Information Processing Letters 158 (2020) 105940

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Approximation algorithms for connected maximum coverage problem for the discovery of mutated driver pathways in cancer[☆]

Dorit S. Hochbaum[1], Xu Rao[*,1]

Department of IEOR, Etcheverry Hall, Berkeley, CA 94709, USA

ARTICLE INFO

Article history:
Received 28 October 2019
Received in revised form 14 February 2020
Accepted 24 February 2020
Available online 26 February 2020
Communicated by R. Lazic

Keywords:
Approximation algorithms
Connected maximum coverage
Bounded radius subgraph

ABSTRACT

This paper addresses the connected maximum coverage problem, motivated by the detection of mutated driver pathways in cancer. The connected maximum coverage problem is NP-hard and therefore approximation algorithms are of interest. We provide here an approximation algorithm for the problem with an approximation bound that strictly improves on previous results. A second approximation algorithm with faster run time, though worse approximation factor, is presented as well. The two algorithms are then applied to submodular maximization over a connected subgraph, with a monotone submodular set function, delivering the same approximation bounds as for the coverage maximization case.

1. Introduction

We address here the connected maximum coverage problem. Given a collection of subsets, each associated with a node in a graph, and an integer k, the goal is to find up to k subsets the union of which is of largest cardinality so that the respective nodes in the graph form a connected subgraph. This problem generalizes the maximum coverage problem which is to find k subsets that jointly cover the most elements.

The motivation for connected maximum coverage problem is the detection of mutations associated with cancer. The search for driver mutations in cancer has accelerated with recent advances in sequencing technologies. These technologies enable measurements in large cohorts of cancer patients and identify their individual list of gene mutations. Each mutation in the list is thus associated with a subset of patients whose profiles include this mutation. The goal is to “explain” the cancer with up to k mutations, where explanation means that these mutations cover jointly the largest possible collection of patients. These selected mutations are considered to be the driver mutations that are most associated with the specific type of cancer. The connectivity requirement is motivated by the belief that cancer is a disease of pathways [1]. Pathways are connected subnetworks in a large gene interaction network. Therefore, the goal is to identify mutations that not only deliver large coverage of the set of patients, but also form a connected subnetwork in the interaction network.

In the graph representation of the driver mutations' detection problem, each node of the graph corresponds to a protein and its associated gene (mutation), and thus, also associated with a subset of patients whose profiles include the mutation. The edges of the graph represent the pairwise protein-to-protein interactions. The detection of k driver mutations is then the maximum coverage problem on this graph.

[☆] Research supported in part by NSF award No. CMMI-1760102.

[*] Corresponding author.
E-mail addresses: hochbaum@ieor.berkeley.edu (D.S. Hochbaum), xrao@berkeley.edu (X. Rao).

[1] Department of Industrial Engineering and Operations Research, University of California, Berkeley, USA.

https://doi.org/10.1016/j.ipl.2020.105940

Formally, the input to an instance of the connected maximum coverage problem is a graph $G = (V, E)$, where each node $v \in V$ be associated with a set $S_v \subset P$ (possibly empty), for P a universal set, and an integer k. The goal is to find a subset $V' \subset V$ of cardinality less than or equal to k, that induces a connected subgraph in G such that $|\bigcup_{v \in V'} S_v|$ is maximized.

The connected maximum coverage problem is easily seen to be NP-hard as it is a special case of the *maximum coverage* problem [2]. The maximum coverage problem was studied by Hochbaum and Pathria [3] who showed that the problem is NP-hard and devised a simple greedy algorithm with an approximation factor of $1 - 1/e = 0.632121\ldots$. This approximation factor is the best-possible attained in polynomial time for maximum coverage problem under the assumption that $P \neq NP$ [4].

Problems closely related to the connected maximum coverage problem include the problem of finding a k-nodes connected subgraph that maximizes the **sum** of weights of the nodes. This problem was studied by Hochbaum and Pathria in [5] who showed that this problem is in general NP-hard, but solvable in polynomial time for a tree graph. Seufert et al. (2010) [6] addressed this NP-hard connected maximum sum of weights problem (on general graphs) and provided an $1/(5(1 + \epsilon))$-approximation algorithm.

Vandin et al. [7] studied the connected maximum coverage problem and provided an approximation algorithm. Bomersbach et al. [8] formulated the connected maximum coverage problem as an integer programming problem and used a branch and cut algorithm to get exact solutions. Both of these papers that address the connected maximum coverage problem are motivated by the search for driver mutations in cancer.

Our main contribution here is the improvement of the approximation factor for the connected maximum coverage problem, with an algorithm we call the Bounded Radius Algorithm. The approximation factor achieved by Vandin et al. [7] is $1/(cr_{\text{OPT}})$, where $c = (2e - 1)/(e - 1) = 2.581976\ldots$ and r_{OPT} is the *radius* of the optimal solution subnetwork. The approximation factor of our Bounded Radius Algorithm is $\max\{(1 - 1/e)\,(1/r_{\text{OPT}} - 1/k)\,, 1/k\}$, which is demonstrated to always improve on the factor of $1/(cr_{\text{OPT}})$. We also provide an alternative approximation algorithm, the *Greedy Path Algorithm*, with approximation factor $\max\{(1 - 1/e)\,(1/R - 1/k)\,, 1/k\}$ where R is the *radius* of the input graph G. The Greedy Path Algorithm is always worse than the Bounded Radius Algorithm in terms of approximation bound, but always better in terms of running time.

Another contribution here is the generalization of the connected maximum coverage to a *connected submodular maximization* problem. It is shown here that our two approximation algorithms apply also to the connected submodular maximization, for monotone submodular set functions, delivering the same approximation bounds as for the connected maximum coverage problem.

2. Preliminaries and notations

An undirected connected graph is denoted by $G = (V, E)$. We use the standard notation of $m = |E|$ for the number of edges, and $n = |V|$ for the number of nodes in the graph.

We let the *distance* between two nodes in the graph, $v_1 \in V$ and $v_2 \in V$ be the shortest path length between the two nodes where all the edge weights are of unit length. We denote this distance by $d(v_1, v_2)$. It is easy to find the distance between node v_1 and all other nodes in V in $O(m)$ time using breadth first search (BFS). The level of each node in the BFS tree rooted in v_1 is its distance from v_1. Additional details about BFS are provided in the next subsection.

Definition 1. The *eccentricity* $\epsilon(v)$ of a node $v \in V$ is the longest distance between v and any other node in V.

The eccentricity of node v in G can be found using BFS from v, where $\epsilon(v)$ is equal to the depth of the BFS tree (the deepest level in the tree).

Definition 2. The *radius* of a graph is the minimum eccentricity among all nodes in the graph, $\min_{v \in V} \epsilon(v)$. The argument of this minimum is attained for node v^* called the *center* of the graph. $\epsilon(v^*)$ is then the radius of the graph.

Let the radius of graph G be denoted by $R(G)$. Note that the center of the graph is not necessarily unique.

It is easy to see that $R(G) \leq n/2$ and that this inequality is tight when the graph is an n node path and the number of nodes n is even. We therefore conclude that,

Lemma 1. *The radius of the optimal connected subgraph on k-nodes, r_{OPT}, satisfies $r_{OPT} \leq k/2$.*

In order to find the radius and center of a graph we use breadth-first-search (BFS). The procedure creates the BFS-tree rooted at the node called “root”. The longest distance from the root to another node is attained for a leaf node. Hence, to find the eccentricity of node v it suffices to evaluate the distance from the root v of each one of the leaves (the level of each leaf), and to take the maximum distance. To find the radius of the graph it is sufficient to initiate BFS from each node of the graph, compute the respective eccentricity and take the minimum. The complexity of BFS is known to be $O(m)$ and therefore, finding the radius and the center of a graph takes at most $O(nm)$ steps for a graph on n nodes and m edges.

The parent of a node v in a rooted tree T is denoted by $p(v)$.

The algorithms presented in this paper make calls to a subroutine that merges two sets and measure the size of the union. There are various algorithms of different complexities for finding set unions and the size of the unions, that depend on different assumptions on how the input is given. Here, we assume that for any $v \in V$, subset S_v is represented by an array of ordered indexes of length $|S_v|$, where i is in the array if and only if the ith element of P is in the subset S_v. Then we can traverse the two subsets to get their union set and measure the size of the union, in $O(N)$ time where $N = |P|$.

3. The approximation algorithm for the maximum coverage problem

The maximum coverage problem differs from the connected maximum coverage problem in that it does not require the connectivity of the subgraph selected. We make use here of results for the maximum coverage problem addressed in [3]. The relevant results are reviewed here for the sake of completeness.

Given a family $\mathcal{F}$ of sets, an integer k, the maximum coverage problem is to find k sets such that the size of the union of the selected sets is maximized. Formally, the goal is to identify sets $S_{i_1}, S_{i_2}, \ldots, S_{i_k} \in \mathcal{F}$, such that $|\bigcup_{j=1}^{k} S_{i_j}|$ is maximum. The problem was shown to be NP-hard and the following greedy algorithm of [3] delivers the state-of-the-art approximation for the problem.

Algorithm 1 Maximum coverage greedy algorithm [3].

```
Input $\mathcal{F}, k$.
$\mathcal{S} \leftarrow \emptyset$, $U \leftarrow \emptyset$.
for $i = 1, \ldots, k$ do
    Select set $S_i$ from $\mathcal{F} \setminus \mathcal{S}$ that maximizes $|U \cup S_i|$; $\mathcal{S} \leftarrow \mathcal{S} \cup \{S_i\}$, $U \leftarrow U \cup S_i$.
end for
Output The collection of $k$ sets $\mathcal{S}$.
```

Theorem 1 *([3]). The Maximum Coverage Greedy Algorithm is a $(1 - 1/e)$-approximation algorithm for the maximum coverage problem.*

The actual approximation bound proved in [3] is better for any finite value of k: $1 - (1 - 1/k)^k > 1 - 1/e$.

4. Approximation algorithms for the connected maximum coverage problem

There is a trivial $1/k$-approximate solution for the connected maximum coverage problem. The *singleton greedy algorithm* selects a single node with the associated subset of largest size:

Lemma 2. *The* singleton greedy algorithm *delivers a $1/k$-approximate solution.*

The proof is obvious and therefore omitted.

For small values of k the singleton greedy algorithm gives a good approximate solution. In the following two subsections, we present two approximation algorithms that give better approximation factors than $1/k$ for large k.

First we present an approximation algorithm with an approximation factor of $(1 - 1/e)(1/R - 1/k)$ for $R = R(G)$, the radius of graph G, with the assumption that $k > R$. Next, the approximation bound of the first algorithm is improved to $(1 - 1/e)(1/r_{\text{OPT}} - 1/k)$, where r_{OPT} is the radius of the optimal subgraph. The second algorithm requires $k > r_{\text{OPT}}$, which is always satisfied as the optimal subgraph have at most k nodes. Since $r_{\text{OPT}} \le R$, this improvement is meaningful. We call the first algorithm, the *Greedy Path Algorithm*, and we call the second one, the *Bounded Radius Algorithm*.

4.1. Greedy path algorithm

The input to the Greedy Path Algorithm is the radius R and the center node v^* of graph G, subset S_{v^*}, a subset of nodes L and sets P_ℓ, A_ℓ for each $\ell \in L$. The subset of nodes L is the set of leaves of a BFS tree T rooted at v^*. For each $\ell \in L$, P_ℓ is the set of nodes on the unique path on T between v^* and ℓ (including ℓ but not v^*), and A_ℓ is the union of subsets associated to nodes in P_ℓ, $A_\ell = \cup_{v \in P_\ell} S_v$.

The input parameters are computed first in a preprocessing step. This includes running BFS rooted at each node in V and a dynamic programming procedure. As described earlier, we run n BFS trees to get the center v^*, radius R, and the BFS tree, T, rooted at v^* with its set of leaves L. Consider the BFS tree T rooted at v^* to be directed from parents to children. Then the BFS ordering is a topological ordering, and we can compute P_v, A_v for each v in topological order with the dynamic programming procedure:

Initialize $P_{v^*} := \{v^*\}$, $A_{v^*} := S_{v^*}$;

the dynamic programming recursion is then:

$P_v = \{v\} \cup P_{p(v)}, \;\; A_v = S_v \cup A_{p(v)}$.

The algorithm maintains a set of nodes V' that have been selected, as well as the union U of the subsets corresponding to the nodes in V', $U = \cup_{v \in V'} S_v$.

Algorithm 2 Greedy path algorithm.

```
Input Integer $k$, center $v^*$ and radius $R$ of $G$, $S_{v^*}$, subset $L$ and subsets $A_\ell, P_\ell$ for each $\ell \in L$.
$V' \leftarrow \{v^*\}, U \leftarrow S_{v^*}$.
while $k - |V'| \ge R$ do
    Select $\ell$ from $L$ that maximizes $|U \cup A_\ell|$, $V' \leftarrow V' \cup P_\ell$, $U \leftarrow U \cup A_\ell, L \leftarrow L \setminus \{\ell\}$.
end while
Output $V'$.
```

By the definition of radius and the center, $|P_\ell| \le R$ for any $\ell \in L$ so in any iteration of the while loop, at most R nodes are added to set V'. Therefore, output V' must have a size no more than k. Since the set of nodes V' also induce a connected subgraph, the output of the algorithm is a feasible solution.

Let k' denote the number of iterations of the while loop.

Lemma 3. *The number of iterations of the while loop, k', is at least $(k - R)/R$.*

Proof. The initial size of set V' is 1. At each iteration of while loop, at most R new nodes are added to V' as $|P_\ell| \le R$ for any $\ell \in L$. Hence $|V'| \le 1 + k'R$. On the other hand, the termination condition of the while loop is $|V'| > k - R$ (or equivalently, $|V'| \ge k - R + 1$). Therefore, $1 + k'R \ge k - R + 1$, and thus, $k' \ge (k - R)/R$. □

Let the optimal solution to the connected maximum coverage problem be the set of k nodes V^*. Let z^* be the value of the optimal solution (the size of the union of the sets in V^* or the optimal coverage) and let z_1 be the value of the Greedy Path Algorithm's solution (the size of the union of the sets in V', or the algorithm's coverage).

Theorem 2. $z_1 \ge \left(1 - \frac{1}{e}\right)\left(\frac{1}{R} - \frac{1}{k}\right) z^*$.

Proof. Let z'_{path} be the optimal coverage of k' subsets from $\mathcal{F}_{\text{path}} = \{A_\ell, \ell \in L\}$. We observe that V' can be viewed as the output of the greedy algorithm for the maximum coverage problem if restricted to select k' sets from the family of subsets $\mathcal{F}_{\text{path}} = \{A_\ell, \ell \in L\}$. By Theorem 1,

$$z_1 \geq \left(1 - \frac{1}{e}\right) z'_{\text{path}}. \tag{1}$$

Let z'_{node} be the optimal coverage of k' subsets from $\mathcal{F}_{\text{node}} = \{S_v, v \in V\}$. Since for each subset $S \in \mathcal{F}_{\text{node}}$, there is some subset $A \in \mathcal{F}_{\text{path}}$ such that $S \subseteq A$. So the optimal coverage of k' subsets from $\mathcal{F}_{\text{path}}$ is better than that from $\mathcal{F}_{\text{node}}$, that is,

$$z'_{\text{path}} \geq z'_{\text{node}}. \tag{2}$$

Let z_{node} be the optimal coverage of k subsets from $\mathcal{F}_{\text{node}}$. Then $z'_{\text{node}} \geq \frac{k'}{k} z_{\text{node}}$. We know $k' \geq (k - R)/R$ from Lemma 3, hence,

$$z'_{\text{node}} \geq \frac{\frac{k-R}{R}}{k} z_{\text{node}} = \left(\frac{1}{R} - \frac{1}{k}\right) z_{\text{node}}. \tag{3}$$

Since $z_{\text{node}} \geq z^*$, with inequalities (1), (2) and (3), we derive that

$$\begin{aligned} z_1 &\geq \left(1 - \frac{1}{e}\right) z'_{\text{path}} \geq \left(1 - \frac{1}{e}\right) z'_{\text{node}} \\ &\geq \left(1 - \frac{1}{e}\right) \left(\frac{1}{R} - \frac{1}{k}\right) z_{\text{node}} \\ &\geq \left(1 - \frac{1}{e}\right) \left(\frac{1}{R} - \frac{1}{k}\right) z^*. \quad \square \end{aligned}$$

Complexity: As discussed in the previous section, finding the n BFS tress in the preprocessing step can be done in $O(nm)$ time. The dynamic programming procedure consists of $O(n)$ set union operations so it is $O(nN)$. There are $k' \leq k$ iterations in greedy path algorithm, each requiring $O(n)$ set union operations and $O(n)$ time to pick the subset that has the largest incremental coverage. As noted earlier, we assume the input allows us to do one union operation and measure the size in $O(N)$ time. The total complexity of the Greedy Path Algorithm is then $O\,(n(m + kN))$.

4.2. An improved approximation algorithm – the bounded radius algorithm

We present here the *Bounded Radius Algorithm*, Algorithm 3, which is a $(1 - 1/e)\,(1/r_{\text{OPT}} - 1/k)$-approximation Algorithm.

Let $G(v, r)$ be the induced subgraph of the node set $V(v, r) = \{w \in V : d(v, w) \leq r\}$, which are all the nodes within distance r from node v. As above, let V^* be the optimal solution (node set) for the connected maximum coverage problem. Then, it follows from the definition of the radius of the optimal solution r_{OPT}, that there exists a node u in V^* such that $V^* \subseteq V(u, r_{\text{OPT}})$. One such node u is the center of the graph induced by V^*.

If we were to apply the Greedy Path Algorithm to $G(u, r_{\text{OPT}})$, as we know that the optimal subgraph have at most k nodes so $k > r_{\text{OPT}}$. Then according to Theorem (2) the output would be a $(1 - 1/e)\,(1/r_{\text{OPT}} - 1/k)$-approximate solution. However, there is no easy way to guess the correct node u and the value r_{OPT}. Our improved approximation hence relies on testing all possible combinations of nodes u and radius values r_{OPT} by applying the Greedy Path Algorithm on subgraph $G(v, r)$ for all $v \in V$ and $r = 1, \ldots, R$.

Here, for each selection of $u \in V$ and $r \in \{1, \ldots, R\}$, the algorithm maintains a set of nodes $V'(u, r)$ that have been selected, as well as the union U of the subsets corresponding to the nodes in V', $U = \cup_{v \in V'(u,r)} S_v$.

Algorithm 3 Bounded radius algorithm.

```
Input $G = (V, E)$, $R = R(G)$, $S_v$ for each $v \in V$, integer $k$.
for $u \in V$ do
    $P_u \leftarrow \{u\}$, $A_u \leftarrow S_u$.
    Run BFS with root $u$, $V_r \leftarrow$ set of nodes of level $r$ and the leaves of level less than $r$ in the BFS tree, for $r = 1, \ldots, k$. ($V_r$ are the set of leaves of the subtree induced by deleting all nodes of level more than $r$ from the BFS tree.)
    for $r = 1, \ldots, k$ do
        for $v \in V_r$ do
            if $v$ is at level $r$ then
                $P_v \leftarrow \{v\} \cup P_{p(v)}$, $A_v \leftarrow S_v \cup A_{p(v)}$.
            end if
        end for
        $V'(u, r) \leftarrow \{u\}$, $U \leftarrow S_u$.
        while $k - |V'| \geq R$ do
            Select $\ell$ from $V_r$ that maximizes $|U \cup A_\ell|$, $V'(u, r) \leftarrow V'(u, r) \cup P_\ell$, $U \leftarrow U \cup A_\ell$, $V_r \leftarrow V_r \setminus \{\ell\}$.
        end while
        $z(u, r) \leftarrow |U|$.
    end for
end for
$v^*, r^* \leftarrow \arg\max_{u,r} z(u, r)$.
Output $V'(v^*, r^*)$.
```

Complexity: For each potential center u, it takes $O(m)$ time to run BFS. Then for each potential r, it takes $O(kNn)$ time to apply the loop of the Greedy Path Algorithm. Therefore the total complexity is $O(mn + k^2Nn^2) = O(k^2Nn^2)$.

5. The approximation bounds and a comparison with the results of Vandin et al.

We reviewed here three approximation bounds for the connected maximum coverage problem. These are:

1. The singleton greedy algorithm is a $1/k$-approximation algorithm (Lemma 2). This algorithm provides a solution consisting of a single node, which by default is a connected subgraph.
2. The algorithm of Vandin et al. (2011) [7] provides an approximation bound of $1/(cr_{\text{OPT}})$. (Recall that $c = \frac{2e-1}{e-1} \simeq 2.58$.)
3. The Bounded Radius Algorithm which is a $(1 - 1/e)\,(1/r_{\text{OPT}} - 1/k)$-approximation algorithm.

When $k \leq cr_{\text{OPT}}$ the singleton greedy approximation is best, meaning that $1/k$ is the largest of the three factors.

In Lemma 1 we showed that $k \geq 2r_{\text{OPT}}$, hence for k in the range $[2r_{\text{OPT}}, cr_{\text{OPT}}]$ the singleton approximate is the best. This occurs when the optimal graph has a relatively large diameter and is close to a path. The other extreme is when the optimal subgraph is a star which has a radius of 1 regardless of the value of k.

We show next, that for $k \geq cr_{\text{OPT}}$ the approximation factor provided by our algorithm is the best (greater than) the approximation of [7] as well as the singleton greedy approximation.

Lemma 4. *For $k \geq cr_{OPT}$, $\left(1-\frac{1}{e}\right)(1/r_{OPT} - 1/k) \geq 1/(cr_{OPT})$.*

Proof. The proof relies on simple arithmetic.

$$\begin{aligned} & \left(1-\tfrac{1}{e}\right)(1/r_{\text{OPT}} - 1/k) \geq \left(1-\tfrac{1}{e}\right)\tfrac{1}{r_{\text{OPT}}} - \left(1-\tfrac{1}{e}\right)\tfrac{1}{cr_{\text{OPT}}} \\ = \; & \left(1-\tfrac{1}{e}\right)\left(1-\tfrac{1}{c}\right)\tfrac{1}{r_{\text{OPT}}} = \tfrac{1}{cr_{\text{OPT}}}. \end{aligned}$$

The last equality follows since

$$\begin{aligned} \left(1-\frac{1}{e}\right)\left(1-\frac{1}{c}\right) &= \frac{e-1}{e}\,\frac{(2e-1)-(e-1)}{2e-1} \\ &= \frac{e-1}{2e-1} = \frac{1}{c}. \quad \square \end{aligned}$$

Hence, our results deliver the best approximation factor known for the connected maximum coverage problem, $\max\{(1-1/e)(1/r_{\text{OPT}} - 1/k), 1/k\}$.

6. Generalization to maximizing monotone submodular set function

Consider a generalization of connected maximum coverage problem, which we call connected monotone submodular maximization: for any monotone submodular set function $f : 2^V \to \mathbb{R}$ where $f(\emptyset) = 0$, find a subset of nodes V' of cardinality k, that induces a connected subgraph in G, such that $f(V')$ is maximized. The definition of submodular function and monotone submodular function is given below.

Definition 3. Let Ω be a finite set and $f : 2^\Omega \to \mathbb{R}$. Then f is *submodular* if for all $S, T \subseteq N$,

$$f(S) + f(T) \geq f(S \cup T) + f(S \cap T).$$

Furthermore, if for any $T \subseteq S$, $f(T) \leq f(S)$, then f is *monotone*.

Examples of monotone submodular functions include: linear functions, budget-additive functions, coverage functions, and matroid rank functions. It is known that greedy algorithm is an $(1-1/e)$-approximate algorithm for monotone submodular maximization.

Theorem 3 *([9]). The greedy algorithm for monotone submodular maximization always produce an $1-(1-1/k)^k$-approximate solution for maximizing a monotone submodular function $f(S)$ where $f(\emptyset) = 0$, under the constraint $|S| \leq k$.*

In the two approximation algorithms presented above, the Greedy Path Algorithm as well as the Bounded Radius Algorithm, we incorporate the greedy algorithm of maximum coverage problem to select k' paths from a "center" node. Now for the monotone submodular objective, we generalize these two algorithms by incorporating the greedy algorithm of maximizing monotone submodular set function to select paths. That is, instead of adding the path which has the most increment on coverage in each step, we add the path which has the most increment on the objective f.

Now, we are going to show that the Greedy Path Algorithm and the Bounded Radius Algorithm deliver the same approximation factors respectively as applied to connected maximum coverage problem when applied to connected monotone submodular maximization. Since Lemma 3 does not depend on the objective function, providing a proof similar to Theorem 2 is sufficient.

Let the optimal solution to connected monotone submodular maximization problem be subset V^*. Let $z^* = f(V^*)$ be the value of the optimal solution and let $z_1 = f(V')$ be the value of the Greedy Path Algorithm's solution.

Theorem 4. $z_1 \geq \left(1-\frac{1}{e}\right)\left(\frac{1}{R} - \frac{1}{k}\right) z^*$.

Proof. Let k' to denote the number of iterations of the while loop in the Greedy Path Algorithm. In the preprocessing step, we get a subset of nodes L, and P_ℓ the set of nodes on a path between v^* and ℓ for each $\ell \in L$. Define a new function $g : 2^L \to \mathbb{R}$ where $g(S) = f(\cup_{\ell \in S} P_\ell)$. One can easily verify that g is also monotone submodular with $g(\emptyset) = 0$.

Let $z'_g = \max_{S \subseteq L : |S| = k'} g(S)$. We observe that V' can be viewed as the output of the greedy algorithm for maximizing monotone submodular function g with the restriction of selecting k' subsets from the family of subsets L. By Theorem 3,

$$z_1 \geq \left(1 - \frac{1}{e}\right) z'_g. \tag{4}$$

Let $z'_f = \max_{S \subseteq V : |S| = k'} f(S)$. Since each node is on some path P_ℓ where $\ell \in L$, by monotonicity, we have

$$z'_g \geq z'_f. \tag{5}$$

Let $z_f = \max_{S \subseteq V : |S| = k} f(S)$. Then by submodularity, $z'_f \geq \frac{k'}{k} z_f$. We know $k' \geq (k-R)/R$ from Lemma 3, hence,

$$z'_f \geq \frac{\frac{k-R}{R}}{k} z_f = \left(\frac{1}{R} - \frac{1}{k}\right) z_f. \tag{6}$$

Since $z_f \geq z^*$, with inequalities (4), (5) and (6), we derive that

$$\begin{aligned} z_1 &\geq \left(1-\frac{1}{e}\right) z'_g \geq \left(1-\frac{1}{e}\right) z'_f \\ &\geq \left(1-\frac{1}{e}\right)\left(\frac{1}{R} - \frac{1}{k}\right) z_f \geq \left(1-\frac{1}{e}\right)\left(\frac{1}{R} - \frac{1}{k}\right) z^*. \quad \square \end{aligned}$$

Theorem 4 shows that the Greedy Path Algorithm is an $\left(1-\frac{1}{e}\right)\left(\frac{1}{R}-\frac{1}{k}\right)$-approximation algorithm for connected monotone submodular maximization problem. With the same reasoning in Section 4.2, we show the Bounded Radius Algorithm is an $\left(1-\frac{1}{e}\right)\left(\frac{1}{r_{\mathrm{OPT}}}-\frac{1}{k}\right)$-approximation algorithm for connected monotone submodular maximization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] W.C. Hahn, R.A. Weinberg, Modelling the molecular circuitry of cancer, Nat. Rev. Cancer 2 (5) (2002) 331, https://doi.org/10.1038/nrc795.

[2] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, New York, NY, USA, 1978.

[3] D.S. Hochbaum, A. Pathria, Analysis of the greedy approach in problems of maximum k-coverage, Nav. Res. Logist. (NRL) 45 (6) (1998) 615–627, https://doi.org/10.1002/(SICI)1520-6750(199809)45:6<615::AID-NAV5>3.0.CO;2-5.

[4] U. Feige, A threshold of $\ln n$ for approximating set cover, J. ACM 45 (4) (1998) 634–652, https://doi.org/10.1145/285055.285059.

[5] D.S. Hochbaum, A. Pathria, Node-optimal connected k-subgraphs, UC Berkeley, Unpublished manuscript.

[6] S. Seufert, S. Bedathur, J. Mestre, G. Weikum, Bonsai: growing interesting small trees, in: 2010 IEEE International Conference on Data Mining, IEEE, 2010, pp. 1013–1018.

[7] F. Vandin, E. Upfal, B.J. Raphael, Algorithms for detecting significantly mutated pathways in cancer, J. Comput. Biol. 18 (3) (2011) 507–522, https://doi.org/10.1089/cmb.2010.0265.

[8] A. Bomersbach, M. Chiarandini, F. Vandin, An efficient branch and cut algorithm to find frequently mutated subnetworks in cancer, in: International Workshop on Algorithms in Bioinformatics, Springer, 2016, pp. 27–39.

[9] G.L. Nemhauser, L.A. Wolsey, M.L. Fisher, An analysis of approximations for maximizing submodular set functions, Math. Program. 14 (1) (1978) 265–294, https://doi.org/10.1007/BF01588971.