. 1: Tradeoff between shallowness and lightness of a topology for r and $T = \{t_1, t_2, t_3, t_4\}$ embedded into $(\mathbb{R}^2, l_1)$.

the time needed for splitting the signal. Following the model in [1], the *delay* of an r-v-path ($v \in V(A)$) in A is given by its length plus b times the number of bifurcations on the path:

$$delay_A(v) := \left(\sum_{e=(v',w') \in E\left(A_{[r,v]}\right)} dist(v', w') \right) + b \cdot (|E(A_{[r,v]})| - 1). \quad (1)$$

Restricting the out-degrees of Steiner vertices to 2 prevents saving vertex delays by using high-fanout Steiner vertices. Each sink $t \in T$ is associated with a delay bound or *required arrival time* $rat(t) \in \mathbb{R}_+$. A topology A meets the required arrival times if its *worst slack* is non-negative:

$$wsl(A) := \min_{t \in T}\{rat(t) - delay_A(t)\} \geq 0.$$

Figure 1 shows examples of topologies.

In the *shallow light Steiner arborescence problem with vertex delays* (SLAP) we wish to compute a topology A for $T + r$ and positions $p(s) \in M$ for each Steiner point s such that $wsl(A) \geq 0$ and $cost(A)$ is as small as possible or to decide that a topology with $wsl(A) \geq 0$ does not exist.

The existence is easy to check. As shown in [1], we can always place all Steiner points at the root location $p(r)$ as indicated in Figure 1(a). This way all paths are shortest possible w.r.t *dist*. For each sink $t \in T$ an upper bound $bif(t)$ for the number of bifurcations on an r-t-path is imposed by $rat(t)$ and $dist(r,t)$, the minimum possible delay for covering the distance:

$$bif(t) := \left\lfloor \frac{rat(t) - dist(r,t)}{b} \right\rfloor \text{ if } b > 0 \text{ and } bif(t) := |T| - 1 \text{ if } b = 0. \quad (2)$$

For the case $b = 0$, where the delays depend only on the distances, a topology with non-negative worst slack exists if and only if $rat(t) \geq dist(r,t)$ for all $t \in T$. Any topology consisting of shortest paths would be feasible.

Otherwise, $b > 0$. Since the subtree rooted at the unique child of r is a binary tree in which each sink $t \in T$ has depth exactly $|E(A_{[r,t]})| - 1$, by Kraft's inequality [9] a topology with non-negative worst slack exists if and only if

$$\sum_{t \in T} 2^{-bif(t)} \leq 1. \tag{3}$$

Such a topology can be computed in $\mathcal{O}(|T| \log |T|)$ time using Huffman-coding [6], which iteratively takes two vertices with maximum *bif*-value and replaces them by a Steiner vertex with position $p(r)$ and a suitable required arrival time.

SLAP in $(\mathbb{R}^2, l_1)$ is known as the *repeater tree problem* [1,5]. In [1] a greedy algorithm was proposed that starts with an empty topology A and adds sinks in non-increasing order of their *bif*-value, subdividing an edge in A for which the resulting topology maximizes the worst slack minus the cost weighted by some adjustment factor. Although it proved to be effective in practice, theoretical results are only known for the cases $rat \equiv \infty$, where it provides a $\frac{3}{2}$-approximation [7], and $dist \equiv 0$, where the worst slack is maximized.

For the special case $b = 0$ the first significant result was given in [2], providing a bicriteria approximation achieving path lengths within $(1+\epsilon) \cdot \max\{dist(r,t) : t \in T\}$ and cost within $1 + \frac{2}{\epsilon}$ times the cost of a minimum spanning tree. By a modification of this algorithm, [8] achieved a length of at most $(1+\epsilon) \cdot dist(r,t)$ for each r-t-path ($t \in T$).
In [3] the cost bound was improved further to $3 + 2 \cdot \lceil \log(\frac{2}{\epsilon}) \rceil$ using Steiner vertices. However, the Steiner points are not embedded into $(M, dist)$ but into an extended metric space, making this improvement less interesting for practical problems. For the Euclidean space $(\mathbb{R}^2, l_2)$ [3] construct instances for which the cost of each topology A with $delay_A(t) \leq (1+\epsilon) \cdot rat(t)$ ($t \in T$) varies from the cost of a minimum spanning tree by a factor of $\Omega(\frac{1}{\epsilon})$ for each $\epsilon > 0$.

Our problem is loosely related to delay or hop constrained tree problems (see [4,12] for recent references), where edge costs are unrelated to their lengths, which makes it more difficult to trade off delays and costs. [13] have proved *NP*-hardness of computing a rectilinear Steiner tree rooted at a vertex r with minimum cost in which all paths are shortest paths. This problem, also known as the *Rectilinear Steiner Arborescence Problem*, has a 2-factor approximation algorithm (see [11]). [10] have shown that the hop constrained tree problem in graphs cannot be approximated within a constant factor unless $P = NP$. But the proof uses non-metric edge weights violating the triangle inequality and does not bound out-degrees, so that it does not apply to our problem.

In Section 2 we prove that there is no constant factor approximation algorithm for SLAP unless $P = NP$. Then, in Section 3 we develop a new bicriteria algorithm for SLAP, generalizing algorithms from [8,3] for $b = 0$. For $b = 0$ and $(M, dist) = (\mathbb{R}^2, l_1)$ we adapt the algorithm of [3] so that Steiner vertices are embedded into $(\mathbb{R}^2, l_1)$ obtaining an algorithm which guarantees bounds of $delay_A(t) \leq (1+\epsilon) \cdot rat(t)$ for all $t \in T$ and $cost(A) \leq (2 + \lceil \log(\frac{2}{\epsilon}) \rceil) \cdot cost(A_c)$, where A_c is an initial short topology. Finally, we demonstrate in Section 4 that the new algorithm achieves significant improvements over the industrially employed algorithm from [1] on practical instances from VLSI design.

2 Non-Approximability

Although the question of existence of a feasible solution is easy to answer, it is very hard to find an optimum solution as the next theorem shows.

Theorem 1. *There is no constant factor approximation algorithm for SLAP unless P=NP.*

Proof. Assume, there is an approximation algorithm with approximation ratio $\alpha > 0$. We use this algorithm to decide an *NP*-complete variant of SATISFIABILITY. Let $\mathcal{C}$ be a set of clauses over variables $X = \{x_1, \dots, x_n\}$ where $n = 2^k$ for some $k \in \mathbb{N}$ and each literal appears in at most two clauses. It is *NP*-complete to decide if a set of clauses of this special form is satisfiable (the proof immediately follows from [14]). Furthermore, we may assume that $|\mathcal{C}| \leq 2 \cdot n$.

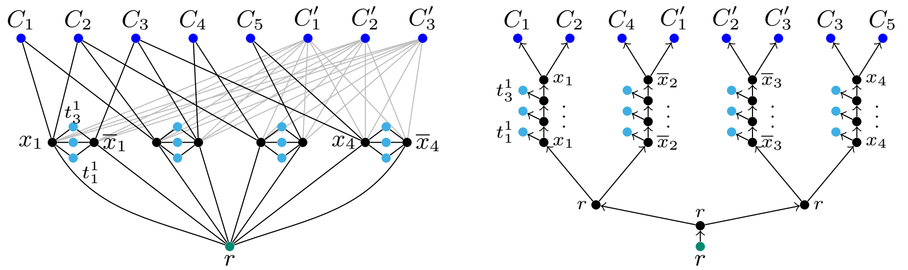
Define $\overline{X} := \{\overline{x}_1, \dots, \overline{x}_n\}$ and let $\mathcal{C}'$ be a set of $2n - |\mathcal{C}|$ elements. We define (M, dist) as the metric closure of an undirected graph G which is defined as follows (see also Figure 2(a)). Let $m := \lceil 6\alpha n - 4n - 2k \rceil + 1$, $\epsilon := \frac{1}{2(\alpha-1)n}$, and $V(G) = \mathcal{C} \cup \mathcal{C}' \cup \{r\} \cup X \cup \overline{X} \cup \{t^i_j : i \in \{1, \dots, n\}, j \in \{1, \dots, m\}\}$. Include edges

- $\{r, \lambda\}$ for all $\lambda \in X \cup \overline{X}$ with length 1,
- $\{\lambda, C\}$ for all $\lambda \in X \cup \overline{X}$, $C \in \mathcal{C}$ such that $\lambda \in C$ with length 1,
- $\{\lambda, C'\}$ for all $\lambda \in X \cup \overline{X}$, $C' \in \mathcal{C}'$ with length 1,
- $\{\lambda, t^i_j\}$ for all $\lambda \in X \cup \overline{X}$ s.t. $\lambda = x_i$ or $\lambda = \overline{x}_i$, $j \in \{1, \dots, m\}$ with length ϵ.

Let $T := \mathcal{C} \cup \mathcal{C}' \cup \{t^i_j : i \in \{1, \dots, n\}, j \in \{1, \dots, m\}\}$, $b = 1$, $p(t) = t$ for all $t \in T \cup \{r\}$, $\mathit{rat}(C) = k + m + 3$ for $C \in \mathcal{C} \cup \mathcal{C}'$, $\mathit{rat}(t^i_j) = 1 + \epsilon + j + k$ for all $i \in \{1, \dots, n\}, j \in \{1, \dots, m\}$. Note that $\sum_{t \in T} 2^{-\mathit{bif}(t)} = 1$ and by (3), a topology with non-negative worst slack for the constructed instance exists. Because equality holds, the number of Steiner vertices on an r-t-path is uniquely determined by $\mathit{bif}(t)$ for every $t \in T$ and a feasible solution cannot have a Steiner vertex simultaneously at $p(x_i)$ and $p(\overline{x}_i)$ for $i = 1, \dots, n$. The following claim proves the theorem.

Claim: If $\mathcal{C}$ is satisfiable, a topology for $T + r$ with non-negative worst slack and cost at most $3n + nm \cdot \epsilon$ exists (see Figure 2(b)). Otherwise, each topology with non-negative worst slack has cost at least $2n + k + (1 + \epsilon n) \cdot m > \alpha \cdot (3n + nm \cdot \epsilon)$.

The idea is to show that in every topology with non-negative worst slack the set $\mathcal{C} \cup \mathcal{C}'$ must be arranged pairwise such that each pair is connected to a common Steiner point s with $|E(A_{[r,s]})| = k + m$. If $\mathcal{C}$ is not satisfiable we have to place one of these Steiner points and all of its predecessors at $p(r)$. The total cost of a topology gets large that way. If $\mathcal{C}$ is satisfiable we find a pairing such that we can place all of these Steiner points and m of its predecessors at a position of a true literal. We obtain a short topology. A detailed proof of the claim can be found in the appendix. □



(a) The graph G which defines $(M, dist)$ in the proof of Theorem 1 ($m = 3$).

(b) Light topology with $wsl = 0$. Labels indicate positions. $x_1 = x_4 =$ true, $x_2 = x_3 =$ false satisfies all clauses.

Fig. 2: Graph G and a light topology with non-negative worst slack in the proof of Theorem 1. The corresponding instance of SATISFIABILITY is $X = \{x_1, x_2, x_3, x_4\}$, $C_1 = \{x_1, x_2\}$, $C_2 = \{x_1, x_2, x_3\}$, $C_3 = \{\overline{x}_1, \overline{x}_2, x_4\}$, $C_4 = \{\overline{x}_2, x_3\}$, $C_5 = \{\overline{x}_3, x_4\}$. For simplicity, we chose $m = 3$.

3 Bicriteria Approximation Algorithms

We have seen there is no algorithm with a constant approximation ratio unless $P{=}NP$. We now relax the constraint that the computed topology should have a non-negative worst slack. Instead, we wish to obtain a bicriteria approximation algorithm, i.e. an algorithm that computes a topology A such that $cost(A)$ is at most a factor of β away from optimum while $delay_A(t) \leq \alpha \cdot rat(t)$ for each sink t and constants $\alpha, \beta \geq 1$.

3.1 An Algorithm for General Metric Spaces

Let $\epsilon > 0$ and let $T + r, p, rat, b$ be an instance of SLAP for which a topology with non-negative worst slack exists. The following algorithm is inspired by [8], which was developed for $b = 0$.

Algorithm 1. Let A_c be any (light) topology for $T + r$. A_c can be obtained from an approximate minimum Steiner tree by directing all edges away from r and applying local transformations such that all degree constraints are met. Let r' be the successor of r in A_c and let $\overleftrightarrow{A_c}$ be the directed graph with vertex set $V(A_c)$ and edge set $E(A_c) \cup \overleftarrow{E}(A_c)$ where $\overleftarrow{E}(A_c) := \{(w, v) : (v, w) \in E(A_c)\}$. Note that $\overleftrightarrow{A_c}$ is Eulerian. The idea is to perform an Eulerian walk in $\overleftrightarrow{A_c} - r$ starting at r'. During the walk we keep track of a branching B and an estimate $d(v)$ on the delay of the r-v path in the final topology for each vertex v. Initially, set $B := A_c - r$ (see Fig. 3(a) for an example) and $d(r') := dist(r, r')$. Throughout the whole algorithm, for vertices $v \in V(B)$ that are not roots ($|\delta^-_B(v)| = 1$) we recursively set $d(v) := d(u) + b + dist(u, v)$, where $(u, v) \in E(B)$. By construction, each forward edge $(v, w) \in E(A_c)$ is visited prior to its backward counterpart

$(w,v) \in \overleftarrow{E}(A_c)$ and when (w,v) is visited, the tour finished visiting vertices in the subtree of A_c rooted at w.

When we visit a forward edge $(v,w) \in E(A_c)$, we do nothing if $w \in V(A_c)\backslash T$. Otherwise, $w \in T$ is a leaf and we check whether

$$d(w) > (1+\epsilon)\cdot rat(w). \tag{I}$$

If this is the case, we delete the edge (v,w). The sink w becomes a new root of B and we set $d(w) = dist(r,w) + b \cdot bif(w)$ (see Fig. 3(a) and 3(b)).

When we visit a backward edge $(w,v) \in \overleftarrow{E}(A_c)$, we check whether it is better to merge the (current) subtree of B rooted at v with the connected component of B containing w. More precisely, we check whether

$$d(v) > d(w) + dist(w,v) + b. \tag{II}$$

Note that by the definition of d, this can only be the case if the edge (v,w) is not in B anymore. If condition (II) is true, we

- delete the edge currently entering v (unless v is a root of B),
- subdivide the edge currently entering w by a Steiner vertex placed at $p(w)$ and connect it to w if w is not a root of B,
- create a new Steiner point s placed at $p(w)$, connect it to v and w, and set $d(s) = d(w)$ if w is a root (see Fig. 3(c) for an illustration). The vertex s is the new root of the connected component of B containing v and w.

When we have finished our Eulerian walk, we make sure that $|\delta^+(s)| = 2$ for all $s \in V(B)\backslash T$. If $|\delta^+(s)| + |\delta^-(s)| \leq 1$ for a Steiner point s, we delete it. If $s \in V(B)\backslash T$ has both out-degree and in-degree equal to one, delete it and connect its predecessor with its successor.

Let T' be the set of roots of connected components of B (e.g. boxed vertices in Fig. 3(c)). Note that $r' \in T'$ unless there are no sinks left in the connected component of B containing r' after the Eulerian walk. Set $rat'(t') := d(t') + b$ for $t' \in T'$. Let $bif' : T' \to \mathbb{N}$ be defined analogously to bif in (2).

We have $\sum_{t' \in T'} 2^{-bif'(t')} \leq \frac{1}{2} + \frac{1}{2} \cdot \sum_{t \in T} 2^{-bif(t)} \leq 1$ and hence, a topology A' with non-negative worst slack for the instance $T' + r, p, rat', b$ exists as (3) holds. We do not need to be overly careful in bounding the cost of A' and can place all Steiner vertices at $p(r)$. Thus, we can compute A' by Huffman-coding or the greedy-algorithm from [1] as described in the introduction.

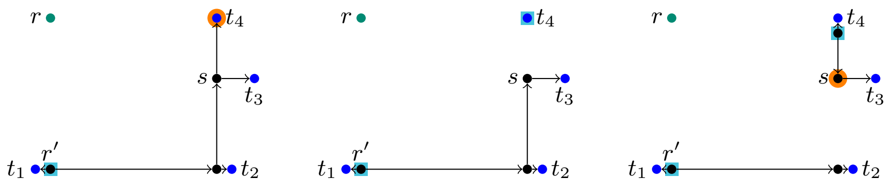
Finally, the algorithm returns $A := A' + B$ (Fig. 1(c) in our example).

Theorem 2. *Let $T + r, p, rat, b$ be an instance of SLAP satisfying (3), $\epsilon > 0$, and A_c a topology for $T + r$. Algorithm 1 computes in $\mathcal{O}(n \log n + \Psi(A_c))$ time a topology A with*

$$wsl(A) \geq -2 \cdot b - \epsilon \cdot \max\{rat(t) : t \in T\} \text{ and} \tag{4}$$

$$cost(A) < \left(1 + \frac{2}{\epsilon}\right) \cdot cost(A_c) + \frac{4b \cdot n}{\epsilon}, \tag{5}$$

where $n := |T|$ and $\Psi(A_c)$ is the time needed to query $dist(v,w)$ for all $(v,w) \in E(A_c)$ and $dist(r,t)$ for all $t \in T$.



(a) Initial branching B, when A_c is given by Fig. 1(b). When visiting $(s, t_4) \in E(A_c)$, we check if $d(t_4) > (1+\epsilon) \cdot \mathit{rat}(t_4)$.

(b) If so, a new connected component of B with root t_4 is created.

(c) When visiting $(t_4, s) \in \overleftarrow{E}(A_c)$, we reconnect s if (II) holds.

Fig. 3: The branching B at different stages of the Eulerian walk. The wide orange circles mark the head vertex of the currently visited edge in $E(\overleftrightarrow{A_c})$. The sky-blue boxes mark roots in B.

Note that $\Psi(A_c) = \mathcal{O}(|E(A_c)|) = \mathcal{O}(n)$ in many applications, e.g. in $(\mathbb{R}^2, l_1)$.

Proof. The algorithm uses only distances of edges in $E(A_c)$ and distances from r to all $t \in T$. Now, the running time follows from the fact that the Eulerian walk takes $\mathcal{O}(n)$ time, including all transformations of B and necessary updates of d. Huffman-coding for constructing A' runs in $\mathcal{O}(n \log n)$ time.

Now we prove inequality (4). Let $t \in T$ be a sink. After the first visit of t, $d(t) \leq (1+\epsilon) \cdot \mathit{rat}(t)$ by (I). Note that $d(t)$ increases only if an edge on the path from the root of its containing connected component and t is subdivided by a Steiner point during its second visit, i.e. after checking (II). Due to the subdivision, this can happen at most once. With $\mathit{rat}'(t') = d(t') + b$ for all $(t' \in T')$, we conclude that $\mathit{delay}_A(t) \leq (1+\epsilon) \cdot \mathit{rat}(t) + 2b$.

For the proof of inequality (5) first note that $\mathit{cost}(B) \leq \mathit{cost}(A_c) - \mathit{dist}(r, r')$ holds at the end of the Eulerian walk. Since $\mathit{cost}(A)$ is equal to the sum of $\mathit{cost}(B)$ and $\mathit{cost}(A')$, it suffices to estimate $\mathit{cost}(A')$. Let $T_1 := \{t_1, \ldots, t_k\}$ be the set of sinks for which condition (I) was true when we traversed the edge entering it ordered by the time they are traversed by the Eulerian walk (i.e. we visited t_i before t_{i+1} for all $1 \leq i \leq k-1$). Let T' be the set of roots of B at the end of the Eulerian walk (as defined in the algorithm). By construction, for each $t' \in T' \backslash \{r'\}$ it holds that $p(t') = p(t_i)$ $(i \in \{1, \ldots, k\})$, where t_i is the unique sink from T_1 in the connected component of B rooted at t'. Hence, $\mathit{cost}(A') = \sum_{t' \in T'} \mathit{dist}(r, t') \leq \mathit{dist}(r, r') + \sum_{i=1}^{k} \mathit{dist}(r, t_i)$.
In the remaining part of the proof we show that

$$\sum_{i=1}^{k} \mathit{dist}(r, t_i) < \frac{2}{\epsilon} \cdot \mathit{cost}(A_c) + \frac{4b \cdot n}{\epsilon}.$$

Define $t_0 := r$ and $d(r) := 0$ at any time of the algorithm. Consider the time when we visit a sink $t_i \in T_1$ ($i \in \{2, \ldots, k\}$) in the Eulerian walk. Let P_i be the $t_{i-1} - t_i$-subtour of the Eulerian walk produced by the algorithm and let $w \in V(P_i)$ be the lowest common ancestor of t_{i-1} and t_i in A_c. Let $V\left(A_{c[w,t_{i-1}]}\right) = \{w_1, w_2, \ldots, w_l\}$, $w_1 = w$, $w_l = t_{i-1}$ (in that order). For $y, z \in V(A_c)\backslash\{r\}$ denote by $d^y(z)$ the value of $d(z)$ at the time of the Eulerian walk just before the edge in $\overleftarrow{E}(A_c)$ leaving y is visited. By convention let $d^r(z)$ denote the value for $d(z)$ at the very beginning of the algorithm.
Due to condition (II), $d^x(x) \leq d^x(y) + dist(x, y) + b \leq d^y(y) + dist(x, y) + b$ for all edges $(x, y) \in E(A_c)$. Consequently, for all $i > 1$ it holds that

$$\begin{aligned} d^w(w) &\leq d^{w_2}(w_2) + dist(w, w_2) + b \\ &\leq d^{w_3}(w_3) + (dist(w, w_2) + dist(w_2, w_3)) + 2b \\ &\leq \ldots \\ &\leq d^{t_{i-1}}(t_{i-1}) + cost(A_{c[w,t_{i-1}]}) + b \cdot |E\left(A_{c[w,t_{i-1}]}\right)|. \end{aligned} \tag{6}$$

Now consider the time when the edge in $E(A_c)$ entering t_i is visited (i.e. the time the algorithm determines that $d(t_i) > (1 + \epsilon) \cdot rat(t_i)$). Let $d^*_A(z)$ denote the value of $d(z)$ at that time for all $z \in V(A_c)$. No edge of $E\left(A_{c[w,t_i]}\right)$ has been deleted since the first traversal of the first edge of $A_{c[w,t_i]}$ by the choice of t_{i-1} and t_i. Hence, $d^*(w) = d^w(w)$ and $d^*(t_i) \leq d^*(w) + cost(A_{c[w,t_i]}) + b \cdot |E\left(A_{c[w,t_i]}\right)|$. Since $t_i \in T_1$, it holds that $d^*(t_i) > (1 + \epsilon) \cdot rat(t_i)$ and with (6):

$$\begin{aligned} (1 + \epsilon) \cdot rat(t_i) &< d^*(t_i) \\ &\leq d^*(w) + cost(A_{c[w,t_i]}) + b \cdot |E\left(A_{c[w,t_i]}\right)| \\ &\leq d^{t_{i-1}}(t_{i-1}) + cost(P_i) + b \cdot |E(P_i)|. \end{aligned}$$

For each $i > 1$, t_{i-1} has become the root of a new connected component of B after we have visited t_{i-1} which implies $d^{t_{i-1}}(t_{i-1}) = dist(r, t_{i-1}) + b \cdot bif(t_{i-1})$. If $i = 1$, the inequality $(1 + \epsilon) \cdot rat(t_i) < d^{t_{i-1}}(t_{i-1}) + cost(P_i) + b \cdot |E(P_i)|$ is trivial. Using $rat(.) \geq dist(r, .) + b \cdot bif(.)$ we obtain

$$(1+\epsilon)(dist(r, t_i) + b \cdot bif(t_i)) < dist(r, t_{i-1}) + b \cdot bif(t_{i-1}) + cost(P_i) + b \cdot |E(P_i)|.$$

Summing up over all $i = 1, \ldots, k$ yields

$$\begin{aligned} &(1 + \epsilon) \sum_{i=1}^{k} dist(r, t_i) + (1 + \epsilon) \sum_{i=1}^{k} b \cdot bif(t_i) \\ &\qquad < \sum_{i=0}^{k-1} dist(r, t_i) + \sum_{i=0}^{k-1} b \cdot bif(t_i) + \sum_{i=1}^{k} (cost(P_i) + b \cdot |E(P_i)|). \end{aligned}$$

Now note that the P_i are pairwise disjoint parts of the Eulerian walk through A_c. We conclude that $\sum_{i=1}^{k} cost(P_i) \leq 2 \cdot cost(A_c)$ and $\sum_{i=1}^{k} |E(P_i)| \leq 2 \cdot |E(A_c)| = 4n - 2$. By combining these inequalities we obtain $\sum_{i=1}^{k} dist(r, t_i) < \frac{2 \cdot cost(A_c)}{\epsilon} + \frac{4b \cdot n}{\epsilon}$ which concludes the proof of Theorem 2. □

Remark 3. If $(M, dist) = (\mathbb{R}^2, l_1)$, we can find in $\mathcal{O}(|T| \cdot \log(|T|))$ time a minimum spanning tree to initialize A_c and by Algorithm 1 a topology A satisfying (4) and

$$cost(A) \leq \frac{3}{2}\left(1 + \frac{2}{\epsilon}\right) cost(SMT) + \frac{4b \cdot |T|}{\epsilon},$$

where SMT is a minimum Steiner tree for $T \cup \{r\}$ and $\frac{3}{2}$ the Steiner ratio [7].

3.2 An Algorithm for $(\mathbb{R}^2, l_1)$ and $b = 0$

If the number of bifurcations of a path does not influence its delay ($b = 0$) and the metric space is $(\mathbb{R}^2, l_1)$, we can prove a much better cost bound.

Theorem 4. *Let $(M, dist) = (\mathbb{R}^2, l_1)$ and let $T + r, p, rat, b$ be an instance of SLAP such that $b = 0$ and $rat(t) \geq ||p(t) - p(r)||_1$ for all $t \in T$. For any topology A_c for $T + r$ and for each $\epsilon > 0$ we can compute a topology A for $T + r$ in $\mathcal{O}(n \cdot \log n)$ time, where $n = |T|$, such that*

$$wsl(A) \geq -\epsilon \cdot \max\{rat(t) : t \in T\} \quad \text{and}$$

$$cost(A) \leq \begin{cases} \left(2 + \left\lceil \log\left(\frac{2}{\epsilon}\right)\right\rceil\right) \cdot cost(A_c) & \text{if } 0 < \epsilon \leq 2 \\ \left(1 + \frac{2}{\epsilon}\right) \cdot cost(A_c) & \text{if } \epsilon > 2. \end{cases}$$

The proofs of Theorem 4 and Lemma 5 are based on Lemma 3.1 of [3]. Since the tree they compute in their Section 2 contains Steiner points not belonging to the metric space they are working in, we use a similar algorithm as [11] to compute a topology with the same properties for the $(\mathbb{R}^2, l_1)$ case:

Algorithm 2. Let F be any Steiner tree for $T + r$. W.l.o.g. we may assume that n is a power of 2 (if this is not the case, add $2^{\lceil \log(n) \rceil} - n$ new sinks placed at $p(r)$). F induces a Hamiltonian cycle through T with cost at most $2 \cdot cost(F)$ which contains a perfect matching M on T with cost at most $cost(F)$.

For each edge $e = \{t, t'\} \in M$ we include a Steiner point s and edges (s, t), (s, t') and place s at the median of r, t, and t':

$$p(s) = (\text{median}(p(r)_x, p(t)_x, p(t')_x), \text{median}(p(r)_y, p(t)_y, p(t')_y)).$$

A shortest r-s path together with (s, t) ((s, t')) forms a shortest r-t-path (r-t'-path). The total cost of edges added in this iteration is equal to the cost of M. The Steiner vertex s with required arrival time $dist(r, s) + b \cdot \min\{bif(t), bif(t')\}$ replaces t, t' in $|T|$.

This way we obtain a new instance with $n/2$ sinks. The subdivided Hamiltonian cycle induces a cycle through the set of new sinks with cost at most $2 \cdot cost(F)$. After $\log(n) - 1$ reductions, there are exactly two sinks t_1, t_2 left. In each iteration we have included edges of cost at most $cost(F)$. Since, by construction, t_1 and t_2 are placed within the bounding box of $T \cup \{r\}$, a minimum cost topology for $r + \{t_1, t_2\}$ provides a feasible topology with cost bounded by $cost(F)$. All paths contained in the topology A for $T + r$ found that way are shortest paths.

Lemma 5. *Let $(M, dist) = (\mathbb{R}^2, l_1)$ and let $T+r, p, rat, b$ be an instance of SLAP such that $b = 0$. Furthermore, let F be a Steiner tree for $T \cup \{r\}$, $\alpha \geq 1$, and $\eta > 0$ with*

$$\alpha \cdot \eta \geq \sum_{t \in T} ||p(r) - p(t)||_1 = \sum_{t \in T} dist_A(r, t). \tag{7}$$

The topology A computed by Algorithm 2 fulfills $cost(A) \leq \eta + \lceil \log(\alpha) \rceil \cdot cost(F)$ and can be computed in $\mathcal{O}(n)$ time, where $n = |T|$.

Proof. As in Algorithm 2 we assume that n is a power of 2. If $\lceil \log(\alpha) \rceil \geq \log(n)+1$, the statement is trivial. Assume $\lceil \log(\alpha) \rceil \leq \log(n)$. For $i \in \{0, \ldots, \log(n)\}$ let E_i denote the set of edges $(v, w) \in E(A)$ for which $|E\left(A_{[r,v]}\right)|$ is equal to $\log(n) - i$. Note that the number of sinks reachable from the endpoint of an edge in E_i is 2^i. Consequently,
$\sum_{t \in T} dist_A(r, t) = \sum_{i=0}^{\log(n)} \sum_{e \in E_i} |\{t \in T : e \in E(A_{[r,t]})\}| \cdot cost(e) = \sum_{i=0}^{\log(n)} 2^i cost(E_i)$.
Thus $\alpha \cdot \eta \geq \sum_{i=\lceil \log(\alpha) \rceil}^{\log(n)} 2^i \cdot cost(E_i) \geq \alpha \cdot \sum_{i=\lceil \log(\alpha) \rceil}^{\log(n)} cost(E_i)$ and hence,
$cost(A) = \sum_{i=0}^{\lceil \log(\alpha) \rceil - 1} cost(E_i) + \sum_{i=\lceil \log(\alpha) \rceil}^{\log(n)} cost(E_i) \leq \lceil \log(\alpha) \rceil \cdot cost(F) + \eta$.
The running time is obvious. □

Proof (of Theorem 4). We use Algorithm 1 and use Lemma 5 to compute A' at the very end. Let A be the output. Theorem 2 implies the claimed properties on the worst slack of A. If $\epsilon > 2$, Theorem 2 implies the claimed cost bound. Let $0 < \epsilon \leq 2$. As seen in the proof of Theorem 2, $\sum_{t \in T'} ||p(r) - p(t)||_1 \leq \frac{2}{\epsilon} \cdot cost(A_c)$. By Lemma 5 ($F := A_c, \alpha := \frac{2}{\epsilon}, \eta := cost(A_c)$), $cost(A') \leq \left\lceil \log\left(\frac{2}{\epsilon}\right) \right\rceil \cdot cost(A_c) + cost(A_c)$. We conclude that the returned topology has cost at most $\left(2 + \left\lceil \log\left(\frac{2}{\epsilon}\right) \right\rceil\right) \cdot cost(A_c)$. The claim about the running time follows from Remark 3 and Theorem 4. □

Remark 6. We can use ideas of Algorithm 2 to improve Huffman-coding in the case $(M, dist) = (\mathbb{R}^2, l_1)$ and $b > 0$. Instead of placing a Steiner vertex replacing sinks t, t' in T at position $p(r)$, we may also place them at the median of r, t, and t'. Whenever the cardinality of the set of sinks with maximum *bif*-value, $T_{\max bif}$, is larger than 2, we can compute a matching in $T_{\max bif}$ with low cost and select the sink pairs according to the matching instead of arbitrary. In order to achieve that $|T_{\max bif}|$ is large, we can compute $H \in \mathbb{N}$ minimal such that $\sum_{t \in T} 2^{-\min\{bif(t), H\}} \leq 1$ and decrease all *bif*-values to $\min\{bif(t), H\}$.

4 Experimental Results

We used Algorithm 1 with improved Huffman-coding as in Remark 6 on instances arising as repeater topology problems in VLSI-design provided by IBM. Here, an electrical signal is distributed from one logic gate to a set of destination gates on a chip (see [1,5] for details).

Excluding trivial instances with one or two sinks, there were 718 379 instances with at least 3 and up to 169 150 sinks. 3 656 of them had more than 100 sinks.

$\|T\|$		$\epsilon = 0$ wsl (in ps)	cost ratio	$\epsilon = 0.1$ wsl (ps)	cost r.	$\epsilon = 0.3$ wsl (ps)	cost r.	$\epsilon = 1.0$ wsl (ps)	cost r.
≤ 100	max	0.000	4.385	0.000	3.393	0.000	2.747	0.000	2.224
	min	-9.726	0.995	-82.281	0.995	-284.963	0.998	-497.640	0.998
	av	-0.141	1.030	-0.330	1.022	-0.795	1.012	-1.567	1.002
	total		1.077		1.049		1.019		1.002
≥ 101	max	0.000	9.229	0.000	3.305	0.000	2.673	0.000	1.888
	min	-9.726	1.000	-164.175	1.000	-556.700	1.000	-1497.640	1.000
	av	-0.149	1.456	-6.772	1.316	-19.511	1.198	-61.107	1.083
	total		1.427		1.218		1.099		1.060
all	max	0.000	9.229	0.000	3.393	0.000	2.747	0.000	2.224
	min	-9.726	0.995	-164.175	0.995	-556.700	0.998	-1497.640	0.998
	av	-0.679	1.032	-0.363	1.023	-0.891	1.013	-1.870	1.003
	total		1.093		1.054		1.013		1.004

wsl: $\min\{0, wsl(A)\} - \min\{0, wsl(A_{\text{huf}})\}$ in picoseconds, cost ratio: $cost(A)/cost(A_c)$.
Table 1: Comparison of the results of Algorithm 1 A with A_c and A_{huf}.

The values for b varied between 4 and 10 picoseconds, depending on the chip-technology. The initial topologies A_c were computed by heuristics guaranteeing minimum Steiner trees for up to eight sinks and $\frac{3}{2}$-approximations otherwise.

In Table 1 we compare the cost of topologies A computed by Algorithm 1 with the cost of A_c and the best possible worst slack attained by A_{huf}, the topology arising from Huffman-coding. In addition, we ran an optimized variant of the greedy algorithm in [1], as it is used at IBM, to obtain a reference topology A_{ref}. This comparison is shown in Table 2.

Running times are negligible for all compared algorithms. Algorithm 1 needs roughly 2 minutes to compute topologies for all 718 379 instances on a 3GHz Xeon machine. More than 90 % of the time was spent on computing A_c.

In both tables, for a topology A generated by Algorithm 1 and a respective reference topology A', the column "cost ratio" shows the maximum, minimum, and average of the ratios $cost(A)/cost(A')$ as well as the ratio of the total costs added up over all instances. The columns "wsl" show the maximum, minimum and average worst slack difference $\min\{0, wsl(A)\} - \min\{0, wsl(A')\}$ in picoseconds. We ran Algorithm 1 with four different values for ϵ (0, 0.1, 0.3, and 1.0). Table 1 shows that for $\epsilon = 0$ we achieve near-feasible solutions at 10% higher total cost compared to the short topologies A_c. With higher values of ϵ the worst slack decreases moderately, except for a few instances, while the costs approach the costs of A_c. As A_c is not minimum, its length can be underpriced by A.

Table 2 shows that the greedy algorithm [1] is not able to bound the worst slack tightly. It looses almost a nanosecond on some instances. In contrast, Algorithm 1 with $\epsilon = 0$ guarantees near-optimum worst slacks and slight improvements w.r.t. average cost and worst slack. The total netlength, which is dominated by a few very large instances, is only 2.3% larger. By increasing ϵ we achieve a large improvement in average and total cost while the average worst slack decreases moderately below the reference worst slack.

$\|T\|$		$\epsilon = 0$		$\epsilon = 0.1$		$\epsilon = 0.3$		$\epsilon = 1.0$	
		wsl (in ps)	cost ratio	wsl (ps)	cost r.	wsl (ps)	cost r.	wsl (ps)	cost r.
≤ 100	max	919.533	2.957	885.573	2.936	783.098	2.048	664.193	1.647
	min	-9.726	0.176	-77.096	0.176	-253.053	0.176	-466.440	0.176
	av	0.002	0.969	-0.188	0.962	-0.653	0.954	-1.425	0.946
	total		1.015		0.986		0.960		0.944
≥ 101	max	358.295	6.136	350.783	2.679	328.639	1.980	219.752	1.426
	min	-9.726	0.202	-164.175	0.190	-477.800	0.190	-1447.180	0.190
	av	2.244	1.114	-2.697	1.018	-15.436	0.938	-57.031	0.859
	total		1.166		0.995		0.929		0.866
all	max	919.533	6.136	885.573	2.936	783.098	2.048	664.193	1.647
	min	-9.726	0.176	-164.175	0.176	-477.800	0.176	-1447.180	0.176
	av	0.013	0.970	-0.201	0.962	-0.728	0.954	-1.708	0.947
	total		1.023		0.987		0.959		0.940

wsl: $\min\{0, wsl(A)\} - \min\{0, wsl(A_{\text{ref}})\}$ in picoseconds, cost ratio: $cost(A)/cost(A_{\text{ref}})$.
Table 2: Comparison of the results of Algorithm 1 A with A_{ref}.

References

1. Bartoschek, C., Held, S., Maßberg, J., Rautenbach, D., Vygen, J.: The Repeater Tree Construction Problem. Information Processing Letters 110, 1079–1083 (2010).
2. Cong, J., Kahng, A.B., Robins, G., Sarrafzadeh, M., Wong, C.K.: Provably good performance-driven global routing. IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems 11 (6), 739–752 (1992).
3. Elkin, M., Solomon, S.: Steiner Shallow-Light Trees are Exponentially Lighter than Spanning Ones. Proc. 52nd FOCS, 373–382 (2011).
4. Gouveia, L., Simonetti, L., Uchoa, E.: Modeling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs. Mathematical Programming A 128 (1-2), 123–148 (2011).
5. Hrkic, M., Lillis, J.: Generalized Buffer Insertion. Handbook of Algorithms for VLSI Physical Design Automation. Alpert, C., Sapatnekar, S., Mehta, D.D., Editors, CRC Press, 557–567 (2007).
6. Huffman, D.A.: A Method for the Construction of Minimum-Redundancy Codes. Proc. of the IRE 40(9), 1098–1101 (1952).
7. Hwang, F.K.: On steiner minimal trees with rectilinear distance. SIAM Journal of Applied Mathematics 30, 104–114 (1976).
8. Khuller, S., Raghavachari, B., Young, N.: Balancing Minimum Spanning Trees and Shortest-Path Trees. Algorithmica 14, 305–321 (1995).
9. Kraft, S.G.: A device for quantizing grouping and coding amplitude modulated pulses. Master's thesis, MIT, Cambridge (1949).
10. Manyem, P., Stallmann, M.: Some Approximation Results in Multicasting. Working Paper, North Carolina State University (1996).
11. Rao, S., Sadayappan, P., Hwang, F.K., Shor, P.: The Rectilinear Steiner Arborescence Problem. Algorithmica 7, 277–288 (1992).
12. Ruthmair, M., Raidl, G.R.: A layered graph model and an adaptive layers framework to solve delay-constrained minimum tree problems. 15th IPCO, 376–388 (2011).
13. Shi, W., Su, C.: The Rectilinear Steiner Arborescence Problem is NP-Complete. Proc. 11th ACM-SIAM symp. on Discrete algorithms, 780–787 (2000).
14. Tovey, C.A.: A Simplified NP-Complete Satisfiability Problem. Discrete Applied Mathematics 8(1), 85–89 (1984).

A Appendix: Details on Claim from Theorem 1

Claim (in proof of Theorem 1). If $\mathcal{C}$ is satisfiable, a topology for $T + r$ with non-negative worst slack and cost at most $3n + nm \cdot \epsilon$ exists (see Figure 2(b)). Otherwise, each topology with non-negative worst slack has cost at least $2n + k + (1 + \epsilon n) \cdot m > \alpha \cdot (3n + nm \cdot \epsilon)$, because at least $m + k$ Steiner points must be located at $p(r)$.

Proof. We prove that $\mathcal{C}$ is satisfiable if and only if there is a topology A for $T+r$ with non-negative worst slack in which no Steiner point s with $|E(A_{[r,s]})| = k+m$ is placed at position r.

First, we assume that $\mathcal{C}$ is satisfiable. Given a satisfying truth assignment we can define a function $\phi' : \mathcal{C} \to X \cup \overline{X}$, $\phi'(C) = \{\lambda \in C : \lambda \text{ is a true literal}\}$. Note that $|\phi'^{-1}(\lambda)| \leq 2$ for each true literal λ.
By choice of $|\mathcal{C}'|$ we can extend ϕ' to a function $\phi : \mathcal{C} \cup \mathcal{C}' \to X \cup \overline{X}$ such that $|\phi^{-1}(\lambda)| = 2$ for each true literal λ. For each such λ place $m + 1$ Steiner points s_j^λ at position λ ($j = 1, \ldots, m + 1$) and include edges $(s_j^\lambda, s_{j+1}^\lambda)$, (s_j^λ, t_j^i) for $j = 1, \ldots, m$ where $\lambda \in \{x_i, \overline{x}_i\}$. Also include edges (s_{m+1}^λ, C) for each $C \in \phi^{-1}(\lambda)$. The resulting branching can be extended to a topology A for $T + r$ by a balanced binary tree for $\{s_1^\lambda : \lambda \text{ is a true literal}\} + r$ in which all inner vertices are positioned at r. Figure 2(b) depicts this.
It is easy to see that $\mathit{wsl}(A) = 0$ and $\mathit{cost}(A) = 3n + nm \cdot \epsilon$. We have $\{s \in V(A) \backslash (T \cup \{r\}) : |E(A_{[r,s]})| = k + m\} = \{s_m^\lambda : \lambda \text{ is a true literal}\}$. None of these vertices is placed at position r.

Conversely, let A be a topology for $T + r$ with $\mathit{wsl}(A) \geq 0$ in which no Steiner point s with $|E(A_{[r,s]})| = k + m$ is placed at position r. If there is a sink t' for which $A_{[r,t']}$ is not a shortest path, $|E(A_{[r,t']})| - 1$ must be at most $\left\lfloor \mathit{rat}(t') - \sum_{e=(v,w)\in E\left(A_{[r,t']}\right)} \mathit{dist}_G(v, w) \right\rfloor < \mathit{bif}(t)$. By (3), we get a contradiction. Hence, all Steiner points must be placed in $\{r\} \cup X \cup \overline{X}$. Since

$$1 \geq \sum_{t\in T} 2^{-\min\{\mathit{bif}(t), |E(A_{[r,t']})|\}} \geq \sum_{t\in T} 2^{-\mathit{bif}(t)} = 1,$$

$\mathit{bif}(t') = |E(A_{[r,t']})|$ for all $t \in T$ which implies that the predecessor of terminal t_m^i ($i \in \{1, \ldots, n\}$) must be a Steiner point s with $|E(A_{[r,s]})| = k + m$ and which reaches a sink in $\mathcal{C} \cup \mathcal{C}'$. Thus, it must be placed in $\{x_i, \overline{x}_i\}$. Its successor must be positioned at the same point. Let

$$S' := \{s \in V(A)\backslash(T \cup \{r\}) : |E(A_{[r,s]})| = k + m + 1\}$$

Since the successors of elements in S' are exactly the sinks in $\mathcal{C} \cup \mathcal{C}'$, $|S'| = |\mathcal{C} \cup \mathcal{C}'|/2 = n$. We conclude that for each $i = 1, \ldots, n$ there is exactly one $\lambda \in \{x_i, \overline{x}_i\}$ such that there is a vertex of S' with position λ.

We use this property to define a truth assignment of $\{x_1, \ldots, x_n\}$ as follows:

$$\text{for each } i = 1, \ldots, n \text{ set } x_i \text{ to } \begin{cases} \text{true} & \text{if there is } s \in S' \text{ such that } p(s) = x_i \\ \text{false} & \text{otherwise.} \end{cases}$$

Let $C \in \mathcal{C}$ and let $s \in S'$ be the predecessor of C in A. Since $A_{[r,c]}$ is a shortest path, $p(s)$ corresponds to a true literal containing C. Thus, $\mathcal{C}$ is satisfiable.

Assume that $\mathcal{C}$ is not satisfiable. Let A be any topology for $T + r$ with non-negative worst slack. As seen before, all paths contained in A are shortest paths. Let l be the number of Steiner points placed at r. All of the $|T| - 1 - l = 2n + mn - 1 - l$ other Steiner points are placed with distance 1 to the root. Since $\mathcal{C}$ is not satisfiable, at least one Steiner point s such that $|E(A_{[r,s]})| = k + m$ and hence all Steiner points from which s is reachable in A must be placed at position r. Thus, $l \geq k + m$. For $v \in V(A) \backslash \{r\}$ let $\mathit{pred}_A(v)$ denote the position of the predecessor of v in A. We have

$$\begin{aligned} \mathit{cost}(A) &= \sum_{v \in V(A)\backslash\{r\}} \mathit{dist}_G(v, \mathit{pred}_A(v)) \\ &\geq \sum_{v \in V(A)\backslash\{r\}} (\mathit{dist}_G(r, v) - \mathit{dist}_G(r, \mathit{pred}_A(v))) \\ &= \sum_{t \in T} \mathit{dist}_G(r, t) - \sum_{s \in V(A)\backslash(T \cup \{r\})} \mathit{dist}_G(r, s) \\ &\geq 4n + (1 + \epsilon) \cdot nm - 2n - mn + 1 + l \\ &> 2n + k + (1 + \epsilon n) \cdot m. \end{aligned}$$

By choice of m and ϵ, $m > 6\alpha n - 4n - 2k \Rightarrow 2n + k + (1 + \epsilon n) \cdot m > \alpha \cdot (3n + nm \cdot \epsilon)$.
□