A linear time algorithm for the backup 2-center problem on a tree

Hung-Lung Wang[1], Bang Ye Wu[2,*], and Kun-Mao Chao[1,3,4,†]
[1]Department of Computer Science and Information Engineering
[3]Graduate Institute of Biomedical Electronics and Bioinformatics
[4]Graduate Institute of Networking and Multimedia
National Taiwan University, Taipei, Taiwan 106
[2]Department of Computer Science and Information Engineering
Shu-Te University, YenChau, Kaohsiung, Taiwan 824

Abstract

In this paper, we are concerned with the problem of deploying two servers in a tree network, where each server may fail with a given probability. Once a server fails, the other server will take the whole responsibility of service for all vertices (clients). Here, we assume that the servers do not fail simultaneously. In the backup 2-center problem, we want to deploy two servers at the vertices such that the expected distance from a farthest vertex to the nearest surviving server is minimum. We shall propose an $O(n)$-time algorithm for the backup 2-center problem, where n is the number of vertices in the given tree network.

Keywords: location problem, center, median, tree

1 Introduction

Facility location problem is an important topic in the fields of transportation and communication. The k-center problem and the k-median problem are two classic problems in this line of investigation. Given are a graph G with positive edge lengths, a server (supply) set Γ, and a client (demand) set Δ. Usually, $\Gamma \in \{V, E\}$ and $\Delta \in \{V, E\}$, where V and E denote the vertex set and the edge set of the given network G, respectively. The k-center problem (respectively, the k-median problem) is to seek k points $x_1, x_2, ..., x_k$ from Γ so that $\max_{y\in\Delta} \min_{1\leq i\leq k} d(x_i, y)$ (respectively, $\sum_{y\in\Delta} \min_{1\leq i\leq k} d(x_i, y)$) is minimum, where $d(x, y)$ denotes the distance from x to y in the network. One can further categorize the problems by $\Gamma/\Delta/k$. For example $E/V/2$ refers to selecting two points in E such that the objective function is optimal over V.

The k-center problem in general graphs, for arbitrary k, is NP-hard [15]. Therefore, many researchers focused on tree networks. The current best results of $V/V/k$, $E/V/k$, and $V/E/k$ for the k-center problem on trees are all solved in $O(n)$ time by Frederickson [9], and $E/E/k$ is solved in $O(kn \log(2n/k))$ in [10]. Another generalization is the weighted version of the k-center problem, in which the weights of all the vertices are described by a function $w(\cdot)$ and the weighted distance from vertex u to v is $w(u)d(u, v)$. The current known results for the weighted k-center problem on trees take $O(n \log^2 n)$ time for $V/V/k$ [22], and $O(kn \log n)$ time for both $V/V/k$ and $E/V/k$ [14]. Some results for special graphs can refer to [6, 13, 19].

In practice, uncertainties always play roles of influence. The minmax-regret facility problem [2, 3, 4, 5, 7, 8, 17, 20] is one model under such consideration. Another model, called a reliability model [23, 24, 25], deals with the situation where some servers may sometimes fail and the clients originally served by these serves have to request from surviving servers. In this paper, we consider the reliability-based formulations, in which each server may fail with a given probability, of the 2-center problem ($V/V/2$) and the 2-median problem ($V/V/2$) on a tree network. Once a server fails, the other server will take the whole responsibility of services. Here, we assume that the servers do not fail simultaneously. Note that a server may fail but it still act as a client. That is,

*bangye@mail.stu.edu.tw
†kmchao@csie.ntu.edu.tw

the distance between any pair of vertices does not change when servers fail. Also, it is not required that servers are deployed at different vertices.

The backup 2-center problem is defined as follows. Let p_1 and p_2 be the failure probabilities of the two servers. To simplify the presentation, we assume that $p_1 = p_2 = p$. The case $p_1 \neq p_2$ can be handled in a similar way. For any vertex $x \in U \subseteq V$, let $\phi(x, U) = \max_{u \in U}\{d(x, u)\}$ denote the eccentricity of a vertex x to a set U. The vertex c with $\phi(c, V) = \min_{x \in V} \phi(x, V)$ is a center of G. The traditional 2-center problem is to find a set S of two vertices such that $\max_{v \in V}\{\min_{s \in S} d(v, s)\}$ is minimum. For any vertex pair (v_1, v_2), let $\Pi(v_1, v_2) = (V_1, V_2)$ in which $V_1 = \{v | d(v, v_1) \leq d(v, v_2)\}$ and $V_2 = V - V_1$. Note that, in a tree, each of V_1 and V_2 induces a connected component.

Given (v_1, v_2) and let $(V_1, V_2) = \Pi(v_1, v_2)$. For a failure probability $0 \leq p < 1$, the expected value of the longest distance from any client to its nearest surviving server, in condition that not both servers fail, can be calculated by

$$\frac{1}{1-p^2}\left((1-p)^2 \max\{\phi(v_1, V_1), \phi(v_2, V_2)\}\right.$$
$$\left.+p(1-p)(\phi(v_1, V) + \phi(v_2, V))\right).$$

Since p is a given constant, the objective function of the backup 2-center problem can be defined by

$$\begin{aligned}\psi_p(v_1, v_2) &= (1-p)\max\{\phi(v_1, V_1), \phi(v_2, V_2)\} \\ &+ p(\phi(v_1, V) + \phi(v_2, V)). \qquad (1)\end{aligned}$$

An example is given in Figure 1 to illustrate the difference between the backup 2-center and the traditional 2-center. In the traditional 2-center problem, the two severs are always deployed at different vertices, but in the backup 2-center problem, it is possible to deploy the two servers at the same vertex namely the center of the given network. If $p = 0$, the backup 2-center is (x_2, x_6), which is the 2-center of T. If $p = 0.3$, the backup 2-center is (x_3, x_5). If $p = 0.9$, the backup 2-center is (x_4, x_4), where x_4 is the center of T.

The rest of this paper is organized as follows. Section 2 proves some basic properties and gives an $O(n)$-time algorithm for the backup 2-center problem on a vertex-unweighted tree. Concluding remarks are given in Section 3.

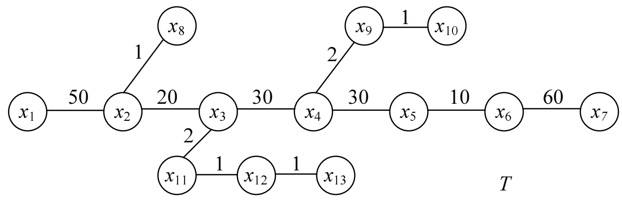

Figure 1: A tree T with center x_4 and 2-center (x_2, x_6).

2 The backup 2-center problem on vertex-unweighted trees

In the following, we denote the given tree network by T, the vertex set of T by V, a path with endpoints x and y by $\mathcal{P}[x, y]$, the center of T by c, and the backup 2-center of T by (c_1, c_2). Here we use "center" to mean the discrete center of T, unless we explicitly state it as the continuous center. Before introducing the algorithm, we summarize some basic properties of the tree center and the diameter which will be used later.

The following three properties are related to the diameter of a tree, which can be shown easily (cf. P. 156 [29]).

Property 1: Let $\mathcal{P}[x, y]$ be a diameter of T. For any vertex v, $\phi(v, V) = \max\{d(v, x), d(v, y)\}$.

Property 2: For any vertex v of T, the farthest vertex to v in T must be an endpoint of a diameter of T.

Property 3: Let c be a center of T. For any two vertices x and y, if x is on the path between c and y, then $\phi(x, V) \leq \phi(y, V)$.

Lemma 4: For $x \in V$, let $d(x, v) = \phi(x, V)$. All centers of T lie on $\mathcal{P}[x, v]$.

Proof: Omitted. □

Now we turn to explore the properties of the backup 2-center. Let the center edge of two vertices x and y be the edge which contains the continuous center of $\mathcal{P}[x, y]$.

Lemma 5: If $\psi_p(c_1, c_2) < \psi_p(c, c)$, then any diameter of T contains the center edge of c_1 and c_2.

Proof: Let $\Pi(c_1, c_2) = (V_1, V_2)$, and T_1 and T_2 be the subtree induced by V_1 and V_2, respectively. Suppose to the contrary that there exists a diameter $\mathcal{D}$ of T which does not contain the center edge of c_1 and c_2. This implies that $\mathcal{D} \subset T_1$ or $\mathcal{D} \subset T_2$. Without loss of generality, we assume that $\mathcal{D} \subset T_1$. By Property 1, we have $\phi(c_1, V_1) = \phi(c, V)$, and thus

$$\max\{\phi(c_1, V_1), \phi(c_2, V_2)\} \geq \phi(c, V).$$

Therefore, by (1),

$$\begin{aligned} & \psi_p(c_1, c_2) \\ = \; & (1-p)\max\{\phi(c_1, V_1), \phi(c_2, V_2)\} \\ & +p(\phi(c_1, V) + \phi(c_2, V)) \\ \geq \; & (1-p)\phi(c, V) + p(\phi(c, V) + \phi(c, V)) \\ = \; & \psi_p(c, c). \end{aligned}$$

This contradicts the assumption that $\psi_p(c_1, c_2) < \psi_p(c, c)$ and the lemma follows. $\square$

From the above lemma, we know that the center edge of c_1 and c_2 must lie on the intersection of all diameters. If the intersection of the diameters of T contains no edge, then the backup 2-center must be (c, c) and can be identified in linear time. Therefore, in the following we assume that the backup 2-center consists of two different vertices.

Lemma 6: If $\psi_p(c_1, c_2) < \psi_p(c, c)$, then $c \in \mathcal{P}[c_1, c_2]$.

Proof: Let $(V_1, V_2) = \Pi(c_1, c_2)$ and (u, v) be the edge where $u \in V_1$ and $v \in V_2$. Without loss of generality, we assume that $c \in V_1$, and let x be the farthest vertex to c_1 in T. Suppose to the contrary that c does not lie on $\mathcal{P}[c_1, c_2]$. This implies that $x \in V_1$, since otherwise $u \in \mathcal{P}[c_1, x]$, and by Lemma 4 $c \in \mathcal{P}[c_1, x]$ and therefore $c \in \mathcal{P}[c_1, u] \subset \mathcal{P}[c_1, c_2]$, which contradicts that $c \notin \mathcal{P}[c_1, c_2]$. Thus, $\phi(c_1, V_1) = \phi(c_1, V)$, and $\max\{\phi(c_1, V_1), \phi(c_2, V_2)\} \geq \phi(c_1, V_1) = \phi(c_1, V) \geq \phi(c, V)$. Similar to the proof of Lemma 5, we can show that $\psi_p(c_1, c_2) \geq \psi_p(c, c)$. It is also a contradiction. $\square$

Lemma 7: Let (u, v) be the center edge of c_1 and c_2, then $c_1 \in \mathcal{P}[c_1', u]$ and $c_2 \in \mathcal{P}[c_2', v]$, where c_1' and c_2', respectively, are the centers of the two subtrees after removing $e = (u, v)$. Moreover assume with loss of generality that $c \in V_1$. Then $c_1 \in \mathcal{P}[c_1', c]$.

Proof: First, we show that any center c of T is on $\mathcal{P}[c_1', c_2']$. For otherwise, by Lemma 4, the farthest vertex to c_1' is in V_1, and there are two possible situations:
(i) If $\phi(c_1', V) > \phi(c, V)$, we have $\phi(c_1', V_1) = \phi(c_1', V) > \phi(c, V) \geq \phi(c, V_1)$, a contradiction.
(ii) If $\phi(c_1', V) = \phi(c, V)$, i.e. c_1' is also a center, we have $\phi(c_1', V_1) = \phi(c_1', V) = \phi(c, V) > \phi(c, V_1)$. The last inequality holds because both c_1' and c are centers, and thus the farthest vertices to c and c_1' are different (two end vertices of a diameter). Since the farthest vertex to c_1' in T is in V_1, that of c lies in V_2. Therefore $\phi(c, V) > \phi(c, V_1)$ and we have a contradiction.

Suppose to the contrary that $c_1 \notin \mathcal{P}[c_1', u]$. Let x be the vertex nearest to c_1 on $\mathcal{P}[c_1', u]$. Since x is on $\mathcal{P}[c_1, c_1']$, by Property 3, $\phi(x, V_1) < \phi(c_1, V_1)$. Since c is on both $\mathcal{P}[c_1, c_2]$ and $\mathcal{P}[c_1', c_2']$, x is on $\mathcal{P}[c_1, c]$. Also by Property 3, $\phi(x, V) < \phi(c_1, V)$. Hence, we have

$$\begin{aligned} & (1-p)\max\{\phi(c_1, V_1), \phi(c_2, V_2)\} \\ & +p(\phi(c_1, V) + \phi(c_2, V)) \\ > \; & (1-p)\max\{\phi(x, V_1), \phi(c_2, V_2)\} \\ & +p(\phi(x, V) + \phi(c_2, V)) \\ \geq \; & \psi_p(x, c_2), \end{aligned}$$

the last inequality holds, because for any $v_1, v_2 \in V$, $\max\{\phi(v_1, V_1), \phi(v_2, V_2)\} \leq \max\{\phi(v_1, V_1'), \phi(v_2, V_2')\}$, where $(V_1, V_2) = \Pi(v_1, v_2)$ and (V_1', V_2') is any bipartition of V. Therefore it is a contradiction, and c_1 is on $\mathcal{P}[c_1', u]$. Similarly, we can show that c_2 is on $\mathcal{P}[c_2', v]$. By Lemma 6, c lies between c_1 and c_2. Thus if V_1 contains c, then c_1 lies on $\mathcal{P}[c_1', c]$. $\square$

In the following, we call $\mathcal{P}[c_1', c]$ and $\mathcal{P}[c_2', v]$ the candidate paths with respect to e, which is mentioned in Lemma 7.

Lemma 8: For each diameter of T, there is a backup 2-center on it.

Proof: Omitted. $\square$

Based on Lemmas 7 and 8, we give an $O(n^2)$-time algorithm for the backup 2-center problem in Figure 2, and then improve the time bound to $O(n)$.

In the $O(n^2)$-time algorithm BACKUP2CENTER1, we first compute a center c and a diameter of T, and then look for the pair of vertices on the candidate paths with minimum cost.

Let $\mathcal{D} = (x_1, x_2, ..., x_h)$ be the diameter we compute. For convenience, we root the tree at

Algorithm BACKUP2CENTER1(T,p)
Input: A tree T and a real p, where $0 \leq p < 1$.
Output: The backup 2-center of T.

find a diameter $\mathcal{D} = (x_1, x_2, ..., x_h)$
  and a center c, root T at x_h
cost $\leftarrow \psi_p(c, c)$; *backup2center* $\leftarrow (c, c)$
for $i = 1$ to $h - 1$
  compute $(r_i, \overline{r}_i)$
  for $x \in \mathcal{P}_i$, compute $\phi(x, V_i)$ **end for**
  for $y \in \overline{\mathcal{P}}_i$, compute $\phi(y, \overline{V}_i)$ **end for**
  for $x \in \mathcal{P}_i$, find $f(i, x)$ **end for**
  for $y \in \overline{\mathcal{P}}_i$, find $f(i, y)$ **end for**
end for
cost $\leftarrow \min\{\min_{i,x} \psi'_p(x, f(i, x)),$
  $\min_{j,y} \psi'_p(y, f(j, y)), cost\}$
backup2center $\leftarrow$the vertex pair
  with minimum cost

Figure 2: An $O(n^2)$-time algorithm for the backup 2-center.

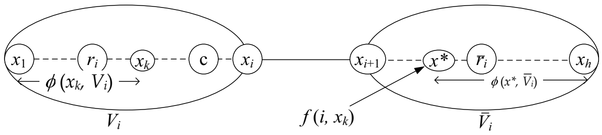

Figure 3: The partner $f(i, x_k)$ of x_k.

x_h, and let T_i be the subtree rooted at x_i, V_i be the vertex set of T_i, and $\overline{V}_i = V - V_i$. Also, let r_i and $\overline{r}_i$, $\mathcal{P}_i$ and $\overline{\mathcal{P}}_i$, be the centers and candidate paths of the two subtrees induced by V_i and $\overline{V}_i$, respectively.

For an edge (x_i, x_{i+1}) on $\mathcal{D}$ and a vertex x_k on $\mathcal{P}_i$, we search the partner $f(i, x_k)$ of x_k on $\overline{\mathcal{P}}_i$. Here the partner $f(i, x_k)$ of x_k is defined to be a vertex $x^* \in \overline{\mathcal{P}}_i$ such that $\phi(x^*, \overline{V_i}) \leq \phi(x_k, V_i)$ and

$$\psi'_p(i, x_k, x^*) = \min_{x \in \overline{\mathcal{P}}_i} \psi'_p(i, x_k, x),$$

where

$$\begin{aligned}\psi'_p(i, x_k, x) &= (1-p)\max\{\phi(x_k, V_i), \phi(x, \overline{V_i})\} \\ &\quad + p(\phi(x_k, V) + \phi(x, V)).\end{aligned}$$

In fact, the partner of $x_k \in \mathcal{P}_i$ is the vertex x closest to c on $\overline{\mathcal{P}}_i$ and satisfying $\phi(x, \overline{V_i}) \leq \phi(x_k, V_i)$ since, by Property 3, $\phi(x, V)$ decreases when x moves toward the center. An illustration is given in Figure 3.

Suppose both x_k and x_{k+1} are on $\mathcal{P}_i$. We have $\phi(x_{k+1}, V_i) \geq \phi(x_k, V_i)$, and therefore $f(i, x_{k+1})$ is on $\mathcal{P}[f(i, x_k), c]$. By this property, for edge (x_i, x_{i+1}), the partners of all vertices on $\mathcal{P}_i$ can be found by a one-pass scan from $\overline{r}_i$ to x_{i+1}. As a result, the partners of all $x_k \in \mathcal{P}_i$ can be found in linear time. Also, the partners of all $x_l \in \overline{\mathcal{P}}_i$ can be computed similarly. The time complexity of BACKUP2CENTER1 is therefore $O(n^2)$ since the center and the diameter of a tree can be computed in linear time [29].

Algorithm BACKUP2CENTER1 spends $O(n^2)$ time to compute the candidate paths of all edges and find the vertex pair with optimal ψ'_p value on the paths. In order to improve the time bound, we give the following lemmas.

First, the candidate paths of all edges can be computed in linear time by Lemma 9.

Lemma 9: Given a diameter $\mathcal{D}$ of T, $(r_i, \overline{r}_i)$ can be computed in $O(n)$ time for $1 \leq i < h$.

Proof: Omitted. □

Although the candidate paths of all edges on the diameter can be identified in $O(n)$ time, there are still $O(n^2)$ pairs of vertices needed to examine since for each edge we need to find all the partners of the vertices on the candidate path and a vertex may be on more than one candidate path. In order not to process a vertex repeatedly, we give the following lemmas. Because of the symmetry of V_i and $\overline{V}_i$, some of the following lemmas only describe the properties of V_i and omit the part of $\overline{V}_i$.

Lemma 10: $\phi(x_k, V_i) = d(x_k, x_1)$ for any $j < k \leq i$, in which $x_j = r_i$.

Proof: Omitted. □

Let $\mathcal{R}_i$ be the subset of $\mathcal{P}_i$ such that $f(i, x)$ exists for all $x \in \mathcal{R}_i$.

Lemma 11: If $x_k \in \mathcal{R}_i$, then $x_l \in \mathcal{R}_i$ for $x_l \in \mathcal{P}_i$ and $l \geq k$.

Proof: Omitted. □

By Lemma 11, we know that $\mathcal{R}_i$ is a connected subgraph of $\mathcal{D}$ (i.e. a path) and can be identified by a pair of vertices (x_l, x_r), where x_l and x_r are the two vertices in $\mathcal{R}_i$ with the smallest and the largest indices respectively. In the following we call x_l the left boundary of $\mathcal{R}_i$ and x_r the right boundary of $\mathcal{R}_i$.

Lemma 12: For $x \in \mathcal{R}_i$, if $x \in \mathcal{P}_{i+1}$, then $x \in \mathcal{R}_{i+1}$.

Proof: Omitted. □

Lemma 13: For $1 \leq i < h$, $\mathcal{R}_i$ can be found in amortized constant time.

Proof: By Lemma 11, we only need to identify the two boundaries of each $\mathcal{R}_i$, and the procedure is described as follows. First we compute the candidate paths of all edges, which can be done in $O(n)$ time by Lemma 9. For $i = 1$, there is only one vertex on $\mathcal{P}_i$ and we can see whether $f(1, x_1)$ exists by checking $\overline{\mathcal{P}}_i$. For $i \geq 2$, we first identify the right boundary, and then the left one. Let x_c be the center of T, the right boundary of $\mathcal{R}_i$ is $x_{\min\{c,i\}}$. If $i \leq c$ and $f(i, x_i)$ does not exist, by Lemma 11 we know that $\mathcal{R}_i$ is empty. Otherwise, by Lemma 12 the left boundary can be found by searching from the left boundary of $\mathcal{R}_{i-1}$ toward r_i. The total time for finding all the $\mathcal{R}_i$ boundaries is $O(n)$, and thus each costs amortized constant time. □

Lemma 14 : $\psi'_p(i, x_k, f(i, x_k)) \leq \psi'_p(j, x_k, f(j, x_k))$, where $i < j$, $x_k \neq r_j$, and $x_k \in \mathcal{R}_i \cap \mathcal{R}_j$.

Proof: Omitted. □

Corollary 15 : $\psi'_p(i, x_i, f(i, x_i)) = \min_{j \geq i} \psi'_p(j, x_i, f(j, x_i))$, where $x_i \in \mathcal{R}_i \cap \mathcal{R}_j$, and $x_i \neq r_j$.

Based on the above lemmas, we propose an algorithm Backup2center as in Figure 4. The correctness and time complexity are both shown in Theorem 16.

Theorem 16 : Backup2center computes the backup 2-center of T in $O(n)$ time.

Proof: By Lemma 14, we need only to compute the partners of the vertices in $(\mathcal{R}_i - \mathcal{R}_{i-1}) \cup \{r_i\}$ for each $i \in \{1, 2, ..., h-1\}$.

$f(i, r_i)$ can be found iteratively. $f(i+1, r_{i+1})$ is on the path between $f(i, r_i)$ and the center of T because $\phi(r_{i+1}, V_{i+1}) \geq \phi(r_i, V_i)$. The total time for this case is $O(n)$.

For vertices in $\mathcal{R}_i - \mathcal{R}_{i-1}$, there are two types. The first is the vertex x_i, and $f(i, x_i)$ can be computed by a procedure similar to $f(i, r_i)$. By Corollary 15, we know that $\psi'_p(i, x_i, f(i, x_i)) = \min_{j \geq i} \psi'_p(j, x_i, f(j, x_i))$, and thus it is not necessary to compute $\psi'_p(j, x_i, f(j, x_i))$ for $j > i$. The second type of vertices are those on $\mathcal{P}[x_k, x_{k'}]$, where x_k and $x_{k'}$ are the left boundaries of $\mathcal{R}_i$ and $\mathcal{R}_{i-1}$, respectively. $f(i, x_{k'-1})$ can be found by searching from $f(i-1, x_{k'})$. For $j < k' - 1$, $f(i, x_j)$ can be found iteratively similar to $f(i, r_i)$. Thus the total time is $O(n)$. □

Algorithm Backup2center(T, p)
Input: A tree T and a failure probability $0 \leq p < 1$.
Output: The backup 2-center of T.

```
1  find a diameter D = (x_1, x_2, ..., x_k) and a center c;
   root T at x_h
2  cost ← ψ_p(c, c); backup2center ← (c, c)
3  compute (r_i, r̄_i), d(x_i, x_1), d(x_i, x_h), φ(r_i, V_i),
   φ(r̄_i, V̄_i) for 1 ≤ i < h
4  compute R_i for 1 ≤ i < h
5  compute ψ'_p(1, x_1, f(1, x_1))
6  for i = 2 to h − 1
7      do for x_k ∈ R_i − R_{i−1}
8          do compute ψ'_p(i, x_k, f(i, x_k))
9      end for
10     compute ψ'_p(i, r_i, f(i, r_i))
11 end for
12 root T at x_1 and repeat step 3 to 11
13 backup2center ← (x, y), where
   ψ_p(x, y) = min{min_{i,j} ψ'_p(i, x_j, f(i, x_j)), ψ_p(c, c)}
```

Figure 4: An $O(n)$-time algorithm for the backup 2-center.

By the description of the algorithm, we know that the failure probability is used only when computing ψ'_p. Thus in the case where $p_1 \neq p_2$, it suffices to take into account the permutation of the two servers and redefine the objective function to be

$$\begin{aligned} & \psi_{\{p_1,p_2\}}(v_1, v_2) \\ = & \ (1-p_1)(1-p_2)\max\{\phi(v_1, V_1), \phi(v_2, V_2)\} \\ + & \ p_2(\phi(v_1, V)) + p_1(\phi(v_2, V)), \end{aligned}$$

and consequently

$$\begin{aligned} & \psi'_{\{p_1,p_2\}}(i, x_k, x) \\ = & \ (1-p_1)(1-p_2)\max\{\phi(x_k, V_i), \phi(x, \overline{V_i})\} \\ + & \ p_1(\phi(x_k, V)) + p_2(\phi(x, V)). \end{aligned}$$

Therefore, we conclude this section by the following theorem.

Theorem 17: Given failure probabilities p_1 and p_2, the backup 2-center of a vertex-unweighted tree can be found in linear time.

3 Conclusions

In this paper, we consider the backup 2-center problem on trees and solve it in $O(n)$ time. In the backup 2-center problem, the underlying graph is a vertex-unweighted tree and a natural generalization is to consider the vertex-weighted tree. The backup 2-center problem on vertex-weighted trees can be solved by a naive $O(n^3)$-time method but the exact time bound is still unknown. we are also interested in the backup k-center problem. All of them will be our future works.

4 Acknowledgements

We thank Hsiao-Fei Liu for verifying the proof. Hung-Lung Wang and Kun-Mao Chao were supported in part by NSC grants 94-2213-E-002-018 and 95-2221-E-002-126-MY3 from the National Science Council, Taiwan. Bang Ye Wu was supported in part by NSC grants 94-2213-E-366-006 and 95-2221-E-366-003 from the National Science Council, Taiwan.

References

[1] V. Auletta, D. Parente, and G. Persiano, Dynamic and static algorithms for optimal placement of resources in a tree, *Theoretical Computer Science*, 165, pp. 441–461, 1996.

[2] I. Averbakh and O. Berman, Minimax regret p-center location on a network with demand uncertainty, *Location Science*, 5, pp. 247–254, 1997.

[3] I. Averbakh and O. Berman, Algorithms for the robust 1-center problem on a tree, *European Journal of Operationl Research*, 123, pp. 292–302, 2000.

[4] I. Averbakh and O. Berman, Minimax regret median location on a network under uncertainty, *Informs Journal on Computing*, 12, pp. 104–110, 2000.

[5] I. Averbakh and O. Berman, An improved algorithm for the minmax regret median problem on a tree, *Networks*, 41:2, pp. 97–103, 2003.

[6] B. Ben-Moshe, B. Bhattacharya, Q. Shi, and A. Tamir, Efficient algorithms for center problems in cactus graphs, *International Symposium on Symbolic and Algebraic Computation* pp. 693–703, 2005.

[7] R. E. Burkard, H. Dollani, A note on the robust 1-center problem on trees, *Annals of Operations Research*, 110, pp. 69-82, 2002.

[8] B. T. Chen, C. S. Lin, Minmax-regret robust 1-median location on a tree, *Networks*, 31, pp. 93–103, 1998.

[9] G. N. Frederickson, Parametric search and locating supply centers in trees, *Workshop on Algorithms and Data Structures* pp. 299–319, 1991.

[10] G. N. Frederickson and D. B. Johnson, Finding kth paths and p-centers by generating and searching good data structures, *Journal of Algorithms* 4, pp. 61–80, 1983.

[11] B. Gavish and S. Sridhar, Computing the 2-median on tree networks in $O(n \log n)$ time, *Networks*, 26, pp. 305–317, 1995.

[12] A. J. Goldman, Optimal center location in simple networks, *Transportation Science*, 5, pp. 212–221, 1979.

[13] Y. Gurevich, L. Stockmeyer and U. Vishkin, Solving NP-hard problems on graphs that are almost trees and an application to facility location problems, *Journal of ACM* 31:3, pp. 459–473, 1984.

[14] M. Jeger and O. Kariv, Algorithms for finding p-centers on a weighted tree (for relatively small p), *Networks* 15, pp. 381–389, 1985.

[15] O. Kariv and S. L. Hakimi, An algorithmic approach to network location problems, Part I. The p-centers, *SIAM Journal on Applied Mathematics* 37, pp. 441–461, 1996.

[16] O. Kariv and S. L. Hakimi, An algorithmic approach to network location problems, Parrt II: The p-medians, *SIAM Journal on Applied Mathematics*, 37, pp. 539–560, 1979.

[17] P. Kouvelis, G. Vairaktarakis, and G. Yu, Robust 1-median location on a tree in the presence of demand and transportation cost uncertainty. Working Paper 93/94-3-4. Department of Management Science and Information Systems, Graduate School of Business, The University of Texas at Austin, 1994.

[18] S. C. Ku, C. J. Lu, B. F. Wang, T. C. Lin, Efficient algorithms for two generalized 2-median problems on trees, *International Symposium on Algorithms and Computation* , pp. 768–778, 2001.

[19] Y. F. Lan, Y. L. Wang and H. Suzuki, A linear-time algorithm for solving the center problem on weighted cactus graphs, *Information Processing Letters* 71, pp. 205–212, 1999.

[20] T. C. Lin, H. I Yu, B. F. Wang, Improved algorithms for the minmax-regret 1-center problem, *International Symposium on Algorithms and Computation*, pp. 537–546, 2006.

[21] N. Megiddo and A. Tamir, New results on the complexity of p-center problems, *SIAM Journal on Computing* 12:4, pp. 751–758, 1983.

[22] N. Megiddo, A. Tamir, E. Zemel and R. Chandrasekaran, An $O(n \log^2 n)$ algorithms for the kth longest path in a tree with applications to location problems, *SIAM Journal on Computing*, 10:2, pp. 328–337, 1981.

[23] L. V. Snyder and M. S. Daskin, Reliability models for facility location: The expected failure cost case, *Transportation Science*, 39:3, pp. 400–416, 2005.

[24] L. V. Snyder, Facility location under uncertainty: A review, *IIE Transactions*, 38:7, pp. 537–554, 2006.

[25] L. V. Snyder and M. S. Daskin, Stochastic p-robust location problems, *IIE Transactions*, 38:11, 971-985, 2006.

[26] A. Tamir, An $O(pn^2)$ algorithm for the p-median and related problems on tree graphs, *Operations Research Letters*, 19, pp. 59–64, 1996.

[27] A. Tamir, D. Perez-Brito, and J. A. moreno-Perez, A polynomial algorithm for the p-centdian problem on a tree, *Networks*, 32, pp. 255–262, 1998.

[28] B. F. Wang, S. C. Ku, and K. H. Shi, Cost-optimal parallel algorithms for the tree bisector problem and applications, *IEEE Transactions on Parallel and Distributed Systems*, 12, no. 9, pp. 888–898, 2001.

[29] B. Y. Wu and K. M. Chao, *Spanning Trees and Optimization Problems*, Chapman and Hall/CRC Press, USA, 2004.