Budgeted Out-Tree Maximization with Submodular Prizes

Gianlorenzo D'Angelo
Gran Sasso Science Institute, L'Aquila, Italy

Esmaeil Delfaraz
Gran Sasso Science Institute, L'Aquila, Italy

Hugo Gilbert
Université Paris-Dauphine, Université PSL, CNRS, LAMSADE, 75016 Paris, France

Abstract

We consider a variant of the prize collecting Steiner tree problem in which we are given a *directed graph* $D = (V, A)$, a monotone submodular prize function $p : 2^V \to \mathbb{R}^+ \cup \{0\}$, a cost function $c : V \to \mathbb{Z}^+$, a root vertex $r \in V$, and a budget B. The aim is to find an out-subtree T of D rooted at r that costs at most B and maximizes the prize function. We call this problem *Directed Rooted Submodular Tree* (**DRST**).

For the case of undirected graphs and additive prize functions, Moss and Rabani [SIAM J. Comput. 2007] gave an algorithm that guarantees an $O(\log |V|)$-approximation factor if a violation by a factor 2 of the budget constraint is allowed. Bateni et al. [SIAM J. Comput. 2018] improved the budget violation factor to $1 + \varepsilon$ at the cost of an additional approximation factor of $O(1/\varepsilon^2)$, for any $\varepsilon \in (0, 1]$. For directed graphs, Ghuge and Nagarajan [SODA 2020] gave an optimal quasi-polynomial time $O\left(\frac{\log n'}{\log \log n'}\right)$-approximation algorithm, where n' is the number of vertices in an optimal solution, for the case in which the costs are associated to the edges.

In this paper, we give a polynomial time algorithm for **DRST** that guarantees an approximation factor of $O(\sqrt{B}/\varepsilon^3)$ at the cost of a budget violation of a factor $1 + \varepsilon$, for any $\varepsilon \in (0, 1]$. The same result holds for the edge-cost case, to the best of our knowledge this is the first polynomial time approximation algorithm for this case. We further show that the unrooted version of **DRST** can be approximated to a factor of $O(\sqrt{B})$ without budget violation, which is an improvement over the factor $O(\Delta\sqrt{B})$ given in [Kuo et al. IEEE/ACM Trans. Netw. 2015] for the undirected and unrooted case, where Δ is the maximum degree of the graph. Finally, we provide some new/improved approximation bounds for several related problems, including the additive-prize version of **DRST**, the maximum budgeted connected set cover problem, and the budgeted sensor cover problem.

2012 ACM Subject Classification Theory of computation → Routing and network design problems

Keywords and phrases Prize Collecting Steiner Tree, Directed graphs, Approximation Algorithms, Budgeted Problem

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2022.9

Related Version *Full Version*: https://arxiv.org/abs/2204.12162

Funding *Gianlorenzo D'Angelo*: Partially supported by the Italian MIUR PRIN 2017 Project ALGADIMAR "Algorithms, Games, and Digital Markets".

1 Introduction

Prize collecting Steiner tree problems (**PCSTP**) have been extensively studied due to their applications in designing computer and telecommunication networks, VLSI design, computational geometry, wireless mesh networks, and cancer genome studies [5, 8, 15, 23, 32]. Very interesting polynomial-time constant/poly-logarithmic approximation algorithms have been proposed for many variants of **PCSTP** when the graph is undirected [1, 2, 10, 12, 18, 21, 29]. However, these problems are usually much harder on directed graphs. For instance,

33rd International Symposium on Algorithms and Computation (ISAAC 2022).
Editors: Sang Won Bae and Heejin Park; Article No. 9; pp. 9:1–9:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

there is a simple polynomial-time 2-approximation algorithm for the *undirected Steiner tree* problem, but no quasi-polynomial-time algorithm for the *directed Steiner tree* problem achieving an approximation ratio of $o\left(\frac{\log^2 k}{\log\log k}\right)$ exists, unless $NP \subseteq \bigcap_{0<\varepsilon<1} \mathrm{ZPTIME}(2^{n^\varepsilon})$ or the Projection Game Conjecture is false [13], where k is the number of terminal nodes.

Some of the most relevant variants of **PCSTP** are represented by prize collecting problems with budget constraints. In such problems, we are usually given a graph with prizes and costs on the nodes and the goal is to find a tree that maximizes the sum of the prize of its nodes, while keeping the total cost bounded by a given budget. Guha et al. [14] introduced the case in which the graph is undirected and the goal is to find a tree that contains a distinguished vertex, called root, respects the budget constraint, and maximizes the prize, we call this problem *Undirected Rooted Additive Tree* (**URAT**). They gave an algorithm that achieves an $O(\log^2 n)$-approximation factor, where n is the number of nodes in the graph, but the computed solution requires a factor-2 violation of the budget constraint. Moss and Rabani [27] and Bateni et al. [2] further investigated **URAT** and improved the results from the approximability point of view. The former paper improved the approximation factor to $O(\log n)$, with the same budget violation, and the latter one improved the budget violation factor to $1+\varepsilon$ to obtain an approximation factor of $O\left(\frac{1}{\varepsilon^2}\log n\right)$, for any $\varepsilon \in (0,1]$. Kortsarz and Nutov [22] showed that the unrooted version of **URAT**, so does **URAT**, admits no $o(\log\log n)$-approximation algorithm, unless $NP \subseteq DTIME(n^{\mathrm{polylog}(n)})$, even if the algorithm is allowed to violate the budget constraint by a factor equal to a universal constant.

In this paper, we consider a generalization of **URAT** on directed graphs. We are given a directed graph, where each node is associated with a cost, and the prize is defined by a monotone submodular function on the subsets of nodes, and the goal is to find an *out-tree* (a.k.a. *out-arborescence*) rooted at a specific vertex r with the maximum prize such that the total cost of all vertices in the out-tree is no more than a given budget. We term this problem *Directed Rooted Submodular Tree* (**DRST**). A closely related problem, called *Submodular Tree Orienteering* (**STO**), has been recently introduced by Ghuge and Nagarajan [11]. **STO** is the same problem as **DRST** except that edges and not nodes have costs. They provided a tight quasi-polynomial-time $O(\frac{\log n'}{\log\log n'})$-approximation algorithm that requires $(n\log B)^{O(\log^{1+\varepsilon} n')}$ time, where n' is the number of vertices in an optimal solution and B is the budget constraint.

Contribution. By extending some ideas of Kuo et al. [23] and Bateni et al. [2], we design a polynomial-time $O(\sqrt{B}/\varepsilon^3)$-approximation algorithm for **DRST**, violating the budget constraint B by a factor of at most $1+\varepsilon$, for any $\varepsilon \in (0,1]$ (Section 4). Our technique can be used to obtain the same result for **STO** (Section 6). To our knowledge, this is the first polynomial-time approximation algorithm for **STO**. We also show that, for any $1+\varepsilon$ budget violation, with $\varepsilon \in (0,1]$, our approach provides an $O(\sqrt{B}/\varepsilon^2)$-approximation algorithm for the special cases of **DRST** and **STO** where the prize function is additive (Section 7). We also consider the unrooted version of **DRST** and give an $O(\sqrt{B})$-approximation algorithm without budget violation (Section 5), which is an improvement over the factor $O(\Delta\sqrt{B})$ [23] for the undirected and unrooted version of **DRST**, where Δ is the maximum node-degree.

Finally, we study some variants of **DRST** on undirected graphs. We show that, for any $1+\varepsilon$ budget violation, **URAT** admits an $O(\Delta/\varepsilon^2)$-approximation algorithm, while its quota version admits a 2Δ-approximation algorithm. Next, we present some approximation results for some variants of the connected maximum coverage problem, which improve over the bounds given by Ran et al. [30]. Finally, we provide two approximation algorithms for the Budgeted Sensor Cover problem, which result in an improvement to the literature [23, 30, 33, 34]. We discuss these results in Section 7.

Related Work. Many variants of Prize collecting Steiner Tree problems have been investigated. Here we list those that are more closely related to our study. Further related work is reported in the Appendix.

Kuo et al. [23] studied the unrooted version of **DRST** on undirected graphs called *Maximum Connected Submodular function with Budget constraint* (**MCSB**). They provided an $O(\Delta\sqrt{B})$-approximation algorithm for **MCSB**, where Δ is the maximum degree of the graph. Vandin et al. [32] provided a $(\frac{2e-1}{e-1}r)$-approximation algorithm for a special case of the same problem, where r is the radius of an optimal solution. This problem coincides with the connected maximum coverage problem in which each set has cost one. Ran et al. [30] presented an $O(\Delta \log n)$-approximation algorithm for a special case of the connected maximum coverage problem. Hochbaum and Rao [15] investigated **MCSB** in which each vertex costs 1 and provided an approximation algorithm with factor $\min\{1/((1-1/e)(1/R-1/B)), B\}$, where R is the radius of the graph. Chen et al. [4] investigated the edge-cost version of **MCSB**. One of the applications of **MCSB** is a problem in wireless sensor networks called *Budgeted Sensor Cover Problem* (**BSCP**), where the goal is to find a set of B connected sensors to maximize the number of covered users, for a given B. Kuo et al. [23] provided a $5(\sqrt{B}+1)/(1-1/e)$-approximation algorithm for **BSCP**, which was improved by Xu et al. [33] to $\lfloor\sqrt{B}\rfloor/(1-1/e)$. Huang et al. [17] proposed a $8(\lceil 2\sqrt{2}C\rceil+1)^2/(1-1/e)$-approximation algorithm for **BSCP**, where $C = O(1)$.

Johnson et al. [18] introduced an edge-cost variant of **DRST** on undirected graphs, where the prize function is additive, called **E-URAT**. They showed that there exists a $(5+\varepsilon)$-approximation algorithm for the unrooted version of **E-URAT** using Garg's 3-approximation algorithm [9] for the k-MST problem, and observed that a 2-approximation for k-MST would lead to a 3-approximation for **E-URAT**. This observation along with the Garg's 2-approximation algorithm [10] for k-MST yield a 3-approximation algorithm for the unrooted version of **E-URAT**. Recently, Paul et al. [29] provided a polynomial-time 2-approximation algorithm for **E-URAT**.

2 Notation and problem statement

For an integer k, let $[k] := \{1, \ldots, k\}$. A directed *path* is a directed graph made of a sequence of distinct vertices $(v_1, \ldots, v_k)$ and a sequence of directed edges (v_i, v_{i+1}), $i \in [k-1]$. An *out-tree* (a.k.a. out-arborescence) is a directed graph in which there is exactly one directed path from a specific vertex r, called *root*, to each other vertex. If a subgraph T of a directed graph D is an out-tree, then we say that T is an out-tree of D.

Let $D = (V, A)$ be a directed graph with n nodes, $c : V \to \mathbb{Z}^+$ be a cost function on nodes, $p : 2^V \to \mathbb{R}^+ \cup \{0\}$ be a monotone submodular prize function on the subsets of nodes, $r \in V$ be a root vertex, and B be an integer budget. For any subgraph D' of D, we denote by $V(D')$ and $A(D')$ the set of nodes and edges in D', respectively. Given $S \subseteq V$, we denote the cost of S by $c(S) = \sum_{v \in S} c(v)$ and we use shortcuts $c(D') = c(V(D'))$ and $p(D') = p(V(D'))$ for a subgraph D' of D. In the Directed Rooted Submodular Tree problem (**DRST**), the goal is to find an out-tree T of D rooted at r such that $c(T) \leq B$ and $p(T)$ is maximum. Throughout the paper, we denote an optimal solution to **DRST** by T^*.

Given two nodes u and v in V, a path in D from u to v with the minimum cost is called a *shortest path* and its cost, denoted by $dist(u, v)$, is called the *distance* from u to v in D.

An algorithm is a bicriteria (β, α)-approximation algorithm for **DRST** if, for any instance I of the problem, it returns a solution Sol_I such that $p(Sol_I) \geq \frac{OPT_I}{\alpha}$ and $c(Sol_I) \leq \beta B$, where OPT_I is the optimum for I.

3 Results and Techniques

Our main result is given in the next theorem.

▶ **Theorem 1.** ***DRST*** *admits a polynomial-time bicriteria* $\left(1+\varepsilon, O\left(\frac{\sqrt{B}}{\varepsilon^3}\right)\right)$*-approximation algorithm, for any* $\varepsilon \in (0,1]$.

Our approach combines and extends techniques given by Kuo et al. [23] and Bateni et al. [2]. To illustrate our techniques, we now consider the case in which costs are unitary, i.e. $c(v)=1$, for each $v \in V$, and the prize function is additive, i.e. $p(S)=\sum_{v\in S} p(\{v\})$, for any $S \subseteq V$. In this case, the distance from a node u to a node v is equal to the minimum number of nodes in a path from u to v and the cost of a tree T is equal to its size, $c(T)=|V(T)|$. W.lo.g. we also assume that the distance from r to any node is at most B. We will give the proof for the general case in Section 4.

The algorithm works as follows. For any vertex u, we denote as V_u the set of all nodes that are at a distance no more than $\lfloor\sqrt{B}\rfloor$ from u, $V_u := \{v \mid dist(u,v) \le \lfloor\sqrt{B}\rfloor\}$. We first select a subset S_u of V_u of at most $\lfloor\sqrt{B}\rfloor$ nodes with the maximum prize, $S_u := \arg\max\{p(S) : S \subseteq V_u, |S| \le \lfloor\sqrt{B}\rfloor\}$.[1] We then compute a minimal inclusion-wise out-tree T_u rooted at u that spans all nodes in S_u. Note that $|V(T_u)| \le B$ since the distance from u to any node in S_u is at most $\lfloor\sqrt{B}\rfloor$. Let z be a node such that $p(T_z)$ is maximum. If $z=r$, then we take T_z as our solution, otherwise we compute a solution by adding to T_z a shortest path P from r to z and removing the edges in $A(T_z)\setminus A(P)$ incoming the nodes in $V(T_z)\cap V(P)$. Let T be our solution and T^* be an optimal solution.

We will prove (Lemma 6) that any out-tree $\hat{T}$ can be covered by at most $N = O(|\hat{T}|/m)$ out-subtrees $\{\hat{T}_i\}_{i=1}^N$ with at most m nodes each, where m is any positive integer less than $|\hat{T}|$. By applying this claim to an optimal solution T^* and by setting $m=\lfloor\sqrt{B}\rfloor$, we obtain

$$p(T^*) = p\left(\bigcup_{i=1}^{N} V(T_i^*)\right) \le Np(T'),$$

where $p(T')=\max\{p(T_i^*) \mid i\in[N]\}$, $|T'| \le \lfloor\sqrt{B}\rfloor$, and $N = O(|T^*|/m) = O(\sqrt{B})$. Let w be the root of T'. Recall that S_w is a set of at most $\lfloor\sqrt{B}\rfloor$ nodes that are at a distance no more than $\lfloor\sqrt{B}\rfloor$ from w and have the maximum prize and T_w contains all the nodes in S_w. Since $|T'| \le \lfloor\sqrt{B}\rfloor$, we have

$$p(T') \le p(S_w) \le p(T_w) \le p(T_z) \le p(T).$$

Since $N=O(\sqrt{B})$, we conclude that $p(T^*)=O(\sqrt{B})p(T)$.

Note that the cost of T is upper-bounded by $2B$, as both the cost of T_z and that of a shortest path from r to z are at most B. We can use the trimming procedure introduced by Bateni et al. [2] to obtain an out-subtree of T with cost at most $(1+\varepsilon)B$ by loosing an approximation factor of $O(1/\varepsilon^2)$, for any $\varepsilon\in(0,1]$ (see Lemma 2). This shows Theorem 1 for the unit-cost, additive-prize case. In the case in which the prize is a general monotone submodular function, the trimming procedure by Bateni et al. cannot be applied. We show how to generalize this procedure to the case of any monotone submodular prize function by loosing an extra approximation factor of $O(1/\varepsilon)$.

[1] This step can be done in polynomial time since function p is additive. If p is monotone and submodular, this step consists in solving the submodular maximization problem. See Section 4 for more details.

We can use the same approach to obtain a polynomial-time bicriteria $\left(1+\varepsilon, O\left(\frac{\sqrt{B}}{\varepsilon^2}\right)\right)$-approximation algorithm for the case of additive prize function and edge-cost. More importantly, we can obtain a polynomial-time bicriteria $\left(1+\varepsilon, O\left(\frac{\sqrt{B}}{\varepsilon^3}\right)\right)$-approximation algorithm for **STO**, i.e. for the edge-cost case where the prize function is monotone submodular. To the best of our knowledge, this is the first polynomial-time approximation algorithm for **STO**.

Finally, for the unrooted version the same approach with some minor changes achieves an $O(\sqrt{B})$-approximation with no budget violation.

4 Approximation Algorithm for DRST

We now introduce our polynomial-time approximation algorithm for **DRST**. We start by defining a procedure that takes as input an out-tree of a directed graph D and returns another out-tree of D which has a smaller cost but preserves the same prize-to-cost ratio (up to a bounded multiplicative factor).

Bateni et al. [2] introduced a similar procedure for the case of undirected graphs and additive prize function. In their case, we are given an undirected graph $G = (V, E)$, a distinguished vertex $r \in V$ and a budget B, where each vertex $v \in V$ is assigned with a prize $p'(v)$ and a cost $c'(v)$. For a tree T, the prize and cost of T are the sum of the prizes and costs of the nodes of T and are denoted by $p'(T)$ and $c'(T)$, respectively. A graph G is called *B-proper* for the vertex r if the cost of reaching any vertex from r is at most B. Bateni et al. proposed a trimming process that leads to the following lemma.

▶ **Lemma 2** (Lemma 3 in [2]). *Let T be a tree rooted at r with the prize-to-cost ratio $\gamma = \frac{p'(T)}{c'(T)}$. Suppose the underlying graph is B-proper for r and for $\varepsilon \in (0, 1]$ the cost of the tree is at least $\frac{\varepsilon B}{2}$. One can find a tree T' containing r with the prize-to-cost ratio at least $\frac{\varepsilon\gamma}{4}$ such that $\varepsilon B/2 \leq c'(T') \leq (1+\varepsilon)B$.*

We now generalize this trimming process to the case in which the underlying graph is directed and the prize function is monotone and submodular by borrowing ideas from [2].

We introduce some additional definitions. Let T be an out-tree rooted at r. A *full* out-subtree of T rooted at some node v is an out-subtree of T containing all the vertices that are reachable from r through v in T. The set of *strict* out-subtrees of T is the set of all full out-subtrees of T other than T itself. The set of *immediate* out-subtrees of T is the set of all full out-subtrees rooted at the children of r in T. A directed graph $D = (V, A)$ is *B-appropriate* for a node r if $dist(r, v) \leq B$ for any node $v \in V$.

▶ **Lemma 3.** *Let $D = (V, A)$ be a B-appropriate graph for a node r. Let T be an out-tree of D rooted at r with the prize-to-cost ratio $\gamma = \frac{p(T)}{c(T)}$, where p is a monotone submodular function. Suppose that $\frac{\varepsilon B}{2} \leq c(T) \leq hB$, where $h \in (1, n]$ and $\varepsilon \in (0, 1]$. One can find an out-subtree $\hat{T}$ rooted at r with the prize-to-cost ratio at least $\frac{\varepsilon^2\gamma}{32h}$ such that $\varepsilon B/2 \leq c(\hat{T}) \leq (1+\varepsilon)B$.*

Proof. We run the following initial trimming procedure. We iteratively remove a strict out-subtree T' from T that satisfies two conditions: (i) the prize-to-cost ratio of $T \setminus T'$ is at least γ, and (ii) $c(T \setminus T') \geq \frac{\varepsilon}{2}B$. We repeat this process until no such strict out-subtree exists. Let T_{-} be the remaining out-tree after applying this process on T.

Now if $c(T_{-}) \leq (1+\varepsilon)B$, the desired out-subtree is obtained. Suppose it is not the case. A full out-subtree T' is called *rich* if $c(T') \geq \frac{\varepsilon}{2}B$ and the prize-to-cost ratio of T' and all its strict out-subtrees are at least γ. We claim that if there exists a rich out-subtree, then we can find the desired out-subtree $\hat{T}$.

▷ Claim 4. Given a rich out-subtree T', the desired out-subtree $\hat{T}$ can be found.

Proof. We first find a rich out-subtree T'' of T' such that the strict out-subtrees of T'' are not rich, i.e., $c(T'') \geq \frac{\varepsilon}{2}B$ while the cost of any strict out-subtree of T'' (if any exist) is less than $\frac{\varepsilon}{2}B$. Let C be the total cost of the immediate out-subtrees of T''. We distinguish between two cases:

1. If $C < \frac{\varepsilon}{2}B$, then let $\hat{T}$ be the union of T'' and a shortest path P from r to the root r'' of T''. $\hat{T}$ has cost at most $C + B \leq (1+\varepsilon)B$ and prize at least $\gamma(\frac{\varepsilon}{2}B)$. This implies that $\hat{T}$ has ratio at least $\frac{\gamma\varepsilon}{2(1+\varepsilon)} \geq \frac{\gamma\varepsilon}{4} \geq \frac{\gamma\varepsilon^2}{32h}$.
2. If $C \geq \frac{\varepsilon}{2}B$, we proceed as follows. Since each immediate out-subtree of T'' has a cost strictly smaller than $\frac{\varepsilon}{2}B$, we can partition all the immediate out-subtrees of T'' into M groups $S_1, \dots, S_M$ in such a way that for each $i \in [M-1]$ the total cost of immediate out-subtrees in S_i is at least $\frac{\varepsilon}{2}B$, and for each $i \in [M]$ it is at most εB. We can always group in this way since the cost of each immediate out-subtree of T'' is less than $\frac{\varepsilon}{2}B$ while $C \geq \frac{\varepsilon}{2}B$. Since the total cost of all the immediate out-subtrees of T'' is upper bounded by hB, then the number of selected groups M is at most

$$M \leq \left\lceil \frac{hB}{\frac{\varepsilon}{2}B} \right\rceil = \left\lceil \frac{2h}{\varepsilon} \right\rceil \leq \left\lfloor \frac{2h}{\varepsilon} \right\rfloor + 1 \leq \left\lfloor \frac{4h}{\varepsilon} \right\rfloor \leq \frac{4h}{\varepsilon}.$$

We now add the root r'' of T'' to each group S_i and denote the new group by S'_i, i.e., $S'_i = S_i \cup \{r''\}$, for any $i \in [M]$. By the monotonicity and submodularity of p, we have $\sum_{i=1}^{M} p(S'_i) \geq p(S'_1) + \sum_{i=2}^{M} p(S_i) \geq p(S'_1 \cup \bigcup_{i=2}^{M} S_i) = p(T'')$. Now among $S'_1, \dots, S'_M$, we select the group S'_z that maximizes the prize, i.e., $z = \arg\max_{i \in [M]} p(S'_i)$. We know that

$$p(S'_z) \geq \frac{1}{M}\sum_{i=1}^{M} p(S'_i) \geq \frac{p(T'')}{M} \geq \frac{\varepsilon}{4h}p(T'') \geq \frac{\varepsilon}{4h} \cdot \frac{\gamma\varepsilon}{2}B = \frac{\gamma\varepsilon^2}{8h}B.$$

In case $z = M$ and $c(S'_M) < \frac{\varepsilon}{2}B$, we select a subset of immediate out-subtrees from $\bigcup_{i=1}^{M-1} S_i$ with the total cost of at least $\frac{\varepsilon}{2}B$ and at most $\varepsilon B - c(S'_M)$, and add it to S'_z.
Finally, let $\hat{T}$ be the union of a shortest path P from r to r'', S'_z, and the edges from r'' to the roots of the out-subtrees in S_z (see Figure 1). By monotonicity, $\hat{T}$ has the total prize at least $p(\hat{T}) \geq p(S'_z) \geq \frac{\gamma\varepsilon^2}{8h}B$. Note that $c(\hat{T}) \leq (1+\varepsilon)B$ as $c(S'_z \setminus \{r''\}) = c(S_z) \leq \varepsilon B$ and the shortest path from r to r'' costs at most B (since the graph is B-appropriate). This implies that the prize-to-cost ratio of $\hat{T}$ is at least $\frac{\gamma\varepsilon^2}{8h(1+\varepsilon)} \geq \frac{\gamma\varepsilon^2}{16h} \geq \frac{\gamma\varepsilon^2}{32h}$. ◁

It only remains to consider the case when there is no rich out-subtree. Since T_- is not rich and $c(T_-) \geq \frac{\varepsilon}{2}B$, the ratio of at least one strict out-subtree of T_- is less than γ. Now we find a strict out-subtree T' with ratio less than γ such that the ratio of all of its strict out-subtrees (if any exist) is at least γ. We first need to show that $c(T_- \setminus T') < \frac{\varepsilon}{2}B$.

▷ Claim 5. $c(T_- \setminus T') < \frac{\varepsilon}{2}B$.

Proof. By the submodularity of p, we know that $p(T_- \setminus T') + p(T') \geq p(T_-)$. This implies that

$$\frac{p(T_- \setminus T')}{c(T_- \setminus T')} \geq \frac{p(T_-) - p(T')}{c(T_-) - c(T')}. \tag{1}$$

Let $\gamma' = \frac{p(T')}{c(T')}$ be the prize-to-cost ratio of T'. We know that

$$p(T_-) - p(T') = c(T_-)\gamma - c(T')\gamma' > c(T_-)\gamma - c(T')\gamma, \tag{2}$$

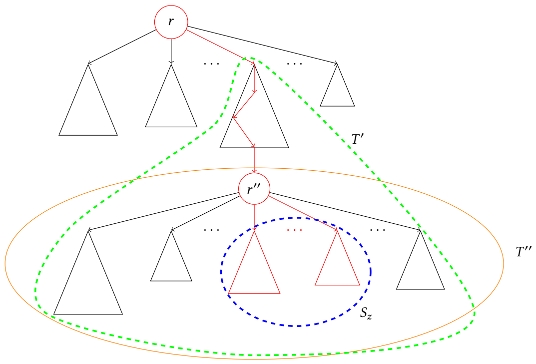

Figure 1 T_- is rooted at r, which is the whole out-tree. The green dashed closed curve represents the rich out-subtree T'. The orange circle represents T'' rooted at r'', where its strict out-subtrees are not rich, i.e., the cost of any strict out-subtree of T'' is less than $\frac{\varepsilon}{2}B$. The blue dashed circle represents the partition S_z, which maximizes the prize and costs at most εB. The red out-subtree represents $\hat{T}$, which is the union of a shortest path from r to r'', S_z and the edges from r'' to the immediate out-subtrees of T'' in S_z. Note that for the sake of simplicity, in this figure we suppose that the shortest path from r to r'' is included in T_-.

where the inequality holds because $\gamma' < \gamma$. By Equations (1) and (2), we have $\frac{p(T_- \setminus T')}{c(T_- \setminus T')} > \gamma$. As the prize-to-cost ratio of $T_- \setminus T'$ is more than γ but T' has not been removed from T during the initial phase, then $c(T_- \setminus T') < \frac{\varepsilon}{2}B$. This concludes the proof of the claim. $\lhd$

We know that $c(T_-) > (1+\varepsilon)B$ and the cost from r to the root of T' is at most B. Then by Claim 5, the total cost of immediate out-subtrees of T' is at least $\frac{\varepsilon}{2}B$. Also, the cost of an immediate out-subtree of T' is less than $\frac{\varepsilon}{2}B$, otherwise, we have a rich out-subtree. As the ratio and cost of T_- are at least γ and $\frac{\varepsilon}{2}B$, respectively, then $p(T_-) \geq \frac{\gamma\varepsilon}{2}B$. Now we distinguish between two cases:

1. If $p(T') \geq \frac{\gamma\varepsilon}{4}B$, by similar reasoning as above, we group the immediate out-subtrees of T' into M groups $S_1, \ldots, S_M$ in such a way that for each $i \in [M-1]$ the total cost of immediate out-subtrees in S_i is at least $\frac{\varepsilon}{2}B$, and for each $i \in [M]$ it is at most εB. Now define a new group $S'_i = S_i \cup \{r'\}$, for any $i \in [M]$. Let $z = \arg\max_{i \in [M]} p(S'_i)$. Then the group S'_z, which maximizes the prize is selected. We know that $M \leq \frac{4h}{\varepsilon}$. Hence, $p(S'_z) \geq \frac{\varepsilon}{4h} p(T') \geq \frac{\varepsilon}{4h} \cdot \frac{\gamma\varepsilon}{4} B = \frac{\gamma\varepsilon^2}{16h}B$.
   Note that in case $z = M$ and $c(S'_M) < \frac{\varepsilon}{2}B$, we select a subset of immediate out-subtrees from $\bigcup_{i=1}^{M-1} S_i$ with the total cost of at least $\frac{\varepsilon}{2}B$ and at most $\varepsilon B - c(S'_M)$, and add it to S'_z. Let $\hat{T}$ be the union of a shortest path P from r to r', S'_z, and the edges from r' to the roots of the out-subtrees in S_z. The cost of $\hat{T}$ is at most $(1+\varepsilon)B$ and the prize-to-cost ratio is at least $\frac{\gamma\varepsilon^2}{16h(1+\varepsilon)} \geq \frac{\gamma\varepsilon^2}{32h}$.
2. If $p(T') < \frac{\gamma\varepsilon}{4}B$, we proceed as follows. Consider the out-subtree $T'' = T_- \setminus T'$, which is rooted at r. Recall that by Claim 5, we have $c(T'') < \frac{\varepsilon}{2}B$. We connect a subset of immediate out-subtrees $T'_1, \ldots, T'_q$ of T' with cost $\frac{\varepsilon}{2}B - c(T'') \leq c(\bigcup_{i=1}^{q} T'_i) \leq \varepsilon B - c(T'')$ to the root of T'' through the root of T'. Since the cost of each immediate out-subtree of T' is less than $\frac{\varepsilon}{2}B$ (otherwise, we have a rich out-subtree) and $c(T') > (1+\frac{\varepsilon}{2})B$, a subset of immediate out-subtrees $T'_1, \ldots, T'_q$ of T' with such a cost can be found. We call the resulting out-subtree $\hat{T}$ and observe that $c(\hat{T}) \geq \frac{\varepsilon}{2}B$ (see Figure 2). We now bound the prize-to-cost ratio of $\hat{T}$. First note that by the submodularity of p, $p(T'') + p(T') \geq p(T_-)$. Thus by the subcase assumption and the monotonicity of p, we have $p(\hat{T}) \geq p(T'') \geq \frac{\gamma\varepsilon}{4}B$. Since $\frac{\varepsilon}{2}B - c(T'') \leq c(\bigcup_{i=1}^{q} T'_i) \leq \varepsilon B - c(T'')$ and the graph is B-appropriate, $c(\hat{T}) \leq (1+\varepsilon)B$.

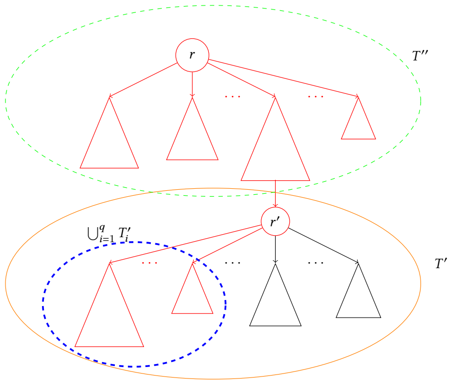

Figure 2 T_- is rooted at r, which is the whole out-tree. The green dashed circle represents T'' rooted at r. The orange circle represents T' rooted at r', where its cost is more than $(1+\frac{\varepsilon}{2})B$. The blue dashed circle represents a subset of immediate out-subtrees $T_1', \ldots, T_q'$ of T' with cost $\frac{\varepsilon}{2}B - c(T'') \leq c(\bigcup_{i=1}^{q} T_i') \leq \varepsilon B - c(T'')$. The red out-subtree represents $\hat{T}$, which is the union of T'', the edge from T'' to r', $\bigcup_{i=1}^{q} T_i'$ and the edges from r' to $T_1', \ldots, T_q'$.

Therefore, the prize-to-cost ratio of the resulting out-subtree $\hat{T}$ is $\frac{\gamma\varepsilon}{4(1+\varepsilon)} \geq \frac{\gamma\varepsilon}{8} \geq \frac{\gamma\varepsilon^2}{32h}$. The proof is complete. ◀

To propose our algorithm, we need a last element. Let $U = \{x_1, \ldots, x_n\}$ be a ground set, $c: U \to Z^+$ be a cost function, $f: 2^U \to \mathbb{R}^+ \cup \{0\}$ be a monotone submodular function, and K be an integer budget. In the Submodular Maximization problem (**SM**), we are looking for a subset $S \subseteq U$ such that $|S| \leq K$ and $f(S)$ is maximum. Nemhauser et al. [28] provided a greedy algorithm that starts from $S := \emptyset$ and runs K iterations in which, at each iteration, it adds to S the element x which maximizes $f(S \cup \{x\}) - f(S)$. This algorithm guarantees a $(1-e^{-1})$-approximation for **SM**. We denote by **RSM** the rooted variant of **SM** in which, additionally, a specific element $v \in U$ is required to be included in the solution, that is we are looking for a subset $S \subseteq U$ such that $|S| \leq K$, $v \in S$ and $f(S)$ is maximum. We can run Nemhauser et al. [28]'s approach for **RSM** with a minor change: we initialize $S := \{v\}$ and run $K-1$ greedy iterations. We call this approach **Greedy**. It can be shown that **Greedy** guarantees a $(1-e^{-1})$-approximation algorithm for **RSM** (see e.g. [23]).

Now we can propose our approximation algorithm for **DRST**, which is reported in Algorithm 1. In words, Algorithm 1 first computes the maximal inclusion-wise B-appropriate subgraph for r of a given graph D by removing all the nodes at a distance larger than B from r. Let $D = (V, A)$ be the resulting directed graph. For each node u, it computes the set V_u of all nodes that are at a distance no more than $c(u) + \lfloor\sqrt{B}\rfloor$ from u. Let S_u^* be a subset of V_u such that $|S_u^*| \leq \lfloor\sqrt{B}\rfloor + 1$, $u \in S_u^*$, and $p(S_u^*)$ is maximum. Finding S_u^* requires to solve an instance I_u^{RSM} of **RSM** where the elements are V_u, the budget is $\lfloor\sqrt{B}\rfloor + 1$, the specific element is u, and profits are defined by function $p(\cdot)$. Using **Greedy**, Algorithm 1 computes in polynomial time an approximate solution S_u to I_u^{RSM} with $u \in S_u$, $|S_u| \leq \lfloor\sqrt{B}\rfloor + 1$ and $p(S_u) \geq (1-e^{-1})p(S_u^*)$. Finally, for each $u \in V$, Algorithm 1 computes a spanning out-tree T_u rooted in u that spans all the nodes in S_u. Let z be a node such that $p(T_z)$ is maximum, i.e., $z = \arg\max_{u \in V} p(T_u)$. Then, we have $c(T_z \setminus \{z\}) \leq B$ as $|S_z \setminus \{z\}| \leq \lfloor\sqrt{B}\rfloor$ and $dist(z, v) \leq c(z) + \lfloor\sqrt{B}\rfloor$ for any $v \in S_z$. If $z = r$, then Algorithm 1 defines $T = T_z$. Otherwise, it computes a shortest path P from r to z and defines T as the union of T_z and P. Since the obtained graph might not be an out-tree, Algorithm 1 removes the possible edges incoming the nodes in $V(T_z) \cap V(P)$ that belong only to T_z. The obtained out-tree T has a cost of at

Algorithm 1 DRST-Algo.

Input: Directed graph $D = (V, A)$; monotone submodular prize function $p : 2^V \to \mathbb{R}^+ \cup \{0\}$; cost function $c : V \to \mathbb{Z}^+$; root $r \in V$; budget B; and $\varepsilon' \in (0, 1]$.
Output: Out-tree T of D rooted at r such that $c(T) \leq (1 + \varepsilon')B$.

```
Remove from D all the nodes at a distance more than B from r;
for u ∈ V do
    V_u := {v | dist(u, v) ≤ c(u) + ⌊√B⌋};
    Define an instance I_u^RSM of RSM with elements V_u, specific element u, budget
    ⌊√B⌋ + 1, profits p(S), for each S ⊆ V_u;
    Let S_u be a (1 − e^{−1})-approximate solution to I_u^RSM, computed by using Greedy;
    Let T_u be a minimal inclusion-wise out-tree rooted at u spanning all nodes in S_u;
end for
z := arg max_{u∈V} p(T_u);
Let P be a shortest path from r to z;
T := P ∪ T_z;
A(T) := A(T) \ {(v, w) ∈ A(T_z) \ A(P) : w ∈ V(T_z) ∩ V(P)};
Apply the trimming process in Lemma 3 with ε = ε' to T;
return T.
```

most $2B$ as $dist(r, z) \leq B$ and $c(T_z \setminus \{z\}) \leq B$. Therefore, Algorithm 1 applies the trimming process in Lemma 3 to T to reduce the cost to $(1 + \varepsilon)B$, where $\varepsilon \in (0, 1]$ and outputs the resulting out-tree.

In the next theorem, we show that Algorithm 1 guarantees a bicriteria approximation.

▶ **Theorem 1.** ***DRST*** *admits a polynomial-time bicriteria* $\left(1 + \varepsilon, O\left(\frac{\sqrt{B}}{\varepsilon^3}\right)\right)$*-approximation algorithm, for any* $\varepsilon \in (0, 1]$.

For our analysis, we need to decompose an optimal out-tree into a bounded number of out-subtrees of bounded cost as in the following lemma, which is similar to Claim 3 in Kuo et al. [23] on the unrooted problem and undirected graphs.

▶ **Lemma 6.** *For any out-tree* $\hat{T} = (V, A)$ *rooted at* r *with cost* $c(\hat{T})$ *and any* $m \leq c(\hat{T})$*, there exist* $N \leq 5\lfloor \frac{c(\hat{T})}{m} \rfloor$ *out-subtrees* $T^i = (V^i, A^i)$ *of* $\hat{T}$*, for* $i \in [N]$*, where* $V^i \subseteq V$*,* $A^i = (V^i \times V^i) \cap A$*,* $c(V^i) \leq m + c(r_i)$*,* r_i *is the root of* T^i*, and* $\bigcup_{i=1}^{N} V^i = V$.

Proof. An out-subtree T' of $\hat{T}$ rooted at r' is called *feasible* if $c(V(T') \setminus \{r'\}) \leq m$; it is called *infeasible* otherwise.

Let us consider the following procedure called **Proc**. **Proc** takes as input an out-tree T', **Proc**(T'), and visits the vertices on T' from the leaves to the root. In this visiting process when **Proc** encounters a vertex v such that T'_v is the first infeasible full out-subtree, it removes T'_v from T', i.e., $T' = T' \setminus T'_v$. **Proc** iteratively repeats this process for the new tree T'. Finally, **Proc** returns all infeasible full out-subtrees that have been found in the visit.

Let $I_1, \ldots, I_s$ be the set of all infeasible full out-subtrees that have been returned after running **Proc**$(\hat{T})$ and let I_{s+1} be the possible feasible out-subtree rooted at r that remains after the visit of **Proc**$(\hat{T})$. We have $\cup_{i \in [s+1]} V(I_i) = V(\hat{T})$ and $V(I_i) \cap V(I_j) = \emptyset$, for $i \neq j$.

For each $i \in [s]$, let us consider the infeasible out-subtree I_i, let v_i be the root of I_i, and let I_u be the full out-subtree of I_i rooted at u, for each child u of v_i. Each out-subtree I_i is further divided into out-subtrees as follows:

- for all children u of v_i such that $c(I_u) \geq m/2$, we generate an out-subtree I_u, observe that $c(I_u) \leq m + c(u)$ because I_u is feasible. If all the children of v_i are in this category, we generate a further out-subtree made of only node v_i.
- All children u of v_i such that $c(I_u) < m/2$ are partitioned into groups of cost between $m/2$ and m, plus a possible group of cost smaller than $m/2$. It is always possible to partition the nodes in this way since $c(I_u) < m/2$ for all such nodes. Then, for each of these groups, we generate an out-subtree by connecting v_i to the roots of the out-subtrees in the group. All the generated out-subtrees have the same root v_i and cost at most $m + c(v_i)$.

The generated out-subtrees cover all the nodes in I_i. We add I_{s+1} to the set of generated out-subtrees, if it exists. Let $T^1, \ldots, T^N$ be the set of generated out-subtrees. Since $I_1, \ldots, I_{s+1}$ cover all the nodes of $\hat{T}$, then so do $T^1, \ldots, T^N$. Moreover, each generated out-subtree T^j costs at most $m + c(r_j)$, where r_j is the root of T^j.

We now bound the number N of generated out-subtrees. Given an infeasible out-subtree I_i, for some $i \in [s]$, each out-subtree generated from I_i costs at least $m/2$, except for the possible out-subtree made of only the root node of I_i and a possible out-subtree of cost smaller than $m/2$. Note that, by construction, at most one of these two additional out-subtrees can be generated. Hence, for each $i \in [s]$, the number s_i of out-subtrees generated from I_i is

$$s_i \leq \left\lfloor \frac{c(I_i)}{m/2} \right\rfloor + 1 \leq 2 \left\lfloor \frac{c(I_i)}{m} \right\rfloor + 2 \leq 4 \left\lfloor \frac{c(I_i)}{m} \right\rfloor .$$

Since $I_1, \ldots, I_{s+1}$ are disjoint, then the overall number of generated out-subtrees is at most $N \leq 1 + \sum_{i \in [s]} s_i \leq 1 + \sum_{i \in [s]} 4 \left\lfloor \frac{c(I_i)}{m} \right\rfloor \leq 5 \left\lfloor \frac{c(\hat{T})}{m} \right\rfloor$. ◀

Now we are ready to prove Theorem 1.

Proof of Theorem 1. By applying Lemma 6 to an optimal solution T^* and by setting $m = \lfloor \sqrt{B} \rfloor$, we obtain $N \leq 5\lfloor \sqrt{B} \rfloor$ out-subtrees $T^i = (V^i, A^i)$ of T^*, for $i \in [N]$, where $V^i \subseteq V(T^*)$, $A^i = (V^i \times V^i) \cap A(T^*)$, $c(V^i) \leq c(r_i) + \lfloor \sqrt{B} \rfloor$, r_i is the root of T^i, and $\bigcup_{i=1}^{N} V^i = T^*$. Let $p(T') = \max\{p(T^i) : i \in [N]\}$ and w be the root of T'. The submodularity of p implies $p(T^*) = p\left(\bigcup_{i=1}^{N} V(T^i)\right) \leq Np(T')$, which implies

$$p(T) \geq p(T_z) \geq p(S_w) \geq (1 - e^{-1})p(S_w^*) \geq (1 - e^{-1})p(T') \geq \frac{1 - e^{-1}}{N} p(T^*) \geq \frac{1 - e^{-1}}{5\lfloor \sqrt{B} \rfloor} p(T^*), \quad (3)$$

where the first two inequalities hold by the definitions of z and S_w and by the monotonicity of function p; The Third inequality holds because S_w is a $(1 - e^{-1})$-approximate solution for instance I_w^{RSM}; The fourth inequality holds as (i) T' contains nodes at a distance no more than $c(w) + \lfloor \sqrt{B} \rfloor$ from w and contains at most $1 + \lfloor \sqrt{B} \rfloor$ nodes (since the minimum cost of a node is at least 1) and (ii) $p(S_w^*) = \max\{p(S) : |S| \leq 1 + \lfloor \sqrt{B} \rfloor \text{ and } dist(w, v) \leq c(w) + \lfloor \sqrt{B} \rfloor, \text{ for all } v \in S\}$.

Before the trimming process in Lemma 3, the ratio between the prize and the cost of T is at least $\gamma = \frac{1-e^{-1}}{10\sqrt{B}B} p(T^*)$ as $c(T) \leq 2B$. After applying the trimming process in Lemma 3 (with $h = 2$) to T, the cost of T is at most $(1 + \varepsilon)B$ and its prize-to-cost ratio is:

$$\frac{p(T)}{c(T)} \geq \frac{\varepsilon^2 \gamma}{64} = \alpha \frac{\varepsilon^2}{\sqrt{B}B} p(T^*),$$

where $\alpha = \frac{1-e^{-1}}{640}$. As $c(T) \geq \varepsilon B/2$, we have $p(T) \geq \frac{\alpha \varepsilon^3}{2\sqrt{B}} p(T^*)$, which concludes the proof. ◀

5 The unrooted version of DRST

Here we consider the unrooted version of **DRST**, denoted by **DUST**, in which the goal is to find an out-tree T of D such that $c(T) \le B$ and $p(T)$ is maximum. Note that T can be rooted at any vertex. By guessing the root of an optimal solution, we can apply the algorithm in the previous section to obtain a bicriteria $(1+\varepsilon, O(\frac{\sqrt{B}}{\varepsilon^3}))$ approximation. We now show that **DUST** admits an $O(\sqrt{B})$-approximation algorithm with no budget violations. To do this, we first provide an unrooted version of Lemma 3 in which it is not necessary to violate the budget constraint when each vertex costs at most half of the budget. This trimming process follows the same procedure as that of Lemma 3, but we include it for the sake of completeness.

▶ **Lemma 7.** *Let T be an out-tree with the prize-to-cost ratio $\gamma = \frac{p(T)}{c(T)}$, where p is a monotone submodular function. Suppose $\frac{B}{2} \le c(T) \le hB$, where $h \in (1, n]$ and the cost of each vertex is at most $\frac{B}{2}$. One can find an out-subtree $\hat{T} \subseteq T$ with the prize-to-cost ratio at least $\frac{\gamma}{32h+8}$ such that $B/4 \le c(\hat{T}) \le B$.*

Proof. In the initial step, we remove a strict out-subtree T' of T if (i) the prize-to-cost ratio of $T \setminus T'$ is at least γ, and (ii) $c(T \setminus T') \ge \frac{B}{4}$. This process is performed iteratively, until no such out-subtree exists. Let T_- be the remaining out-subtree after applying this iterative process on T.

If $c(T_-) \le B$, the desired out-subtree is obtained and we are done. Suppose it is not the case. A full out-subtree T' is called *rich* if $c(T') \ge \frac{B}{4}$ and the prize-to-cost ratio of T' and all its strict out-subtrees are at least γ. As in Lemma 3, we claim that the lemma follows from the existence of a rich out-subtree.

▷ Claim 8. Given a rich out-subtree T', the desired out-subtree $\hat{T}$ can be found.

Proof. Let T'' be the lowest rich out-subtree of T' such that the strict out-subtrees of T'' are not rich, i.e., $c(T'') \ge \frac{B}{4}$ while the cost of strict out-subtrees of T'' (if any exist) is less than $\frac{B}{4}$. Let C be the total cost of the immediate out-subtrees of T''. We distinguish between two cases:

1. If $C < \frac{B}{4}$, then $c(T'') \le \frac{3B}{4}$ as the root of T'' costs at most $\frac{B}{2}$. Since T'' has the prize-to-cost ratio at least γ and cost at least $\frac{B}{4}$ (as it is rich), $\hat{T} = T''$ is the desired out-subtree.
2. If $C \ge \frac{B}{4}$, we first group the immediate out-subtrees of T'' into M groups $S_1, \dots, S_M$ in such a way that for each $i \in [M-1]$ the total cost of immediate out-subtrees in S_i is at least $\frac{B}{4}$, and for each $i \in [M]$ it is at most $\frac{B}{2}$. As $c(T_-) \le hB$, we have

$$M \le \left\lceil \frac{hB}{B/4} \right\rceil = \lceil 4h \rceil \le 4h + 1.$$

For each $i \in [M]$, let $S'_i = S_i \cup \{r''\}$, where r'' is the root of T''. Let $z = \arg\max_{i \in [M]} p(S'_i)$. Hence by the submodularity and monotonicity of p, we have

$$p(S'_z) \ge \frac{\sum_{i=1}^{M} p(S'_i)}{4h+1} \ge \frac{p(S'_1) + \sum_{i=2}^{M} p(S_i)}{4h+1} \ge \frac{p(S'_1 \cup \bigcup_{i=2}^{M} S_i)}{4h+1} = \frac{p(T'')}{4h+1} \ge \frac{\gamma}{16h+4} B,$$

where the last inequality holds as $p(T'') \ge \gamma \frac{B}{4}$ (since T'' is rich).
In case $z = M$ and $c(S'_M) < \frac{B}{4}$, we select a subset of immediate out-subtrees from $\bigcup_{i=1}^{M-1} S_i$ with the total cost of at least $\frac{B}{4}$ and at most $\frac{B}{2} - c(S_M)$, and add it to S_z.
Let $\hat{T}$ be the union of r'', the edges from r'' to the roots of the out-subtrees in S_z, and S_z. The cost of $\hat{T}$ is at most B. Hence the prize-to-cost ratio of $\hat{T}$ is at least $\frac{\gamma}{16h+4} \ge \frac{\gamma}{32h+8}$. ◀

It only remains to consider the case when there is no rich out-subtree. Since T_- is not rich and $c(T_-) \geq \frac{B}{4}$, the ratio of at least one of the strict out-subtrees of T_- is less than γ. Now we find an out-subtree T' with ratio less than γ such that the ratio of all of its strict out-subtrees (if any exist) is at least γ. Since the ratio of T' is less than γ and T' is not removed in the initial process, $c(T_- \setminus T') < \frac{B}{4}$ (this can be shown by the same argument as that of Claim 5). As $c(T_-) > B$ and the cost of the root of T' is at most $\frac{B}{2}$, the total cost of the immediate out-subtrees of T' is at least $\frac{B}{4}$. Also, the cost of an immediate out-subtree of T' is less than $\frac{B}{4}$, otherwise we have a rich out-subtree. As the ratio and cost of T_- are at least γ and $\frac{B}{4}$, respectively, then $p(T_-) \geq \frac{\gamma}{4}B$. We distinguish between two cases.

1. If $p(T') \geq \frac{\gamma}{8}B$, by the similar reasoning as above, we partition the immediate out-subtrees of T' into M groups $S_1, \ldots, S_M$ in such a way that for each $i \in [M-1]$ the total cost of immediate out-subtrees in S_i is at least $\frac{B}{4}$, and for each $i \in [M]$ it is at most $\frac{B}{2}$. For each $i \in [M]$, let $S'_i = S_i \cup \{r'\}$ where r' is the root of T'. Let $z = \arg\max_{i \in [M]} p(S'_i)$. As $M \leq 4h+1$, by the submodularity and monotonicity of p we have:

$$p(S'_z) \geq \frac{\sum_{i=1}^{M} p(S'_i)}{4h+1} \geq \frac{p(S'_1) + \sum_{i=2}^{M} p(S_i)}{4h+1} \geq \frac{p(S'_1 \cup \bigcup_{i=2}^{M} S_i)}{4h+1} = \frac{p(T')}{4h+1} \geq \frac{\gamma}{32h+8}B,$$

   where the last inequality holds as $p(T') \geq \frac{\gamma}{8}B$.
   Note that in case $z = M$ and $c(S'_M) < \frac{B}{4}$, we select a subset of immediate out-subtrees from $\bigcup_{i=1}^{M-1} S_i$ with the total cost of at least $\frac{B}{4}$ and at most $\frac{B}{2} - c(S_M)$, and add it to S_z. Let $\hat{T}$ be the union of r', the edges from r' to the roots of the out-subtrees in S_z and S_z. The cost of $\hat{T}$ is at most B and its prize-to-cost ratio is at least $\frac{\gamma}{32h+8}$.
2. If $p(T') < \frac{\gamma}{8}B$, we proceed as follows. Consider the out-subtree $T'' = T_- \setminus T'$. Recall that by the above discussion we have $c(T'') < \frac{B}{4}$. Thus we find a subset S of the immediate out-subtrees of T' with cost between $\frac{B}{4} - c(T'') \leq c(S) \leq \frac{B}{2} - c(T'')$. Note that such set S can be found as each immediate out-subtree of T' costs less than $\frac{B}{4}$ (otherwise we have a rich subtree) and $c(T'') > \frac{3B}{4}$ (as $c(T_-) > B$ and $c(T'') < \frac{B}{4}$). Then let $\hat{T}$ be the union of T'', the edge from T'' to r' in T_-, S, and the edges from r' to the roots of the out-subtrees in S, where r' is the root of T'. We now bound the prize-to-cost ratio of $\hat{T}$. Recall that $T'' = T_- \setminus T'$. First note that by the submodularity' properties $p(T'') + p(T') \geq f(T_-)$. Thus by the case assumption and monotonicity, we have $f(\hat{T}) \geq p(T'') \geq \frac{\gamma}{8}B$. Since $\frac{B}{4}B - c(T'') \leq c(S) \leq \frac{B}{2} - c(T'')$ and $c(r') \leq \frac{B}{2}$, $c(\hat{T}) \leq B$. Therefore, the prize-to-cost ratio of $\hat{T}$ is at least $\frac{\gamma}{8} \geq \frac{\gamma}{32h+8}$.

The proof is complete. ◀

▶ **Theorem 9.** ***DUST*** *admits a polynomial-time $O(\sqrt{B})$-approximation algorithm.*

Proof. We follow arguments similar to those in Theorem 4 from Bateni et al. [2], but for the sake of completeness the proof is provided here.

An out-tree is called *flat* if each vertex of the out-tree costs no more than $\frac{B}{2}$. Let x be a vertex of an out-tree with the largest cost. An out-tree is called *saddled* if $c(x) > \frac{B}{2}$ and the cost of every other vertex of the out-tree is no more than $\frac{B-c(x)}{2}$. Let T^*_f (resp. T^*_s) be the optimal flat (resp. saddled) out-tree, i.e, a flat (resp. saddled) out-tree with cost at most B maximizing the prize. We first show that given an optimal solution T^* to **DUST**, then either $p(T^*_f) \geq \frac{p(T^*)}{2}$ or $p(T^*_s) \geq \frac{p(T^*)}{2}$.

▷ Claim 10. Either $p(T^*_f) \geq \frac{p(T^*)}{2}$ or $p(T^*_s) \geq \frac{p(T^*)}{2}$, where T^* is an optimal solution to **DUST**.

Proof. If T^* has only one vertex, then it is either flat or saddled and we are done. If T^* has more than one vertex and it is neither flat nor saddled, then we proceed as follows. Let x and y be two vertices in T^* with the maximum cost and the second maximum cost, respectively. Since T^* is not flat then $c(x) > \frac{B}{2}$ and $c(y) \leq \frac{B}{2}$. Also as T^* is not saddled, $c(y) > \frac{B-c(x)}{2}$, and, since the cost of T^* is at most B, y is the only node with a cost higher than $\frac{B-c(x)}{2}$. By removing the edge e adjacent to y on the path between x and y, we can partition T^* into two out-subtrees T_x^* and T_y^* that contain x and y, respectively. Clearly, each vertex in T_y^* costs no more than $\frac{B}{2}$, then T_y^* is flat. Also, each vertex in T_x^* except x costs at most $\frac{B-c(x)}{2}$, implying that T_x^* is saddled. By the submodularity of p, $p(T_x^*) + p(T_y^*) \geq p(T^*)$, meaning that one of T_x^* and T_y^* has at least half of the optimum prize $p(T^*)$, which concludes the claim. ◁

Now we restrict Algorithm 1 to only flat and saddled out-trees. Indeed, we can reduce the case of saddled out-trees to flat out-trees as follows. We first find a vertex x with the maximum cost. We then set the cost of x to zero and define a new budget $B' = B - c(x)$. Note that the cost of any other vertex in the optimal saddled out-tree T_s^* is at most half of the remaining budget. This means that we only need to find an approximation solution when restricted to flat out-trees.

Since for the new instance no other vertex except x with cost more than $\frac{B}{2}$ can be contained in the final solution, we remove all vertices with cost more than $\frac{B}{2}$ and run Lines 1-8 of Algorithm 1 on the new resulting graph to achieve an out-tree T with cost $c(T) \leq 2B$ (as $c(T \setminus \{z\}) = B$ and $c(z) \leq B$) and prize $p(T) \geq \frac{1-e^{-1}}{5\sqrt{B}} p(T_f^*) \geq \frac{1-e^{-1}}{10\sqrt{B}} p(T^*)$. So, the prize-to-cost ratio of T is $\gamma \geq \frac{p(T)}{2B}$. As T is flat, we can apply Lemma 7 to achieve an out-subtree $\hat{T}$ of T with the cost $B/4 \leq c(\hat{T}) \leq B$ and the prize-to-cost ratio $\frac{p(\hat{T})}{c(\hat{T})} \geq \frac{\gamma}{32h+8} = \frac{\gamma}{72}$ as $h \leq 2$. This implies that

$$p(\hat{T}) \geq \frac{\gamma}{72} \cdot \frac{B}{4} = \frac{\gamma}{288} B \geq \frac{p(T)}{576} \geq \frac{1-e^{-1}}{5760\sqrt{B}} p(T^*). \qquad ◀$$

6 Submodular Tree Orienteering

Recently, Ghuge and Nagarajan [11] studied the *Submodular Tree Orienteering* problem (**STO**), which is similar to **DRST** with the only difference that the costs are associated to the edges of a directed graph instead of the nodes. In particular, in **STO**, we are given a directed graph $D = (V, A)$, a vertex $r \in V$, a budget B, a monotone submodular function $p : 2^V \to \mathbb{R}^+$, and a cost $c : A \to \mathbb{Z}^+$, and the goal is to find an out-subtree T of D rooted at r such that $\sum_{e \in A(T)} c(e) \leq B$ and $p(T) = p(V(T))$ is maximum. Ghuge and Nagarajan [11] proposed an $O\left(\frac{\log k}{\log \log k}\right)$-approximation algorithm for **STO** that runs in $(n \log B)^{O(\log^{1+\epsilon} k)}$ time for any constant $\epsilon > 0$, where $k \leq |V|$ is the number of vertices in an optimal solution.

Here we first show that **DRST** can be reduced to **STO**, preserving the approximation factor, by assigning the cost of each node v to all edges entering v.

▶ **Theorem 11.** *There is an $O(\frac{\log k}{\log \log k})$-approximation algorithm for **DRST** that runs in $(n \log B)^{O(\log^{1+\epsilon} k)}$ time for any constant $\epsilon > 0$, where $k \leq n = |V|$ is the number of vertices in an optimal solution.*

Proof. To prove the theorem, we show that one can transform an instance $J = \langle D' = (V', A'), p', c', r', B' \rangle$ of **DRST** to an instance $I = \langle D = (V, A), p, c, r, B \rangle$ of **STO** as follows. We set $V = V'$, $r = r'$, $A = A' \setminus \{(v, r') | (v, r') \in A'\}$, $B = B' - c'(r')$ and for any subset

$S \subseteq V$, $p(S) = p'(S)$. For any $e = (i, j) \in A$ in I, we set $c(e) = c'(j)$. The theorem follows by observing that any out-subtree T of D is an out-subtree for D', $c'(T) = \sum_{v \in V(T)} c'(v) = \sum_{e=(u,v) \in A(T)} c(e) + c'(r') = c(T) + c'(r')$, and $p'(T) = p(T)$. ◀

Moreover, we show that Algorithm 1 can be used to approximate **STO**. To our knowledge, this is the first polynomial-time bicriteria approximation algorithm for **STO**.

▶ **Theorem 12.** *There exists a polynomial-time bicriteria $(1 + \varepsilon, O(\frac{\sqrt{B}}{\varepsilon^3}))$-approximation algorithm for **STO**, for any $\varepsilon \in (0, 1]$.*

Proof. We first transform an instance $I_S = \langle D_S = (V_S, A_S), p_S, c_S, r, B\rangle$ of **STO** to an instance $I_D = \langle D_D = (V_D, A_D), p_D, c_D, r, B\rangle$ of **DRST**, where $V_D = V_S \cup V_A$, $V_A = \{v_e : e \in A_S\}$, $A_D = \{(i, v_e), (v_e, j) : e = (i, j) \in A_S\}$, $p_D(S) = p_S(S \cap V_S)$, for each $S \subseteq V_D$, $c_D(v) = 0$ for each $v \in V_S$, and $c_D(v_e) = c_S(e)$ for each $v_e \in V_A$.

Let T^*_S be an optimal solution for I_S and let T^*_D be the out-subtree of D' corresponding to T^*_S (i.e. $V(T^*_D) = V(T^*_S) \cup \{v_e : e \in A(T^*_S)\}$, $A(T^*_D) = \{(i, v_e), (v_e, j) : e = (i, j) \in A(T^*_S)\}$). We observe that $p_D(T^*_D) = p_S(T^*_S)$, $c_D(T^*_D) = c_S(T^*_S)$, and T^*_D is an optimal solution for I_D, since if there exists an out-subtree $\bar{T}_D$ of D_D with $p_D(\bar{T}_D) > p_D(T^*_D)$, then we can construct an out-subtree $\bar{T}_S = (V(\bar{T}_D) \cap V_S, \{e \in A_S : v_e \in V(\bar{T}_D) \cap V_A\})$ of D_S such that $p_S(\bar{T}_S) > p_S(T^*_S)$.

We decompose T^*_D as in Lemma 6,[2] with $m = \lfloor\sqrt{B}\rfloor$; let T'_D be the out-subtree that maximizes the prize among those returned by the lemma, and let w be the root of T'_D. We have that $p_D(T'_D) \geq \frac{1}{5\lfloor\sqrt{B}\rfloor} p_D(T^*_D)$ and $c(T'_D) \leq c(w) + \lfloor\sqrt{B}\rfloor$. It follows that the distance from w to any other node in T'_D is at most $c(w) + \lfloor\sqrt{B}\rfloor$.

We now show that $|V(T'_D) \cap V_S| \leq \lfloor\sqrt{B}\rfloor + 1$. Since the cost of nodes in V_S is equal to 0, then $c((V(T'_D) \cap V_A) \setminus \{w\}) = c(V(T'_D) \setminus \{w\}) \leq \lfloor\sqrt{B}\rfloor$. Therefore, as the cost of each edge in A_S is at least 1, $|(V(T'_D) \cap V_A) \setminus \{w\}| \leq \lfloor\sqrt{B}\rfloor$. For every node in $(V(T'_D) \cap V_S) \setminus \{w\}$, there exists a distinct node in $V(T'_D) \cap V_A$, which means that $|(V(T'_D) \cap V_S) \setminus \{w\}| = |V(T'_D) \cap V_A|$. If $w \in V_S$, then $|V(T'_D) \cap V_A| = |(V(T'_D) \cap V_A) \setminus \{w\}| \leq \lfloor\sqrt{B}\rfloor$. If $w \in V_A$, then $|(V(T'_D) \cap V_A)| \leq \lfloor\sqrt{B}\rfloor + 1$. In both cases $|V(T'_D) \cap V_S| \leq \lfloor\sqrt{B}\rfloor + 1$.

Let T_D be the output of lines 1-11 of Algorithm 1 for instance I_D. We have that

$$p_D(T_D) \geq (1 - e^{-1}) p_D(S^*_w) \geq (1 - e^{-1}) p_D(T'_D) \geq \frac{1 - e^{-1}}{5\lfloor\sqrt{B}\rfloor} p_D(T^*_D) = \frac{1 - e^{-1}}{5\lfloor\sqrt{B}\rfloor} p_S(T^*_S),$$

where the second inequality is due to the fact that (i) T'_D contains nodes at a distance no more than $c(w) + \lfloor\sqrt{B}\rfloor$ from w and contains at most $\lfloor\sqrt{B}\rfloor + 1$ nodes in V_S, and (ii) $p_D(S) = p_D(S \cap V_S)$, for each $S \subseteq V_D$, and therefore $p_D(S^*_w) = \max\{p_D(S) : |S \cap V_S| \leq \lfloor\sqrt{B}\rfloor + 1$ and $dist(w, v) \leq c(w) + \lfloor\sqrt{B}\rfloor$, for all $v \in S\}$. The other inequalities are analogous to those in (3).

The cost of T_D is at most $2B$, as in Theorem 1 we can trim T_D to reduce its cost to $(1 + \varepsilon)B$ and maintaining a prize of $p_D(T_D) = \frac{\alpha\varepsilon^2}{\sqrt{B}} p(T^*_S)$, for some constant α and any arbitrary $\varepsilon > 0$.

Let us consider the out-subtree T_S of D_S corresponding to T_D, $T_S = (V(T_D) \cap V_S, \{e \in A_S : v_e \in V(T_D) \cap V_A\})$, then $p_S(T_S) = p_D(T_D)$ and $c_S(T_S) = c_D(T_D)$, which concludes the proof. ◀

7 Further Results on Some Variants of DRST

In this section, we provide approximation results on some variants of **DRST**. Due to space constraints, here we only state our results, all the details are given in a long version of the paper [6].

[2] Note that the Lemma 3 and 6 hold even if node costs are allowed to be equal to 0.

Additive prize function and Directed Tree Orienteering (DTO). We consider the special case of **DRST** in which the prize function is additive, i.e., for any $S \subseteq V$, $p(S) = \sum_{v \in S} p(\{v\})$, called **DRAT**. We show that there exists a polynomial-time bicriteria $(1+\varepsilon, O(\sqrt{B}/\varepsilon^2))$-approximation algorithm for **DRAT** (Theorem B.1 in [6]). By using the reduction in Theorem 12, it follows that this result also holds for **DTO**, which is the special case of **STO** in which the prize function is additive [11].

Undirected graphs. All our results hold also in the case in which the input graph is undirected and the output graph is a tree. In particular, our $O(\sqrt{B})$-approximation algorithm for the unrooted case improves over the factors $O((\Delta+1)\sqrt{B})$ [23] and $\min\{1/((1-1/e)(1/R - 1/B)), B\}$ [15], where R is the radius of the input graph G. For the case in which the prize function is additive, we show that there exists a polynomial-time bicriteria approximation algorithm whose approximation factor only depends on the the maximum degree Δ of the given graph. In particular, it guarantees a bicriteria $(1+\varepsilon, 16\Delta/\varepsilon^2)$-approximation (Theorem B.2 in [6]).

Quota problem. We consider the problem in which we are given an undirected graph $G = (V, E)$, a cost function $c : V \to \mathbb{R}^+$, a prize function $p : V \to \mathbb{R}^+$, a quota $Q \in \mathbb{R}^+$, and a vertex r, and the goal is to find a tree T such that $p(T) \geq Q$, $r \in V(T)$ and $c(T)$ is minimum. We prove that this problem admits a 2Δ-approximation algorithm (Theorem B.3 in [6]).

Maximum Weighted Budgeted Connected Set Cover (MWBCSC). Let X be a set of elements, $\mathcal{S} \subseteq 2^X$ be a collection of sets, $p : X \to \mathbb{R}^+$ be a prize function, $c : \mathcal{S} \to \mathcal{R}^+$ be a cost function, $G_{\mathcal{S}}$ be a graph on vertex set $\mathcal{S}$, and B be a budget. In **MWBCSC**, the goal is to find a subcollection $\mathcal{S}' \subseteq \mathcal{S}$ such that $c(\mathcal{S}') = \sum_{S \in \mathcal{S}'} c(S) \leq B$, the subgraph induced by $\mathcal{S}'$ is connected and $p(\mathcal{S}') = \sum_{x \in X_{\mathcal{S}'}} p(x)$ is maximum, where $X_{\mathcal{S}'} = \bigcup_{S \in \mathcal{S}'} S$. We show that **MWBCSC** admits a polynomial-time αf-approximation algorithm, where f is the maximum frequency of an element and α is the performance ratio of an algorithm for the unrooted version of **MCSB** with additive prize function (Theorem B.4 in [6]). Moreover, one can have a polynomial-time $O(\log n)$-approximation algorithm for **MWBCSC** under the assumption that if two sets have an element in common, then they are adjacent in $G_{\mathcal{S}}$ (Corollary B.2 in [6]). This last result is an improvement over the factor $2(\Delta+1)\alpha/(1-e^{-1})$ by Ran et al. [30].

Budgeted Sensor Cover Problem (BSCP). In **BSCP**, we are given a set $\mathcal{S}$ of sensors, a set $\mathcal{P}$ of target points in a metric space, a sensing range R_s, a communication range R_c, and a budget B. A target point is covered by a sensor if it is within distance R_s from it. Two sensors are connected if they are at a distance at most R_c. The goal is to find a subset $\mathcal{S}' \subseteq \mathcal{S}$ such that $|\mathcal{S}'| \leq B$, the number of covered target points by $\mathcal{S}'$ is maximized and $\mathcal{S}'$ induces a connected subgraph. We give a $2f$-approximation algorithm for **BSCP** (Theorem B.5 in [6]), where f is the maximum number of sensors that cover a target point. We also show that, under the assumption that $R_s \leq R_c/2$, **BSCP** admits a polynomial-time $8/(1-e^{-1})$-approximation algorithm (Theorem B.8 in [6]), which improves the factors $8(\lceil 2\sqrt{2}C \rceil + 1)^2/(1-1/e)$ [17] and $8(\lceil 4C/\sqrt{3} \rceil + 1)^2/(1-1/e)$ [34], where $C = R_s/R_c$. Note that Huang et al. [17] do not assume that $R_s/R_c \leq R_c/2$, however, our technique improves over their result if $R_s \leq R_c/2$.

References

1 Aaron Archer, MohammadHossein Bateni, MohammadTaghi Hajiaghayi, and Howard J. Karloff. Improved approximation algorithms for prize-collecting steiner tree and TSP. *SIAM J. Comput.*, 40(2):309–332, 2011.

2 Mohammad Hossein Bateni, Mohammad Taghi Hajiaghayi, and Vahid Liaghat. Improved approximation algorithms for (budgeted) node-weighted steiner problems. *SIAM J. Comput.*, 47(4):1275–1293, 2018.

3 Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999.

4 Xuefeng Chen, Xin Cao, Yifeng Zeng, Yixiang Fang, and Bin Yao. Optimal region search with submodular maximization. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI*, pages 1216–1222, 2020.

5 Xiuzhen Cheng, Yingshu Li, Ding-Zhu Du, and Hung Q Ngo. Steiner trees in industry. In *Handbook of combinatorial optimization*, pages 193–216. Springer, 2004.

6 Gianlorenzo D'Angelo, Esmaeil Delfaraz, and Hugo Gilbert. Budgeted out-tree maximization with submodular prizes. *CoRR*, abs/2204.12162, 2022.

7 Kiril Danilchenko, Michael Segal, and Zeev Nutov. Covering users by a connected swarm efficiently. In *Algorithms for Sensor Systems - 16th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2020, , Revised Selected Papers*, volume 12503 of *Lecture Notes in Computer Science*, pages 32–44. Springer, 2020.

8 Xiaofeng Gao, Junwei Lu, Haotian Wang, Fan Wu, and Guihai Chen. Algorithm design and analysis for wireless relay network deployment problem. *IEEE Trans. Mob. Comput.*, 18(10):2257–2269, 2019.

9 N Garg. A 3 factor approximation algorithm for the minimum tree spanning k vertices. In *Proc of 37th Symp. on Foundations of Computer Science*, pages 302–309, 1996.

10 Naveen Garg. Saving an epsilon: a 2-approximation for the k-mst problem in graphs. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 396–402. ACM, 2005.

11 Rohan Ghuge and Viswanath Nagarajan. Quasi-polynomial algorithms for submodular tree orienteering and other directed network design problems. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1039–1048. SIAM, 2020.

12 Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.

13 Fabrizio Grandoni, Bundit Laekhanukit, and Shi Li. O($\log^2$k/log log k)-approximation algorithm for directed steiner tree: A tight quasi-polynomial-time algorithm. *CoRR*, abs/1811.03020, 2018.

14 Sudipto Guha, Anna Moss, Joseph Naor, and Baruch Schieber. Efficient recovery from power outage (extended abstract). In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 574–582. ACM, 1999.

15 Dorit S. Hochbaum and Xu Rao. Approximation algorithms for connected maximum coverage problem for the discovery of mutated driver pathways in cancer. *Inf. Process. Lett.*, 158:105940, 2020.

16 Chien-Chung Huang, Mathieu Mari, Claire Mathieu, Joseph S. B. Mitchell, and Nabil H. Mustafa. Maximizing covered area in the euclidean plane with connectivity constraint. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019*, volume 145 of *LIPIcs*, pages 32:1–32:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

17 Lingxiao Huang, Jian Li, and Qicai Shi. Approximation algorithms for the connected sensor cover problem. *Theor. Comput. Sci.*, 809:563–574, 2020.

18 David S. Johnson, Maria Minkoff, and Steven Phillips. The prize collecting steiner tree problem: theory and practice. In David B. Shmoys, editor, *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 760–769. ACM/SIAM, 2000.

19 Samir Khuller, Manish Purohit, and Kanthi K. Sarpatwar. Analyzing the optimal neighborhood: Algorithms for partial and budgeted connected dominating set problems. *SIAM J. Discret. Math.*, 34(1):251–270, 2020.

20 Philip N. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted steiner trees. *J. Algorithms*, 19(1):104–115, 1995.

21 Jochen Könemann, Sina Sadeghian Sadeghabad, and Laura Sanità. An LMP o(log n)-approximation algorithm for node weighted prize collecting steiner tree. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 568–577. IEEE Computer Society, 2013.

22 Guy Kortsarz and Zeev Nutov. Approximating some network design problems with node costs. *Theor. Comput. Sci.*, 412(35):4482–4492, 2011.

23 Tung-Wei Kuo, Kate Ching-Ju Lin, and Ming-Jer Tsai. Maximizing submodular set function with connectivity constraint: Theory and application to networks. *IEEE/ACM Trans. Netw.*, 23(2):533–546, 2015.

24 Ioannis Lamprou, Ioannis Sigalas, and Vassilis Zissimopoulos. Improved budgeted connected domination and budgeted edge-vertex domination. *Theor. Comput. Sci.*, 858:1–12, 2021.

25 Heungsoon Felix Lee and Daniel R Dooly. Algorithms for the constrained maximum-weight connected graph problem. *Naval Research Logistics (NRL)*, 43(7):985–1008, 1996.

26 Shi Li and Bundit Laekhanukit. Polynomial integrality gap of flow LP for directed steiner tree. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022*, pages 3230–3236. SIAM, 2022.

27 Anna Moss and Yuval Rabani. Approximation algorithms for constrained node weighted steiner tree problems. *SIAM J. Comput.*, 37(2):460–481, 2007.

28 George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Math. Program.*, 14(1):265–294, 1978.

29 Alice Paul, Daniel Freund, Aaron M. Ferber, David B. Shmoys, and David P. Williamson. Budgeted prize-collecting traveling salesman and minimum spanning tree problems. *Math. Oper. Res.*, 45(2):576–590, 2020.

30 Yingli Ran, Zhao Zhang, Ker-I Ko, and Jun Liang. An approximation algorithm for maximum weight budgeted connected set cover. *J. Comb. Optim.*, 31(4):1505–1517, 2016.

31 Stephan Seufert, Srikanta J. Bedathur, Julián Mestre, and Gerhard Weikum. Bonsai: Growing interesting small trees. In Geoffrey I. Webb, Bing Liu, Chengqi Zhang, Dimitrios Gunopulos, and Xindong Wu, editors, *ICDM 2010, The 10th IEEE International Conference on Data Mining*, pages 1013–1018. IEEE Computer Society, 2010.

32 Fabio Vandin, Eli Upfal, and Benjamin J. Raphael. Algorithms for detecting significantly mutated pathways in cancer. *J. Comput. Biol.*, 18(3):507–522, 2011.

33 Wenzheng Xu, Yueying Sun, Rui Zou, Weifa Liang, Qiufen Xia, Feng Shan, Tian Wang, Xiaohua Jia, and Zheng Li. Throughput maximization of UAV networks. *IEEE/ACM Transactions on Networking*, 30(2):881–895, 2022. doi:10.1109/TNET.2021.3125982.

34 Nan Yu, Haipeng Dai, Guihai Chen, Alex X. Liu, Bingchuan Tian, and Tian He. Connectivity-constrained placement of wireless chargers. *IEEE Trans. Mob. Comput.*, 20(3):909–927, 2021.

35 Alexander Zelikovsky. A series of approximation algorithms for the acyclic directed steiner tree problem. *Algorithmica*, 18(1):99–110, 1997.

36 Chenyang Zhou, Anisha Mazumder, Arun Das, Kaustav Basu, Navid Matin-Moghaddam, Saharnaz Mehrani, and Arunabha Sen. Relay node placement under budget constraint. In Paolo Bellavista and Vijay K. Garg, editors, *Proceedings of the 19th International Conference on Distributed Computing and Networking, ICDCN*, pages 35:1–35:11. ACM, 2018.

A Further Related Work

Zelikovsky [35] provided the first approximation algorithm for Directed Steiner Tree Problem (**DSTP**) with factor $O(k^{\varepsilon}(\log^{1/\varepsilon} k))$ that runs in $O(n^{1/\varepsilon})$, where $|U| = k$. Charikar et al. [3] proposed a better approximation algorithm for **DSTP** with a factor $O(\log^3 k)$ in quasi-polynomial time. Grandoni et al. [13] improved this factor and provided a randomized $O(\frac{\log^2 k}{\log\log k})$-approximation algorithm $n^{O(\log^5 k)}$ time. Also, they showed that, unless $NP \subseteq \bigcap_{0<\varepsilon<1} \mathrm{ZPTIME}(2^{n^{\varepsilon}})$ or the Projection Game Conjecture is false, there is no quasi-polynomial time algorithm for **DSTP** that achieves an approximation ratio of $o(\frac{\log^2 k}{\log\log k})$. Ghuge and Nagarajan [11] showed that their approximation algorithm results in a deterministic $O(\frac{\log^2 k}{\log\log k})$-approximation algorithm for **DSTP** in $n^{O(\log^{1+\varepsilon} k)}$ time. Very recently, Li and Laekhanukit [26] showed that the lower bound on the integrality gap of the flow LP is polynomial in the number of vertices.

Danilchenko et al. [7] investigated a closely related problem to **BSCP**, where the goal is to place a set of connected disks (or squares) such that the total weight of target points in the plane is maximized. They provided a polynomial-time $O(1)$-approximation algorithm for this problem. **MCSB** is also closely related to the budgeted connected dominating set problem, where the goal is to select at most B connected vertices in a given undirected graph to maximize the profit function on the set of selected vertices. Khuller et al. [19] investigated this problem in which the profit function is a *special submodular function*. Khuller et al. [19] designed a $\frac{12}{1-1/e}$-approximation algorithm. By generalizing the analysis of Khuller et al. [19], Lamprou et al. [24] showed that there is a $\frac{11}{1-e^{-7/8}}$-approximation algorithm for the budgeted connected dominating set problem. They also showed that for this problem we cannot achieve in polynomial time an approximation factor better than $\left(\frac{1}{1-1/e}\right)$, unless $P = NP$.

Lee and Dooly [25] provided a $(B-2)$-approximation algorithm for **URAT**, where each vertex costs 1. Zhou et al. [36] studied a variant of **E-URAT** in the wireless sensor networks and provided a 10-approximation algorithm. Seufert et al. [31] investigated a special case of the unrooted version of **URAT**, where each vertex has cost 1 and we aim to find a tree with at most B nodes maximizing the accumulated prize. This coincides with the unrooted version of **E-URAT** when the cost of each edge is 1 and we are looking for a tree containing at most $B-1$ edges to maximize the accumulated prize. Seufert et al. [31] provided a $(5+\varepsilon)$-approximation algorithm for this problem. Similarly, Huang et al. [16] investigated this variant of **E-URAT** (or **URAT**) in the plane and proposed a 2-approximation algorithm.

The quota variant of **URAT** also has been studied, which is called **Q-URAT**. Here we wish to find a tree including a vertex r in a way that the total cost of the tree is minimized and its prize is no less than some *quota*. By using Moss and Rabbani [27]'black box and the ideas of Könemann et al. [21], and Bateni et al. [2], we have an $O(\log n)$-approximation algorithm for **Q-URAT**. This bound is tight [27]. The edge cost variant of **Q-URAT**, called **EQ-URAT**, has been investigated by Johnson et al. [18]. They showed that by adapting an α-approximation algorithm for the k-MST problem, one can have an α-approximation algorithm for **EQ-URAT**. Hence, the 2-approximation algorithm of Garg [10] for the k-MST problem results in a 2-approximation algorithm for **EQ-URAT**.

The prize collecting variants of **URAT** have also been studied. Könemann et al. [21] provided a Lagrangian multiplier preserving $O(\ln n)$-approximation algorithm for **NW-PCST**, where the goal is to minimize the cost of the nodes in the resulting tree plus the penalties of vertices not in the tree. Bateni et al. [2] considered a more general case of **NW-PCST** and provided an $O(\log n)$-approximation algorithm. There exists no $o(\ln n)$-approximation algorithm for **NW-PCST**, unless $NP \subseteq \mathrm{DTIME}(n^{\mathrm{Polylog}(n)})$ [20]. The edge cost variant of **NW-PCST** has been investigated by Goemans and Williamson [12]. They provided a 2-approximation algorithm for **EW-PCST**. Later, Archer et al. [1] proposed a $(2-\varepsilon)$-approximation algorithm for **EW-PCST** which was an improvement upon the long standing bound of 2.

Table 1 A summary of the best bounds on some variants of prize collecting problems.

Problem	Best Bound
STO	$O(\frac{\log n}{\log\log n})$ [11] (tight)
DTO	$O(\frac{\log n}{\log\log n})$ [11] (tight)
DSTP	$O(\frac{\log^2 k}{\log\log k})$ [11, 13] (tight)
NW-PCST	$O(\log n)$ [2, 21] (tight)
EW-PCST	$2-\varepsilon$ [1]
URAT	$(1+\varepsilon, O(\frac{\log n}{\varepsilon^2}))$ [2, 21, 27]
E-URAT	2 [29]
Q-URAT	$O(\log n)$ [2, 21, 27] (tight)
EQ-URAT	2 [10, 18]