Bicriteria Approximation Tradeoff for the Node-Cost Budget Problem

Yuval Rabani[1,⋆] and Gabriel Scalosub[2,⋆⋆]

[1] Computer Science Dept., Technion - Israel Institute of Technology, Haifa 32000, Israel.
rabani@cs.technion.ac.il
[2] School of Electrical Engineering, Tel Aviv University, Ramat Aviv 69978, Israel.
gabriels@eng.tau.ac.il

Abstract. We consider an optimization problem consisting of an undirected graph, with cost and profit functions defined on all vertices. The goal is to find a connected subset of vertices with maximum total profit, whose total cost does not exceed a given budget. The best result known prior to this work guaranteed a $(2, O(\log n))$ bicriteria approximation, i.e. the solution's profit is at least a fraction of $\frac{1}{O(\log n)}$ of an optimum solution respecting the budget, while its cost is at most twice the given budget. We improve these results and present a bicriteria tradeoff that, given any $\varepsilon \in (0, 1]$, guarantees a $(1 + \varepsilon, O(\frac{1}{\varepsilon} \log n))$-approximation.

1 Introduction

We consider the following problem: Given an undirected graph $G = (V, E)$, a non negative cost function c defined on V, a non negative profit function π defined on V, and a budget B, our aim is to find a connected set of vertices $T \subseteq V$ such that their overall cost does not exceed B, while maximizing the overall profit obtained from T. We refer to this problem as the *Node-Cost Budget Problem* (or the *Budget Problem (BP)* for short). This problem can also be cast as a rooted problem, where we specify a special vertex $r \in V$ as a root, and impose the restriction that $r \in T$. An algorithm for solving the rooted problem can easily be transformed into one which solves the unrooted problem by simply enumerating over all vertices in V as candidates for r, and picking the best among the various solutions. We shall hereafter focus on the rooted version of the problem. In what follows we denote the size of V by n. We note that previous results for this problem all give bicriteria approximation guarantees, where an (α, β)-approximate solution [1] is one which guarantees at least a β fraction of the optimal profit possible using a given budget, while violating the budget restriction by a factor of at most α.

⋆ Work at the Technion supported by BSF grant number 99-00217, by ISF grant number 386/99, by IST contract number 32007 (APPOL), and by the Fund for the Promotion of Research at the Technion.

⋆⋆ Supported in part by BSF grant number 99-00217. This work was done while the author was with the Computer Science Dept. at the Technion, Israel.

1.1 Our Results

We present, for every $\varepsilon \in (0,1]$, a $(1+\varepsilon, O(\frac{1}{\varepsilon}\log n))$ bicriteria approximation algorithm for BP. This improves upon previous results discussed below. Our algorithm consists of meticulously defining a collection of instances, and finding approximate solutions to these instances using extensions of the methods described in [2]. Upon finding a collection of approximate solutions to these instances, we show that these can be used to find a collection of feasible solutions to our initial problem. We prove that all these solutions violate the budget restriction by a factor of at most $1+\varepsilon$, while at least one has sufficient profit.

1.2 Related Work

The budget problem was introduced by Guha, Moss, Naor, and Schieber [3]. They motivated the problem by its application to power-outage recovery. Guha et al. presented a $(2, O(\log^2 n))$ bicriteria approximation algorithm for the problem. Their algorithm is based on an $O(\log n)$-approximation algorithm for the node weighted Steiner tree problem, devised by Klein and Ravi [4]. The node weighted Steiner tree problem consists of an undirected graph with costs assigned to vertices, where given a subset T of V, one seeks a connected subset of V which contains all the nodes in T such that its overall cost is minimal.

Moss and Rabani [2] improved these results by presenting a $(2, O(\log n))$ bicriteria approximation. Their algorithm is based on a tree packing devised by using a primal-dual algorithm for approximating the prize collecting problem. This problem consists of an undirected graph with costs and profits defined over its set of vertices. The aim is to find a connected subset S of V so as to minimize the sum of the costs of the vertices in S (the cost) and the profits of the vertices not in S (the penalty). Given the tree packing, an averaging argument then enables them to pick out a tree from the packing with good features.

A closely related problem to BP is the node-cost quota problem. In this problem we are given a graph with costs and profits defined over its set of vertices, and a quota Q. The aim is to find a connected subset S of V whose profit is at least Q, and whose cost is minimal. The best upper bound for this problem guarantees an $O(\log n)$-approximation (see [2]). Furthermore, assuming $P \neq NP$ this result is tight, up to a constant factor because the problem is at least as hard as set cover [5].

There has also been a considerable amount of work concerning edge-costs versions of similar problems, such as variants of the prize-collecting Steiner tree problem [6–8] , the k-MST problem [9], and the constrained minimum spanning tree problem [10].

The best approximation lower bound for BP is based on the tight lower bound for the budgeted maximum coverage problem, where due to an approx-

imation preserving reduction from this problem to BP, one can obtain a lower bound of $1 - \frac{1}{e}$ on the approximation ratio of BP [11, 5].

2 Notation and Preliminaries

Given a connected set of vertices $T \subseteq V$, we will speak in terms of any spanning tree induced by T. We may assume without loss of generality that the cost and the profit of the root of an instance is 0. Otherwise, we may solve an altered instance where we assign cost 0 to the root, and strengthen the budget restriction to be at most the original budget from which we subtract the cost of the root in the original problem. Any solution to the altered problem induces a solution with the same value to the original problem and vice-versa, up to re-adding the profit of the root. Let us begin by introducing the concept of *distance* and *reachability* among vertices.

Given a graph $G = (V, E)$ and a non-negative cost function c defined on V, we define the *cost-distance* $d(u, v)$ between vertices u and v to be the minimum total cost of the inner vertices on any path connecting u and v, with respect to the cost function c. Formally:

$$d(u,v) = \min \left\{ \sum_{i=2}^{\ell-1} c(v_i) \;\middle|\; \begin{array}{l} \{v_i\}_{i=1}^{\ell} \text{ is a } u\text{–}v \text{ path} \\ \text{such that } v_1 = u \text{ and } v_\ell = v \end{array} \right\}.$$

If $u = v$ or $(u, v) \in E$ we define $d(u, v) = 0$. If there is no path connecting u and v we define $d(u, v) = \infty$. We further denote by $\mathrm{path}(u, v)$ the set of inner vertices of a path achieving minimal cost. Formally:

$$\mathrm{path}(u,v) = \arg\min \left\{ \sum_{i=2}^{\ell-1} c(v_i) \;\middle|\; \begin{array}{l} \{v_i\}_{i=1}^{\ell} \text{ is a } u\text{–}v \text{ path} \\ \text{such that } v_1 = u \text{ and } v_\ell = v \end{array} \right\} \setminus \{u, v\}.$$

In addition, we say that a vertex v is *reachable with cost at most* p from $u \neq v$ if $d(u, v) + c(v) \leq p$, and any vertex v is reachable from itself with cost 0.

By the above definitions, for any instance of BP, we may assume without loss of generality that all vertices in G are reachable from the root r with cost at most B, since any vertex not reachable from r with cost at most B will not be part of any feasible solution. In particular we may assume G is connected.

For any $S \subseteq V$ we let $c(S) = \sum_{v \in S} c(v)$ denote the cost of S and $\pi(S) = \sum_{v \in S} \pi(v)$ denote the profit obtained by S. We further denote the *density* of S by $\gamma(S) = \frac{\pi(S)}{c(S)}$. For any subtree T of G and any $v \in T$, denote by $CH_T(v)$ the set of children of v in T. We further denote by T_u the subtree of T rooted at vertex $u \in T$, i.e. T_u consists of all vertices in T such that the path connecting them to the root r, contains u.

3 Finding Good Candidate Solutions

In finding a good approximate solution we make use of the notion of tree packing. This enables us to find a connected set of vertices with good properties.

Definition 1 (Tree Packing). *Given a graph $G = (V, E)$, a* tree packing *in G relative to a function $d : V \mapsto \mathbb{Q}^+$ is an assignment of weights λ to a set $\mathcal{T}$ of trees in G, which satisfies $\sum_{T \in \mathcal{T} | v \in T} \lambda_T \leq d(v)$, where λ_T is the weight of tree T in the packing.*

In what follows, for every $S \subseteq V$ we let $\partial S = \{v \in V | \exists u \in S \text{ s.t. } (u, v) \in E\}$. Consider the integer program for BP, denoted IP:

$$\begin{aligned}
&\text{maximize} \quad \textstyle\sum_{i \in V} \pi(i) d_i \\
&\text{subject to} \\
&\qquad \textstyle\sum_{i \in V} c(i) d_i \leq B && (1) \\
&\qquad d_i \leq \textstyle\sum_{j \in \partial S} d_j \quad \forall S \subseteq V \setminus \{r\}, \forall i \in S && (2) \\
&\qquad d_r = 1 && (3) \\
&\qquad d_i \in \{0, 1\} \qquad \forall i \in V \setminus \{r\}. && (4)
\end{aligned}$$

In the above program, vertex i is part of the solution if and only if its associated variable d_i satisfies $d_i = 1$. Constraint (2) ensures the connectivity of the output, i.e. that the vertices for which $d_i = 1$ make up one connected component. Notice that assuming G is not trivial (i.e. $n > 1$), considering constraint (2) for the case where $S = V \setminus \{r\}$ we have $\partial S = \{r\}$ which implies $d_i \leq 1$ for all $i \in V \setminus \{r\}$. Let us denote by OPT_B the value of an optimal solution for IP. If we replace constraints (4) by

$$d_i \geq 0 \qquad \forall i \in V \setminus \{r\} \tag{5}$$

we obtain a linear programming relaxation for BP which we denote by LP. This linear program can be solved in polynomial time using the ellipsoid algorithm [12]. Let d denote an optimal solution to LP. As shown in the following theorem, we can use such a solution to find a tree packing such that every vertex is covered sufficiently by the packing:

Theorem 1 ([2]). *Let $G = (V, E)$ be an undirected graph with non-negative node weights $d : V \mapsto \mathbb{Q}^+$, considered rooted at $r \in V$. Assume d satisfies inequalities (2) and (3). Then, there exists a polynomial time algorithm that computes a tree packing in G of trees containing r such that for every node $v \in V$*

$$\frac{d(v)}{c \log n} \leq \sum_{T \in \mathcal{T} | v \in T} \lambda_T \leq d(v)$$

for some constant c independent of n.

Let $\mathcal{T}$ be the support of the tree packing guaranteed by Theorem 1, let $\mathcal{L} = \{T \in \mathcal{T} | c(T) \leq B\}$ and $\mathcal{H} = \{T \in \mathcal{T} | c(T) > B\}$. The following lemma will serve as a starting point in our quest for finding a good approximate solution.

Lemma 1 (A good tree exists in the packing [2]). *Given the support $\mathcal{T}$ of the packing guaranteed by Theorem 1, at least one of the following conditions holds:*

1. *$\exists T \in \mathcal{L}$ such that $\pi(T) \geq \frac{1}{2c \log n} OPT_B$;*
2. *$\exists T \in \mathcal{H}$ such that $\gamma(T) = \frac{\pi(T)}{c(T)} \geq \frac{1}{2c \log n} \frac{OPT_B}{B}$.*

Furthermore a tree T satisfying one of the above conditions can be found in time polynomial in the size of the original input.

In what follows we focus our attention on the case where we have a tree T which satisfies condition 2 in Lemma 1. Note that the cost of such a tree may very well exceed the available budget. In the following section we presents a trimming process, which under some conditions on the underlying instance, enables us to give a good upper bound on the cost of the resulting tree.

3.1 Candidate Solutions with High Density

Consider an instance of BP with graph $G = (V, E)$ such that all vertices are reachable from the root $r \in V$ with cost at most d, and a budget restriction B, where $d \leq B$. In what follows we show that for any $\alpha > 0$ and any subtree T of G rooted at r such that $\gamma(T) \geq \alpha \cdot \frac{\mathrm{OPT}_B}{B}$ and $c(T) > B$, T can be trimmed into a tree T^H satisfying $\pi(T^H) \geq \frac{\alpha}{4} \cdot \mathrm{OPT}_B$ and $c(T^H) \leq B + d$.

Let T be any subtree of G rooted at r such that $\gamma(T) \geq \alpha \cdot \frac{\mathrm{OPT}_B}{B}$ and $c(T) > B$. In order to obtain T^H, we will accumulate subtrees of T, while making sure the overall cost remains within a certain range. The following lemma, whose proof is omitted due to space constraints, gives a sufficient condition for the resulting tree having sufficient profit:

Lemma 2. *Let $T \subseteq G$ be a tree such that $\gamma(T) \geq \alpha \cdot \frac{\mathrm{OPT}_B}{B}$ and $c(T) > B$. For any set of vertices $U \subseteq T$ such that for every $u \in U$, $\gamma(T_u) \geq \gamma(T)$, if $\sum_{u \in U} c(T_u) \geq \frac{B}{2}$ then $\sum_{u \in U} \pi(T_u) \geq \frac{\alpha}{2} \cdot OPT_B$.*

The following corollary is an immediate consequence of Lemma 2:

Corollary 1. *Let $T \subseteq G$ be a tree such that $\gamma(T) \geq \alpha \cdot \frac{\mathrm{OPT}_B}{B}$ and $c(T) > B$. For any $v \in T$, and any set of vertices $U \subseteq CH_T(v)$, if U satisfies $\frac{B}{2} \leq \sum_{u \in U} c(T_u) \leq B$ and for every $u \in U$, $\gamma(T_u) \geq \gamma(T)$, then the set $T^H = \bigcup_{u \in U} T_u \cup \{v\} \cup \mathrm{path}(r, v) \cup \{r\}$ is a tree which satisfies $\pi(T^H) \geq \frac{\alpha}{2} \cdot OPT_B$ and $c(T^H) \leq B + d$.*

Proof. Note that by the assumption that $U \subseteq CH_T(v)$ for some $v \in T$, we are guaranteed to have for every $u \neq u'$ in U, $T_u \cap T_{u'} = \emptyset$. It therefore follows that $c(\bigcup_{u \in U} T_u) = \sum_{u \in U} c(T_u)$ and $\pi(\bigcup_{u \in U} T_u) = \sum_{u \in U} \pi(T_u)$. Furthermore, since $U \subseteq CH_T(v)$, then clearly T^H is a tree. Since $\bigcup_{u \in U} T_u \subseteq T^H$ then by Lemma 2 we are guaranteed to have $\pi(T^H) \geq \frac{\alpha}{2} \cdot OPT_B$. On the other hand, since every node is reachable from the root $r \in V$ with cost at most d, the cost of the path $\{r\} \cup \text{path}(r, v) \cup \{v\}$ is at most d. It therefore follows that the overall cost of T^H is at most $B + d$, as required. □

The following lemma, whose proof is omitted due to space constraints, guarantees we can trim a high density tree such that the resulting tree carries a sufficiently large profit, while having a limited cost.

Lemma 3. *If* $T = \arg\max \{\gamma(T') | T' \in \mathcal{T} \setminus \mathcal{L}\}$ *satisfies* $\gamma(T) \geq \frac{1}{2c \log n} \frac{OPT_B}{B}$ *for some constant* c*, then there exists a polynomial trimming algorithm* TRIM*, such that* $T^H = \text{TRIM}(T)$ *satisfies* $\pi(T^H) = \Omega\left(\frac{1}{\log n}\right) OPT_B$ *and* $c(T^H) \leq B + d$*.*

3.2 An Algorithm for Finding a Good Candidate

Clearly any tree T satisfying condition 1 in Lemma 1, would suffice for our purpose, since such a tree does not violate the budget constraint, while carrying at least an $\Omega\left(\frac{1}{\log n}\right)$ fraction of the optimal profit. On the other hand, by Lemma 3, we can trim any tree T satisfying condition 2 in Lemma 1, so as to obtain a tree with cost at most $B + d$, carrying at least an $\Omega(\frac{1}{\log n})$ fraction of the optimal profit. Algorithm EXTRACT described in Algorithm 1 therefore summarizes the method for finding a subtree $T \subseteq G$ such that $c(T) \leq B + d$ and $\pi(T) = \Omega\left(\frac{1}{\log n}\right) \text{OPT}_B$.

Algorithm 1 EXTRACT (tree packing $\mathcal{T}$, budget B)

1: set $\mathcal{L} = \{T \in \mathcal{T} | c(T) \leq B\}$
2: set $\mathcal{H} = \mathcal{T} \setminus \mathcal{L}$
3: set $T^L = \arg\max \{\pi(T) | T \in \mathcal{L}\}$.
4: set $T = \arg\max \{\pi(T)/c(T) | T \in \mathcal{H}\}$
5: set $T^H = \text{TRIM}(T)$
6: return $\arg\max \left\{\pi(T^L), \pi(T^H)\right\}$

The above proves the following lemma:

Lemma 4. *Given any instance of BP with graph* $G = (V, E)$ *such that all vertices are reachable from the root* $r \in V$ *with cost at most* d*, and a budget*

restriction B, where $d \leq B$, algorithm EXTRACT *produces a solution T such that $c(T) \leq B + d$ and $\pi(T) = \Omega\left(\frac{1}{\log n}\right) \mathrm{OPT}_B$.*

4 Structure of an Optimal Solution

In this section, we study the structure of an optimal solution. We show that assuming all optimal solutions have sufficiently large cost, there exists a decomposition of an optimal solution into disjoint subtrees, such that at least one of them has sufficiently small cost, while carrying at least an $O\left(\frac{1}{\log n}\right)$ fraction of the profit attained by the optimal solution. Our study later motivates the definition of a sequence of instances of BP, each corresponding to one of these subtrees, such that at least one of these instances can be transformed into a $(1+\varepsilon, \log n)$-approximate solution.

First note that if there exists an optimal solution T^*, such that $c(T^*) \leq \frac{B}{2}$, then by applying the algorithm of [2] over the same instance, with a budget restriction of $B' = \frac{B}{2}$, we are guaranteed to obtain a $(1, O(\log n))$-approximate solution. We can therefore assume that for every optimal solution T^*, $c(T^*) > \frac{B}{2}$.

Let T^* be any such optimal solution, and let $\varepsilon \in (0, 1]$. Define $k = \lceil \frac{1+\varepsilon}{2\varepsilon} \rceil = \Theta\left(\frac{1}{\varepsilon}\right)$, and let $\overline{\sigma} = \overline{\sigma}(k) = (\sigma_1, \ldots, \sigma_{k-1})$ be a sequence of proportions, $\sigma_i \leq 1$ for all $i = 1, \ldots, k-1$, such that $\sigma_1 \leq \frac{1}{2}$. In what follows we let $\rho_i = \prod_{j=1}^{i} \sigma_j$, $i = 1, \ldots, k-1$ and define $\rho_0 = 1$. We now describe a recursive partition of T^* into k disjoint connected components $\{T_i^*\}_{i=0}^{k-1}$ according to $\overline{\sigma}$. Let $r_0 = r$ and let $G_0^* = T^*$. Given G_{i-1}^*, let $r_i \in G_{i-1}^*$ be such that

$$c(T_{r_i}^*) \geq \rho_i B \tag{6}$$

$$c(T_u^*) \leq \rho_i B \qquad \forall u \in CH_{T^*}(r_i), \tag{7}$$

and define $G_i^* = T_{r_i}^*$.

First note that for every $i = 1, \ldots, k-1$, there exists a node $r_i \in G_{i-1}^*$ which satisfies conditions (6) and (7). To see this, note that by our assumption that $\sigma_1 \leq \frac{1}{2}$ we have $\rho_1 B \leq \frac{B}{2}$. On the other hand, we have assumed that $c(T^*) > \frac{B}{2}$, hence $c(T_{r_0}^*) = c(T^*) \geq \rho_0 B$. Since $\rho_i \geq \rho_{i+1}$, assuming we have found a node $r_i \in G_{i-1}^*$ satisfying (6) and (7), we are guaranteed that r_i also satisfies $c(T_{r_i}^*) \geq \rho_{i+1} B$. Consider any maximal path of nodes $r_i = v_0, \ldots, v_\ell$ in G_i^*, such that for every $j = 0, \ldots, \ell$, $c(T_{v_j}^*) \geq \rho_{i+1} B$. Such a path necessarily exists since the tree is finite, and $c(T_{v_0}^*) \geq \rho_{i+1} B$. By maximality if follows that we can pick $r_{i+1} = v_\ell$, which would satisfy both condition (6) and condition (7).

After having defined subtrees $G_0^*, \ldots, G_{k-1}^*$ as described above, we can assume without loss of generality that for all $i = 0, \ldots, k-2$, $G_i^* \neq G_{i+1}^*$, and

define $T_i^* = G_i^* \setminus G_{i+1}^*$ for all $i = 0, \ldots, k-2$ and $T_{k-1}^* = G_{k-1}^*$. We call such a partition a $\overline{\sigma}$*-partition* of T^*. Note that such a partition can be identified by its corresponding sequence of roots $r_0, \ldots, r_{k-1}$. See Figure 1 for the schematics of a $\overline{\sigma}$-partition for $k = 4$.

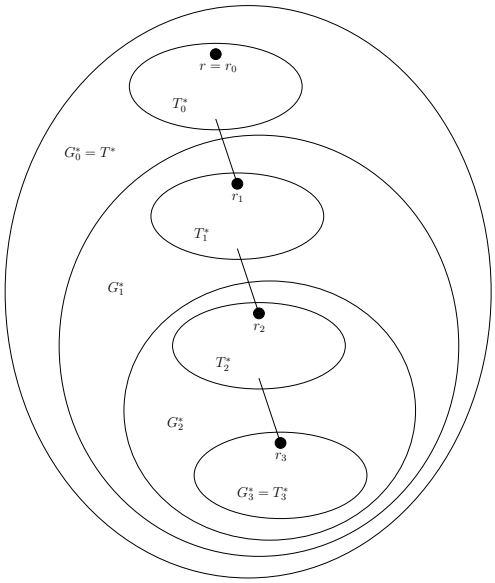

Fig. 1. Schematic $\overline{\sigma}$-partition for $k = 4$

We will first bound the reachability of vertices in a component T_i^* from r_i.

Lemma 5. *For any k, given a $\overline{\sigma}$-partition of T^*, for every $i = 1, \ldots, k-1$, all vertices in G_i^* are reachable from r_i with cost at most $\rho_i B$.*

Proof. Let v be a vertex in G_i^*. If $v = r_i$ we are done. Otherwise, there exists a vertex $u \in CH_{T^*}(r_i)$ such that $v \in T_u^*$. In particular, $\mathrm{path}(r_i, v) \cup \{v\} \subseteq T_u^*$, which by the definition of reachability and the choice of r_i yields

$$d(r_i, v) + c(v) = c(\mathrm{path}(r_i, v) \cup \{v\}) \leq c(T_u^*) \leq \rho_i B$$

□

Since T_i^* is a subset of G_i^* we have the following corollary:

Corollary 2. *Given a $\overline{\sigma}$-partition of T^*, for every $i = 1, \ldots, k-1$, all vertices in T_i^* are reachable from r_i with cost at most $\rho_i B$.*

We now turn to bound the cost of every component T_i^*.

Lemma 6. *For any k, given a $\overline{\sigma}$-partition of T^*, the following holds:*

1. $c(T_0^*) \leq (1-\rho_1)B$
2. $c(T_i^*) \leq B - d(r,r_i) - \rho_{i+1}B$ *for all* $i = 1,\ldots,k-2$.
3. $c(T_{k-1}^*) \leq B - d(r,r_{k-1})$

Proof. To prove part 1 simply note that by the feasibility of T^*, its cost is upper bounded by B, and since by the choice of r_1, $c(G_1^*) \geq \rho_1 B$, the result follows from the definition of T_0^*. In an even simpler manner, part 3 also follows from the feasibility of T^* and the definition of T_{k-1}^* as its subtree. As for part 2, since $\rho_i B \leq c(G_i^*) \leq B - d(r,r_i)$ then by the definition of T_i^* we have

$$\begin{aligned} c(T_i^*) &= c(G_i^* \setminus G_{i+1}^*) \leq \\ &\leq B - d(r,r_i) - \rho_{i+1}B \end{aligned}$$

which completes the proof. □

5 The Algorithm

5.1 Road Map

The structural analysis of an optimal solution presented in the previous section motivates the definition of a sequence of instances of BP, each corresponding to a different component in the decomposition.

Let T^* be an optimal solution to an instance I of BP, with budget restriction B, and let $k = \Theta\left(\frac{1}{\varepsilon}\right)$, as defined in section 4. For every $i = 0,\ldots,k-1$ and $v \in V$ we define a new instance of BP, $I_{i,v}$. In instance $I_{i,v}$ we actually "guess" that v is the root of T_i^* (i.e. r_i). We then consider all vertices with distance d_i from v, and a budget restriction B_i. Using Lemma 4, we can conclude that we can find a tree $T'_{i,v}$ with cost at most $B_i + d_i$. Our choices of B_i and d_i will be such that two conditions are met; First, for every i and every v, $T'_{i,v}$ can be extended to a solution $T_{i,v}$ to the original instance, with overall cost at most $(1+\varepsilon)B$. Second, for every i, T_i^* is a feasible solution to at least one of the new instances. It follows that at least one of solutions $T_{i,v}$ carries a profit of at least an $\Omega\left(\frac{\varepsilon}{\log n}\right)$ fraction of the optimal profit.

5.2 Detailed Description

Let I be any instance of BP over a graph $G = (V,E)$, with a budget restriction B. Given any $v \in V$ and $i \in \{0,\ldots,k-1\}$, every instance $I_{i,v}$ will be determined by two parameters, whose values are motivated by Corollary 2

and Lemma 6; d_i representing a reachability radius around v, and a budget B_i. Specifically, we consider the subset $V_{i,v} \subset V$ consisting of all vertices with distance at most d_i from v, and let instance $I_{i,v}$ be the instance over the subgraph of G induced by $V_{i,v}$, with budget restriction B_i. Table 1 shows the different values of d_i and B_i, depending on i.

i	d_i	B_i	$B_i + d_i$
0	$(1-\rho_1)B$	$(1-\rho_1)B$	$2(1-\rho_1)B$
$1,\ldots,k-2$	$\rho_i B$	$B - d(r,v) - \rho_{i+1}B$	$(1+(1-\sigma_{i+1})\rho_i)B - d(r,v)$
$k-1$	$\rho_{k-1}B$	$B - d(r,v)$	$(1+\rho_{k-1})B - d(r,v)$

Table 1. Values of reachability distance and budget of the new instances $I_{i,v}$

For every instance $I_{i,v}$, we solve the corresponding LP, compute the tree packing $\mathcal{T}_{i,v}$ which corresponds to $I_{i,v}$ implied by Theorem 1, and let $T'_{i,v} = \textsc{Extract}(\mathcal{T}_{i,v})$. Note that by Corollary 2 and Lemma 6, for every $i = 0,\ldots,k-1$, T^*_i is a feasible solution for I_{i,r_i}. The following lemma is an immediate consequence:

Lemma 7. *Given the above notation, for every $v \in V$ and $i = 0,\ldots,k-1$, $c(T'_{i,v}) \leq B_i + d_i$, as specified in Table 1. Furthermore, for every $i = 1,\ldots,k-1$, $\pi(T'_{i,r_i}) = \Omega\left(\frac{1}{\log n}\right)\pi(T^*_i)$.*

For every $v \in V$ and $i = 0,\ldots,k-1$, let $T_{i,v} = T'_{i,v} \cup \mathrm{path}(r,v) \cup \{r\}$. Note that every $T_{i,v}$ is a tree rooted at r. The following corollary follows immediately from Lemma 7:

Corollary 3. *Given the above notation*

1. $c(T_{0,v}) \leq 2\,(1-\rho_1)\,B$, *for all* $v \in V$.
2. $c(T_{i,v}) \leq (1+(1-\sigma_{i+1})\,\rho_i)\,B$, *for all* $i = 1,\ldots,k-2$ *and* $v \in V$.
3. $c(T_{k-1,v}) \leq (1+\rho_{k-1})\,B$, *for all* $v \in V$.
4. $\pi(T_{i,r_i}) = \Omega\left(\frac{1}{\log n}\right)\pi(T^*_i)$ *for all* $i = 0,\ldots,k-1$.

The following lemma, whose proof is omitted due to space constraints, ensures we can bound the cost of every $T_{i,v}$ constructed above by $(1+\varepsilon)B$:

Lemma 8. *For $\overline{\sigma}$ defined by*

$$\sigma_i = \begin{cases} \dfrac{k-i}{k-i+1} & i = 2, \ldots, k-1 \\ \dfrac{k-1}{2k-1} & i = 1, \end{cases} \tag{8}$$

we have $c(T_{i,v}) \leq (1+\varepsilon)B$ for all $i = 0, \ldots, k-1$ and $v \in V$.[3]

We can now prove our main result, as to the performance of algorithm BPA described in Algorithm 2.

Algorithm 2 BPA ($G = (V, E)$, costs c, profits π, budget B)

```
1: S_1 ← MR(G, c, π, B/2)        ▷ Apply the algorithm appearing in [2] with half the budget
2: S_2 ← ∅
3: for every i ∈ {0, ..., k − 1} and v ∈ V do
4:     let I_{i,v} be the appropriate instance
5:     solve the LP corresponding to I_{i,v}
6:     let T_{i,v} be its corresponding tree packing
7:     T'_{i,v} ← EXTRACT(T_{i,v})
8:     T_{i,v} ← T'_{i,v} ∪ path(r, v) ∪ {r}
9:     if π(T_{i,v}) > π(S_2) then
10:        S_2 ← T_{i,v}
11:    end if
12: end for
13: T ← arg max {π(S_1), π(S_2)}
14: return T
```

Theorem 2. *For any $\varepsilon \in (0, 1]$, algorithm* BPA *produces a $(1+\varepsilon, O(\frac{1}{\varepsilon}\log n))$-approximate solution to BP.*

Proof. The first candidate solution considered by the algorithm (line 1) is the solution produced by applying the algorithm proposed by Moss and Rabani [2]. If there exists an optimal solution with cost at most $\frac{B}{2}$, then this candidate is guaranteed to be a $(1, O(\log n))$-approximate solution. The remainder of the algorithm is designed to deal with the case where every optimal solution has cost greater than $\frac{B}{2}$.

By the analysis presented in Section 4 and the pigeonhole principle we are guaranteed to have at least one subtree T_i^* contributing at least a $\frac{1}{k}$ fraction of the optimum's profit. Let m be the index of such a subtree. Consider the instance

[3] Note that by our choice, we have $\sigma_1 \leq \frac{1}{2}$, which guarantees our solutions correspond to the decomposition described in Section 4.

corresponding to I_{m,r_m}, and let T'_{m,r_m} be the candidate solution produced by the algorithm in the appropriate iteration.

By Lemma 4 and the fact that $k = \Theta(\frac{1}{\varepsilon})$, the subtree T'_{m,r_m} produced in line 7 satisfies

$$\pi(T'_{m,r_m}) = \Omega\left(\frac{1}{\log n}\right)\pi(T^*_m) = \Omega\left(\frac{\varepsilon}{\log n}\right)\mathrm{OPT}_B\,.$$

Since $T'_{m,r_m} \subseteq T_{m,r_m}$, we are guaranteed to have $\pi(T_{m,r_m}) = \Omega\left(\frac{\varepsilon}{\log n}\right)\mathrm{OPT}_B$.

Since by Lemma 8 the cost of every candidate solution produced by the algorithm in line 8 is at most $(1+\varepsilon)B$, and as shown above at least one of them has profit at least $\Omega\left(\frac{\varepsilon}{\log n}\right)OPT_B$, the result follows. □

References

1. Marathe, M., Ravi, R., Sundaram, R., Ravi, S., Rosenkrantz, D., (III), H.H.: Bicriteria network design problems. Journal of Algorithms **28**(1) (1998) 142–171
2. Moss, A., Rabani, Y.: Approximation algorithms for constrained node weighted steiner tree problems. SIAM Journal on Computing **37**(2) (2007) 460–481
3. Guha, S., Moss, A., Naor, J., Schieber, B.: Efficient recovery from power outage. In: Proceedings of the 31st ACM Symposium on Theory of Computing. (1999) 574–582
4. Klein, P., Ravi, R.: A nearly best-possible approximation algorithm for node-weighted steiner trees. Journal of Algorithms **19**(1) (1995) 104–114
5. Feige, U.: Threshold of $\ln n$ for approximating set cover. Journal of the ACM **45**(4) (1998) 634–652
6. Goemans, M.X., Williamson, D.P.: A general approximation technique for constrained forest problems. SIAM Journal on Computing **24**(2) (1995) 296–317
7. Johnson, D., Minkoff, M., Phillips, S.: The prize collecting steiner tree problem: theory and practice. In: Proceedings of the 11th annual ACM-SIAM symposium on Discrete algorithms. (2000) 760–769
8. Jain, K., Hajiaghayi, M.: The prize-collecting generalized steiner tree problem via a new approach of primal-dual schema. In: Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms. (2006) 631–640
9. Garg, N.: Saving an epsilon: a 2-approximation for the k-mst problem in graphs. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing. (2005) 396–402
10. Ravi, R., Goemans, M.: The constrained minimum spanning tree problem. In: Proceedings of the 5th Scandinavian Workshop on Algorithmic Theory. (1996) 66–75
11. Khuller, S., Moss, A., Naor, S.: The budgeted maximum coverage problem. Information Processing Letters **70**(1) (1999) 39–45
12. Grötschel, M., Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Second corrected edn. Springer-Verlag (1993)