An Improved Approximation Algorithm for the Uniform Cost-Distance Steiner Tree Problem

Ardalan Khazraei[1] and Stephan Held[2]

[1] Hasso-Plattner-Institute, University of Potsdam, Germany
[2] Research Institute for Discrete Mathematics, University of Bonn, Germany
Ardalan.Khazraei@hpi.de held@dm.uni-bonn.de

Abstract. The cost-distance Steiner tree problem asks for a Steiner tree in a graph that minimizes the total cost plus a weighted sum of path delays from the root to the sinks. We present an improved approximation for the uniform cost-distance Steiner tree problem, where the delay of a path corresponds to the sum of edge costs along that path.
Previous approaches deploy a bicriteria approximation algorithm for the length-bounded variant that does not take the actual delay weights into account. Our algorithm modifies a similar algorithm for the single-sink buy-at-bulk problem by Guha et al. [7], allowing a better approximation factor for our problem. In contrast to the bicriteria algorithms it considers delay weights explicitly. Thereby, we achieve an approximation factor of $(1+\beta)$, where β is the approximation factor for the Steiner tree problem. This improves the previously best known approximation factor for the uniform cost-distance Steiner tree problem from 2.87 to 2.39.
This algorithm can be extended to the problem where the ratio of edge costs to edge delays throughout the graph is bounded from above and below. In particular, this shows that a previous inapproximability result [15] requires large variations between edge delays and costs.
Finally, we present an important application of our new algorithm in chip design. The cost-distance Steiner tree problem occurs as a Lagrangean subproblem when optimizing millions of Steiner trees with mutually depending path length bounds. We show how to quickly approximate a continuous relaxation of this problem with our new algorithm.

1 Introduction

Steiner trees and arborescences that balance the objectives of minimum total length and shortest path lengths in the tree play an important role in the interconnect optimization of computer networks or chips [4, 12, 14, 9, 10, 1, 3]. In fact, the decreasing feature sizes have turned interconnect optimization into a key problem in the physical design of computer chips [1, 3].

In this work, we consider a special case of the *cost-distance Steiner tree problem* that was introduced in [14]. In its general form, we are given a graph G with edge costs $c : E(G) \to \mathbb{R}_{\geq 0}$ and edge delays $d : E(G) \to \mathbb{R}_{\geq 0}$, a root $r \in V(G)$, a terminal set $T \subseteq V(G) \setminus \{r\}$, and delay weights $w : T \to \mathbb{R}_{\geq 0}$. For a tree A rooted at r spanning the terminals $T \cup \{r\} \subseteq V(G)$ we consider the objective function

$$\sum_{e \in E(A)} c(e) + \sum_{t \in T} w(t) \cdot d(E(A_{[r,t]})), \tag{1}$$

where $A_{[r,t]}$ is the unique r-t-path in A and $d(E(A_{[r,t]})) = \sum_{e \in E(A_{[r,t]})} d(e)$. We shall refer to the first term in the objective function as the *connection cost* and second term as the *delay cost*.

In [14] a $\mathcal{O}(\log |T|)$-factor approximation algorithm was presented, and in [15] it was shown that the problem cannot be approximated better than $\Omega(\log \log |T|)$ unless

NP $\subseteq$ DTIME($|T|^{\mathcal{O}(\log\log\log|T|)}$). Here, we consider the *uniform cost-distance Steiner tree problem* that arises if there is a fixed relation $\theta c = d$ for some constant $\theta \in \mathbb{R}_{>0}$ for which our algorithm provides an improved approximation from a previous constant factor. We may assume without loss of generality that the constant θ is equal 1 as it can otherwise be factored into the delay weights.

We later extend this to the *bounded-ratio cost-distance Steiner tree problem* that arises if there are constants $\Theta \geq \theta > 0$ such that $\theta c(e) \leq d(e) \leq \Theta c(e)$ for all $e \in E(G)$ and provide an approximation factor based on the parameter Θ/θ. This in particular shows that the inapproximability result in [15] relies on allowing this parameter to increase indefinitely with the input size. The conditions imply that $c(e) = 0$ if and only if $d(e) = 0$. We assume that edges with $c(e) = 0 = d(e)$ are contracted in a preprocessing step, i.e. $c, d > 0$.

As noted in [14], a constant-factor approximation for the uniform case can be achieved with the bicriteria algorithm balancing minimum spanning trees and shortest path trees [12], leading to a 3.57-factor approximation. Using a variant that constructs shallow-light Steiner arborescences instead of (terminal) spanning trees [9], the approximation factor improves to 2.87 as shown implicitly in [8] and explicitly in [17].

Many approaches consider a single Steiner tree problem with bounded path lengths [9, 10, 1, 3], and modify the bicriteria approximation algorithm for the shallow-light spanning tree problem by [4], which was later refined by [12]. Note that relaxing the bounds on the path lengths via Lagrangean relaxation results in the uniform cost-distance Steiner arborescence problem.

A related problem is the *single-sink buy-at-bulk problem* where a set of demands need to be routed from a set of sources to a single sink. K different types of pipes are available for installation on any of the edges. Each pipe type has a different cost and capacity. This problem has been extensively studied in the literature with a series of improving constant approximation factors of 292 [7], 216 [18], 153.6 [11], and 40.82 [6]. For the splittable case, the best factor is 20.41 [6]. In [18] the single-sink buy-at-bulk problem was related to the *generalized deep discount problem*, where the pipe capacities are replaced by *'incremental'* costs per unit of flow. With only a single pipe type it reduces to the uniform cost-distance Steiner tree problem. The approximation factor of the generalized deep discount problem is within a factor of two of the approximation factor for the single-sink buy-at-bulk problem [18].

1.1 Our Contribution

Here, we propose a new algorithm that achieves an approximation factor of $(1 + \beta)$, where β is the approximation factor for the Steiner tree problem. Using $\beta = \ln(4)+\epsilon$ [2], this improves the best known approximation factor from 2.87 to 2.39. In contrast to the employment of the bicriteria approximation algorithm for shallow-light arborescences [8, 17], our algorithm uses the delay weights explicitly.

We also show NP hardness of the uniform cost-distance Steiner tree problem for the special case $V(G) = \{r\} \cup T$, $c \equiv d : E(G) \to \{1, 2\}$. Demonstrating that the hardness stems not only from the Steiner tree problem.

The bounded-ratio and uniform cost-distance Steiner tree problems have important applications. We show how they arise as Lagrangean subproblems in a fast approximation algorithm [10] for computing (fractional) Steiner trees for all nets in a VLSI netlist with mutual delay constraints minimizing the total length of all trees.

The remainder of this paper is organized as follows. In Section 2, we show that the uniform cost-distance Steiner tree problem is NP-hard even if $V(G) = T \cup \{r\}$,

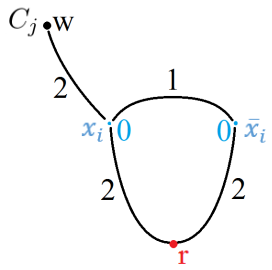

Fig. 1. graph construction used to reduce 3SAT to uniform cost-distance spanning tree

$c : E(G) \to \{1, 2\}$, and $w(t) = w > 0$ $(t \in T)$. Then, in Section 3, we briefly summarize how to obtain a 2.87-factor approximation algorithm from bicriteria algorithms for shallow-light Steiner trees. Our main result, a $(1 + \beta)$-factor approximation algorithm for the uniform cost-distance Steiner tree problem is presented in Section 4. Finally, a significant application in chip design is given in Section 6, followed by Conclusions.

2 Hardness of the spanning tree case

The NP-hardness of the uniform cost-distance Steiner tree problem follows directly from the NP-hardness of the ordinary Steiner tree problem, when setting all delay weights to zero. Here, we show that the special case $V(G) = \{r\} \cup T$ and $c : E(G) \to \{1, 2\}$ is already NP-hard. Thus, the difficulty arises from the two-folded objective function and not only from the Steiner tree problem.

Note that hardness of the buy-at-bulk problems can still stem from the Steiner tree problem when all vertices are demand vertices, because demands can be chosen as zero, leaving vertices unconnected. In contrast, in the cost-distance Steiner tree problem all terminals have to be connected to r regardless of their delay weight.

Theorem 1. *The cost-distance Steiner tree problem is NP-hard even if* $c : E(G) \to \{1, 2\}$, $d(e) = c(e)$ $(e \in E(G))$, *and* $V(G) = \{r\} \cup T$.

Proof. We prove the statement by reduction from 3-SAT. Let $\phi = C_1 \wedge C_2 \wedge \ldots \wedge C_m$ be a 3-SAT formula on variables $x_1, x_2, \ldots, x_n$. We construct a graph G as follows:

$$V(G) := \{C_1, \ldots, C_m, x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n, r\}, \text{ and} \tag{2}$$

$$E(G) := \{\{r, x_i\}, \{r, \bar{x}_i\}, \{x_i, \bar{x}_i\} : i = 1, \ldots, n\} \cup \{\{C_j, \alpha\} : \alpha \text{ is a literal in } C_j\} \tag{3}$$

Edge costs are chosen as $c(\{x_i, \bar{x}_i\}) = 1$ for $i \in \{1, \ldots, n\}$, and $c(e) = 2$ for all other edges. The delay weights are chosen uniformly as $w(C_j) = w$ $(j \in \{1, \ldots, m\})$ for an arbitrary constant $w > 0$ and zero for all other nodes. Here, any arbitrary choice of w works as long as it is positive. Figure 1 shows exemplified how the root, one pair of literals, and a clause are connected and the edge costs and weights are chosen. We claim that ϕ is satisfiable, if and only if G has a spanning tree A with objective value at most $3n + 2m + 4mw$.

To prove the claim, assume first that we are given a satisfying truth assignment for ϕ. We will construct a desired spanning tree A from the truth assignment. Add all edges between a true literal and the root to $E(A)$ and add all edges of length one.

Finally, connect each clause node to one of its true literals in the truth assignment. Then, the connection cost of the tree is $3n + 2m$. The path from each clause to the root consists of an edge of length two to the literal and another edge of length two to the root, so it has length four, resulting in the objective value of $3n + 2m + 4mw$.

For the reverse direction, observe that a connection cost of $3n + 2m$ is the cost of a minimum spanning tree and, thus, the least possible connection cost. Also, the distance of the clause nodes to the root in the graph is four, therefore the above objective value is simultaneously minimum in both factors. Thus, a tree with this value must be a minimum spanning tree and it selects all edges of length one. So at most one literal per variable is adjacent to the root. The connected literals give us a truth assignment. Now the spanning tree must also contain a shortest r-C_i-path for all $i = 1, \dots, m$. Consequently, each clause node must be connected to a literal adjacent to the root, i.e. a true literal. This means each clause evaluates to true and ϕ is satisfiable. □

By Theorem 1, the uniform cost-distance Steiner tree problem in graphs is NP-hard even when restricting to the case where $T = V(G) \setminus \{r\}$ rendering the Steiner tree to become a spanning tree. We therefore turn our attention to approximation algorithms.

3 Approximation factors from Shallow-Light Steiner Trees

Previous constant-factor approximation algorithms for the uniform cost-distance Steiner tree problem are based on bicriteria approximation algorithms for shallow-light Steiner trees. For a rooted Steiner tree, being shallow refers to bounding path lengths in the tree between terminals and the root, and being light refers to bounding the sum of edge lengths of the tree. Instead of delay weights w, they are given a delay bound $\bar{d}(t)$ for every $t \in T$. Sometimes this is a uniform deadline $\bar{d}(t) = \Delta \in \mathbb{R}_{\geq 0}$ [4], or the distance from the root $\bar{d}(t) = \mathrm{dist}_{(G,c)}(r, t)$ [12].

These algorithms [4, 12, 9, 8, 3] take as input an initial approximate minimum length Steiner tree A_0 and a parameter $\delta > 0$. Then, they proceed roughly as follows. They traverse A_0 along an Eulerian walk in the directed graph $\overleftrightarrow{A_0}$ which has each edge of A_0 replaced by two opposite arcs, starting at r and build an auxiliary graph H that is initialized with A_0. Whenever they enter a terminal $t \in T$ for which the current r-t-distance in H w.r.t. d is greater than $(1+\delta)\bar{d}(t)$, they add a shortest path in (G, d) to H. Finally, they return a shortest path tree in H. The length of the resulting tree is at most $\left(1 + \frac{2}{\delta}\right)$ times the length of the input tree, and it violates the delay bounds by at most a factor $(1 + \delta)$ (see [8] for more details).

We can employ a simple lower bound for the minimum total cost of a cost-distance Steiner tree that is also valid in the non-uniform case. The connection cost is minimized for a Steiner minimum tree T^{SMT} in (G, c) and the delay cost is minimized for a shortest path tree T^{SP} in (G, d). Thus, with $C_{SMT} := \sum_{e \in E(T^{SMT})} c(e)$ and $D_{SP} := \sum_{t \in T} w_t \mathrm{dist}_{(G,d)}(r, t)$ a lower bound for the objective value is

$$C_{SMT} + D_{SP}. \tag{4}$$

As proposed but not explicitly conducted in [14], we can use a bicriteria algorithm to derive a constant factor approximation for the uniform cost-distance problem. The bicriteria algorithm returns a solution with total cost $(1+\frac{2}{\delta})\beta C_{SMT} + (1+\delta)D_{SP}$. Now choose δ such that $(1 + \delta) = (1 + \frac{2}{\delta})\beta$ (see also [8, 17]), equalizing the approximation factors for the delay and for the total length. Using a spanning tree for A_0 [12], where

Algorithm 1: Cost-Distance Steiner Tree Approximation

Input : A graph G, $c, d : E(G) \to \mathbb{R}_{\geq 0}$, $T \subset V(G)$, $r \in V(G)$, a weight threshold $\mu > 0$.
Output : A Steiner arborescence on the terminal set T rooted at r.

1 $A_0 :=$ β-approximate minimum cost Steiner arborescence for $T \cup \{r\}$ in $(\overleftrightarrow{G}, c)$ rooted at r, satisfying $\delta^+(t) = \emptyset$ for all $t \in T$, $|\delta^+_{A_0}(v)| = 2$ for all $v \in V(A_0) \setminus T$.
2 $H := A_0$; $\mathcal{T} = \emptyset$.
3 **for** $v \in V(A_0)$ *in reverse topological order* **do**
4 Let H_v be the sub-arborescence of H rooted at v.
5 **if** $w(V(H_v) \cap T) > \mu$
6 $T' := V(H_v) \cap T$; $A' := H_v$.
7 $E(H) := E(H) \setminus \big((E(A') \cup \delta^-(v)\big)$.
8 $\mathcal{T} = \mathcal{T} \cup \{T'\}$.
9 **for** $T' \in \mathcal{T}$ **do**
10 Let A' be the Steiner tree connecting T' according to Step 6 or $(T', \emptyset)$ if $|T'| = 1$.
11 Select a "port" vertex $t \in T'$ and a shortest r-t-path P in (G, d) such that the arborescence A'' rooted at r, which results from re-orienting $P + A'$, minimizes

$$\sum_{e \in E(A'')} c(e) + \sum_{t \in T'} w(t) \cdot d(E(A''_{[r,t]})). \tag{5}$$

  $E(H) := E(H) \cup E(A'')$.
12 Return shortest path arborescence in H w.r.t. d rooted at r and covering T.

$\beta = 2$, this yields an approximation factor of 3.57. With the currently best known approximation factor $\beta = \ln(4) + \epsilon$ [2], this gives an approximation factor of 2.87.

Note that the bicriteria algorithm does not take the actual delay weights w into account. It returns the same solutions for $w \equiv 0$ or $w \nearrow \infty$, for which the optimum solutions would be T^{SMT} or approach T^{SP}.

4 Improved Approximation for the Uniform Cost-Distance Steiner Tree Problem

We show how to improve the approximation factor for the uniform cost-distance Steiner tree problem by employing the delay weights algorithmically.

Algorithm 1 shows the overall algorithm, which we will now describe in detail. The solution to the cost-distance Steiner tree problem is a tree. Our algorithm formally computes a Steiner arborescence in $\overleftrightarrow{G}$ rooted at r, where $\overleftrightarrow{G}$ is the directed graph G with each edge replaced by two oppositely oriented arcs of cost $c(\overleftarrow{e}) := c(\overrightarrow{e}) := c(e)$ and delay $d(\overleftarrow{e}) := d(\overrightarrow{e}) := d(e)$ for $e \in E(G)$. It can easily be turned into the desired Steiner tree by neglecting its orientation. Note that the description of the algorithm uses separate functions for cost and delay that we use later for the bounded-ratio case in Section 5. However, the analysis of the uniform case in this section assumes $c \equiv d$.

We partition the terminal sets into clusters of bounded delay weight based on a threshold parameter $\mu > 0$. To this end, we first compute a light Steiner tree for $T \cup \{r\}$ in (G, c) by a β-factor approximation algorithm and turn it into an arborescence A_0 rooted at r. By inserting dummy vertices and edges of zero cost and zero delay, as well as short-cutting Steiner vertices with out-degree one, we can assume that A_0 is a full binary tree, i.e. $|\delta^+_{A_0}(v)| = 2$ ($v \in V(A_0) \setminus T$), with leave set T. Here, we use $\delta^+_{A_0}(v)$ and $\delta^-_{A_0}(v)$ to denote the set of edges leaving vertex v and the set of edges entering vertex v in the graph A_0, respectively. Note that formally A_0 is not a subgraph of $\overleftrightarrow{G}$ any more.

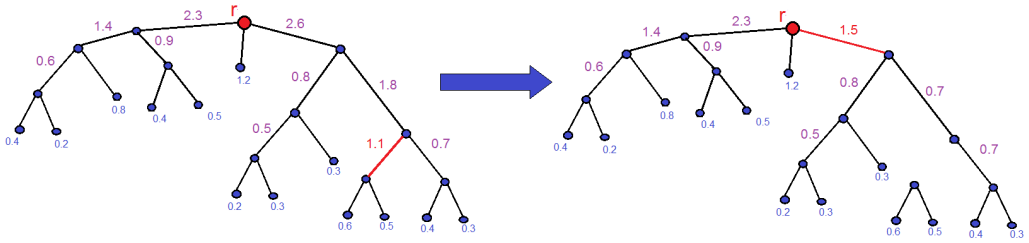

Fig. 2. Splitting off sub-arborescences of aggregate delay weight higher than the threshold μ, here with $\mu = 1$. Traversing in reverse topological order we remove the red edge into a sub-arborescence with delay weight $1.1 > \mu$.

Then, we traverse the vertices of A_0 in reverse topological order. Whenever the delay weight of the terminals spanned by the sub-arborescence H_v rooted at the currently visited vertex $v \in V(A_0)$ is larger than μ, we split off the sub-arborescence rooted at v as in Figure 2. The sub-arborescences are stored for later reuse.

This clustering results in a partition of $T = T_r \dot\cup T_1 \dot\cup \ldots \dot\cup T_k = T_r \dot\cup \dot\bigcup_{T' \in \mathcal{T}} T'$ and arborescences $A_1, \ldots, A_k$, where A_i connects the terminals in T_i $(i = 1, \ldots, k)$ rooted at some vertex r_i where the sub-arborescence was cut off such as in Figure 3. i.e., r_i corresponds to some vertex v and T_i to some set T' according to Step 6. T_r consists of the terminals that are still residing in the component H_r when the for-loop in Step 3 terminates.

The following proposition provides a key characteristic of the clusters.

Proposition 1. *Each cluster $T' \in \mathcal{T}$ generated in Step 6 is either a heavy singleton $\{t\}$ $(t \in T)$ with delay weight $w(t) > \mu$ or a cluster $T' \in \mathcal{T}$ with delay weight $\mu < w(T') \leq 2\mu$.*

Proof. If a visited vertex is not a heavy singleton, then its at most two sub-arborescences have weight at most μ. Otherwise, they would have been split off. In turn, the entering edge is removed only if $w(T') > \mu$. □

After the clusters are determined, we connect each cluster $T' \in \mathcal{T}$ to the root r in Step 11. To this end we consider each $t \in T'$ as a possible "port" vertex for which we select an r-t-path P minimizing $d(E(P))$. The other vertices in T' are connected through the initial sub-arborescence A' after re-orienting arcs if necessary. For each cluster, we pick a port vertex minimizing the total cost (5) for the cluster. Finally, we return a shortest path tree in H w.r.t. the edge delays rooted in r and containing all terminals T.

Next, we analyze the cost caused by a cluster after choosing the terminal and shortest path through which we connect it in Step 11.

Lemma 1. *Let $T' \in \mathcal{T}$, and let A' be the sub-arborescence stored with T'. In the uniform case $c \equiv d$, the terminal $t \in T'$ and the path P chosen in Step 11 provide a shortest path tree in $P \cup A'$ that has a total cost of at most*

$$\big(1 + \frac{1}{\mu}\big) \sum_{t \in T'} w(t) \mathrm{dist}_{(G,d)}(r, t) + \big(1 + \mu\big) c(E(A')).$$

Proof. If T' is a singleton $\{t\}$, the statement becomes evident since $c(E(A')) = 0$, $\frac{w(t)}{\mu} > 1$ and $c \equiv d$. Otherwise, let us randomly choose a "port" terminal $t \in T'$

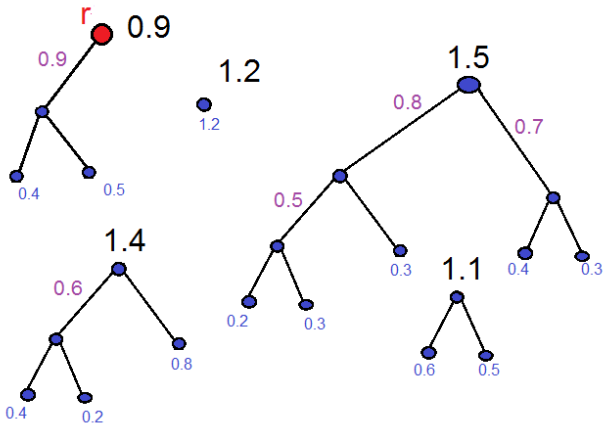

Fig. 3. The clusters at the end of the splitting-off process with $\mu = 1$.

together with a shortest r-t-path P in (G, d). We will show that the expected value of this choice satisfies the desired cost bound. Thus, the deterministic best choice of terminal and path in the algorithm must also satisfy this bound.

Let $W := w(T')$. The terminal $t \in T'$ will be selected with probability p_t proportional to its delay weight:

$$p_t := \frac{w(t)}{W}.$$

The expected value of the connection cost can be upper bounded by the expected value of $c(E(P))$ plus $c(E(A'))$. The expected value of the delay cost can be upper bounded by the expected value of $W \cdot d(E(P))$ plus the expected delay cost for serving $T' \setminus \{t\}$ through A'.

The delay cost on P has the expected value

$$\mathbb{E}\left(W \cdot d(E(P))\right) = W \sum_{t \in T'} p_t \cdot \mathrm{dist}_{(G,d)}(r, t) = \sum_{t \in T'} w(t) \cdot \mathrm{dist}_{(G,d)}(r, t).$$

This is the delay cost of a shortest path tree connecting T' to the root, i.e. the minimum possible delay cost for T'.

For the delay cost through A', consider an edge $(u, v) \in E(A')$ in the sub-arborescence A' before re-orienting it. The edge separates the sub-arborescence A'_v of $A' - (u, v)$ containing v from the rest of A'. We compute the expected delay cost contribution of (u, v). It is the product of $d(u, v)$ and the delay weight served through (u, v) or (v, u) if it is re-oriented. Let $T'_v := V(A'_v) \cap T'$ and $W_v := w(T'_v)$.

With probability $\frac{W_v}{W}$ one of the terminals in T'_v is chosen by the random selection as the port t. In this case, the delay weight served by (u, v) equals the sum of the weights of all other terminals, $W - W_v$. Otherwise, the port is chosen from $T' \setminus T'_v$ and the delay weight served by (u, v) (after reverting it) is W_v (see Figure 4). Therefore, the expected delay weight served by (u, v) is

$$\frac{W_v}{W} \cdot (W - W_v) + \frac{W - W_v}{W} \cdot W_v = \frac{2W_v(W - W_v)}{W} \leq \frac{W}{2} \leq \mu,$$

where the last inequality follows from Proposition 1. This means that the delay cost in A' has expected value at most $\mu \sum_{e \in E(A')} d(e) = \mu d(E(A'))$.

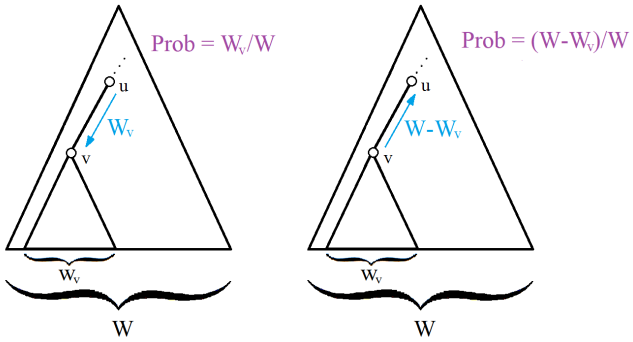

Fig. 4. The delay weight on the edge (u, v) is W_v with probability (W_v/W) if the port terminal is chosen from A'_v. It is $W - W_v$ with probability $(W - W_v)/W$ if the port is chosen from $A' \setminus A'_v$.

The connection cost of P has expected value

$$\mathbb{E}\left(c(E(P))\right) = \sum_{t \in T'} \frac{w(t)}{W} d(E(P_t)) = \sum_{t \in T'} \frac{w(t)}{W} \text{dist}_{(G,d)}(r,t) \leq \frac{1}{\mu} \sum_{t \in T'} w(t) \text{dist}_{(G,d)}(r,t),$$

where P_t denotes a shortest r-t-path in (G, d), and we are using that $W \geq \mu$ and $c(E(P) = d(E(P))$. As the expected cost of serving T' through a randomly chosen t is at most

$$\sum_{t \in T'} w(t) \text{dist}_{(G,d)}(r,t) + \mu c(E(A')) + \frac{1}{\mu} \sum_{t \in T'} w(t) \text{dist}_{(G,d)}(r,t) + c(E(A'))$$

$$= \big(1 + \frac{1}{\mu}\big) \sum_{t \in T'} w(t) \text{dist}_{(G,d)}(r,t) + \big(1 + \mu\big) c(E(A')),$$

this cannot be exceeded by the cheapest choice for t in Step 11. □

Theorem 2. *Given an instance $(G, c \equiv d, w)$ of the uniform cost-distance Steiner tree problem, a Steiner tree A, whose objective value (1) is at most*

$$(1 + \mu)\beta C_{SMT} + (1 + \frac{1}{\mu}) D_{SP},$$

where $C_{SMT} := \sum_{e \in E(T^{SMT})} c(e)$ and $D_{SP} := \sum_{t \in T} w_t \text{dist}_{(G,d)}(r,t)$, can be computed in $\mathcal{O}(\Lambda + |T|^2 + m + n \log n)$ time. Here, Λ is the running time for computing a β-approximate minimum Steiner tree for $T \cup \{r\}$, $n = |V(G)|$, and $m = |E(G)|$.

Proof. We run Algorithm 1. The resulting objective value is given by the cost of the root component H_r, just before the for loop in Step 9 starts adding clusters in $\mathcal{T}$, plus the cost of all added clusters. As the total delay weight in any sub-arborescence of $H_r - r$ is at most μ, the total cost of H_r is at most $\mu d(E(H_r)) + c(E(H_r)) = (\mu + 1)c(E(H_r))$.

As we use edges from the initial approximate minimum Steiner tree at most once in either $E(H_r)$ or one of the edge sets $E(A')$ of the stored sub-arborescences, we can simply add the above cost of H_r and the cluster costs according to Lemma 1. This gives the desired solution cost of at most

$$(1 + \mu)\beta C_{SMT} + (1 + \frac{1}{\mu}) D_{SP}.$$

The running time of the algorithm is given by the running time $\Lambda + \mathcal{O}(n)$ for computing the β-approximate minimum Steiner tree and transforming it into the arborescence A_0 plus the running time for clustering and selecting the port terminal and shortest path for all clusters. The clustering can be accomplished in time $\mathcal{O}(|E(A_0)|) \leq \mathcal{O}(|T|)$.

Rerouting a cluster tentatively connects a cluster through a terminal in the cluster. To this end we once compute a shortest path tree rooted at r covering all terminals using Dijkstra's algorithm with Fibonacci heaps in time $\mathcal{O}(m + n \log n)$ [5]. As the arborescence stored for the cluster has size $\mathcal{O}(|T|)$, the total runtime of all the rerouting steps is $\mathcal{O}(|T|^2 + m + n \log n)$. Thus, the overall running time is bounded by $\mathcal{O}(\Lambda + |T|^2 + m + n \log n)$. □

Corollary 1. *The uniform cost-distance Steiner tree problem can be approximated in polynomial time within a factor* $(1 + \beta)$*, where* β *is the Steiner ratio.*

Proof. Let OPT be the total cost of an optimum solution. Setting $\mu = \frac{1}{\beta}$ yields a solution with total cost at most

$$(1 + \mu)\beta C_{SMT} + (1 + \frac{1}{\mu})D_{SP} = (1 + \beta)(C_{SMT} + D_{SP}) \leq (1 + \beta)OPT.$$

□

Using [2] to compute A_0 with $\beta = \ln(4) + \epsilon \leq 1.39$ we get the following result.

Corollary 2. *The uniform cost-distance Steiner tree problem can be approximated within a factor 2.39 in polynomial time.*

Many practical applications [10, 1, 3] require a faster running time than provided by the Steiner tree approximation algorithm in [2]. We can lower the quality of the initial Steiner arborescence to get the following result.

Corollary 3. *Given an instance* (G, c, d, w) *of the uniform cost-distance Steiner tree problem, a Steiner tree* A*, whose cost is at most* 3 *times the optimum cost can be computed in* $\mathcal{O}(|T|^2 + m + n \log n)$*, where* $n = |V(G)|$ *and* $m = |E(G)|$*.*

Proof. Compute A_0 by a terminal spanning tree algorithm in time $\mathcal{O}(m + n \log n)$ [13]. Thereby, $\beta = 2$ and the approximation factor is 3. □

Algorithm 1 is similar to [7] for the single-sink buy-at-bulk problem with a single pipe type ($K = 1$), we would like to point out a few key differences. Both start from a light arborescence and split off sub-arborescences that are heavy w.r.t. the delay weight and demand respectively. However, we reuse the initial sub-arborescence to connect terminals within a cluster. To this end, we introduce a novel computation of the expected delay cost across the edges of the stored sub-arborescences. In contrast, new pipes between the initial root of the sub-arborescence and the port vertex are inserted in [7]. Furthermore, our clustering enforces an upper bound of 2μ on the delay weights (except for heavy singletons), which is crucial to obtain the desired approximation factor.

5 Bounded-Ratio Cost-Distance Steiner Trees

The approximation algorithms in Sections 3 and 4 for the uniform cost-distance Steiner can be extended to the bounded-ratio cost-distance Steiner tree problem, where $\theta c(e) \leq d(e) \leq \Theta c(e)$ for two constants $\theta \leq \Theta$. For the bicriteria algorithm (and Algorithm 1), we compute the initial light Steiner tree w.r.t. (G, c), i.e. aiming for minimum connection cost. Then, for the bicriteria algorithm in Sections 3, the splitting into sub-instances is then conducted based on delays in (G, d).

Similarly, the choice of the port vertex $t \in T'$ in Step 11 of Algorithm 1 is done based on the combined cost function (4). With a β-approximate Steiner tree A_0 and a delay violation tolerance $\delta > 0$, the bicriteria algorithm computes a solution of objective value at most

$$\left(1 + \frac{2}{\delta}\frac{\Theta}{\theta}\right)\beta C_{SMT} + (1+\delta)D_{SP}. \tag{6}$$

For given C_{SMT} and D_{SP}, we can choose δ such that the ratio of the final cost (6) to the trivial lower bound (4) is minimized. This leads to a choice of $\delta = \sqrt{2\beta\frac{C_{SMT}}{D_{SP}}\frac{\Theta}{\theta}}$ and to the approximation factor

$$\frac{D_{SP} + \beta C_{SMT} + 2\sqrt{2\beta C_{SMT} D_{SP}\frac{\Theta}{\theta}}}{D_{SP} + C_{SMT}}. \tag{7}$$

It turns into an approximation factor of $\frac{1}{2}\left(\beta + 1 + \sqrt{(\beta-1)^2 + 8\beta\frac{\Theta}{\theta}}\right)$, independent of C_{SMT} and D_{SP}. The same value arises when choosing δ such that $(1+\frac{2}{\delta}\frac{\Theta}{\theta})\beta = 1+\delta$.

Similarly, following the proof from Section 4, Algorithm 1 achieves a solution of cost at most

$$(1+\mu\Theta)\beta C_{SMT} + (1+\frac{1}{\mu\theta})D_{SP}.$$

Setting $\mu = \frac{1}{\sqrt{\beta\frac{C_{SMT}}{D_{SP}}\frac{\Theta}{\theta}}}$, results in an approximation factor of

$$\frac{D_{SP} + \beta C_{SMT} + 2\sqrt{\beta C_{SMT} D_{SP}\frac{\Theta}{\theta}}}{D_{SP} + C_{SMT}},$$

which compared to (7) avoids a factor 2 under the square root. We can state the following generalization of Theorem 2.

Theorem 3. *Given an instance (G, c, d, w) of the bounded-ratio cost-distance Steiner tree problem, a Steiner tree A, whose objective value (1) is at most $\frac{\beta+1+\sqrt{(\beta-1)^2+4\beta\frac{\Theta}{\theta}}}{2}$ times the optimum, can be computed in $\mathcal{O}(\Lambda + |T|^2 + m + n\log n)$ time, where Λ is the running time for computing a β-approximate minimum Steiner tree for $T \cup \{r\}$, $n = |V(G)|$, and $m = |E(G)|$.*

For $\theta = \Theta$, it matches the previous ratio $(1 + \beta)$.

6 Application: delay constrained Steiner netlists

We now present an interesting application of (uniform) cost-distance Steiner trees in chip design. Here, Steiner trees need to be chosen for millions of interconnects subject

to mutual signal delay constraints. The graph G can be considered a (possibly 3-dimensional) grid graph with holes. Holes arise from blockages that prevent interconnect routing.

The Steiner tree instances and their mutual delay dependencies are defined through a hypergraph D with $V(D) \subseteq V(G)$, which is often called *netlist*. Vertices in $V(D)$ represent start and end points of signal paths or small logic gates such as two-way AND or OR functions. Edges $E(D)$ represent interconnect nets. Each edge $e \in E(D)$ has a designated root $r_e \in e$ ($\subseteq V(G)$), while $T_e := e \setminus \{r_e\}$ is its sink set. This implies an orientation of the hyperedges from the root to the sinks. A path is an alternating sequence of vertices and hyperedges $v_1, e_1, v_2, e_2, \ldots, e_k, v_{k+1}$ such that v_i is the root of e_i and v_{i+1} is a sink of e_i. As signal paths are intercepted by registers where signal paths start and end, the hypergraph is acyclic, i.e. there are no cycles $v_1, e_1, v_2, e_2, \ldots, e_k, v_{k+1} = v_1$. Each vertex $v \in V(D)$ has a fixed delay $d(v) \in \mathbb{R}_{\geq 0}$, where d is extended from $E(G)$ to $d : E(G) \cup V(D) \to \mathbb{R}_{\geq 0}$.

Given a Steiner tree $A \subseteq G$ connecting the root r_e of a hyperedge $e \in E(D)$ with its sinks T_e, the delay from r_e to a sink $v \in T_e$ is defined as $d_A(r_e, v) := d(r_e) + \sum_{f \in E(A_{[r_e,v]})} d(f)$.

The *delay-constrained Steiner netlist problem* is to compute a Steiner tree A_e for each $e \in E(D)$, minimizing the total connection cost $\sum_{e \in E(D)} c(E(A_e))$ such that all (maximal) paths $v_1, e_1, v_2, e_2, \ldots, e_k, v_{k+1}$ of D satisfy

$$\sum_{i=1}^{k} d_{A_{e_i}}(v_i, v_{i+1}) \leq \Delta,$$

where Δ is the given cycle time of the chip.

Using vertex potentials $a : V(D) \to \mathbb{R}$ for longest paths, it is easy to see that the path delay constraints can be represented in a compact form using $|V(D)|$ variables and $2|V(D)| + \sum_{e \in E(D)} |e|$ constraints:

$$\begin{aligned} a(v) &\geq 0 && (v \in V(D) : \delta^-(v) = \emptyset) \\ a(v) &\leq \Delta && (v \in V(D) : \delta^+(v) = \emptyset) \\ a(r_e) + d_{A_e}(r_e, v) &\leq a(v) && (e \in E(D), v \in e \setminus \{r_e\}). \end{aligned} \tag{8}$$

The delay-constrained Steiner netlist problem is a special case of the timing-constrained global routing problem, where G is additionally equipped with edge capacities $u : E(G) \to \mathbb{R}_{\geq 0}$. Now, the Steiner trees also have to be packed w.r.t. the edge capacities. Recently, a global resource sharing model for timing-constrained global routing was proposed by [10]. It considers the feasibility problem for the constraints (8), the global routing capacities, and a guessed upper bound on the total wire length. In this model $C := |V(D)| + |E(D)|$ customers/variables consume from $R := \sum_{e \in E(D)} |e| + |E(G)| + 1$ resources/inequality constraints. The algorithm by [16] efficiently approximates an optimum solution to the fractional relaxation via Lagrangean relaxation and multiplicative-weight updates. The output is a convex combination of Steiner trees for each $e \in E(D)$. The fractionality arises as the average of finitely many discrete solutions.

However, it requires an approximate oracle for the cost-distance Steiner tree problem that occurs as a Lagrangean subproblem, where prices $c : E(G) \to \mathbb{R}_{\geq 0}$, penalizing routing congestion and the total tree length, and prices $w : \cup_{e \in E(D)} (e \setminus r_e) \to \mathbb{R}_{\geq 0}$, penalizing delays signal delays, are maintained by the resource sharing algorithm. Details can be found in [16, 10]. In this general setting, [14] still provides the best

known approximation factor $\mathcal{O}(\log |T|)$, while only oracles for constantly bounded terminal sizes or heuristics were presented by [10].

The special case of neglecting the packing aspect, i.e. $u \equiv \infty$, is still relevant. Several recent practical algorithms for delay-bounded Steiner trees in chip design [1, 3] compute delay-constrained Steiner trees on a net-by-net basis without considering mutual delay dependencies or routing congestion. We can improve over such approaches by directly considering delay constraints on signal paths. In this special case, the resource sharing algorithm [16, 10] maintains a single price penalizing the total length of all trees and delay prices $w : \cup_{e \in E(D)}(e \setminus r_e) \to \mathbb{R}_{\geq 0}$ so that the required oracle function corresponds to a uniform cost-distance Steiner tree problem with $\theta c(e) = d(e)$ $(e \in E(G))$ for some θ depending on the current resource price for the total length. Using Theorem 2 as an oracle for [16, 10] yields the following results.

Theorem 4. *Given $\omega > 0$, we can compute a fractional solution to the delay-bounded Steiner netlist problem that violates the path delay constraints by at most a factor $(1 + \omega)2.39$ and deviates from the length of an optimum solution by at most a factor $(1 + \omega)2.39$. The running time of the algorithm is*

$$\mathcal{O}\left(|E(D)|(m + n \log n) + (\Lambda + |e_{\max}|n) \log R \left((C + R) \log \log R + (C + R)\omega^{-1}\right)\right),$$

where $m := |E(G)|$ and $n := |V(G)|$.

Proof. The approximation factor follows immediately from Theorem 8 in [16] and Theorem 2.

The resource sharing algorithm makes $\mathcal{O}\left(\log R \left((C + R) \log \log R + (C + R)\omega^{-1}\right)\right)$ calls to the oracle. For each hyperedge $e \in E(D)$, the shortest path trees w.r.t. (G, c) needs to be computed only once in the beginning so that the running time (taking $\mathcal{O}(|E(D)|(m + n \log n))$ time in total) per oracle call is only $(\Lambda + |e_{\max}|n)$. □

Note that the number of oracle calls is near-linear in the instance size, leading to a fast algorithm in practice. While the solution is only fractional, it can be discretized using randomized rounding as in [16]. The models and algorithms in [16, 10] constitute state-of-the-art algorithms for global routing on industrial instances. Thus, our improved approximation ratio for uniform cost-distance Steiner trees is of theoretical relevance and likewise also of practical interest.

As modern metal stacks exhibit routing layers with different metal widths and spacings, it would be reasonable to have different cost and delay parameters on different layers of the 3d grid graph G. As the number of layers is typically bounded by a small constant 10-16, the oracle to solve the Lagrangean subproblem would correspond to the bounded-ratio cost-distance Steiner tree problem, for which Algorithm 1 also provides a constant approximation factor.

7 Conclusion

We presented an improved approximation factor $(1 + \beta)$ for the uniform cost-distance Steiner tree problem, which for $\beta = \ln(4) + \epsilon$ [2] results in a factor of 2.39. The best previous factor based on bicriteria algorithms for shallow-light Steiner arborescences was 2.87. We achieve this by using delay weights explicitly.

The practical importance of the bounded-ratio and uniform cost-distance Steiner tree problems is demonstrated applications from chip design, where these problems occur as Lagrangean subproblems. In the application Steiner trees for all nets of minimum total length have to be computed subject to mutual delay dependencies.

References

1. C.J. Alpert, W.K. Chow, K. Han, A.B. Kahng, Z. Li, D. Liu, and S. Venkatesh, *Prim-Dijkstra Revisited: Achieving Superior Timing-driven Routing Trees.* International Symposium on Physical Design (ISPD'18), 10–17, 2018.
2. J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità, *Steiner tree approximation via iterative randomized rounding.* Journal of the ACM 60, Article 6, 2013.
3. G. Chen and E.F.Y. Young, *SALT: Provably Good Routing Topology by a Novel Steiner Shallow-Light Tree Algorithm* IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, early online access, 2019.
4. J. Cong, A.B. Kahng, G. Robins, M. Sarrafzadeh, and C.K. Wong, *Provably good performance-driven global routing.* IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems 11 (6), 739–752, 1992.
5. M.L. Fredman and R.E. Tarjan, *Fibonacci heaps and their uses in improved network optimization problems.* Journal of the ACM 34, 596–615, 1987.
6. F. Grandoni and T. Rothvoß, *Network design via core detouring for problems without a core.* International Colloquium on Automata, Languages, and Programming (ICALP'10), 490–502, 2010.
7. S. Guha, A. Meyerson, and K. Mungala, *A Constant Factor Approximation for the Single Sink Edge Installation Problem.*, SIAM Journal on Computing 38 (6), 2426–2442, 2009.
8. L. Guo, N. Zou, and Y. Li, *Approximating the Shallow-Light Steiner Tree Problem When Cost and Delay are Linearly Dependent.*, Symposium on Parallel Architectures, Algorithms and Programming (PAAP'14), 99–103, 2014.
9. S. Held and D. Rotter, *Shallow-Light Steiner Arborescences with Vertex Delays.* Conference on Integer Programming and Combinatorial Optimization (IPCO'13), 229–241, 2013.
10. S. Held, D. Müller, D. Rotter, R. Scheifele, V. Traub, and J. Vygen, *Global Routing with Timing Constraints.* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 37(2), 406–419, 2018.
11. R. Jothi and B. Raghavachari, *Improved approximation algorithms for the single-sink buy-at-bulk network design problems.* Journal of Discrete Algorithms 7(2), 249–55, 2009.
12. S. Khuller, B. Raghavachari, N. Young, *Balancing Minimum Spanning and Shortest Path Trees.* Algorithmica 14(4), 305–321, 1995.
13. K. Mehlhorn, *A faster approximation algorithm for the Steiner problem in graphs.* Information Processing Letters 27, 125–128, 1988.
14. A. Meyerson, K. Munagala, and S. Plotkin, *Cost-distance: Two metric network design.*, SIAM Journal on Computing 38(4), 1648–1659, 2008.
15. Julia Chuzhoy, Anupam Gupta, Joseph (Seffi) Naor, Amitabh Sinha *On the Approximability of Some Network Design Problems*
16. D. Müller, K. Radke, and J. Vygen, *Faster min-max resource sharing in theory and practice.* Mathematical Programming Computation 3, 1–35, 2011.
17. D. Rotter, *Timing-Constrained Global Routing with Buffered Steiner Trees.*, Dissertation thesis, University of Bonn, 2017.
18. K. Talwar, *The single-sink buy-at-bulk LP has constant integrality gap*, International Conference on Integer Programming and Combinatorial Optimization (IPCO'02), 475–486, 2002.