Quasi-Polynomial Time Approximation Algorithm for Low-Degree Minimum-Cost Steiner Trees

J. Könemann and R. Ravi

GSIA, Carnegie Mellon University,
Pittsburgh PA 15213, USA.
{jochen,ravi}@cmu.edu

Abstract. In a recent paper [5], we addressed the problem of finding a minimum-cost spanning tree T for a given undirected graph $G = (V, E)$ with maximum node-degree at most a given parameter $B > 1$. We developed an algorithm based on Lagrangean relaxation that uses a repeated application of Kruskal's MST algorithm interleaved with a combinatorial update of approximate Lagrangean node-multipliers maintained by the algorithm.
In this paper, we show how to extend this algorithm to the case of Steiner trees where we use a primal-dual approximation algorithm due to Agrawal, Klein, and Ravi [1] in place of Kruskal's minimum-cost spanning tree algorithm. The algorithm computes a Steiner tree of maximum degree $O(B + \log n)$ and total cost that is within a constant factor of that of a minimum-cost Steiner tree whose maximum degree is bounded by B. However, the running time is quasi-polynomial.

1 Introduction

We consider the minimum-degree Steiner tree problem (B-ST) where we are given an undirected graph $G = (V, E)$, a non-negative cost c_e for each edge $e \in E$ and a set of terminal nodes $R \subseteq V$. Additionally, the problem input also specifies positive integers $\{B_v\}_{v \in V}$. The goal is to find a minimum-cost Steiner tree T covering R such each node v in T has degree at most B_v, i.e. $\deg_T(v) \leq B_v$ for all $v \in V$. We present an algorithm for the problem and give a proof of the following theorem.

Theorem 1. *There is a primal-dual approximation algorithm that, given a graph $G = (V, E)$, a set of terminal nodes $R \subseteq V$, a nonnegative cost function $c : E \to \mathcal{R}^+$, integers $B_v > 1$ for all $v \in V$, and an arbitrary $b > 1$ computes a Steiner tree T that spans the nodes of R such that*

1. $\deg_T(v) \leq 12b \cdot B_v + \lceil 4 \log_b n \rceil + 1$ *for all* $v \in V$, *and*
2. $c(T) \leq 3\mathtt{OPT}$

where OPT *is the minimum cost of any Steiner tree whose degree at node v is bounded by B_v for all v. Our method runs in $O(n \log(|R|) \cdot |R|^{\lceil 4 \log n \rceil})$ iterations each of which can be implemented in polynomial time.*

The algorithm combines ideas from the primal-dual algorithm for Steiner trees due to Agrawal et al. [1] with local search elements from [2].

2 A linear programming formulation

The following natural integer programming formulation models the problem. Let $\mathcal{U} = \{U \subseteq V : U \cap R \neq \emptyset, R \setminus U \neq \emptyset\}$.

$$
\begin{aligned}
\min \quad & \sum_{e \in E} c_e x_e & \text{(IP)} \\
\text{s.t} \quad & x(\delta(U)) \geq 1 \quad \forall U \in \mathcal{U} & (1) \\
& x(\delta(v)) \leq B_v \quad \forall v \in V & (2) \\
& x \text{ integer}
\end{aligned}
$$

The dual of the linear programming relaxation (LP) of (IP) is given by

$$
\begin{aligned}
\max \quad & \sum_{U \in \mathcal{U}} y_U - \sum_{v \in V} \lambda_v \cdot B_v & \text{(D)} \\
\text{s.t} \quad & \sum_{U : e \in \delta(U)} y_U \leq c_e + \lambda_u + \lambda_v \quad \forall e = uv \in E & (3) \\
& y, \lambda \geq 0
\end{aligned}
$$

We also let (IP-ST) denote (IP) without constraints of type (2). This is the usual integer programming formulation for the Steiner tree problem. Let the LP relaxation be denoted by (LP-ST) and let its dual be (D-ST).

3 An algorithm for the Steiner tree problem

Our algorithm is based upon previous work on the generalized Steiner tree problem by Agrawal et al. [1]. In the following we refer to this algorithm by `AKR`. We first give a fairly complete description of Algorithm `AKR` to compute approximate Steiner trees. Subsequently we present an alternate view of `AKR` which simplifies the presentation of our algorithm for `B-ST` later in this paper.

3.1 The `AKR` algorithm

At each step during the algorithm we maintain a partial Steiner tree. More specifically, we have a set of edges $E' \subseteq E$. We use $\mathcal{C}(E')$ to denote the set of connected components of $G[E']$ and let

$$\mathcal{V}(E') = \{S \in \mathcal{C}(E') : S \in \mathcal{U}\}$$

denote the set of *minimally violated components* for edge set E'. Sets $S \in \mathcal{V}(E')$ are violated in the sense the inequality (1) in (LP-ST) is violated for set S and partial solution E'. S is minimal in the sense that there is no proper subset $S' \subset S$ such that (1) is violated for set S' and partial solution E'. We sometimes refer to sets $S \in \mathcal{V}(E')$ as *moats*.

The algorithm also maintains a feasible dual solution y for (D-ST) and let $y_S = 0$ initially for all $S \in \mathcal{U}$.

The construction of the final Steiner tree is now viewed as a continuous process. With each *time* $t \in [0, t^*]$, we associate a partial primal solution x^t for (LP-ST) and a feasible dual solution y^t for (D-ST). At time t, let E^t be the set of edges that we have included so far and let $\mathcal{V}^t = \mathcal{V}(E^t)$ be the set of minimally violated components. We let y^t be the dual solution a time t.

At any given point $t \in [0, t^*]$ such that x^t is not the incidence vector of a feasible Steiner tree, we increase y_S for all $S \in \mathcal{V}^t$ simultaneously until a constraint of type (3) for edge $e \in E \setminus E^t$ becomes tight. Assume that this happens at time $t' \geq t$. The dual update is

$$y_S^{t'} = \begin{cases} y_S^t + (t' - t) & : \quad S \in \mathcal{V}^t \\ y_S^t & : \quad \text{otherwise.} \end{cases}$$

If the tightness of e is caused by two colliding active moats $S_1, S_2 \in \mathcal{V}^t$, we *merge* them and create a new moat $S = S_1 \cup S_2$. In this case, there must exist a minimum-c-cost path P connecting terminal nodes $s_1 \in S_1$ and $s_2 \in S_2$ such that every edge $e' \in P$ is tight and $e \in P$. Moreover, all internal nodes v of P (we write $v \in \texttt{int}(P)$) are Steiner nodes. We then say, that P is an s_1, t_1-*Steiner path* and we let $\mathcal{P}_{S_1,T_1}$ be the set of all Steiner paths that connect terminals in S_1 and T_1, respectively.

We include the edges of P into our solution:

$$x_{e'}^{t'} = \begin{cases} 1 & : \quad e' \in P \\ x_{e'}^t & : \quad \text{otherwise.} \end{cases}$$

Notice, that we do not update the primal solution in the case that the tightness of edge e is caused by one moat alone. This must mean that e connects S to a Steiner node. In this way, our view of the primal-dual algorithm is closer to that of Agrawal, Klein, and Ravi [1] rather than the treatment in [3] where all tight edges are added and unnecessary ones are deleted in a *reverse delete* step at the end.

In the following, let $\mathcal{P}$ be the set of Steiner nodes that Algorithm **AKR** uses to connect all terminals in R and denote by $\mathcal{P}_S$ the subset of paths from $\mathcal{P}$ that use only nodes from $S \subseteq V$. The following lemma is proved inductively.

Lemma 1. *For any $t \in [0, t^*]$ and any active component $S \in \mathcal{V}^t$ we must have*

$$\sum_{P \in \mathcal{P}_S} c(P) \leq 2 \sum_{S' \subseteq S} y_S^t - 2t$$

Proof. The claim is trivially true for $t = 0$.

Now, let $0 \leq t_1 \leq t_2 \leq \cdots \leq t_q \leq t^*$ be those points in time when active moats collide and the partial primal solution changes. We first show that the claim is true for $t \neq t_i$ for all all $1 \leq i \leq q$.

Let $t' = t_i = \max_{j:t_j \leq t} t_j$. By induction, we have that

$$\sum_{P \in \mathcal{P}_S} c(P) \leq 2 \sum_{S' \subseteq S} y_S^{t'} - 2t' \tag{4}$$

Notice, that no Steiner paths are added in the time interval $(t', t]$ and that $y_S^t - y_S^{t'} = (t - t')$. It then follows easily that (4) holds even for t' replaced by t.

Now, let $t = t_i$ for some $1 \leq i \leq q$. Let S_1 and S_2 be the two moats colliding at time t_i and let P be the S_1, S_2 Steiner path that is used to merge S_1 and S_2. It is not hard to see that

$$c(P) = \sum_{e \in P} c_e \leq 2t. \tag{5}$$

Hence,

$$\begin{aligned}
\sum_{P \in \mathcal{P}_{S_1 \cup S_2}} c(P) &\leq \sum_{P \in \mathcal{P}_{S_1}} c(P) + \sum_{P \in \mathcal{P}_{S_2}} c(P) + 2t \\
&\leq \sum_{S' \subseteq S_1} y_{S'}^t + \sum_{S' \subseteq S_2} y_{S'}^t - 4t + 2t \\
&= \sum_{S' \subseteq S} y_{S'}^t - 2t
\end{aligned}$$

where the second inequality follows from our induction hypothesis.

This immediately implies the following corollary.

Corollary 1. *The preceeding algorithm computes a* $(2-2/k)$*-approximate Steiner tree.*

Proof. Assume that the last moat collision in `AKR` happens at time t^*. We then must have

$$\sum_{S \in \mathcal{U}} y_S^{t^*} \leq k \cdot t^* \tag{6}$$

and hence we must have

$$\begin{aligned}
\sum_e c_e x_e^{t^*} &\leq \sum_{P \in \mathcal{P}} c(P) \\
&\leq 2 \sum_{S \in \mathcal{U}} y_S^{t^*} - 2t^* \\
&\leq (2 - 2/k) \cdot \sum_{S \in \mathcal{U}} y_S^{t^*} \\
&\leq (2 - 2/k) \cdot \texttt{OPT}.
\end{aligned}$$

The third inequality uses (6) and the last inequality uses weak duality.

3.2 An alternate view of AKR

Executing **AKR** on an undirected graph $G = (V, E)$ with terminal set $R \subseteq V$ and costs $\{c_e\}_{e\in E}$ is – in a certain sense – equivalent to executing **AKR** on the complete graph H with vertex set R where the cost of edge $e = uv \in R \times R$ is equal to the minimum cost of any u, v-path in G.

Let $\mathcal{P}_{S,T}$ denote the set of s, t-Steiner paths for all $s \in R \cap S$ and $t \in R \cap T$. For an s, t-Steiner path P, we define

$$c^{\lambda}(P) = c(P) + \lambda_s + \lambda_t + 2 \cdot \sum_{v\in \mathtt{int}(P)} \lambda_v$$

and finally, let

$$\mathtt{dist}_{c^{\lambda}}(S, T) = \min_{P\in\mathcal{P}_{S,T}} c^{\lambda}(P)$$

be the minimum c^{λ}-cost for any S, T-Steiner path. We now work with the following dual:

$$\max \quad \sum_{U\subset R} y_U - \sum_{v\in V} \lambda_v \cdot B_v \tag{D2}$$

$$\text{s.t} \quad \sum_{U\subset R, s\in U, t\notin U} y_U \le c^{\lambda}(P) \quad \forall s, t \in R, P \in \mathcal{P}_{s,t} \in E \tag{7}$$

$$y, \lambda \ge 0$$

Let H be a complete graph on vertex set R. We let the length of edge $(s, t) \in E[H]$ be $\mathtt{dist}_{c^{\lambda}}(s, t)$. Running **AKR** on input graph H with length function $\mathtt{dist}_{c^{\lambda}}$ yields a tree in H that corresponds to a Steiner tree spanning the nodes of R in G in a natural way. We also obtain a feasible dual solution for (D2). The following lemma shows that (D) and (D2) are equivalent.

Lemma 2. *(D) and (D2) are equivalent.*

Proof. Let $\{y_U\}_{\mathcal{U}}$ be a dual solution to (D). Define

$$\overline{y}_S = \sum_{U\in\mathcal{U}, U\cap R=S} y_U.$$

It is now easy to check that $\overline{y}$ is feasible for (D2) and that $\sum_{U\subset R} \overline{y}_U = \sum_{U\in\mathcal{U}} y_U$.

We now show how to convert a feasible solution $\{\overline{y}_U\}_{U\subset R}$ to (D2) into a feasible solution $\{y_S\}_{S\in\mathcal{U}}$ for (D). In order to do this, we keep a *budget* b_{uv} for each edge $uv \in E$ and let

$$b_{uv} = c_{uv} + \lambda_u + \lambda_v$$

for all $uv \in E$ initially. For a subset $S \subset R$ and a budget vector $\{b_e\}_{e\in E}$, we let

$$B(S, b) = \{u \in V \setminus R : \exists S, u\text{-path } P \text{ s.t. } b_e = 0 \text{ for all } e \in P\} \cup S.$$

Algorithm 1 An algorithm to convert a solution $\{\overline{y}_U\}_{U \subset R}$ for (D2) into a solution $\{y_S\}_{S \in \mathcal{U}}$ for (D).

```
Given: feasible solution {ȳ_U}_{U⊂R} for (D2).
b_e ← c_e for all e ∈ E
y'_U ← ȳ_U for all U ⊂ R
y_S ← 0 for all S ∈ 𝒰
while ∃U ⊂ R with y'_U > 0 do
   ε ← min{min_{e∈δ(B(U,b))} b_e, y'_U}
   y_{B(U,δ)} ← y_{B(U,δ)} + ε
   y'_U ← y'_U − ε
   b_e ← b_e − ε for all e ∈ δ(B(U,b))
end while
```

In other words, $B(S, b)$ contains S and all Steiner vertices u that are connected to S by at least one path of 0-budget edges.

The algorithm starts with $y'_U = \overline{y}_U$ for all $U \subset R$ and then incrementally constructs $\{y_S\}_{S \in \mathcal{U}}$. At any point in time, we look at a subset U of the terminals such that $y'_{U'} = 0$ for all $U' \subset U$. Given such a subset, we then increase $y_{B(U,b)}$ by the maximum $\epsilon > 0$ such that the resulting set of dual variables is still feasible for (D). The maximum such ϵ equals the minimum budget among the edges in $\delta(B(U, b))$. If $\epsilon > y'_U$ then let $\epsilon = y'_u$.

Afterwards, we decrease the budget b_e of all edges $e \in \delta(B(U, b))$ as well as variable y_U by ϵ. We continue this process until $y'_U = 0$ for all $U \subset R$. Algorithm 1 has the pseudo code for the conversion procedure.

It can be shown by induction that at any point during the algorithm and for any s, t-Steiner path P we must have

$$\sum_{S \in \mathcal{U}, |S \cap \{s,t\}|=1} y_S + \sum_{U \subset R, |U \cap \{s,t\}|=1} y'_U = \sum_{U \subset R, |U \cap \{s,t\}|=1} \overline{y}_U.$$

We leave the details to the reader.

This implies the termination of Algorithm 1. An immediate consequence is the feasibility of $\{y_S\}_{S \in \mathcal{U}}$ for (D) and

$$\sum_{S \in \mathcal{U}} y_S = \sum_{U \subset R} \overline{y}_U.$$

This finishes the proof of the claim.

4 An algorithm for the B-ST problem

In this section, we propose a modification of AKR in order to compute a feasible degree-bounded Steiner tree of low total cost. We start by giving a rough overview over our algorithm.

4.1 Algorithm: Overview

We define the normalized degree $\mathtt{ndeg}_T(v)$ of a node v in a Steiner tree T as

$$\mathtt{ndeg}_T(v) = \max\{0, \deg_T(v) - \beta_v \cdot B_v\} \tag{8}$$

where $\{\beta_v\}_{v \in V}$ are parameters to be defined later.

Algorithm `B-ST` goes through a sequence of Steiner trees $E^0, \ldots, E^t$ and associated pairs of primal (infeasible) and dual feasible solutions x^i, (y^i, λ^i) for $0 \leq i \leq t$. The goal is to reduce the maximum normalized degree of at least one node in V in the transition from one Steiner tree to the next.

In the i^{th} iteration our algorithm passes through two main steps:

Compute Steiner tree. Compute an approximate Steiner tree spanning the nodes of R for our graph $G = (V, E)$ using a modified version of `AKR`. Roughly speaking, this algorithm implicitly assumes a cost function $\widetilde{c}$ that satisfies

$$c(P) \leq \widetilde{c}(P) \leq c^{\lambda^i}(P) \tag{9}$$

for all Steiner paths P.

When the algorithm finishes, we obtain a primal solution x^i together with a corresponding dual solution y^i. In the following we use $\mathcal{P}^i$ to denote the set of paths used by `AKR` to connect the terminals in iteration i.

Notice that using the cost function $\widetilde{c}$ that satisfies (9) ensures that (y^i, λ^i) is a feasible solution for (D2). The primal solution x^i may induce high normalized degree at some of the vertices of V and hence may not be feasible for (IP).

Update node multipliers. The main goal here is to update the node multipliers λ^i such that another run of `AKR` yields a tree in which the normalized degree of at least one node decreases. Specifically, we continue running our algorithm as long as the maximum normalized node-degree induced by x^i is at least $2 \log_b n$ where $b > 1$ is a positive constant to be specified later.

Let Δ^i be the maximum normalized degree of any node in the tree induced by x^i. The algorithm then picks a threshold $d^i \geq \Delta^i - \lceil 4 \log_b n \rceil + 2$. Subsequently we raise the λ values of all nodes that have normalized degree at least $d^i - 2$ in the tree induced by x^i by some $\epsilon^i > 0$. We also implicitly increase the $\widetilde{c}$ cost of two sets of Steiner paths:

1. those paths $P \in \mathcal{P}^i$ that contain nodes of degree at least d^i and
2. those paths $P \notin \mathcal{P}^i$ that contain nodes of degree at least $d^i - 2$.

We denote to the set of all such paths by $\mathcal{L}^i$.

Rerunning `AKR` replaces at least one Steiner path whose $\widetilde{c}$-cost increased with a Steiner path whose length stayed the same. In other words, a path that touches a node of normalized degree at least d^i is replaced by some other path that has only nodes of normalized degree less than $d^i - 2$.

Throughout the algorithm we will maintain that the cost of the current tree induced by x^i is within a constant factor of the dual objective function value induced by (y^i, λ^i). By weak duality, this ensures that the cost of our tree is within a constant times the cost of any Steiner tree that satisfies the individual degree bounds. However, we are only able to argue that the number of iterations is quasi-polynomial.

In the following, we will give a detailed description of the algorithm. In particular, we elaborate on the choice of ϵ^i and d^i in the node-multiplier update and on the modification to `AKR` that we have alluded to in the previous intuitive description.

4.2 Algorithm: A detailed top-level description

We first present the pseudo-code of our `B-ST`-algorithm. In the description of the algorithm we use the abbreviation $\texttt{ndeg}^i(v)$ in place of $\texttt{ndeg}_{E^i}(v)$ for the normalized degree of vertex v in the Steiner tree E^i. We also let Δ^i denote the maximum normalized degree of any vertex in E^i, i.e.

$$\Delta^i = \max_{v \in R} \texttt{ndeg}^i(v).$$

Furthermore, we adopt the notation of [2] and let

$$S_d^i = \{v \in V \,:\, \texttt{ndeg}^i(v) \geq d\}$$

be the set of all nodes whose normalized degrees are at least d in the i^{th} solution.

The following lemma is proved easily by contradiction

Lemma 3. *There is a $d^i \in \{\Delta^i - \lceil 4 \log_b n \rceil + 2, \ldots, \Delta^i\}$ such that*

$$\sum_{v \in S^i_{d^i - 2}} B_v \leq b \cdot \sum_{v \in S^i_{d^i}} B_v$$

for a given constant $b > 1$.

Proof. Suppose for a contradiction that for all $d^i \in \{\Delta^i - \lceil 4 \log_b n \rceil + 2, \ldots, \Delta^i\}$, we have

$$\sum_{v \in S^i_{d^i - 2}} B_v > b \cdot \sum_{v \in S^i_{d^i}} B_v .$$

Note that since we may assume $B_v \leq (n-1)$ for all vertices, we must have $\sum_{v \in V} B_v \leq n(n-1)$. However, since $\sum_{v \in S^i_{\Delta^i}} B_v \geq 1$, we have in this case that

$$\sum_{v \in S^i_{\Delta^i - \lceil 4 \log_b n \rceil}} B_v \geq b^{\lceil 2 \log_b (n) \rceil} \geq n^2$$

a contradiction.

The *low expansion* of $S^i_{d^i - 2}$ turns out to be crucial in the analysis of the performance guarantee of our algorithm.

Finally, we let `mod-AKR` denote a call to the modified version of the Steiner tree algorithm `AKR`. Algorithm 2 has the pseudo code for our method.

Algorithm 2 An algorithm to compute an approximate minimum-cost degree-bounded Steiner tree.

1: Given: primal feasible solution $x^0, \mathcal{P}^0$ to (LP-ST) and dual feasible solution y^0 to (D-ST)
2: $\lambda_v^0 \leftarrow 0, \forall v \in V$
3: $i \leftarrow 0$
4: **while** $\Delta^i > 4\lceil \log_b n \rceil$ **do**
5:     Choose $d^i \geq \Delta^i - \lceil 4 \log_b n \rceil + 2$ s.t. $\sum_{v \in S^i_{d^i-2}} B_v \leq b \cdot \sum_{v \in S^i_{d^i}} B_v$
6:     Choose $\epsilon^i > 0$ and identify swap pair $\langle P^i, \overline{P}^i \rangle$.
7:     $\lambda_v^{i+1} \leftarrow \lambda_v^i + \epsilon^i$ if $v \in S^i_{d^i-2}$ and $\lambda_v^{i+1} \leftarrow \lambda_v^i$ otherwise
8:     $y^{i+1} \leftarrow$ `mod-AKR`$(\mathcal{P}^i, \epsilon^i, y^i, \langle P^i, \overline{P}^i \rangle)$
9:     $\mathcal{P}^{i+1} \leftarrow \mathcal{P}^i \setminus \{P^i\} \cup \{\overline{P}^i\}$
10:     $i \leftarrow i + 1$
11: **end while**

Step 6 of Algorithm 2 hides the details of choosing an appropriate ϵ^i. We lengthen all Steiner paths in $\mathcal{L}^i$. Our choice of ϵ^i will ensure that there exists at least one point in time during the execution of a slightly modified version of `AKR` in step 8 at which we now have the choice to connect two moats using paths P^i and $\overline{P}^i$, respectively. We show that there is a way to pick ϵ^i such that

$$P^i \cap S^i_{d^i} \neq \emptyset \text{ and } \overline{P}^i \cap S^i_{d^i-2} = \emptyset.$$

We now break ties such that $\overline{P}^i$ is chosen instead of P^i and hence, we end up with a new Steiner tree E^{i+1}.

In `mod-AKR`, we prohibit including alternate paths that contain nodes from $S^i_{d^i-2}$ and argue that the dual load that such a non-tree path P' sees does not go up by more than ϵ^i. Hence, we preserve dual feasibility.

We first present the details of Algorithm `mod-AKR` and discuss how to find ϵ^i afterwards.

4.3 Algorithm: `mod-AKR`

Throughout this section and the description of `mod-AKR` we work with the modified dual (D2) as discussed in Section 3.2.

For a r_1, r_2-Steiner path P we let $R_P \subseteq 2^R$ denote all sets $S \subset R$ that contain exactly one of $r_1, r_2 \in R$. For a dual solution y, λ we then define the cut-metric $l_y(P) = \sum_{S \in R_P} y_s$. From here it is clear that (y, λ) is a feasible dual solution iff $l_y(P) \leq c^\lambda(P)$ for all Steiner paths P. We use $l^i(P)$ as an abbreviation for $l_{y^i}(P)$.

At all times during the execution of Algorithm 2 we want to maintain dual feasibility, i.e. we maintain

$$l^i(P) \leq c^{\lambda^i}(P) \tag{10}$$

for all Steiner paths P and for all i. Moreover, we want to maintain that for all i, the cost of any path $P \in \mathcal{P}^i$ is bounded by the dual load that P sees. In other words, we want to enforce that

$$c(P) \leq l^i(P) \tag{11}$$

for all $P \in \mathcal{P}^i$ and for all i. It is easy to see that both (10) and (11) hold for $i = 0$ from the properties of `AKR`.

First, let $\mathcal{P}^i = \mathcal{P}_1^i \cup \mathcal{P}_2^i$ be a partition of the set of Steiner paths used to connect the terminal nodes in the i^{th} iteration. Here, a path $P \in \mathcal{P}^i$ is added to $\mathcal{P}_1^i$ iff $P \cap S_{d^i}^i \neq \emptyset$ and we let $\mathcal{P}_2^i = \mathcal{P}^i \setminus \mathcal{P}_1^i$.

`mod-AKR` first constructs an auxiliary graph G^i with vertex set R. We add an edge (s,t) to G^i for each s,t-path $P \in \mathcal{P}^i \setminus \{P^i\}$. The edge (s,t) is then assigned a length of $l_{st}^{i+1} = l^i(P) + \epsilon^i$ if $P \in \mathcal{P}_1^i$ and $l_{st}^{i+1} = l^i(P)$ otherwise.

Assume that $\overline{P}^i$ is an s',t'-path. We then also add an edge connecting s' and t' to G^i and let its length be the maximum of $l^i(\overline{P}^i)$ and $c(\overline{P}^i)$. Observe, that since $\mathcal{P}^i \setminus \{P^i\} \cup \{\overline{P}^i\}$ is tree, G^i is a tree as well.

Subsequently, `mod-AKR` runs `AKR` on the graph G^i and returns the computed dual solution. We will show that this solution together with λ^{i+1} is feasible for (D2). A formal definition of `mod-AKR` is given in Algorithm 3.

Algorithm 3 `mod-AKR`$(\mathcal{P}^i, \epsilon^i, y^i, \langle P^i, \overline{P}^i \rangle)$: A modified version of `AKR`.

1: Assume $\overline{P}^i$ is an s',t'-Steiner path
2: $G^i = (R, E^i)$ where
   $E^i = \{(s,t) : \exists s,t-\text{path } P \in \mathcal{P}^i \setminus \{P^i\}\} \cup \{(s',t')\}$
3: For all s,t Steiner paths $P \in \mathcal{P}^i \setminus \{P^i\}$:

$$l_{st}^{i+1} = \begin{cases} l^i(P) + \epsilon^i & : \quad P \in \mathcal{P}_1^i \\ l^i(P) & : \quad \text{otherwise} \end{cases}$$

4: $l_{s't'}^{i+1} = \max\{c(\overline{P}^i), l^i(\overline{P}^i)\}$
5: $y^{i+1} \leftarrow$ `AKR`(G^i, l^{i+1})
6: **return** y^{i+1}

We defer the proof of invariants (10) and (11) to the end of the next section.

4.4 Algorithm: Choosing ϵ^i

In this section, we show how to choose ϵ^i. Remember that, intuitively, we want to increase the cost of currently used Steiner paths that touch nodes of normalized degree at least d^i. The idea is to increase the cost of such paths by the smallest possible amount such that other non-tree paths whose length we did not increase can be used at their place. We make this idea more precise in the following.

We first define $\mathcal{K}^i$ to be the set of connected components of

$$G\left[\bigcup_{P\in\mathcal{P}_2^i} P\right].$$

Let H^i be an auxiliary graph that has one node for each set in $\mathcal{K}^i$. Moreover, H^i contains edge (K', K'') iff there is a K', K''-Steiner path in the set $\mathcal{P}_1^i$. It can be seen that each path $P \in \mathcal{P}_1^i$ corresponds to unique edge in H^i. It then follows from the fact that $G[E^i]$ is a tree that H^i must also be a tree.

For $K', K'' \in \mathcal{K}^i$ such that (K', K'') is not an edge of H^i, let C be the unique cycle in $H^i + (K', K'')$. We then use $\mathcal{P}^i(C)$ to denote the set of Steiner paths from $\mathcal{P}^i$ corresponding to edges on C.

For any two connected components $K', K'' \in \mathcal{K}^i$ we let

$$d^i(K', K'') = \min_{\substack{P\in\mathcal{P}_{K',K''}\\ P\cap S^i_{d^i-2}=\emptyset}} c(P). \tag{12}$$

be the cost of the minimum-cost K', K''-Steiner path that *avoids* nodes from $S^i_{d^i-2}$. For a pair of components $K', K'' \in \mathcal{K}^i$ we denote the path that achieves the above minimum by $P_{K',K''}$.

Definition 1. *We say that a path $\overline{P} \notin \mathcal{P}^i$ that contains no nodes from $S^i_{d^i-2}$ is ϵ-swappable against $P \in \mathcal{P}_1^i$ in iteration i if*

1. *$P \in \mathcal{P}^i(C)$ where C is the unique cycle created in H^i by adding the edge corresponding to $\overline{P}$, and*
2. *$c(\overline{P}) \leq l^i(P) + \epsilon$*

We are now looking for the smallest ϵ^i such that there exists a *witness pair* of paths $\langle P^i, \overline{P}^i\rangle$ where $\overline{P}^i$ is ϵ^i-swappable against P^i.

Formally consider all pairs $K', K'' \in \mathcal{K}^i$ such that (K', K'') is not an edge of H^i. Inserting the edge corresponding to $P_{K',K''}$ into H^i creates a unique cycle C. For each such path $P \in \mathcal{P}^i(C)$, let $\epsilon^i_{K',K''}(P)$ be the smallest non-negative value of ϵ such that

$$d^i(K', K'') \leq l^i(P) + \epsilon. \tag{13}$$

We then let $\epsilon^i_{K',K''} = \min_{P\in\mathcal{P}^i(C)} \epsilon^i_{K',K''}(P)$ and define

$$\epsilon^i = \min_{K',K''\in\mathcal{K}^i} \epsilon^i_{K',K''}.$$

We let $\langle P^i, \overline{P}^i\rangle$ be the pair of Steiner paths that defines ϵ^i, i.e. $\overline{P}^i$ is a K', K''-Steiner path such that

1. inserting edge (K', K'') into H^i creates a cycle C and $P^i \in \mathcal{P}^i(C)$, and
2. $c(\overline{P}^i) \leq l^i(P^i) + \epsilon^i$.

We are now in the position to show that (10) and (11) are maintained for our choice of $(P^i, \overline{P}^i)$ and ϵ^i. The following Lemma whose proof is deferred to the full version of this paper shows that `mod-AKR` produces a feasible dual solution (y^{i+1}, λ^{i+1}) for (D2) provided that (y^i, λ^i) was dual feasible.

Lemma 4. *Algorithm 3 produces a feasible dual solution (y^{i+1}, λ^{i+1}) for (D2) given that (y^i, λ^i) is dual feasible for (D2).*

Proof. The proof is by contradiction. Assume there is a Steiner path P such that $l^{i+1}(P) > c^{\lambda^{i+1}}(P)$. First of all, notice that this Steiner path must have endpoints $s \in K'$ and $t \in K''$ for $K', K'' \in \mathcal{K}^i$ such that $K' \neq K''$. The reason for this is that

$$l^{i+1}(P) = l^i(P)$$

for all Steiner paths in $\mathcal{P}^i$ that connect terminals $s, t \in K$ for some $K \in \mathcal{K}^i$. This also implies that $l^{i+1}(P) = l^i(P)$ for all s, t-Steiner paths $P \notin \mathcal{P}^i$ such that $s, t \in K$ and $K \in \mathcal{K}^i$.

Secondly, notice that P must be different from $\overline{P}^i$ since the way we choose the length of the edge corresponding to $\overline{P}^i$ in step 4 of `mod-AKR` ensures that the dual load on this path does not exceed the maximum of the previous load and the cost of the path.

We consider two main cases:

Case 1: ($P \in \mathcal{P}^{i+1} \setminus \{\overline{P}^i\}$) In this case, step 3 of `mod-AKR` determines the dual load of P. We have

$$l^{i+1}(P) = l^i(P) + \epsilon^i.$$

Now, observe that

$$c^{\lambda^{i+1}}(P) = c^{\lambda^i}(P) + |P \cap S^i_{d^i-2}|\epsilon^i \geq c^{\lambda^i}(P) + \epsilon^i$$

which shows that we must have $l^{i+1}(P) \leq c^{\lambda^{i+1}}(P)$.

Case 2: ($P \notin \mathcal{P}^{i+1}$) In this case, we can assume that $\epsilon^i > 0$ since $l^{i+1}(P) = l^i(P)$ for all Steiner paths P otherwise.

We subdivide into two sub cases. Let's first assume that $P \cap S^i_{d^i-2} = \emptyset$. Let C be the cycle in $H^i + P$. Then, there must be some path $P' \in \mathcal{P}^i(C)$ such that

$$l^{i+1}(P) = l^{i+1}(P').$$

It follows from the choice of ϵ^i in (13) and the fact that $P \cap S^i_{d^i-2} = \emptyset$ that

$$l^{i+1}(P') \leq c(P).$$

The right hand side of this inequality is clearly at most $c^{\lambda^i}(P) = c^{\lambda^{i+1}}(P)$. Assume now that $P \cap S^i_{d^i-2} \neq \emptyset$. In this case, we must have that

$$l^{i+1}(P) \leq l^i(P) + \epsilon^i.$$

The claim follows since $l^i(P) \le c^{\lambda^i}(P)$ and

$$c^{\lambda^{i+1}}(P) \ge c^{\lambda^i}(P) + \epsilon^i$$

since P contains nodes from $S^i_{d^i-2}$.

This shows (10). It is clear from the choice of ϵ^i that we include a Steiner path $\overline{P}^i$ into $\mathcal{P}^{i+1}$ only if $l^{i+1}(\overline{P}^i) \ge c(\overline{P}^i)$. (11) now follows since the dual load on any path is non-decreasing as we progress.

4.5 Analysis: Performance guarantee

In this section we show that the cost of the tree computed by Algorithm 2 is within a constant factor of any Steiner tree satisfying all degree bounds. We ensure this by way of weak duality. In particular, our goal is to prove the inequality

$$\sum_{P \in \mathcal{P}^i} c(P) \le 3 \sum_{S \subset R} y^i_S - 3 \sum_{v \in V} B_v \cdot \lambda^i_v \tag{14}$$

for all iterations i of our algorithm.

First, we observe the following simple consequence of the `AKR` algorithm.

Lemma 5. *Assume that Algorithm 2 terminates after t iterations. For iteration $0 \le i \le t$, let $l^i_{\max} = \max_{P \in \mathcal{P}^i} l^i(P)$. We then must have*

$$\sum_{P \in \mathcal{P}^i} l^i(P) = 2 \sum_{S \subset R} y^i_S - l^i_{\max}.$$

Proof. Let $r = |R|$ and let $\mathcal{P}^i = \{P^i_1, \ldots, P^i_{r-1}\}$ be the paths computed by `mod-AKR` in iteration $i-1$. Also let y^i be the corresponding dual solution returned by `mod-AKR`. W.l.o.g. we may assume that

$$l^i(P^i_1) \le \ldots \le l^i(P^i_{r-1}).$$

From the `AKR`algorithm it is not hard to see that

$$\begin{aligned}
\sum_{S \subset R} y^i_S &= \frac{1}{2} \cdot \sum_{j=1}^{r-1} (l^i(P^i_j) - l^i(P^i_{j-1})) \cdot (r - j + 1) \qquad (15)\\
&= \frac{1}{2} \cdot \sum_{j=1}^{r-1} l^i(P^i_j)\,((r-j+1) - (r-j)) + \frac{1}{2} l^i(P^i_{r-1}) \\
&= \frac{1}{2} \cdot \sum_{j=1}^{r-1} l^i(P^i_j) + \frac{1}{2} l^i(P^i_{r-1})
\end{aligned}$$

where we define $l^i(P^i_0) = 0$. The last equality (15) can be restated as

$$\sum_{P \in \mathcal{P}^i} l^i(P) = 2 \sum_{S \subset R} y^i_S - l^i_{\max}$$

and that yields the correctness of the lemma.

We now proceed with proving (14) for all $1 \le i \le t$. Notice that Lemma 5 together with (11) implies (14) for $i = 0$. We concentrate on the case $i \ge 1$.

The proof is based on the following invariant that we maintain inductively for all $0 \le i \le t$:

$$3 \cdot \sum_{v \in V} B_v \lambda_v^i \le \sum_{S \subset R} y_S^i. \tag{Inv}$$

Since, $\lambda_v^0 = 0$ for all $v \in V$ by definition, (Inv) holds for $i = 0$.

Growing λ_v^i by ϵ^i at nodes $v \in S^i_{d^i-2}$ decreases the right hand side of (14) by $3 \cdot \epsilon \cdot \sum_{v \in S^i_{d^i-2}} B_v$. Still the cost of the Steiner tree E^{i+1} is potentially higher than the cost of the old tree E^i. We must show that the first term on the right hand-side of (14), i.e. $3 \cdot \sum_{S \subset R} y_S^i$ grows sufficiently to compensate for the decrease in the second term and the increased Steiner tree cost. In order to show this we need the following technical lemma that lower-bounds the number of paths that contain nodes of degree at least d^i in terms of the number of nodes of normalized degree at least $d^i - 2$.

Lemma 6. *In each iteration $1 \le i \le t$ we must have*

$$|\mathcal{P}_1^i| \ge \alpha \cdot \sum_{v \in S^i_{d^i-2}} B_v$$

for an arbitrary parameter $\alpha > 0$ by setting $\beta_v \ge 2\alpha b + 1/B_v$ for all $v \in V$ in the definition of $\mathtt{ndeg}_T(v)$ in (8).

Proof. We first define a set of marked edges

$$M \subseteq \bigcup_{v \in S^i_{d^i}} \delta(v)$$

and then show that each Steiner path that contains nodes from $S^i_{d^i}$ has at most two marked edges. This shows that the cardinality of the set of marked edges is at most twice the number of paths in $\mathcal{P}_1^i$, i.e.

$$|M| \le 2 \cdot |\mathcal{P}_1^i|. \tag{16}$$

In the second part of the proof we argue that M is sufficiently large.

First, we include all edges that are incident to terminal nodes from $S^i_{d^i}$ into M. Secondly, we also mark edges $uv \in E^i$ that are incident to non-terminal nodes in $S^i_{d^i}$ and that in addition satisfy that there is no Steiner path

$$P = \langle P_1, uv, P_2 \rangle \in \mathcal{P}^i$$

such that both P_1 and P_2 contain nodes from $S^i_{d^i}$.

It is immediately clear from this definition that each Steiner path $P \in \mathcal{P}^i$ has at most two edges from M.

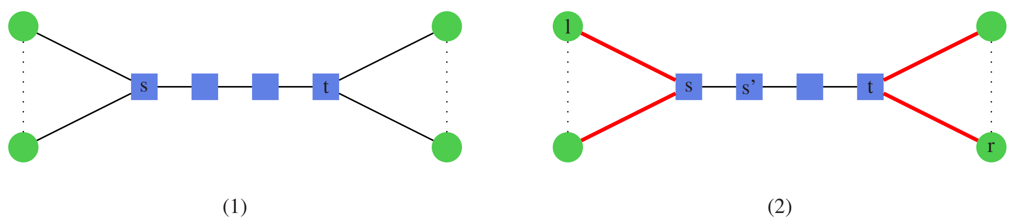

Fig. 1. Figure (1) shows a Steiner tree where circles represent terminals and squares represent Steiner nodes. We assume that there are exactly two nodes of high normalized degree: s and t. Figure (2) shows the set M of marked edges in red. Notice that the edge between Steiner nodes s and s' is not marked since there must be a Steiner path connecting a terminal node l on the left side and a terminal node r on the right side. This Steiner path has the form $\langle P_{ls}, ss', P_{s'r}\rangle$ and P_{ls} contains node s which has high normalized degree.

We now claim that M contains at least

$$b\alpha \cdot \sum_{v \in S^i_{d^i}} B_v \tag{17}$$

edges. To see this, we let T be the tree on node set $S^i_{d^i}$ that is induced by E^i: For $s, t \in S^i_{d^i}$ we insert the edge st into T iff the unique s,t-path in E^i has no other nodes from $S^i_{d^i}$. We let $P_e \subseteq E^i$ be the path that corresponds to an edge $e \in E[T]$.

Define $E^i_{d^i} \subseteq E^i$ to be the set of tree edges that are incident to nodes of normalized degree at least d^i, i.e.

$$E^i_{d^i} = \bigcup_{v \in S^i_{d^i}} \delta(v).$$

Now let $U \subseteq E^i$ be the set of unmarked tree edges that are incident to nodes of normalized degree at least d^i, i.e. $U = E^i_{d^i} \setminus M$.

First observe that, by definition of M, for each unmarked edge $e \in U$ there must be an edge $e^t \in E[T]$ such that e is an edge on the path P_{e^t}. Moreover, for all $e_t \in E[T]$ there are at most two unmarked edges on the path P_{e^t}. Since T has $|S^i_{d^i}| - 1$ edges we obtain

$$|U| \leq 2 \cdot (|S^i_{d^i}| - 1). \tag{18}$$

Each node in $S^i_{d^i}$ has at least $\beta_v B_v + d^i$ edges incident to it. On the other hand, since E^i is a tree, at most $(|S^i_{d^i}| - 1)$ of the edges in $E^i_{d^i}$ are incident to exactly two nodes from $S^i_{d^i}$. Hence, we obtain

$$|E^i_{d^i}| \geq \left(\sum_{v \in S^i_{d^i}} \beta_v B_v + d^i \right) - (|S^i_{d^i}| - 1) = \left(2\alpha b \cdot \sum_{v \in S^i_{d^i}} B_v \right) + d^i \cdot |S^i_{d^i}| + 1 \tag{19}$$

where the last equality uses the definition of β_v.

Now observe that $|M| = |E^i_{d^i}| - |U|$ and hence

$$|M| \geq \left(2\alpha b \cdot \sum_{v \in S^i_{d^i}} B_v \right) + |S^i_{d^i}|(d^i - 2) - 1. \tag{20}$$

using (18) and (19). Notice that $d^i \geq \Delta^i - \lceil 4 \log_b n \rceil + 2$ and $\Delta^i > \lceil 4 \log_b n \rceil$ and hence $d^i \geq 3$. This together with (20) and the fact that $S^i_{d^i}$ is non-empty implies

$$|M| \geq 2\alpha b \cdot \sum_{v \in S^i_{d^i}} B_v. \tag{21}$$

Combining (16) and (21) yields $|\mathcal{P}^i_1| \geq \alpha b \cdot \sum_{v \in S^i_{d^i}} B_v$. Using the fact that $\sum_{v \in S^i_{d^i-2}} B_v \leq b \cdot \sum_{v \in S^i_{d^i}} B_v$ finishes the proof of the lemma.

The following claim now presents the essential insight that ultimately yields the validity of (14).

Lemma 7. *Let α be as in Lemma 6. We then must have*

$$\sum_{S \subset R} y^{i+1}_S \geq \sum_{S \subset R} y^i_S + \frac{\alpha}{2}\epsilon^i \cdot \sum_{v \in S^i_{d^i-2}} B_v$$

for all $0 \leq i \leq t$.

Proof. We can use (15) to quantify the change in dual in iteration i.

$$\begin{aligned}
\sum_{S \subset R} (y^{i+1}_S - y^i_S) &= \frac{1}{2} \cdot \sum_{j=1}^{r-1} (l^{i+1}(P^i_j) - l^i(P^i_j)) + \frac{1}{2}(l^{i+1}(P^i_{r-1}) - l^{i+1}(P^i_{r-1})) \\
&\geq \frac{\epsilon^i}{2} \cdot |\mathcal{P}^i_1|
\end{aligned}$$

where the inequality follows from the fact that we increase the length of all paths in $\mathcal{P}^i_1$ by ϵ^i and the length of all other paths are non-decreasing as we progress. An application of Lemma 6 finishes the proof.

As `mod-AKR` finishes with cut metric l^{i+1}, we obtain

$$l^{i+1}(\mathcal{P}^{i+1}) = \sum_{P \in \mathcal{P}^{i+1}} l^{i+1}(P) \leq 2 \sum_{S \subset R} y^{i+1}_S \tag{22}$$

from Lemma 5. Observe that the real cost of the Steiner tree E^{i+1} is much smaller than $l^{i+1}(\mathcal{P}^{i+1})$. In fact, notice that we have

$$\begin{aligned}
c(\mathcal{P}^{i+1}) &\leq l^{i+1}(\overline{P}^i) + c(\mathcal{P}^i \setminus \{P^i\}) \\
&\leq l^{i+1}(\overline{P}^i) + l^i(\mathcal{P}^i \setminus \{P^i\})
\end{aligned} \tag{23}$$

where the last inequality follows from (11), i.e. the l-cost of a Steiner path in $\mathcal{P}^i$ always dominates its c-cost. Also, observe that

$$\begin{aligned} l^{i+1}(\mathcal{P}^i \setminus \{P^i\}) &= l^i(\mathcal{P}^i \setminus \{P^i\}) + \epsilon^i \cdot |\mathcal{P}_i^1| \\ &\geq l^i(\mathcal{P}^i \setminus \{P^i\}) + \alpha\epsilon^i \cdot \sum_{v \in S^i_{d^i-2}} B_v \end{aligned} \tag{24}$$

using Lemma 6. Combining (22), (23) and (24) yields

$$\begin{aligned} c(\mathcal{P}^{i+1}) &\leq l^{i+1}(\mathcal{P}^{i+1}) - \alpha\epsilon^i \cdot \sum_{v \in S^i_{d^i-2}} B_v \\ &\leq 2 \cdot \sum_{S \subset R} y_S^{i+1} - \alpha\epsilon^i \cdot \sum_{v \in S^i_{d^i-2}} B_v\,. \end{aligned}$$

We can now add (Inv) to the last inequality and get

$$c(\mathcal{P}^{i+1}) \leq 3 \sum_{S \subset R} y_S^{i+1} - 3 \cdot \sum_{v \in V} B_v\,\lambda_v^i - \alpha\epsilon^i \cdot \sum_{v \in S^i_{d^i-2}} B_v\,.$$

Finally notice that $\lambda_v^{i+1} = \lambda_v^i + \epsilon^i$ if $v \in S^i_{d^i-2}$ and $\lambda_v^{i+1} = \lambda_v^i$ otherwise. Now choose $\alpha \geq 3$ and it follows that

$$c(\mathcal{P}^{i+1}) \leq 3 \sum_{S \subset R} y_S^{i+1} - 3 \cdot \sum_{v \in V} B_v\,\lambda_v^{i+1}.$$

We have to show that (Inv) is maintained as well. Observe that the left hand side of (Inv) increases by $3\epsilon^i \cdot \sum_{v \in S^i_{d^i-2}} B_v$. We obtain from Lemma 7 that

$$\sum_{S \subset R} y_S^{i+1} - y_S^i \geq \frac{\alpha}{2} \cdot \epsilon^i \cdot \sum_{v \in S^i_{d^i-2}} B_v\,.$$

Choosing $\alpha \geq 6$ shows that the right hand side of (Inv) increases sufficiently and (Inv) holds in iteration $i+1$ as well.

4.6 Analysis: Running time

In this section, we show that Algorithm 2 terminates within a quasi-polynomial number of steps. We accomplish this by showing that there will be only a quasi-polynomial number of iterations of the main loop in Algorithm 2.

For a Steiner tree $\mathcal{P}$ in path representation, we define its potential value as

$$\Phi(\mathcal{P}) = \sum_{P \in \mathcal{P}} |R|^{\max_{v \in P} \mathtt{ndeg}_{\mathcal{P}}(v)}$$

where $\mathtt{ndeg}_{\mathcal{P}}(v)$ is the normalized degree of node v in the Steiner tree defined by $\mathcal{P}$. The proof of the following lemma is a direct adaptation of the arguments in [8] via the above potential function.

Lemma 8. *Algorithm 2 terminates after* $O(n \log(|R|) \cdot |R|^{\lceil 4 \log n \rceil}))$ *iterations.*

Proof. For a Steiner tree $\mathcal{P}$ in path representation, we define its potential value as

$$\Phi(\mathcal{P}) = \sum_{P \in \mathcal{P}} |R|^{\max_{v \in P} \mathtt{ndeg}_{\mathcal{P}}(v)}$$

where $\mathtt{ndeg}_{\mathcal{P}}(v)$ is the normalized degree of node v in the Steiner tree defined by $\mathcal{P}$.

Now notice that an iteration of Algorithm 2 swaps out a path $P^i \in \mathcal{P}_1^i$ that has at least one node of normalized degree at least d^i. Hence, the maximum normalized degree of any node on P^i must be at least d^i. The path $\overline{P}^i$ on the other hand does not touch any node of normalized degree more than $d^i - 3$.

This means that a swap may potentially raise the maximum normalized node-degree of at most $|R| - 1$ Steiner paths to $d^i - 1$. The reduction in the potential is hence at least

$$(|R|^{d^i} - (|R| - 1)|R|^{d^i - 1}) \geq |R|^{d^i - 1}$$

Using the fact that $d^i \geq \Delta^i - \lceil 4 \log_b n \rceil + 2$, we can lower-bound the right hand side of the last inequality by

$$|R|^{\Delta^i - \lceil 4 \log_b n \rceil + 1} = \Omega\left(\frac{|R|^{\Delta^i + 1}}{|R|^{\lceil 4 \log n \rceil}} \right).$$

The initial potential Φ^i is at most

$$|R| \cdot |R|^{\Delta^i}$$

and the decrease in the potential is at least

$$\Omega\left(\frac{\Phi^i}{|R|^{\lceil 4 \log n \rceil}} \right).$$

In other words, in $O(|R|^{\lceil 4 \log n \rceil})$ iterations, we reduce Φ by a constant factor. Hence, the algorithm needs

$$O(|R|^{\lceil 4 \log n \rceil} \cdot \ln \Phi^0) = O(n \log(|R|) \cdot |R|^{\lceil 4 \log n \rceil})$$

iterations total.

References

1. A. Agrawal, P. Klein and R. Ravi, “When trees collide: an approximation algorithm for the generalized Steiner problem on networks,” *SIAM Journal on Computing*, vol.24, (1995), pp. 440-456.
2. M. Fürer, B. Raghavachari: Approximating the minimum degree Steiner tree to within one of optimal. Journal of Algorithms, **17** (1994) 409–423

3. M. X. Goemans and D. P. Williamson, "A general approximation technique for constrained forest problems," *SIAM J. Computing*, Vol. 24, 1995, pp. 296–317.
4. J. Könemann, R. Ravi: A matter of degree: improved approximation algorithms for degree-bounded minimum spanning trees. Proceedings of the 32nd Symp. on the Theory of Comput. (2000) 537–546. A complete version appears in the SIAM J. on Comput, 31(6), 1783–1793 (2002)
5. J. Könemann, R. Ravi: Primal-dual algorithms come of age: Approximating MST's with nonuniform degree bounds. Proceedings, ACM Symposium on Theory of Computing (2003)
6. R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, H. B. Hunt III: Many Birds with One Stone: Multi-Objective Approximation Algorithms (Extended Abstract). Proceedings of the 25th Symp. on the Theory of Comput. (1993) 438–447
7. R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, H. B. Hunt III: Approximation Algorithms for Degree-Constrained Minimum-Cost Network-Design Problems. Algorithmica **31(1)** (2001) 58–78
8. R. Ravi, B. Raghavachari, and P. N. Klein, "Approximation through local optimality: Designing networks with small degree," *Proc., 12th Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FST & TCS)*, December (1992), LNCS 652, pp. 279-290.