Hindawi
Journal of Applied Mathematics
Volume 2019, Article ID 9520324, 16 pages
https://doi.org/10.1155/2019/9520324

Research Article

On the Location of a Constrained k–Tree Facility in a Tree Network with Unreliable Edges

Abdallah W. Aboutahoun [1,2] **and Eman Fares** [3]

[1] *Applied Mathematics and Information Science Department, Zewail City of Science and Technology, 6th of October City, Giza, Egypt*
[2] *Department of Mathematics, Faculty of Science, Alexandria University, Alexandria, Egypt*
[3] *Department of Basic Sciences, Faculty of Engineering, Pharos University, Alexandria, Egypt*

Correspondence should be addressed to Abdallah W. Aboutahoun; atahoun@zewailcity.edu.eg

Received 20 March 2019; Accepted 24 July 2019; Published 21 August 2019

Academic Editor: Wei-Chang Yeh

Given a tree network T with n vertices where each edge has an independent operational probability, we are interested in finding the optimal location of a reliable service provider facility in a shape of subtree with exactly k leaves and with a diameter of at most l which maximizes the expected number of nodes that are reachable from the selected subtree by operational paths. Demand requests for service originate at perfectly reliable nodes. So, the major concern of this paper is to find a location of a reliable tree-shaped facility on the network in order to provide a maximum access to network services by ensuring the highest level of network connectivity between the demand nodes and the facility. An efficient algorithm for finding a reliable (k, l) – tree core of T is developed. The time complexity of the proposed algorithm is $O(lkn)$. Examples are provided to illustrate the performance of the proposed algorithm.

1. Introduction

The classical location theory is concerned with the location of a service facility on a network. This facility can be a single point or a specified number of points, located at either a vertex or along an edge of the network. The location of the facility depends on the distances between the demand vertices (customers) and their respective server. The location of special types of subgraphs such as paths, trees, or forests is considered as an extensive facility location problem. These facilities formed of connected structures have many applications in the fields such as transportation, communication, and computer sciences. The three criteria for optimality which are extensively studied in the literature are the following:

(1) The median criterion or the minisum criterion in which the sum of the distances from all the vertices of the network to the facility is minimized.

(2) The center criterion or the minimax criterion in which the maximum distance from the demand vertex to the facility is minimized.

(3) The centdian criterion in which the convex combination of the weighted average distance and the maximum weighted distance from the facility to the demand points is minimized.

In recent years, there has been a growing interest in studying the location of facilities formed of connected structures which are also called extensive facilities on a tree network. The location of a path-shaped and tree-shaped facilities under the above three criteria has been studied by a number of authors. In this paper, we are studying the location of a reliable tree-shaped facility on a tree network with unreliable edges. A core of a tree is defined to be a path that is optimal with respect to the property of minimizing the sum of the distances from each vertex in the tree to the path (criterion 1). The generalization of the core of the tree is a k–tree core which is a subtree containing exactly k leaves that minimizes the sum of the distances from each vertex in the tree network to the selected k–tree core facility. The (k, l)–tree core is a subtree with diameter at most l having k leaves which minimizes the sum of the distances or the weighted distances from all vertices in the tree network to this subtree facility.

Finding the core of a tree network has been considered by a several authors. Becker et al. [1] presented two efficient algorithms for finding the l−core of a tree. For unweighted trees they developed an $O(n^2)$ time algorithm, while for weighted trees they provided a procedure with time complexity of $O(n\log^2 n)$. Peng and Lo [2] presented a recursive $O(n\log n)$ time algorithm for finding a core of specified length, that is, a path with length equal to a specified value l, in unweighted trees. Morgan and Slater [3] presented a linear time algorithm for finding the core of a tree network. Minieka and Patel [4] explored some properties of a core of a specified length of a tree network. A 2−core of a tree T is a set of 2 mutually disjoint paths in T that minimizes the sum of the distances of all vertices in T from any of the two paths. Wang [5] developed an $O(n)$ time algorithm for the 2−core problem, where n is the number of vertices in T. Becker et al. [6] considered the problem of finding an optimal location of a path-shaped facility on a tree network using combinations of the center and the median criteria. They studied two problems: (1) finding a path which minimizes the sum of the distances such that the maximum distance from the vertices of the tree to the path is bounded by a fixed constant and such that the length of the path is not greater than a fixed value; (2) finding a path which minimizes the maximum distance with the sum of the distances being not greater than a fixed value and with bounded length. They gave $O(n\log^2 n)$ divide-and-conquer algorithms.

Wang et al. [7] developed two algorithms for finding a (k, l)−tree core of a tree T. The first algorithm has $O(n^2)$ time complexity for the weighted tree (each edge has an arbitrary length). The second algorithm has $O(lkn)$ time complexity for the unweighted tree (the lengths of all edges are 1). Also, Peng et al. [8] presented two algorithms to find a k−tree core of a tree with n vertices. The time complexities of these two algorithms are $O(kn)$ and $O(n\log n)$. Becker et al. [9] provided two algorithms; the first one for unweighted trees has time complexity of $O(n^2)$, whereas the second one for weighted trees has time complexity of $O(n^2\log n)$. Shioura and Uno [10] proposed a linear time algorithm for finding a k−tree core of a tree network. Minieka [11] described methods for finding an optimal location for a path-shaped or tree-shaped facility of a specified size in a tree network. Four optimization criteria were examined: minimizing distancesum, minimizing eccentricity, maximizing distancesum, and maximizing eccentricity. Solution algorithms were presented. Kim et al. [12] studied the problem of locating a subtree facility on a tree network. They considered the cost of establishing the subtree facility in addition to the transportation cost associated with the travel of customers to the subtree facility. The objective function is to select a tree-shaped facility that will minimize the sum of the setup cost and the total transportation cost. Tamir er al. [13] developed an algorithm of $O(n\log n)$ time complexity for finding the optimal location of a tree-shaped facility of a specified size in a tree network with n nodes, using the centdian criterion: a convex combination of the weighted average distance and the maximum weighted distance from the facility to the demand points (nodes of the tree). Tamir et al. [14] studied the location of a subtree facility on a tree network, both discrete and continuous under the condition that existing facilities are already located. They used the center and the median criteria.

The problems of finding a single point, a path, or a subtree on tree network with distance edges have been studied extensively in literature, a single facility location problem on a network with unreliable edges has been also considered by a number of authors. Melachrinoudis and Helander [15] addressed the reliable 1-median (relisum) location problem of a single facility on an undirected tree with unreliable edges. The objective of the problem is to maximize the expected number of nodes reachable by operational paths. They developed two polynomial algorithms; an $O(n^3)$ algorithm which is a modification of the Floyd-Warshall algorithm for finding all pairs of the shortest paths in a graph and an $O(n^2)$ algorithm which set up a depth-first node traversal and the decomposition nature of an operational path. Xue [16] presented a linear time algorithm for the same problem which was displayed by Melachrinoudis and Helander [15]. Helander and Melachrinoudis [17] introduced path reliability measures for the expected number of accidents over a given planning horizon. Reliability refers to the probability of a hazmat transport vehicle completing a journey from an origin to a destination. They presented two different locations of the modeling frameworks: the reliable 1-median and a general framework for considering multiple routes. Santiváñez et al. [18] considered the problem of location a single facility on an undirected network with unreliable edges under the center criterion. The objective function is formally stated as either minimizing the maximum expected number of unsuccessful responses to demand requests over all nodes, called the reli-minmax problem, or maximizing the minimum expected number of successful responses to demand requests over all nodes, called the reli-maxmin problem. The problem is termed the reliable 1-center problem and finds applications in telecommunication and computer networks. As for subproblems of the most general reliable 1-center problem, Eiselt et al. [19] presented the problem of locating p facilities on a network so that the total expected demand disconnected from the facilities is minimized. It is supposed that every edge has a probability of failure and that failures can never occur on two edges simultaneously. Eiselt et al. [20] showed how to optimally locate p facilities on a network with unreliable node or edge in order to minimize the expected demand disconnected from the facilities. Ding and Xue [21] studied the problem of locating a node which maximizes the expected number of nodes that are reachable from it. Such a node is called a most reliable source (MRS) of the network. They presented a linear time algorithm for computing a most reliable source on an unreliable tree network where all links are immune to failures and each node has an independent transmitting probability and an independent receiving probability. Puerto et al. [22] considered the problem of locating two path-shaped facilities minimizing the expected service cost in the long run, assuming that paths may become unavailable and their failure probabilities are known in advance. They provided a polynomial time algorithm that solves the 2 unreliable path location problem on tree networks in $O(n^2)$ time, where n is the number of

vertices. Ding et al. [23] studied the problem of finding the 2-most reliable source (2-MRS) in an unreliable tree network. The 2-MRS problem aims to find a node pair from which the expected number of reachable nodes or the minimum reachability is maximized (i.e., Sum-Max 2-MRS and Min-Max 2-MRS).

The reliable (k,l)−tree core problem is stated as finding the optimal location of a tree-shaped facility (service provider) in an unreliable tree network with a diameter of at most l and having k leaves which maximizes the expected number of nodes that are reachable from it. This problem considers only edge failures while the nodes and the service provider facility are considered perfectly reliable. So, the model we present in this paper is stochastic in the sense that edges fail randomly which may cause a disconnection between demand nodes and service provider facility. The major concern of this work is to find a location of a service provider facility in order to provide a maximum access to network services, but in the context of connectivity, i.e., the probability that an operational path exists between two points of the network, and therefore the connectivity of the network is measured using path reliability.

For any two vertices $v,u \in V$, the vertex v is called reachable from u (or u is called reachable from v) if there exists an operational path between them, i.e., every edge in this path is in an operational state. The objective is to identify a subtree S that has exactly k leaves, diameter of at most l and the cumulative reliability from V to S, $\sum_{v\in V} R(v,S)$ is maximized. This subtree S is the reliable (k,l)−tree core facility of T. The reliable (k,l)−tree core problem is motivated from a distributed database application in computer science; see Wang et al. [7] and Peng et al. [8].

The remainder of this paper is organized into five sections. In Section 2, we present notation, definitions, and some preliminary results. Section 3 is devoted to the formulation and a full description for the proposed problem. In Section 4, we present an efficient algorithm for finding a reliable (k,l)−tree core of T. In Section 5, a numerical example is provided to illustrate the efficiency of the proposed algorithm. Finally, in Section 6, we give a concluding remarks and future research.

2. Definitions and Preliminaries

Let $T = (V,E)$ be a tree network, where $V = \{1,2,3,\ldots,n\}$ is the node set and $E = \{e = (i,j) : i,j \in V\}$ is the edge set. Let $P(v,u)$ be a unique path in tree$T$ from node v to node u and let $d(v,u)$ be its length which is defined in this study to be the number of edges in the path. Let $V(P(v,u))$ and $E(P(v,u))$ denote the set of nodes and edges on the path $P(v,u)$, respectively. Let $p(i,j)$ denote the probability that edge $(i,j) \in E$ in operational state such that $0 < p(i,j) \leq 1$ and $q(i,j) = 1-p(i,j)$ is the probability that the edge $(i,j) \in E$ is in a failed state. All vertices are assumed to be perfectly reliable and any vertex is called a leaf vertex if the number of edges incident to it is equal to 1. The number of edges that can be in a failed state has no limitations, we assume that failures occur independently. Let T^r be the tree obtained by making T rooted at a vertex $r \in V$. The diameter of T^r is $\max_{v,u\in V} d(v,u)$ while the height of T^r is $\max_{u\in V} d(r,u)$, where r is called the root of tree T. Let v be an a vertex in T^r, and we denote by T^r_v a subtree of T^r rooted at a vertex $v \in V$. Let $N(v)$ denote the set of children of v and $f(v)$ denote the parent of v. If the vertex v is adjacent to r, T^r_v is called a subtree of r.

For each edge $e = (i,j)$ we call $tail(e) = i$ the endpoint of e closest to the root r, and we call $head(e) = j$ the endpoint of e farthest to the root. Each edge e divides T^r into two disjoint subtrees denoted by $T^r_{head(e)}$ and $T^r_{tail(e)}$. The subtree $T^r_{head(e)} = T^r_j$ is a subtree of T^r rooted at a vertex j as defined above and the subtree $T^r_{tail(e)} = T^r \backslash T^r_{head(e)}$ is a subtree of T^r rooted also at r but induced by the set of vertices $V\backslash V(T^r_{head(e)})$, where $V(T^r_{head(e)})$ is the set of vertices of the subtree $T^r_{head(e)}$. If v and u are two vertices in an unreliable communication network, we use $R(v,u)$ to denote the reliability that a message can be transmitted correctly from v to u. Let S be a subtree of T and $V(S)$ be the set of nodes of S. The expected number of nodes reachable by operational paths from the nodes of subtree S of T is called the reachability or connectivity of the subtree S.

A reliable (k,l)−tree core is a subtree S with k leaves and with a diameter of at most l which maximizes the expected number of nodes that are reachable by operational paths from the nodes of S. In other words, we seek for a subtree S of T with diameter of at most l, having k leaves, which maximizes the expected number of successful responses to demand requests over all nodes. A demand request is originating at a node v which requires that some entity moves from the service provider located at a subtree S to node v along some path in the tree network.

Definition 1. Under the assumption that edges fail independently, the reliability of a path $P(v,u)$ is defined as the product of the reliability of arcs in the path $E(P(v,u))$, i.e.,

$$R(P(v,u)) = R(v,u) = \prod_{(i,j)\in E(P(v,u))} p(i,j) \tag{1}$$

and the probability that at least one edge in the path $P(v,u)$ has failed deeming the path unusable is

$$1 - R(v,u) = 1 - \prod_{(i,j)\in E(P(v,u))} p(i,j) \tag{2}$$

Note that $R(P(v,u)) = p(v,u)$ when $P(v,u)$ consists of one edge $(v,u) \in E$ and $R(P(v,v)) = 1$ when $v = u$. Also, note that $p(v,u) = p(u,v)$ and $R(v,u) = R(u,v)$ as long as all vertices have the same weight. The path $P(v,u)$ is operational if and only if all edges of $P(v,u)$ are operational simultaneously. Since for any $v,u \in V$ there exists a unique path $P(v,u)$ in T^r connecting v and u, then it always holds that $R(P(v,u)) = R(v,u)$ in T^r.

Definition 2. For a rooted tree T^r, the total connectivity of any vertex $v \in V$ or the expected number of nodes reachable from $v \in V$ is defined by

$$RS(v) = \sum_{u\in V} R(v,u) = 1 + \sum_{u\in V\backslash\{v\}} R(v,u) \tag{3}$$

where $p(v,v) = 1$.

For a vertex $v \in V$ and a path P of T, the reliability from v to P is

$$R(v,P) = \max_{u \in V(P)} R(u,v) \tag{4}$$

Also, for a vertex $v \in V$ and a subtree S of T, the reliability from v to S is

$$R(v,S) = \max_{u \in V(S)} R(u,v) \tag{5}$$

Definition 3. For a rooted tree T^r, the total connectivity of any path P is the sum of the reliabilities from all the vertices of T not in P to the path P

$$RS(P) = \sum_{v \notin P} R(v,P) \tag{6}$$

where $R(v,P)$ is the maximum reliability from $v \in V \backslash V(P)$ to a vertex in P.

Definition 4. For a rooted tree T^r, the total connectivity of any subtree S is the sum of the reliabilities from all the vertices of T not in S to the subtree S

$$RS(S) = \sum_{v \notin S} R(v,S) \tag{7}$$

where $R(v,S)$ is the maximum reliability from $v \in V \backslash V(S)$ to a vertex in S.

Following the same notation developed by Ding and Xue [21], let T^r be a rooted tree at any vertex. For any vertex v in T^r, let T^v_α be the subtree of T^r rooted at vertex v. Let $V_\alpha(v)$ be the set of vertices of T^v_α and let T^v_β be the subtree induced by the set of vertices $V_\beta(v) = V - V_\alpha(v)$ such that $V_\alpha(v) \cup V_\beta(v) = V$ and $V_\alpha(v) \cap V_\beta(v) = \phi$. If v is a leaf, then $V_\alpha(v) = \{v\}$ and $V_\beta(v) = V - \{v\}$. If $v = r$, then $V_\beta(v) = \phi$ and $V_\alpha(v) = V$. The quantities corresponding to T^v_α are often referred as below quantities, while the ones corresponding to T^v_β are referred as upper quantities. The two subtrees T^v_α and T^v_β are connected by the edge $(f(v), v)$, see Figure 1.

Based on the previous decomposition of T^r at node v, Theorem 5 presents two formulas for computing the connectivity of a node v in the subtrees T^v_α and T^v_β. We will use $RS_\alpha(v)$ to denote the expected number of nodes in $V_\alpha(v)$ which can be reached from node v and use $RS_\beta(v)$ to denote the expected number of nodes in$V_\beta(v)$ which can be reached from node v.

Theorem 5 (see [21, 23]). *For any node $v \in V$, the reliability sums $RS_\alpha(v)$ and $RS_\beta(v)$ are given by*

$$RS_\alpha(v) = 1 + \sum_{w \in N(v)} RS_\alpha(w)\, p(v,w) \tag{8}$$

$$RS_\beta(v) = \left(RS_\beta(f(v)) + 1 + \sum_{\substack{z \in N(f(v)) \\ z \neq v}} RS_\alpha(z)\, p(z, f(v)) \right) p(f(v), v) \tag{9}$$

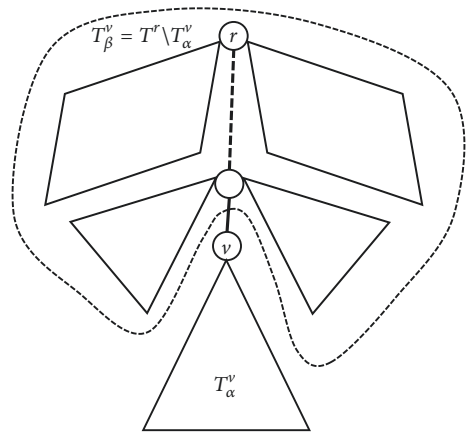

Figure 1: The decomposition of the rooted tree T^r at node v into two subtrees T^v_α and T^v_β.

where $N(v)$ denotes the set of nodes which consists of all children of v. If v is a leaf then $RS_\alpha(v) = 1$ and if v is a root of tree T^r then $RS_\beta(v) = 0$.

Proof. See Ding and Xue [21]. □

Based on the formulas in Theorem 5 for computing the expected number of nodes in the two subtrees T^r_α and T^r_β that can be reached from v, the next lemma gives the combination of $RS_\alpha(v)$ and $RS_\beta(v)$ to find expected number of nodes in $T^r = T^r_\alpha \cup T^r_\beta \cup (f(v), v)$ that can be reached from v which is also called the total connectivity of v.

Lemma 6. *For a rooted tree T^r, the total connectivity of any vertex $v \in V$ or the expected number of nodes reachable from $v \in V$ is given by*

$$RS(v) = RS_\alpha(v) + RS_\beta(v) \tag{10}$$

Proof. See Ding and Xue [21]. □

Theorem 5 and Lemma 6 give the connectivity or the reachability of a node in an unreliable tree network. The next six lemmas give formulas for computing the cumulative reliability of an extensive facility in the form of single edge path-shaped facility, path-shaped facility, and a tree-shaped facility. These formulas generalize the results of Theorem 5 and Lemma 6 to present the connectivity of a path or a subtree instead of a node, also, using the concept of reliability saving which is very useful for computing the cumulative reliability from T to a given path or subtree facility.

Lemma 7. *For each edge $e = (u,v)$, where $u = tail(e)$ and $v = head(e)$, the expected number of nodes reachable by operational paths from the service facility located as a single edge path-shaped facility which is called the relisum of this edge e is given by*

$$RS(e = (u,v)) = (1 - p(u,v))\, RS_\alpha(v) + RS(u) \tag{11}$$

Proof.

$$\begin{aligned} RS\left(e=(u,v)\right) &= RS_{head(e)}\left(v\right) + RS_{tail(e)}\left(u\right) \\ &= RS_{\alpha}\left(v\right) + RS\left(u\right) - p\left(u,v\right) RS_{\alpha}\left(v\right) \\ &= \left(1 - p\left(u,v\right)\right) RS_{\alpha}\left(v\right) + RS\left(u\right) \end{aligned} \tag{12}$$

where,

$$RS_{head(e)}\left(v\right) = RS_{\alpha}\left(v\right) \tag{13}$$

and

$$R_{tail(e)}\left(u\right) = RS\left(u\right) - p\left(u,v\right) RS_{\alpha}\left(v\right) \tag{14}$$

□

The relisum of the path-shaped facility with only one edge $RS(e)$ can be evaluated also by

$$\begin{aligned} RS\left(e=(u,v)\right) &= RS\left(u : p\left(u,v\right)=1\right) \\ &= RS\left(v : p\left(u,v\right)=1\right) \\ &= 2 + \sum_{v_k \in N(v)} RS_{\alpha}\left(v_k\right) p\left(v,v_k\right) \\ &\quad + \sum_{\substack{w_t \in N(u) \\ w_t \neq v}} RS_{\alpha}\left(w_t\right) p\left(\left(u,w_t\right)\right) \\ &\quad + RS_{\beta}\left(u\right) \end{aligned} \tag{15}$$

We now introduce the definition of the reliability saving of a path which is the most important measure for guiding the optimal selection of a path with maximum connectivity or reachability starting from any vertex of the tree.

Definition 8. Given a path $P(v,u)$ and a path $P(u,w)$ with edges disjoint from $P(v,u)$, the reliability saving of $P(u,w)$ with respect to $P(v,u)$ is the increment of the reliable sum obtained by adding $P(u,w)$ to $P(v,u)$, that is,

$$\delta\left(P\left(v,u\right), P\left(u,w\right)\right) = RS\left(P\left(v,w\right)\right) - RS\left(P\left(v,u\right)\right) \tag{16}$$

If the first path consists of only one vertex v, we simply write $\delta(v, P(v,w)) = RS(P(v,w)) - RS(v)$. The reliability saving of a path $P(v,u)$ in the subtree T_{α}^{v} is defined by $\delta_{\alpha}(v, P(v,u)) = RS_{\alpha}(P(v,u)) - RS_{\alpha}(v)$.

The reliability saving of a given path $P(v,u)$ in the subtree T_{α}^{v} is defined recursively as follows:

$$\delta_{\alpha}\left(v, P\left(v,u\right)\right) = \begin{cases} \delta_{\alpha}\left(f\left(u\right), e=(v,u)\right) = \left(1 - p\left(v,u\right)\right) RS_{\alpha}\left(u\right), & v = f\left(u\right) \\ \delta_{\alpha}\left(v, P\left(v, f\left(u\right)\right)\right) + \delta_{\alpha}\left(e=\left(f\left(u\right),u\right)\right), & v \neq f\left(u\right) \end{cases} \tag{17}$$

where $\delta_{\alpha}(f(u), e = (f(u), u))$ gives the reliability saving of a single edge path facility $P(v,u) = (v,u)$ in the subtree T_{α}^{v}.

The following lemma gives two important formulas for calculating the reliability sum and the reliability saving of a given path in the subtree T_{α}^{v}.

Lemma 9. *Let $P(v, u)$ be a path in T_{α}^{v} in the form $\{v = v_1, v_2, \ldots, v_h = u\}$, then we have*

$$\begin{aligned} RS_{\alpha}\left(P\left(v,u\right)\right) &= \sum_{i=1}^{h-1} \left[RS_{\alpha}\left(v_i\right) - RS_{\alpha}\left(v_{i+1}\right) p\left(v_i, v_{i+1}\right)\right] \\ &\quad + RS_{\alpha}\left(v_h\right) \end{aligned} \tag{18}$$

and

$$\delta_{\alpha}\left(v, P\left(v,u\right)\right) = \sum_{i=1}^{h-1} \left(1 - p\left(v_i, v_{i+1}\right)\right) RS_{\alpha}\left(v_{i+1}\right) \tag{19}$$

Proof. The reliability sum $RS_{\alpha}(P(v,u))$ of a given path $P(v,u)$ in the subtree T_{α}^{v} is the sum of the reliability sums $RS_{\alpha}(v_i)$ of the vertices in the path $P(v,u)$ excluding the edges of this path.

$$\begin{aligned} &RS_{\alpha}\left(P\left(v,u\right)\right) \\ &= 1 + \sum_{v_{k_1} \in N(v_1) \setminus \{v_2\}} RS_{\alpha}\left(v_{k_1}\right) p\left(v_1, v_{k_1}\right) + 1 \\ &\quad + \sum_{v_{k_2} \in N(v_2) \setminus \{v_3\}} RS_{\alpha}\left(v_{k_2}\right) p\left(v_2, v_{k_2}\right) + \cdots + 1 \\ &\quad + \sum_{v_{k_{h-1}} \in N(v_{h-1}) \setminus \{v_h\}} RS_{\alpha}\left(v_{k_{h-1}}\right) p\left(v_{h-1}, v_{k_{h-1}}\right) + 1 \\ &\quad + \sum_{v_{k_h} \in N(v_h)} RS_{\alpha}\left(v_{k_h}\right) p\left(v_h, v_{k_h}\right) \end{aligned} \tag{20}$$

which can be written in the form

$$\begin{aligned} &RS_{\alpha}\left(P\left(v,u\right)\right) \\ &= 1 + \sum_{v_{k_1} \in N(v_1)} RS_{\alpha}\left(v_{k_1}\right) p\left(v_1, v_{k_1}\right) \\ &\quad - RS_{\alpha}\left(v_2\right) p\left(v_1, v_2\right) + 1 \\ &\quad + \sum_{v_{k_2} \in N(v_2)} RS_{\alpha}\left(v_{k_2}\right) p\left(v_2, v_{k_2}\right) \\ &\quad - RS_{\alpha}\left(v_3\right) p\left(v_2, v_3\right) + \cdots + 1 \\ &\quad + \sum_{v_{k_{h-1}} \in N(v_{h-1})} RS_{\alpha}\left(v_{k_{h-1}}\right) p\left(v_{h-1}, v_{k_{h-1}}\right) \\ &\quad - RS_{\alpha}\left(v_h\right) p\left(v_{h-1}, v_h\right) + 1 \\ &\quad + \sum_{v_{k_h} \in N(v_h)} RS_{\alpha}\left(v_{k_h}\right) p\left(v_h, v_{k_h}\right) \end{aligned} \tag{21}$$

Hence,

$$\begin{aligned} RS_{\alpha}\left(P\left(v,u\right)\right) &= RS_{\alpha}\left(v_1\right) - RS_{\alpha}\left(v_2\right) p\left(v_1, v_2\right) \\ &\quad + RS_{\alpha}\left(v_2\right) - RS_{\alpha}\left(v_3\right) p\left(v_2, v_3\right) + \cdots \\ &\quad + RS_{\alpha}\left(v_{h-1}\right) - RS_{\alpha}\left(v_h\right) p\left(v_{h-1}, v_h\right) \\ &\quad + RS_{\alpha}\left(v_h\right) \end{aligned}$$

$$= \sum_{i=1}^{h-1} \left[RS_\alpha (v_i) - RS_\alpha (v_{i+1})\, p(v_i, v_{i+1}) \right] + RS_\alpha (v_h) \tag{22}$$

This proves formula (18)

Since $\delta_\alpha(v, P(v,u)) = RS_\alpha(P(v,u)) - RS_\alpha(v)$ and by using formula (18), we get

$$\delta_\alpha (v, P(v,u)) = \sum_{i=1}^{h-1} (1 - p(v_i, v_{i+1}))\, RS_\alpha (v_{i+1}) \tag{23}$$

□

The following lemma shows the using of the reliability saving concept in computing the reliability sum from T^r to a given path.

Lemma 10. *For all $u \in V\backslash\{v\}$, the reliability sum of the path $P(v,u)$ is*

$$RS_\alpha (P(v,u)) = RS_\alpha (v) + \delta_\alpha (v, P(v,u)) \tag{24}$$

Proof. The proof follows immediately from the previous lemma. □

The reliability saving can be calculated recursively by using a bottom up procedure as stated by the following lemma.

Lemma 11. *Let $P(v,u)$ be a path in a tree T_α^v. The reliability saving of the path $P(v,u)$ can be defined recursively by*

$$\delta_\alpha (v, P(v,u)) = \delta_\alpha (v, P(v, f(u))) + \delta_\alpha (f(u), e = (f(u), u)) \tag{25}$$

where $\delta_\alpha(f(u), e = (f(u), u)) = (1 - p(f(u), u)) RS_\alpha(u)$.

Proof.

$$\begin{aligned} &\delta_\alpha (v, P(v,u)) - \delta_\alpha (v, P(v, f(u))) \\ &= \left(RS_\alpha (P(v,u)) - RS_\alpha (v)\right) \\ &\quad - \left(RS_\alpha (P(v, f(u))) - RS_\alpha (v)\right) \\ &= RS_\alpha (P(v,u)) - RS_\alpha (P(v, f(u))) \\ &= \left(1 - p(f(u), u)\right) RS_\alpha (u) \end{aligned} \tag{26}$$

□

For a subtree S and a vertex $v \notin S$, let $T_S(v)$ be the subtree in T induced by the vertex set $V_S(v) = \{w : P(w,v) \cap S = \phi \text{ and } d(w,S) \geq d(v,S)\}$. If we regard T as a tree rooted at S, $T_S(v)$ can be seen as a subtree rooted at v. If the subtree S becomes larger, the reliability sum $RS(S)$ increases strictly. The following lemma shows that the reliability sum of the subtree S increases by adding a path to it.

Lemma 12. *Let $P(v,u)$ be any path in a tree T_α^v and let S be any subtree of T^r which intersects with the path $P(v,u)$ at node v then the relisum of the new subtree $S^* = S \cup P(v,u)$ is*

$$RS(S^*) = RS(S \cup P(v,u)) = RS(S) + \delta_\alpha (v, P(v,u)) \tag{27}$$

Proof.

$$\begin{aligned} &RS(S \cup P(v,u)) - RS(S) \\ &= \sum_{w \in V} \{R(w, S \cup P(v,u)) - R(w,S)\} \\ &= \sum_{w \in T_S(v)} \{R(w, P(v,u)) - R(w,v)\} + \sum_{w \notin T_S(v)} \{R(w,S) - R(w,S)\} \\ &= \sum_{w \in T_S(v)} \{R(w, P(v,u)) - R(w,v)\} + \sum_{w \notin T_S(v)} \{R(w,v) - R(w,v)\} \\ &= \sum_{w \in V} \{R(w, P(v,u)) - R(w,v)\} \\ &= RS(P(v,u)) - RS(v) = \delta_\alpha (v, P(v,u)) \end{aligned} \tag{28}$$

□

We consider increasing $RS(S)$ by adding a path P to a subtree S. The following equation holds for the increase of the reliability by addition of a path to a subtree.

Lemma 13. *Let $P(v,u)$ be a path from vertex v to vertex u in a tree T_α^v and S is any subtree of T^r such that $P(v,u) \cap S = \{v\}$, then*

$$RS(S \cup P(v,u)) - RS(S) = RS_\alpha (P(v,u)) - RS_\alpha (v) \tag{29}$$

Proof. The proof is straight forward by using Lemma 12 and Lemma 10. □

Definition 14. In [9], given a path P having an even (odd) length l, the midpoint of P is a vertex (edge) whose removal divides P into two paths with length $l/2((l-1)/2)$.

3. The Reliable (k,l)–Tree Core Problem

For any subtree S of T with exactly k leaves and with diameter $\leq l$, let c be the unique center of S and suppose that c is a vertex. Let $h = l/2$ be the height of the subtree S in T and orient the tree T into T^c, then S is a subtree rooted at c with height $\leq h$ and having k leaves.

Let c be the midpoint of a path $P(v,u)$ between any two vertices v and u of length $\leq l$. The center of unweighted tree T occurs at the midpoint of the longest path in the tree, this result is presented by Handler [24]. The center of a tree is either of one vertex or an edge and it is unique. The following two sets Γ_1 and Γ_2 define the centers of a subtrees with diameters $\leq l$.

$$\Gamma_1 = \left\{ \begin{matrix} c : c \text{ is the center vertex of the path with even length } P(v,u), \\ \text{where } v,\ u \in V \text{ and } d(v,u) \le l \end{matrix} \right\} \tag{30}$$

$$\Gamma_2 = \left\{ \begin{matrix} e : e \text{ is the center edge of the path with odd length } P(v,u), \\ \text{where } v,\ u \in V \text{ and } d(v,u) \le l \end{matrix} \right\} \tag{31}$$

Therefore, there exists a point in Γ_1 or an edge in Γ_2 that is the center of a (k,l)−tree core of T. All the subtrees of the tree T with diameter $\le l$ will be classified by the midpoints of their diameters. Then, we search for a k leaves subtree in the class of the subtrees of T having a given c as a midpoint. In other words, each $c \in \Gamma_1$ or $c \in \Gamma_2$ is the center of a set of subtrees of T with diameter $\le l$, and the reliable (k,l)−tree core is one of these subtrees having k leaves and maximum reliability sum.

For each path $P(v,u)$ with length less than or equal l and midpoint $c \in \Gamma_1$, the tree T is going to be oriented into a rooted tree at c and denote it by T^c. If $c \in \Gamma_2$ which is an edge, then we contract it into a vertex and we continue to call the resulting vertex c and the resulting rooted tree T^c.

Define

$$\begin{aligned} Can\_Set\_Lvs(c) = \{ & u : u \in V,\ d(u,c) \\ & \le h \text{ and } d(w,c) > h \text{ for every child of } w \text{ of } u \} \end{aligned} \tag{32}$$

to be the candidate set of leaves for the subtree rooted at c, where $h = l/2$ if c is the center of an even length path and $h = (l-1)/2$ if c is the center edge of an odd length path. Note that it is not necessary that the vertices of $Can\_Set\_Lvs(c)$ are leaves in T^c, and it is the set of vertices at a distance h or less from the root c of the tree such that their children vertices are more than h distance away from c. For example, consider the rooted tree T^c in Figure 3. If $h = 3$, we have $Can\_Set\_Lvs(c) = \{v_4, v_6, v_7, v_9, v_{10}, v_{12}, v_{13}, v_{14}, v_{15}, v_{16}, v_{17}, v_{18}, v_{19}\}$. All leaves of the rooted reliable (k,h)−tree core of T^c are vertices in $Can\_Set\_Lvs(c)$. Every vertex $v \in Can\_Set\_Lvs(c)$ is called a candidate with respect to h. The selected k candidates from $Can\_Set\_Lvs(c)$ can not be contained in the same subtree of c. Otherwise, c is not contained in the induced subtree by these vertices.

The stochastic optimization problem considered in this paper is the following: Given a tree network with unreliable edges and given two parameters k and l, the problem is to identify a subtree $S \subseteq T$ with exactly k leaves, diameter $\le l$ and the reliability sum $\sum_{v \in V} R(v,S)$ is maximized. If l exceeds the diameter of T, then the reliable (k,l)−tree core problem is just the k−tree core problem.

Problem 15. Given a rooted tree T^c, find a reliable (k,l)−tree core rooted at c of T^c with maximum reliability sum.

Let T_v^c be a subtree rooted at a vertex v in the tree T^c. If $v \in V$ is a vertex with $d(c,v) \le h$, the candidate $u \in Can\_Set\_Lvs(c) \cap V(T_v^c)$ dominates v if $\delta(v, P(v,u)) > \delta(v, P(v,z))$ for all $z \in Can\_Set\_Lvs(c) \cap V(T_v^c)$ or $\delta(v, P(v,u)) = \delta(v, P(v,z))$ and $u > z$. This means that the path $P(v,u)$ has a maximum reliability saving among all paths from v to vertices in $Can\_Set\_Lvs(c) \cap V(T_v^c)$. If the vertex u dominates the vertex v, then every vertex on the path $P(v,u)$ is also dominated by u. In constructing the reliable subtree (k,h)− tree core we define $\Delta(u)$ to be the farthest ancestor vertex of u that is dominated by it. Given $u \in Can\_Set\_Lvs(c)$, the reliability saving of paths in T^c is defined by

$$\Omega(u) = \begin{cases} \delta(c, P(c,u)) & \text{if } \Delta(u) = c \\ \delta\left(f(\Delta(u)), P\left(f(\Delta(u)), u\right)\right) & \text{otherwise} \end{cases} \tag{33}$$

Sorting the vertices in $Can\_Set\_Lvs(c)$ according to their reliability saving $\Omega(u)$, $u \in Can\_Set\_Lvs(c)$. The rank of a vertex $u \in Can\_Set\_Lvs(c)$ is denoted by $rank(u)$ and it is the number of vertices $z \in Can\_Set\_Lvs(c)$ such that either $\Omega(z) > \Omega(u)$ or $\Omega(z) = \Omega(u)$ and $z \ge u$. We denote the vertex with rank i in $Can\_Set\_Lvs(c)$ by a_i, $1 \le i \le |Can\_Set\_Lvs(c)|$.

The following Lemma states a formula for calculating the reliability sum from the vertices of T^c to the subtree induced by the set of vertices $\{c, a_1, a_2, \ldots, a_i\}$.

Lemma 16.

$$RS\left(\langle \{c, a_1, a_2, \ldots, a_i\} \rangle\right) = RS(c) + \sum_{j=1}^{i} \Omega\left(u_j\right), \quad \text{for all } i = 1, 2, \ldots, |Can\_Set\_Lvs(c)| \tag{34}$$

Proof. Let $S_i = \langle \{c, a_1, a_2, \ldots, a_i\} \rangle$, $i = 1, 2, \ldots, |Can\_Set\_Lvs(c)|$. By definition, a_1 dominates the root c ($\Delta(a_1) = c$). Thus, $\Omega(a_1) = \delta_\alpha(c, P(c,a_1))$ and S_1 is the path connecting the root c with the vertex a_1, $RS(S_1) = RS(c) + \delta_\alpha(c, P(c,a_1)) = RS(c) + \Omega(a_1)$. For any $i > 1$, S_i differs from S_{i-1} only in one path $P(f(\Delta(a_i)), a_i)$. By Lemma 12, $RS(S_i) = RS(S_{i-1}) + \delta_\alpha(f(\Delta(a_i)), P(f(\Delta(a_i)), a_i)) = RS(S_{i-1}) + \Omega(a_i)$ for $1 < i \le |Can\_Set\_Lvs(c)|$. Recursively, it is easy to note that $RS(S_i) = RS(c) + \sum_{j=1}^{i} \Omega(u_j)$, $i = 1, 2, \ldots, |Can\_Set\_Lvs(c)|$. □

If the nonincreasing set of vertices $\Lambda_c = \{a_1, a_2, \ldots, a_k\}$ is contained in $V(T_j^c) \cap Can\_Set\_Lvs(c)$, $j \in N(c)$, then the subtree induced by $\langle \Lambda_c \rangle$ will be a subtree of c rooted at vertex j and does not include the root c. So, let us define

$$MAX\_\delta(c, P(c,u)) = \max_{z \in Can\_Set\_Lvs(c)} \{\delta(c, P(c,z))\} = \delta\left(c, P\left(c, a_1\right)\right)$$

$$SEC\_MAX\_\delta(c, P(c,w))$$

$$= \max_{z \in Can\_Set\_Lvs(c), P(c,a_1) \cap P(c,z) = c} \{\delta(c, P(c,z))\}$$
$$= \delta(c, P(c,a_t)) \tag{35}$$

$MAX\_\delta(c, P(c,u))$ represents the maximum reliability saving of a path connecting the root c to a vertex in $Can\_Set\_Lvs(c)$ which is denoted by a_1. $SEC\_MAX\_\delta(c, P(c,w))$ represents the second maximum reliability saving of a path connecting the root c to a vertex in $Can\_Set\_Lvs(c)$ which is denoted by a_t, where $P(c,a_1) \cap P(c,a_t) = \{c\}$. If $a_t \in \Lambda_c$, then the induced subtree $\langle \Lambda_c \rangle$ represents the required reliable $(k,h)-$ tree core of T^c. If $a_t \notin \Lambda_c$, then Λ_c is included in one subtree T_j^c, $j \in N(c)$ so $\langle \Lambda_c \rangle$ is the subtree rooted at j and the root c is not included in the required subtree. To avoid this situation, a_k in Λ_c will be replaced by a_t, where t is the smallest integer such that

$$\Lambda_c = \begin{cases} \{a_1, a_2, \ldots, a_{k-1}, a_k\} & \text{If } t \le k \\ \{a_1, a_2, \ldots, a_{k-1}, a_t\} & \text{otherwise} \end{cases} \tag{36}$$

Note that Λ_c maximizes $\sum_{u \in S} \Omega(u)$ over all subsets $S \subseteq Can\_Set\_Lvs(c)$ such that $|S| = k$ and the k candidates in S are not all contained in the same subtree of c. The following lemma gives the reliability sum of the optimal subtree $\langle \Lambda_c \rangle$ among all induced subtrees of subsets $S \subseteq Can\_Set\_Lvs(c)$.

Lemma 17.

$$RS(\langle \Lambda_c \rangle) = RS(c) + \sum_{u \in \Lambda_c} \Omega(u) \tag{37}$$

Proof. $\langle \{c, a_1, a_2, \ldots, a_k\} \rangle = \langle \{a_1, a_2, \ldots, a_k\} \rangle$ if $t \le k$, which means the subtree induced by the subset of vertices $\{a_1, a_2, \ldots, a_k\} \subseteq Can\_Set\_Lvs(c)$ contains the first and the second reliability saving paths passing through the root c, thus the lemma holds by using the previous lemma. Let $t > k$, then $\Lambda_c = \langle \{c, a_1, a_2, \ldots, a_{k-1}, a_t\} \rangle$ differs from $\langle \{c, a_1, a_2, \ldots, a_{k-1}\} \rangle$ in only the path $P(c,a_t)$. Then,

$$\begin{aligned} RS(\langle \Lambda_c \rangle) &= RS(\langle \{c, a_1, a_2, \ldots, a_{k-1}\} \rangle) \\ &\quad + \delta_\alpha(c, P(c,a_t)) \\ &= RS(c) + \sum_{j=1}^{k-1} \Omega(a_j) + \Omega(a_t) \\ &= RS(c) + \sum_{u \in \Lambda_c} \Omega(u) \end{aligned} \tag{38}$$

where $\{a_1, a_2, \ldots, a_{k-1}\} \subseteq V(T_j^c) \cap Can\_Set\_Lvs(c)$, $j \in N(c)$ and the path $P(c,a_t)$ has the highest reliability saving rank among all vertices that are not contained in T_j^c, $f(\Delta(a_t)) = c$. Hence, $\delta_\alpha(c, P(c,a_t)) = \Omega(a_t)$. □

Also, $RS(\langle \Lambda_c \rangle)$ can be calculated by a different formula which is stated in the next lemma.

Lemma 18. *The reliability sum of the subtree $S = \langle \Lambda_c \rangle$ can be computed using the following formula:*

$$RS(S) = RS(P(c,a_1)) + \sum_{u \in \Lambda_c \setminus \{a_1\}} \Omega(u) \tag{39}$$

The terms of candidates and values can be generalized to the case that both the root of the tree and the height of the subtrees to be considered are not fixed. The root of the tree can be any vertex i in T and the subtrees considered are those of height $m \le h$. For a given pair of i and m, we define

$$Can\_Set\_Lvs(i,m) = \left\{ \begin{array}{l} v : v \in V(T^i),\ d(i,v) \le m, \text{ and} \\ d(i,u) > m \text{ for every child } u \text{ of } v \end{array} \right\} \tag{40}$$

The value of the candidate $u \in Can\_Set\_Lvs(i,m)$ is denoted by $\Omega(i,m,u)$ which is defined similarly as $\Omega(u)$, the maximum reliability saving of a path from a vertex $u \in Can\_Set\_Lvs(i,m)$ to a vertex $w \in V(T_i^c)$ such that $\Delta(u) = w$.

For each $j \in N(i)$, we define the candidate set

$$\begin{aligned} Can\_Set\_Lvs(i,m,j) &= Can\_Set\_Lvs(i,m) \cap V(T_j^i) \\ &= Can\_Set\_Lvs(j, m-1) \end{aligned} \tag{41}$$

The following lemma shows that the reliability saving value $\Omega(i,m,u)$ of any path can be computed recursively by bottom up procedure.

Lemma 19. *Let i be a vertex in T^c and $j \in N(i)$. We have*

(a)

$$Can\_Set\_Lvs(i,m,j) = \bigcup_{v \in N(j)} Can\_Set\_Lvs(j, m-1, v) \tag{42}$$

(b)

$$\Omega(i,m,u) = \begin{cases} \Omega(j, m-1, u) + (1 - p(i,j)) RS_\alpha(j) & \text{if } u \text{ dominates } j \\ \Omega(j, m-1, u) & \text{otherwise} \end{cases} \tag{43}$$

Proof. From (41), the candidate set of leaves $Can\_Set\_Lvs(i, m, j)$ consists of the set of vertices $u \in V(T_j^i)$ such that $d(i,u) \le m$ and $d(i,w) > m$ for every child w of u as shown in Figure 2. Since the length of the edge (i, j) is 1, a vertex $v \in V(T_j^i)$ and $v \in Can\_Set\_Lvs(i,m)$ if and only if $v \in Can\_Set\_Lvs(j, m-1)$. This shows that $Can\_Set\_Lvs(i,m,j)$ consists of the union of $Can\_Set\_Lvs(j, m-1, v)$ over all $v \in N(j)$. Part (b) follows immediately from definition (33) of $\Omega(u)$ and Lemma 9. □

For any rooted subtree T_i^c of T^c, the set of paths with maximum reliability saving from vertices in $Can\_Set\_Lvs(i,m,j)$

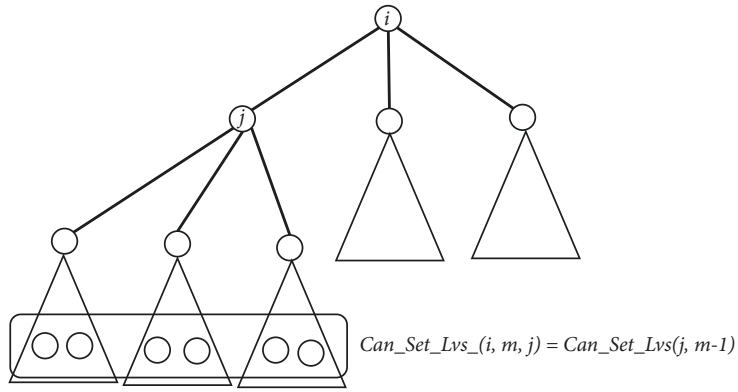

FIGURE 2: The candidate set of leaves $Can\_Set\_Lvs(i, m, j)$ in the subtree T_j^i.

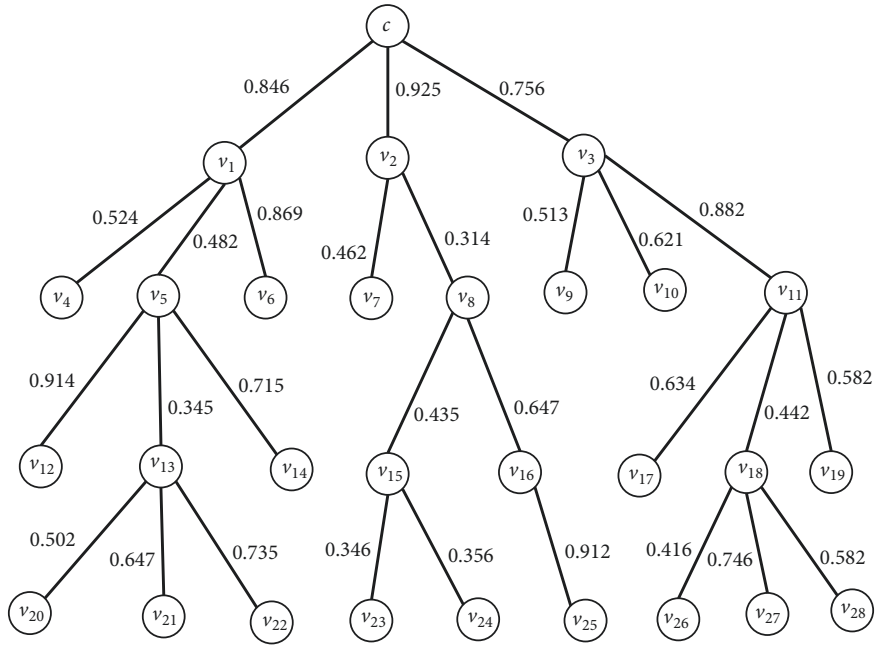

FIGURE 3: A rooted tree network T^c with 29 vertices and 28 edges.

to the vertices of T_j^i can be constructed by using the definition of $\Omega(i, m, u)$ in part (b) of the previous lemma.

The vertices in $Can\_Set\_Lvs(i, m)$ must be sorted in a descending order according to their reliability saving values $\Omega(i, m, u)$ and for any subset $S \subseteq Can\_Set\_Lvs(i, m)$, $rank(u, S)$ denotes the rank of the vertex $u \in S$. A reliable (k, h)–tree core of a rooted tree T^c must contain as a subtree a reliable $(k-1, h)$–tree core.

4. Algorithm

In this section, an algorithm for finding the location of a reliable (k, l)–tree core facility in a tree network with unreliable edges is presented. The algorithm first uses the dynamic programming technique to compute a table Φ in bottom–up fashion. Then, it uses the table to compute Λ_c for each $c \in \Gamma_1 \cup \Gamma_2$ efficiently. Finally, it finds a reliable (k, l)–tree core of T by determining the induced subtree $\langle \Lambda_c \rangle$ of the top ranked k vertices of $Can\_Set\_Lvs(c)$ that maximizes $RS(\langle \Lambda_c \rangle)$. Given a tree network T with unreliable edges, the proposed algorithm identifies a subtree S which satisfies the following:

(1) Orient the input tree T into a rooted tree T^c for every $c \in \Gamma_1 \cup \Gamma_2$.

(2) c is the root of S and has degree at least 2.

(3) S has exactly k leaves.

(4) S has height at most h.

(5) The reliability sum of the subtree S, $RS(S)$ is maximized.

Remark 20. Each reliable (k, l)–tree core $\langle \Lambda_c \rangle$ must contain the two paths $P(c, a_1)$ and $P(c, a_t)$ with the maximum and

the second maximum reliability saving such that $P(c, a_1) \cap P(c, a_t) = \{c\}$.

The algorithm uses dynamic programming technique to compute a table entry $\Phi[i, m, j]$ which stores a set of up to k candidates and their values $\Omega(i, m, j)$.

For the subtree T_u^v and $u \in N(v)$, a recursive formula of $\Phi[v, m, u]$ is given as follows:

$$\begin{aligned}\Phi[v, 1, u] &= \{(u, \delta(v, P(v, u)))\} \\ &= \{(v, (1 - p(v, u)) RS_\alpha(u))\} \quad \text{if } m = 1\end{aligned} \tag{44}$$

and

$$\begin{aligned}\Phi[v, m, u] &= (u, (1 - p(v, u))) \\ &\qquad \text{if } m > 1 \text{ and } u \text{ is a leaf}\end{aligned} \tag{45}$$

When $m > 1$ and u is not a leaf, $\Phi[v, m, u]$ can be recursively computed as follows:

$$\begin{aligned}&\Phi[v, m, u] \\ &\quad = \{\alpha^*, \beta^* + (1 - p(v, u)) RS_\alpha(u)\} \\ &\qquad \cup \{(\alpha, \beta) : (\alpha, \beta) \in Z, 2 \le rank(\alpha, Z) \le k\}\end{aligned} \tag{46}$$

where

$$\begin{aligned}Z &= \bigcup_{w \in N(u), w \ne v} \Phi[u, m - 1, w], \text{ and } (\alpha^*, \beta^*) \\ &\in Z \text{ with } rank(\alpha^*, Z) = 1\end{aligned} \tag{47}$$

Z contains the set of leaves $\alpha \in Can\_Set\_Lvs(v, m, u)$ in the subtree T_u^v of v with their values β; among them the best k leaves can be chosen according to their values of reliability saving.

For example, consider the rooted tree T^c depicted in Figure 3. Let $h = 3$, we show how $\Phi[c, 3, v_1]$ can be computed. At the beginning the set of candidate leaves is $Can\_Set\_Lvs(c, 3, v_1) = \{v_4, v_6, v_{12}, v_{13}, v_{14}\}$. The computation of $\Phi[v_1, 2, v_5]$ requires the computation of $Z = \bigcup_{w \in N(v_5), w \ne v_5} \Phi[v_5, 1, w] = \{(v_{12}, 0.086), (v_{13}^*, 1.88902), (v_{14}, 0.285)\}$. Hence,

$$\begin{aligned}&\Phi[v_1, 2, v_5] \\ &\quad = (v_{13}, 1.88902 + (1 - p(v_1, v_5)) RS_\alpha(v_5)) \\ &\qquad \cup \{(v_{12}, 0.086), (v_{14}, 0.285)\} \\ &\quad = (v_{13}, 1.88902 + (1 - 0.482)(3.62398)) \\ &\qquad \cup \{(v_{12}, 0.086), (v_{14}, 0.285)\} \\ &\quad = \{(v_{13}, 3.76624), (v_{12}, 0.086), (v_{14}, 0.285)\}\end{aligned} \tag{48}$$

Note that $RS_\alpha(v_5)$ in $\Phi[v_1, 2, v_5]$ can be calculated as follows:

$$\begin{aligned}RS_\alpha(v_5) &= 1 + \sum_{w \in N(v)} RS_\alpha(w)\, p(v, w) = 1 \\ &\quad + [RS_\alpha(v_{12})(0.914) + RS_\alpha(v_{13})(0.345) \\ &\quad + RS_\alpha(v_{14})(0.715)] = 1 + (1)(0.914) + [1 \\ &\quad + RS_\alpha(v_{20})(0.502) + RS_\alpha(v_{21})(0.647) \\ &\quad + RS_\alpha(v_{22})(0.735)](0.345) + (1)(0.715) = 1 \\ &\quad + 0.914 + [1 + 0.502 + 0.647 + 0.735](0.345) \\ &\quad + 0.715 = 3.62398\end{aligned} \tag{49}$$

Recursively, to compute $\Phi[c, 3, v_1]$, we need the computation of Z,

$$\begin{aligned}Z &= \bigcup_{w \in N(v_1), w \ne v_1} \Phi[v_1, 2, w] = \Phi[v_1, 2, v_4] \cup \Phi[v_1, 2, \\ &\quad v_5] \cup \Phi[v_1, 2, v_6] = \{(v_4, 0.476), (v_{13}^*, 3.76624), \\ &\quad (v_{12}, 0.086), (v_{14}, 0.285), (v_6, 0.131)\}\end{aligned} \tag{50}$$

Hence,

$$\begin{aligned}\Phi[c, 3, v_1] &= (v_{13}, 3.76624 + (1 - p(c, v_1)) RS_\alpha(v_1)) \\ &\quad \cup \{(v_4, 0.476), (v_{12}, 0.086), (v_{14}, 0.285), (v_6, 0.131)\} \\ &\quad = \{(v_{13}, 4.403763), (v_4, 0.476), (v_{12}, 0.086), \\ &\quad (v_{14}, 0.285), (v_6, 0.131)\}\end{aligned} \tag{51}$$

So, the computation of the table entries $\Phi[c, 3, v_1]$, $\Phi[c, 3, v_2]$ and $\Phi[c, 3, v_3]$ is summarized in Table 1. If $k = 4$ and $h = 3$, a reliable $(4, 3)$−tree core for the considered rooted tree T^c depicted in Figure 3 is shown in Figure 4 with double lines edges. In this case, $\Lambda_c = \{v_{13}, v_{15}, v_{16}, v_{18}\}$, then the induced subtree $\langle \Lambda_c \rangle$ is the optimal reliable $(4, 3)$−tree core in T^c.

If the four leaves in $Can\_Set\_Lvs(c, 3)$ with the highest values are contained in one subtree of c, for example $T_{v_1}^c$. The fourth ranked entry (α_4, β_4) in $\Phi[c, 3, v_1]$ will be replaced with the highest ranked entry in $\Phi[c, 3, v_2] \cup \Phi[c, 3, v_3]$ to construct Λ_c such that the subtree induced by $\langle \Lambda_c \rangle$ will contain c. Figure 5 is a rooted tree at the vertex $v_2 \in \Gamma_1$, so by the same procedure the optimal reliable tree core $\langle \Lambda_{v_2} \rangle$ can be computed (see Algorithm 1).

Let $\Pi = \{v_1, v_2, \ldots, v_k\} \subseteq Can\_Set\_Lvs(c, m)$ be the set of top ranked vertices which are candidate to be the k leaves of the tree-shaped facility rooted at c. If the vertices of the set Π are not contained in one table $\Phi[c, m, j]$, then the induced subtree $\langle \Pi \rangle$ is a rooted reliable (k, h)−tree core of T^c. If the set Π is contained in one subtree of c, for example, the subtree attached to the vertex u through the edge (c, u)

For each $S \subseteq Can\_Set\_Lvs(c)$, note that, for any $S \subseteq Can\_Set\_Lvs(c)$, let $\Theta(S)$ be $\langle S \cup \{c\} \rangle$ which is a rooted subtree with S leaves and root c. Let $\mathsf{F} = \{S : S \subseteq Can\_Set\_Lvs(c), |S| = k\}$.

Table 1: Calculating the table entries $\Phi[c, 3, v_1]$, $\Phi[c, 3, v_2]$, and $\Phi[c, 3, v_3]$.

(v, u)	$\Phi[v_5, 1, v]$	$\Phi[v_1, 2, v]$	$\Phi[c, 3, v_1]$
(v_5, v_{12})	$(v_{12}, 0.086)$		
(v_5, v_{13})	$(v^*_{13}, 1.88902)$		
(v_5, v_{14})	$(v_{14}, 0.285)$		
(v_1, v_4)		$(v_4, 0.476)$	
(v_1, v_5)		$(v_{12}, 0.086)$ $(v^*_{13}, 3.76624)$ $(v_{14}, 0.285)$	
(v_1, v_6)		$(v_6, 0.131)$	
(c, v_1)			$(v_4, 0.476)$ $(v_6, 0.131)$ $(v_{12}, 0.086)$ $(v_{13}, 4.403763)$ $(v_{14}, 0.285)$
(v, u)	$\Phi[v_8, 1, v]$	$\Phi[v_2, 2, v]$	$\Phi[c, 3, v_2]$
(v_8, v_{15})	$(v^*_{15}, 0.96163)$		
(v_8, v_{16})	$(v_{16}, 0.67494)$		
(v_2, v_7)		$(v_7, 0.538)$	
(v_2, v_8)		$(v_{16}, 0.67494)$ $(v^*_{15}, 3.0041)$	
(c, v_2)			$(v_7, 0.538)$ $(v_{16}, 0.67494)$ $(v^*_{15}, 3.1839)$
(v, u)	$\Phi[v_{11}, 1, v]$	$\Phi[v_3, 2, v]$	$\Phi[c, 3, v_3]$
(v_{11}, v_{17})	$(v_{17}, 0.366)$		
(v_{11}, v_{18})	$(v^*_{18}, 1.5312)$		
(v_{11}, v_{19})	$(v_{19}, 0.418)$		
(v_3, v_9)		$(v_9, 0.487)$	
(v_3, v_{11})		$(v_{17}, 0.366)$ $(v^*_{18}, 1.9358)$ $(v_{19}, 0.418)$	
(v_3, v_{10})		$(v_{10}, 0.379)$	
(c, v_3)			$(v_9, 0.487)$ $(v_{10}, 0.379)$ $(v_{17}, 0.366)$ $(v^*_{18}, 3.1944)$ $(v_{19}, 0.418)$

We now verify the correctness of the proposed algorithm. The next lemma is the key in proving Theorem 22.

Lemma 21. *If* $\Xi = \{a_1, a_2, \ldots, a_{k-1}, a_k\}$, *then* $RS(\Theta(\Xi)) = \max_{S \in \mathrm{F}} RS(\Theta(S))$.

Proof. Suppose that $Q = \{a_1, a_2, \ldots, a_{k-1}, b\} \subseteq Can\_Set\_Lvs(c)$, where $\Theta(Q)$ has the maximum reliability sum, *i.e.*, $RS(\Theta(Q)) = \max_{S \in \mathrm{F}} RS(\Theta(S))$. Let $s \in V(\Theta(Q))$ such that b and $a_k \in Can\_Set\_Lvs(c)$ are a descendant vertices of s and assume that s is dominated by a_k, *i.e.*, $\Delta(a_k) = s$. Consider $Q' = Q \setminus \{b\} \cup \{a_k\}$, then $\Theta(Q') = \Theta(Q) \setminus P(w, b) \cup P(s, a_k)$, where w is a child vertex of s on the path $P(s, b)$. Hence,

$$RS\left(\Theta\left(Q'\right)\right) = RS\left(\Theta\left(Q\right)\right) - \delta_\alpha\left(s, P\left(s, b\right)\right) + \delta_\alpha\left(s, P\left(s, a_k\right)\right) \tag{52}$$

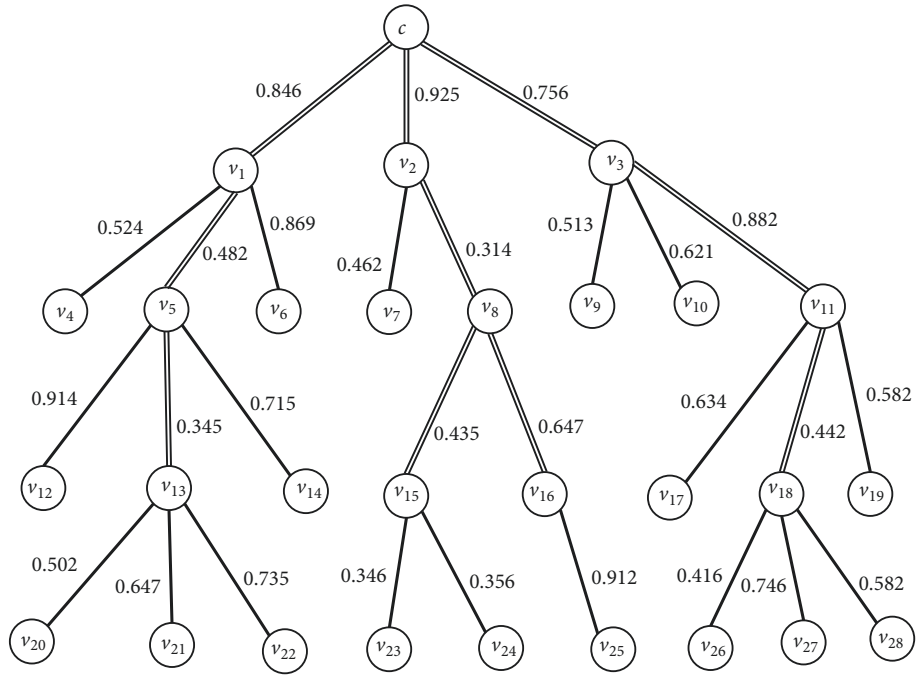

Figure 4: The induced $\langle \Lambda_c \rangle$ optimal reliable $(4, 3)$–tree core of T^c shown in double lines.

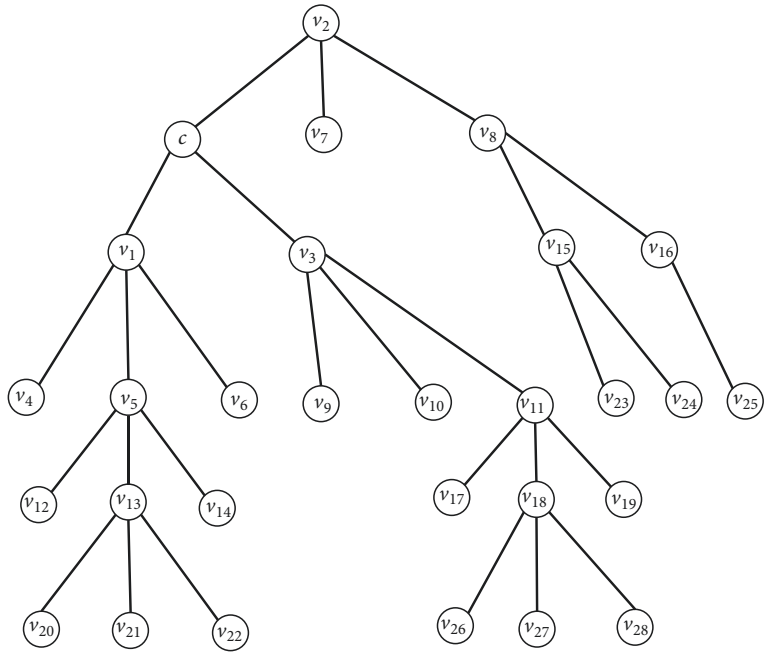

Figure 5: A rooted tree with root v_2 and c is an internal vertex.

Since s is dominated by a_k, then $\delta_\alpha(s, P(s, a_k)) \geq \delta_\alpha(s, P(s, b))$, then

$$RS\left(\Theta\left(Q'\right)\right) \geq RS\left(\Theta\left(Q\right)\right) \tag{53}$$

which contradicts the assumption that $RS(\Theta(Q))$ is the maximum reliability sum.

Suppose that s is not dominated by a_k and b is not a descendant vertex of s but a_k is still a descendant vertex of s. Then, $\delta_\alpha(s, P(s, a_k)) \geq \Omega(a_k) \geq \Omega(b)$. Let $Q' = Q \setminus \{b\} \cup \{a_k\}$. Then, we have

$$\begin{aligned} RS\left(\Theta\left(Q'\right)\right) &= RS\left(\Theta\left(Q\right)\right) \\ &\quad - \delta_\alpha\left(f\left(\Delta\left(b\right)\right), P\left(f\left(\Delta\left(b\right)\right), b\right)\right) \\ &\quad + \delta_\alpha\left(s, P\left(s, a_k\right)\right) \geq RS\left(\Theta\left(Q\right)\right) \end{aligned} \tag{54}$$

which is again a contradiction. □

Inputs: A tree T with an operational probability associated with each edge and two positive integers k, l.
Outputs: A reliable (k,l)−tree core S^* with diameter of at most l and having exactly k leaves which maximizes the reliability sum $RS(S^*)$.
Initialization: $RS^* = 0$ and $S^* = \phi$
begin
For each $c \in \Gamma_1 \cup \Gamma_2$ **do**
Orient the input tree T into a rooted tree T^c.
For each $(u,v) \in E$ **do**
If v is a leaf **then**
$\Phi[u,1,v] =: (v, (1 - p(u,v)))$
Else
$\Phi[u,1,v] =: (v, (1 - p(u,v)) RS_\alpha(T_v^u))$
End If
End For
For $m =: 2 : h$ **do**
For each $j \in N(c)$ **do**
If j is a leaf **then**
$\Phi[c,m,j] =: (c, (1 - p(i,j)))$
Else
$\Phi[c,m,j] = \left\{\left(\alpha^*, \beta^* + (1 - p(i,j)) RS\left(T_j^i\right)\right)\right\} \cup \{(\alpha,\beta) : (\alpha,\beta) \in Z, 2 \le rank(\alpha, Z) \le k\}$
where
$Z = \bigcup_{v \in N(j)} \Phi[j, m-1, v]$ and $(\alpha^*, \beta^*) \in Z, rank(\alpha^*, Z) = 1$
End If
End For
End For
Sort in a decreasing order all the entries in $\bigcup_{v \in N(c)} \Phi[c,m,v]$ according to their reliability savings
If the top k entries are contained in one subtree of c **then**
Replace the entry (α_k, β_k) by the second reliability saving path passing through c
Else
Choose the top k entries of $\bigcup_{v \in N(c)} \Phi[c,m,v]$ and construct the induced subtree $\langle \Lambda_c \rangle$
End If
Calculate the reliability sum of $\langle \Lambda_c \rangle$
$RS^* = \max\{RS^*, RS(\langle \Lambda_c \rangle)\}$
$S^* = \Lambda_c$, where $RS^* = RS(\langle \Lambda_c \rangle)$
End For
End

Algorithm 1

The correctness of the proposed algorithm for finding the reliable tree core is summerized in the following theorem.

Theorem 22. *For each $c \in \Gamma_1 \cup \Gamma_2$, the proposed algorithm correctly finds the subtree $\langle \Lambda_c \rangle$ which is the reliable (k,h)−tree core of the tree T^c.*

Proof. Let

$$\mathsf{F}^* = \left\{S : S \in \mathsf{F},\ S \nsubseteq V\left(T_u^c\right) \text{ for any } u \in N(c)\right\} \tag{55}$$

Note that $\mathsf{F}^* \subseteq \mathsf{F}$, $\Lambda_c \in \mathsf{F}^*$ and for any $S \in \mathsf{F}^*$, $\langle S \rangle$ and $\Theta(S)$ are the same.

Suppose that $a_1 \in V(T_u^c)$, where $u \in N(c)$. Let t be the smallest integer such that $a_t \notin V(T_u^c)$. If $t \le k$, then $\Lambda_c = \{a_1, a_2, \ldots, a_k\}$. Since $\mathsf{F}^* \subseteq \mathsf{F}$ and by Lemma 21, we obtain

$$RS\left(\langle \Lambda_c \rangle\right) = \max_{S \in \mathsf{F}} RS(\Theta(S)) \ge \max_{S \in \mathsf{F}^*} RS(\langle S \rangle) \tag{56}$$

This proves the theorem for the case $t \le k$.

Consider the case in which $t > k$. In this case $\langle \Lambda_c \rangle = \Theta(\{a_1, a_2, \ldots, a_{k-1}\}) \cup P(c, a_t)$. Hence,

$$\begin{aligned} RS\left(\langle \Lambda_c \rangle\right) = {} & RS\left(\Theta\left(\{a_1, a_2, \ldots, a_{k-1}\}\right)\right) \\ & + \delta_\alpha\left(c, P\left(c, a_t\right)\right) \end{aligned} \tag{57}$$

Consider a fixed $S \in \mathsf{F}^*$, and for any $s \in S \backslash \{V(T_u^c)\}$, let $S' = S \backslash \{s\}$. Then $\langle S \rangle = \Theta(S') \cup P(f(\Delta(s)), s)$. Hence,

$$RS(\langle S \rangle) \le RS\left(\Theta\left(S'\right)\right) + \delta_\alpha(c, P(c,s)) \tag{58}$$

From Lemma 21 $\{a_1, a_2, \ldots, a_{k-1}\}$ maximizes $RS(\Theta(A))$ for all $A \subseteq Can\_Set\_Lvs(c)$ of $k-1$ candidates. Hence, $RS(\{\{a_1, a_2, \ldots, a_{k-1}\})) \ge RS(\Theta(S'))$. By definition of t, we have $\delta_\alpha(c, P(c.a_t)) \ge \delta_\alpha(c, P(c,s))$. Therefore,

$$RS\left(\langle \Lambda_c \rangle\right) = RS\left(\Theta\left(\{a_1, a_2, \ldots, a_{k-1}\}\right)\right)$$

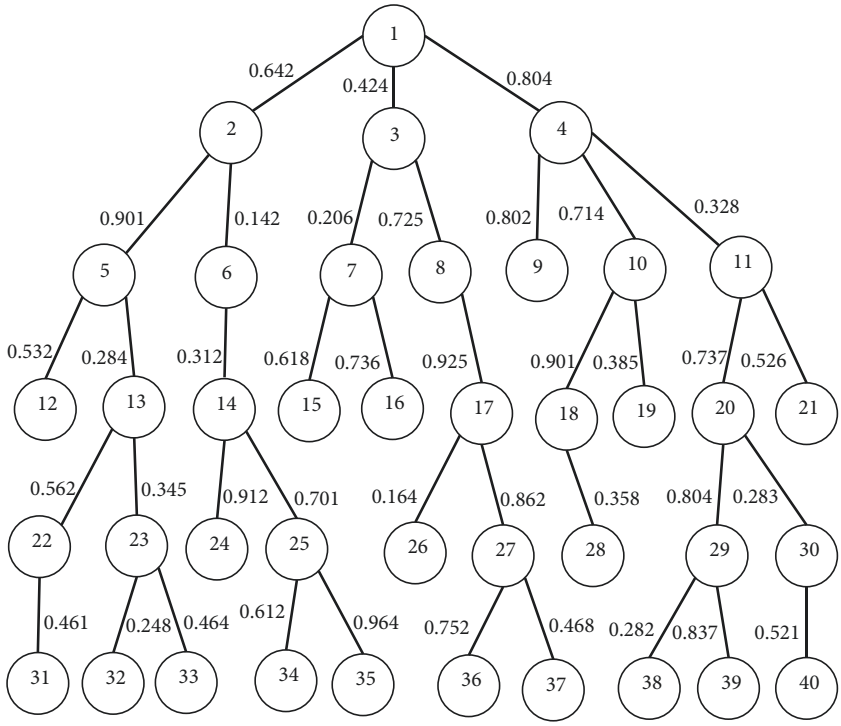

Figure 6: A tree example on which the proposed algorithm is applied where the edge labels are the operational probabilities.

$$+\delta_{\alpha}\left(c, P\left(c, a_{t}\right)\right)$$
$$\geq RS\left(\Theta\left(S'\right)\right)+\delta_{\alpha}\left(c, P\left(c, s\right)\right) \geq RS\left(\langle S\rangle\right) \tag{59}$$

which completes the proof of the theorem. □

Based on the algorithm presented by Wang et al. [7], the time complexity for the subroutine which determines the table Φ is $O(lkn)$. The time required for constructing Λ_c from Φ is $O(k|N(c)|)$. A reliable (k,l)−tree core is the subtree $\langle\Lambda_c\rangle$ that maximizes $RS(\langle\Lambda_c\rangle)$ over all $c \in \Gamma_1 \cup \Gamma_2$. By using the values in Φ, the value of the reliability sum $RS(\langle\Lambda_c\rangle)$ for every $c \in \Gamma_1 \cup \Gamma_2$ can be computed by Lemma 17 or Lemma 18. Therefore, the time complexity of constructing a reliable (k,l)−tree core from the table Φ is $O(lkn)$.

Theorem 23. *The time complexity of the reliable (k,l)−tree core algorithm is $O(lkn)$*

5. Numerical Example

In this section, a numerical example to illustrate the proposed algorithm is considered. A tree network with $n = 40$ vertices and $E = 39$ edges is depicted in Figure 6. The operational probabilities, shown on the arcs, are generated randomly in the interval $(0, 1)$. Γ_1 is the set of centers of all paths of even lengths; in Table 2 a set of paths of length 6 are considered and the tree is rooted at the centers of these paths. Hence, the set Λ_c is constructed using the proposed algorithm and the optimal subtree $\langle\Lambda_c\rangle$ which is a reliable $(4, 3)$−tree core is obtained. The last column of the Table 2 is the reliability sum of the located subtree $\langle\Lambda_c\rangle$. Also, Table 3 considered the case of odd length paths; the paths are centered at the middle edges, where each middle edge is contracted into a single vertex. Similar to the even length paths of Table 2, Table 3 presents a reliable $(4, 3)$−tree core with odd length diameters.

6. Summary and Conclusions

In this paper, we considered the problem of optimally locating a service facility in a tree network with unreliable edges. The service facility considered in this study is an extensive facility or a connected structure which is a special type of subgraph. We considered the location of a tree-shaped (subgraph) facility problem. This tree-shaped facility is a conditional extensive facility, where the constraint is to set a limit to the diameter of the selected facility. So, given two parameters k and l, the objective is to locate a reliable (k,l)−tree core which is a subtree with a diameter $\leq l$ and having k leaves such that the sum of the reliabilities from all the vertices of the tree network to this facility is maximized. The objective of the problem is to maximize the expected number of nodes reachable by operational paths from the located site. An $O(lkn)$ time complexity algorithm is presented based on a modifications to the algorithm developed by Wang et al. [7]. Computational results are provided.

For a future research, the following problems can be considered:

(i) The location of a reliable (k,l)−tree core facility in a tree network with unreliable edges under

Table 2: Some paths of length 6, their centers, the induced subtrees with maximum reliability saving, and the reliability sum of each subtree.

Path $P(u, v)$	Root $c \in \Gamma_1$	Induced subtree $\langle \Lambda_c \rangle$	$R(\langle \Lambda_c \rangle)$
(5, 12), (2, 5), (1, 2), (1, 3), (3, 7), (7, 15)	1	⟨14, 15, 17, 20⟩	24.8535
(6, 14), (2, 6), (1, 2), (1, 3), (3, 8), (8, 17)	1	⟨14, 15, 17, 20⟩	24.8535
(10, 18), (4, 10), (1, 4), (1, 2), (2, 6), (6, 14)	1	⟨14, 15, 17, 20⟩	24.8535
(14, 25), (6, 14), (2, 6), (2, 5), (5, 13), (13, 23)	2	⟨25, 7, 8, 11⟩	24.1866
(20, 30), (11, 20), (4, 11), (4, 10), (10, 18), (18, 28)	4	⟨6, 7, 8, 30⟩	22.7734
(22, 31), (13, 22), (5, 13), (2, 5), (1, 2), (1, 3)	5	⟨32, 2, 3, 12⟩	21.9252
(25, 34), (14, 25), (6, 14), (2, 6), (1, 2), (1, 4)	6	⟨13, 3, 4, 34⟩	20.9972
(37, 27), (17, 27), (8, 17), (3, 8), (1, 3), (1, 2)	8	⟨2, 4, 15, 37⟩	17.8788
(39, 29), (20, 29), (11, 20), (4, 11), (1, 4), (1, 2)	11	⟨2, 3, 19, 40⟩	19.8332
(40, 30), (20, 30), (11, 20), (4, 11), (1, 4), (1, 3)	11	⟨2, 3, 19, 40⟩	19.8332
(22, 13), (5, 13), (2, 5), (1, 2), (1, 3), (3, 7)	2	⟨25, 7, 8, 11⟩	24.1866
(17, 26), (8, 17), (3, 8), (1, 3), (1, 2), (2, 5)	3	⟨5, 10, 11⟩	19.1950
(18, 28), (10, 18), (4, 10), (1, 4), (1, 2), (2, 5)	4	⟨6, 7, 8, 30⟩	22.7734
(22, 31), (13, 22), (5, 13), (2, 5), (2, 6), (6, 14)	5	⟨32, 2, 3, 12⟩	21.9252
(25, 34), (14, 25), (6, 14), (2, 6), (2, 5), (5, 12)	6	⟨13, 3, 4, 34⟩	20.9972

Table 3: Some paths of length 7, their centers, the induced subtrees with maximum reliability saving, and the reliability sum of each subtree.

Path $P(u, v)$	Root $e \in \Gamma_2$	Induced subtree $\langle \Lambda_c \rangle$	$R(\langle \Lambda_c \rangle)$
(3, 7), (1, 3), (1, 2), (2, 6), (6, 14), (14, 25), (25, 34)	(2, 6)	⟨34, 7, 8, 11⟩	23.5746
(4, 10), (1, 4), (1, 2), (2, 6), (6, 14), (14, 25), (25, 35)	(2, 6)	⟨34, 7, 8, 11⟩	23.5746
(4, 11), (1, 4), (1, 2), (2, 6), (6, 14), (14, 25), (25, 34)	(2, 6)	⟨34, 7, 8, 11⟩	23.5746
(5, 12), (2, 5), (1, 2), (1, 4), (4, 10), (10, 18), (18, 28)	(1, 4)	⟨14, 15, 16, 29⟩	23.2456
(8, 17), (3, 8), (1, 3), (1, 4), (4, 11), (11, 20), (20, 30)	(1, 4)	⟨14, 15, 16, 29⟩	23.2456
(6, 14), (2, 6), (1, 2), (1, 4), (4, 10), (10, 18), (18, 28)	(1, 4)	⟨14, 15, 16, 29⟩	23.2456
(17, 26), (8, 17), (3, 8), (1, 3), (1, 4), (4, 11), (11, 21)	(1, 3)	⟨13, 14, 20, 21⟩	22.7339
(17, 26), (8, 17), (3, 8), (1, 3), (1, 4), (4, 11), (11, 20)	(1,3)	⟨13, 14, 20, 21⟩	22.7339
(17,27),(8,17),(3,8),(1,3),(1,4),(4,10),(10,19)	(1,3)	⟨13, 14, 20, 21⟩	22.7339

the condition that existing facilities are already located.

(ii) The biobjective reliable central-median (k, l)-tree core facility problem on tree networks with unreliable edges should be considered. In this problem the two criteria maximin and maxisum will be considered simultaneously and find the set of all the Pareto-Optimal paths.

(iii) Another extension is the multifacility reliable extensive facility network location problem. This problem seeks the location of more than one of tree-shaped facilities in such a way that it maximizes the sum of the reliabilities between nodes and new facilities.

Data Availability

The data used to support the findings of this study are included within the article.

Disclosure

Primary results of this manuscript have been presented previously as conference abstract/talk in International Conference on Mathematics, Trends and Development ICMTD17, Cairo, Egypt, Organized by The Egyptian Mathematical Society.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] R. I. Becker, Y. I. Chang, I. Lari, A. Scozzari, and G. Storchi, "Finding the l-core of a tree," *Discrete Applied Mathematics*, vol. 118, no. 1-2, pp. 25–42, 2002.

[2] S. Peng and W. Lo, "Efficient algorithms for finding a core of a tree with a specified length," *Journal of Algorithms*, vol. 20, no. 3, pp. 445–458, 1996.

[3] C. A. Morgan and P. J. Slater, "A linear algorithm for a core of a tree," *Journal of Algorithms*, vol. 1, no. 3, pp. 247–258, 1980.

[4] E. Minieka and N. H. Patel, "On finding the core of a tree with a specified length," *Journal of Algorithms*, vol. 4, no. 4, pp. 345–352, 1983.

[5] B. Wang, "Finding a 2-core of a tree in linear time," *SIAM Journal on Discrete Mathematics*, vol. 15, no. 2, pp. 193–210, 2002.

[6] R. I. Becker, I. Lari, and A. Scozzari, "Algorithms for central-median paths with bounded length on trees," *European Journal of Operational Research*, vol. 179, no. 3, pp. 1208–1220, 2007.

[7] B. Wang, S. Peng, H. Yu, and S. Ku, "Efficient algorithms for a constrained k-tree core problem in a tree network," *Journal of Algorithms*, vol. 59, no. 2, pp. 107–124, 2006.

[8] S. Peng, A. Stephens, and Y. Yesha, "Algorithms for a core and k-tree core of a tree," *Journal of Algorithms*, vol. 15, no. 1, pp. 143–159, 1993.

[9] R. I. Becker, I. Lari, G. Storchi, and A. Scozzari, "Efficient algorithms for finding the (k,l)-core of tree networks," *Networks*, vol. 40, no. 4, pp. 208–215, 2002.

[10] A. Shioura and T. Uno, "A linear time algorithm for finding ak-tree core," *Journal of Algorithms*, vol. 23, no. 2, pp. 281–290, 1997.

[11] E. Minieka, "The optimal location of a path or tree in a tree network," *Networks*, vol. 15, no. 3, pp. 309–321, 1985.

[12] T. U. Kim, T. J. Lowe, A. Tamir, and J. E. Ward, "On the location of a tree-shaped facility," *Networks. An International Journal*, vol. 28, no. 3, pp. 167–175, 1996.

[13] A. Tamir, J. Puerto, and D. Pérez-Brito, "The centdian subtree on tree networks," *Discrete Applied Mathematics*, vol. 118, no. 3, pp. 263–278, 2002.

[14] A. Tamir, J. Puerto, J. Mesa, and A. Rodríguez-Chía, "Conditional location of path and tree shaped facilities on trees," *Journal of Algorithms*, vol. 56, no. 1, pp. 50–75, 2005.

[15] E. Melachrinoudis and M. E. Helander, "A single facility location problem on a tree with unreliable edges," *Networks*, vol. 27, no. 3, pp. 219–237.

[16] G. Xue, "Linear time algorithms for computing the most reliable source on an unreliable tree network," *Networks*, vol. 30, no. 1, pp. 37–45, 1997.

[17] M. E. Helander and E. Melachrinoudis, "Facility location and reliable route planning in hazardous material transportation," *Transportation Science*, vol. 31, no. 3, pp. 216–226, 1997.

[18] J. Santiváñez, E. Melachrinoudis, and M. E. Helander, "Network location of a reliable center using the most reliable route policy," *Computers & Operations Research*, vol. 36, no. 5, pp. 1437–1460, 2009.

[19] H. A. Eiselt, M. Gendreau, and G. Laporte, "Location of facilities on a network subject to a single-edge failure," *Networks*, vol. 22, no. 3, pp. 231–246, 1992.

[20] H. Eiselt, M. Gendreau, and G. Laporte, "Optimal location of facilities on a network with an unreliable node or link," *Information Processing Letters*, vol. 58, no. 2, pp. 71–74, 1996.

[21] W. Ding and G. Xue, "A linear time algorithm for computing a most reliable source on a tree network with faulty nodes," *Theoretical Computer Science*, vol. 412, no. 3, pp. 225–232, 2011.

[22] J. Puerto, F. Ricca, and A. Scozzari, "Reliability problems in multiple path-shaped facility location on networks," *Discrete Optimization*, vol. 12, pp. 61–72, 2014.

[23] W. Ding, Y. Zhou, G. Chen, H. Wang, and G. Wang, "On the 2-MRS problem in a tree with unreliable edges," *Journal of Applied Mathematics*, vol. 2013, Article ID 743908, 11 pages, 2013.

[24] G. Y. Handler, "Minimax location of a facility in an undirected tree graph," *Transportation Science*, vol. 7, no. 3, pp. 287–293, 1973.