Discrete Optimization 32 (2019) 37–42

Contents lists available at ScienceDirect

Discrete Optimization

www.elsevier.com/locate/disopt

Note

A linear time algorithm for balance vertices on trees

Van Huy Pham [a], Kien Trung Nguyen [b,*], Tran Thu Le [b]

[a] *Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City, Viet Nam*
[b] *Mathematics Department, Teacher College, Can Tho University, Viet Nam*

ARTICLE INFO

Article history:
Received 13 October 2017
Received in revised form 13 July 2018
Accepted 20 November 2018
Available online 10 December 2018

MSC:
90B10
90B80
90C27

Keywords:
Balance vertices
Tree
Complexity

ABSTRACT

The concept of balance vertices was first investigated by Reid (1999). For the main result "the balance vertices of a tree consist of a single vertex or two adjacent vertices", Shan and Kang (2004) and Reid and DePalma (2005) improved the length and technique of the proof. In this paper we further discuss the balance vertices on trees in a generalization context. We do not only provide a simple efficient proof for the relevant result but also develop a linear time algorithm to find the set of balance vertices on the underlying tree.

1. Introduction

The problem of finding a set of vertices (or points) on a network, which satisfies some conditions predetermined by the decision maker, has been studied and become an interesting topic. Many of these problems showed their prominent importance in real life applications. For example, the problem of finding optimal location of new facilities (warehouses, schools, hospitals, mega airports, etc.) can be modeled as a median or center location problem on networks, which was investigated by Kariv and Hakimi [1,2]. Concerning the algorithmic approach, they showed that both median and center problems on general networks are NP-hard and that a p-median is in the vertex set. Then the two authors, Goldman [3] and Hua [4], considered the problem of finding a 1-median on tree networks. They developed linear time algorithms to find the mentioned vertex. Additionally, a universal approach of location theory was proposed by Nickel and Puerto [5].

Reid [6] introduced in 1999 the concept of balance vertices on trees, which is a discrete version of the continuous notion "center of gravity" on an object body. Precisely, a balance vertex partitions the tree

* Corresponding author.
E-mail addresses: phamvanhuy@tdtu.edu.vn (V.H. Pham), trungkien@ctu.edu.vn (K.T. Nguyen), letranthu23041996@gmail.com (T.T. Le).

https://doi.org/10.1016/j.disopt.2018.11.001
1572-5286/

into two subtrees such that the "balance" at this vertex is minimized. He applied the technique of "double orientation" to show that there exists a balance vertex or two balance vertices which are adjacent. Shan and Kang [7] then improved the proof of Reid significantly. Furthermore, Reid and DePalma [8] explored some variants of balance on trees regarding weight, weight-edge, and distance-edge. Then they gave a relationship between distance balance edges and distance balance vertices. Thus, the property on the existence of balance vertices can be considered as a consequence. In the previous paper, the authors did not concern the edge lengths and the vertex weights so far.

In this paper we extend the concept of balance vertices on trees introduced by Reid [6]. Here, vertex weights and edge lengths are taken into account. We can simply, but efficiently, prove the core result concerning the existence property of balance vertices by comparing total weighted distances in the tree partition. Finally, we develop a linear time algorithm to find the set of all balance vertices based on the optimality criteria.

2. Algorithm for the balance vertices on trees

Let a tree T be given, where $V(T)$ and $E(T)$ denote the vertex set and edge set of T, respectively. Moreover, each edge e in $E(T)$ has a positive length ℓ_e and each vertex v in $V(T)$ is associated with a positive weight w_v. The distance between two vertices u and v on the tree is the length of the path connecting them. In the following we propose a generalization of the balance vertex introduced by Reid [6].

For a vertex v, let $\mathcal{P}(v)$ be the set of all pairs of subtrees in T such that a pair in $\mathcal{P}(v)$ has exactly one vertex v in common and the union of this pair is T. Given a pair $(T_1, T_2) \in \mathcal{P}(v)$, we define

$$\mathrm{Diff}_{T_1-T_2}(v) := \sum_{v' \in V(T_1)} w_{v'} d(v', v) - \sum_{v' \in V(T_2)} w_{v'} d(v', v),$$

the difference between the total weighted distance from v to all vertices in T_1 and that to all vertices in T_2. Furthermore, we denote the balance at a vertex v by

$$bal(v) := \min\{|\mathrm{Diff}_{T_1-T_2}(v)| : (T_1, T_2) \in \mathcal{P}(v)\}.$$

Let

$$B(T) = \{v : v \in V(T) \text{ and } bal(v) \leq bal(v'), \forall v' \in V(T)\}$$

be the set of all vertices of T with the smallest balance. We further call a vertex in $B(T)$ a *balance vertex* of T. If we restrict ourselves to the case where all edge lengths and all vertex weights equal 1, then the mentioned concept is indeed the definition of balance vertex in Reid [6]. Therefore, the balance vertex on trees in this investigation is, in fact, an extension of the concept before.

Let us denote by $\mathcal{T}(v)$ the set of all maximal subtrees containing v as a leaf vertex. A trivial, but useful, fact is that, for a vertex v in T, if $T_1 \in \mathcal{T}(v)$ and $(T_1, T_2) \in \mathcal{P}(v)$ such that $\mathrm{Diff}_{T_1-T_2}(v) \geq 0$, we get $bal(v) = \mathrm{Diff}_{T_1-T_2}(v)$. Indeed, let us consider a pair $(T_1', T_2') \in \mathcal{P}(v)$, then either $T_1 \subset T_1'$ or $T_1 \subset T_2'$. Without loss of generality, we can always assume $T_1 \subset T_1'$. As a result, one obtains $\mathrm{Diff}_{T_1'-T_2'}(v) \geq \mathrm{Diff}_{T_1-T_2}(v)$. Hence, (T_1, T_2) is the minimizer of the difference $\mathrm{Diff}_{T_1'-T_2'}(v)$ for $(T_1', T_2') \in \mathcal{P}(v)$ or $bal(v) = \mathrm{Diff}_{T_1-T_2}(v)$. The following result lets us identify which part of a partition contains balance vertices.

Lemma 1. *Given an arbitrary vertex a such that there exists a partition $(L_a, R_a) \in \mathcal{P}(a)$ and $Diff_{L_a-R_a}(a) \geq 0$, then $B(T) \subset V(L_a)$.*

Proof. Assume by contradiction that $B(T) \not\subset V(L_a)$. Then there is a balance vertex, say v^* in $V(R_a)\backslash\{a\}$. Let $(L_{v^*}, R_{v^*}) \in \mathcal{P}(v^*)$ such that $a \in L_{v^*}$ and $L_{v^*} \in \mathcal{T}(v^*)$. We claim that $\mathrm{Diff}_{L_{v^*}-R_{v^*}}(v^*) > 0$. Indeed,

because $L_{v^*} \supset L_a$ and $R_{v^*} \subset R_a$, we can write

$$\begin{aligned}\mathrm{Diff}_{L_{v^*}-R_{v^*}}(v^*) &= \sum_{v\in V(L_{v^*})} w_v d(v^*,v) - \sum_{v\in V(R_{v^*})} w_v d(v^*,v)\\ &> \sum_{v\in V(L_a)} w_v d(v^*,v) - \sum_{v\in V(R_{v^*})} w_v d(a,v)\\ &> \sum_{v\in V(L_a)} w_v d(a,v) - \sum_{v\in V(R_a)} w_v d(a,v)\\ &= \mathrm{Diff}_{L_a-R_a}(a) \geq 0.\end{aligned}$$

As $\mathrm{Diff}_{L_{v^*}-R_{v^*}}(v^*) > 0$ and $L_{v^*} \in \mathcal{T}(v^*)$, we obtain $bal(v^*) = \mathrm{Diff}_{L_{v^*}-R_{v^*}}(v^*)$. Moreover, we get $\mathrm{Diff}_{L_a-R_a}(a) \geq bal(a)$ due to the assumption $\mathrm{Diff}_{L_a-R_a}(a) \geq 0$. Combining these two facts, it implies $bal(v^*) > bal(a)$, which contradicts to the optimality of v^*. Hence, it holds $B(T) \subset V(L_a)$. □

For a vertex v and a subtree X of T, let us denote by

$$D_X(v) := \sum_{v'\in V(X)} w_{v'} d(v',v)$$

the total weighted distance from all vertices in X to v. We denote from here after a subtree in $\mathcal{T}(v)$ by $T^v_{v'}$ which contains a vertex v', $v' \neq v$, for special use in the rest of the paper. For $T^v_{v'} \in \mathcal{T}(v)$, let $\overline{T}^v_{v'}$ be the complementary part of $T^v_{v'}$ in T, i.e., $\overline{T}^v_{v'}$ is induced by $(T\backslash T^v_{v'})\cup\{v\}$. Note that $(T^v_{v'}, \overline{T}^v_{v'}) \in \mathcal{P}(v)$. We obtain the following result.

Lemma 2. *Let v be a given vertex such that a balance vertex v^* is in $T^v_{v^*}$. Then it holds $D_{T^v_{v^*}}(v) > D_{\overline{T}^v_{v^*}}(v)$.*

Proof. Assume that $D_{T^v_{v^*}}(v) \leq D_{\overline{T}^v_{v^*}}(v)$, i.e., $\mathrm{Diff}_{T^v_{v^*}-\overline{T}^v_{v^*}}(v) \leq 0$. By Lemma 1, one obtains $v^* \in V(\overline{T}^v_{v^*})$, which is a contradiction. Hence, it implies $D_{T^v_{v^*}}(v) > D_{\overline{T}^v_{v^*}}(v)$. □

Theorem 1 (*See [6,7]*)**.** *The balance vertices of a tree consist of a single vertex or two adjacent vertices.*

Proof. Assume that there exist two balance vertices in T that are not adjacent, say a and b. Then there is another vertex c on the path connecting a and b. Let us consider two subtrees T^c_a, T^c_b in $\mathcal{T}(c)$. By Lemma 2, we obtain $D_{T^c_a}(c) > D_{\overline{T}^c_a}(c)$ and $D_{T^c_b}(c) > D_{\overline{T}^c_b}(c)$, it is a contradiction as $T^c_b \subset \overline{T}^c_a$ and $T^c_a \subset \overline{T}^c_b$. Hence, the theorem is proved. □

Theorem 1 shows that we can prove the core theorem in the paper of Reid [6] with even less effort than Shan and Kang [7] and Reid and DePalma [8]. In what follows we construct conditions for checking whether a vertex is a balance vertex.

Theorem 2 (*Balance vertex criteria*)**.** *A vertex v^* is a balance vertex on T if and only if the following conditions hold.*

1. *$D_{T^v_{v^*}}(v) > D_{\overline{T}^v_{v^*}}(v)$ for all adjacent vertices v of v^*.*
2. *If Condition 1. holds for an adjacent vertex v of v^*, then*

$$Diff_{T^v_{v^*}-\overline{T}^v_{v^*}}(v) \geq Diff_{T^{v^*}_v-\overline{T}^{v^*}_v}(v^*).$$

Proof. We first prove the necessary condition. Condition 1. is straightforward by Lemma 2. We just prove the second condition. Assume that there exists an adjacent vertex v of v^* such that the Condition 1. is fulfilled, i.e. $D_{T_v^{v'}}(v') > D_{\overline{T}_v^{v'}}(v')$ for all adjacent vertices v' of v. We get $D_{T_v^{v^*}}(v^*) > D_{\overline{T}_v^{v^*}}(v^*)$ since v^* is an adjacent vertex of v. Hence, as $\mathrm{Diff}_{T_v^{v^*}-\overline{T}_v^{v^*}}(v^*) > 0$ and $T_v^{v^*} \in \mathcal{T}(v^*)$, we obtain

$$bal(v^*) = \mathrm{Diff}_{T_v^{v^*}-\overline{T}_v^{v^*}}(v^*). \tag{1}$$

Applying the same argument for the vertex v^* with an adjacent vertex v, it also holds

$$bal(v) = \mathrm{Diff}_{T_{v^*}^{v}-\overline{T}_{v^*}^{v}}(v). \tag{2}$$

By (1) and (2) and the optimality of v^*, it shows $bal(v) \geq bal(v^*)$. Equivalently saying, Condition 2. is correct.

Conversely, if Conditions 1. and 2. hold, we investigate the following cases. In the first case, there is no adjacent vertex of v which satisfies Condition 2. Then the set of balance vertices is exactly $\{v^*\}$. In the second case, we consider an adjacent vertex v with Condition 2. Then the two Eqs. (1) and (2) also hold and thus, $bal(v) = \mathrm{Diff}_{T_{v^*}^{v}-\overline{T}_{v^*}^{v}}(v)$ and $bal(v^*) = \mathrm{Diff}_{T_v^{v^*}-\overline{T}_v^{v^*}}(v^*)$. Therefore, a balance vertex is in $V(T_{v^*}^{v}) \cap V(T_v^{v^*}) = \{v, v^*\}$. As $bal(v^*) \leq bal(v)$ due to the Condition 2., the vertex v^* is a balance vertex. □

We next develop a combinatorial algorithm that finds the balance vertices on T. For simplicity, we denote by $N(v)$ the set of all adjacent vertices of v. The idea of the algorithm is that, we start from a vertex r and identify a subtree T_s^r in $\mathcal{T}(r)$ which maximizes $D_{T_v^r}(r)$ for $v \in N(r)$. Then we search balance vertices along the subtree T_s^r. The next step is to consider T_t^s in $\mathcal{T}(s)$ which maximizes $D_{T_v^s}(s)$ for $v \in N(s)$. If $t = r$, we stop and apply Condition 2. of Theorem 2 to identify which vertices in $\{s, t\}$ are balance vertices. Otherwise, we continue to search for balance vertices in T_t^s. The details can be described in the following two steps.

Step 1: Initialization.

In the initial step it is possible to start from an arbitrary vertex of T. However, to speed up the process, we can start from a centroid r of T, which can be found in linear time (see [2]). Considering the tree T rooted at r, let us denote by $\widetilde{w}(v)$ the total weight of v and its descendants, and $Dis(v)$ by the total weighted distance from its descendants to itself. We can label all vertices v in V by $\left(Dis(v), \widetilde{w}(v)\right)$ in linear time; see the survey of Puerto et al. [9]. Note that $\widetilde{w}(r) = W := \sum_{v \in V} w_v$ is the total weight of vertices in T.

After completing this step, we compute

$$D_{T_v^r}(r) := Dis(v) + \widetilde{w}(v)d(r, v)$$

for all $v \in N(r)$ and find $s := arg\max_{v \in N(r)} D_{T_v^r}(r)$. We know that there exists a balance vertex which is exactly in T_s^r and go to **Step** 2.

Step 2: A linear time algorithm.

After identifying the subtree T_s^r, we search for balance vertices along this subtree by Algorithm 1. In each iteration we first reroot the tree T at s and compute $D_{T_v^s}(s)$ for both cases, say $v = r$ or $v \in N(s)\backslash\{r\}$. Due to Lemma 2, if $r = \arg\max_{v \in N(s)} D_{T_v^s}(s)$, we know that $B(T) \subset \{r, s\}$. Otherwise, we continue the new iteration with the rooted tree T_s^r where $r := s$ and $s := \arg\max_{v \in N(s)} D_{T_v^s}(s)$. In the final computation, we check if a vertex is a balance vertex by the second condition in Theorem 2.

We next analyze the complexity of Algorithm 1. At the current vertex s, we can calculate $D_{T_v^s}(s)$ in constant time for each $v \in N(s)$. Hence, this yields exactly $deg(s)$ calculations, where $deg(s)$ is the degree of s. As we access each vertex of the tree at most once, the complexity of the algorithm is $O(\sum_{v \in V(T)} deg(v)) = O(n)$, where n is the number of vertices in the tree.

Algorithm 1 Finds balance vertices of a tree.

Input: An instance of the tree T after completing **Step 1**.
Set $flag := 0$ and $W := \widetilde{w}(r)$.
while $flag = 0$ **do**
  for $v \in N(s)$ **do**
    if $v = r$ **then**
      Set $D_{T_r^s}(s) := Dis(r) - Dis(s) + (W - 2\widetilde{w}(s))d(r, s)$.
      Update $Dis(s) := Dis(s) + D_{T_r^s}(s)$ and $\widetilde{w}(s) := W$.
    else
      Set $D_{T_v^s}(s) := Dis(v) + \widetilde{w}(v)d(s, v)$.
    end if
    if $r = \arg\max_{v \in N(s)} D_{T_v^s}(s)$ **then**
      Set $flag := 1$.
    else
      Set $r := s$ and $s := \arg\max_{v \in N(s)} D_{T_v^s}(s)$.
    end if
  end for
end while
Compare the balances at r and s to determine $B(T)$.
Output: The set of balance vertices $B(T)$.

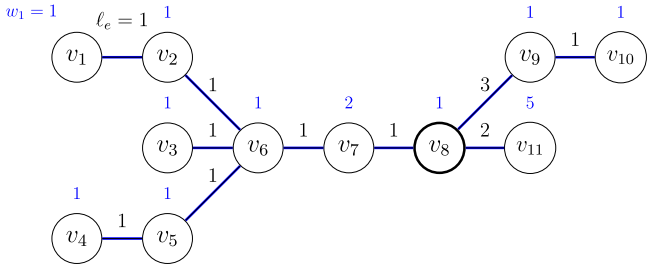

Fig. 1. An instance of a tree network.

Table 1
Vertices and its labels.

i	1	2	3	4	5	6	7	8	9	10	11
$\widetilde{w}(v_i)$	1	2	1	1	2	16	10	8	2	1	5
$Dis(v_i)$	0	1	0	0	1	42	25	17	1	0	0

Theorem 3. *The set of balance vertices on a tree can be found in linear time.*

Let us illustrate the algorithm by the following numerical example.

Example 1. Considering an instance of a tree in Fig. 1, where vertex weights and edge lengths are given.

Step 1: We first find a centroid, say v_6, and set $r := v_6$. By applying the algorithms in [9], we label all vertices in Table 1. We can further compute $D_{T_{v_2}^{v_6}}(v_6) = 3$, $D_{T_{v_3}^{v_6}}(v_6) = 1$, $D_{T_{v_5}^{v_6}}(v_6) = 3$, and $D_{T_{v_7}^{v_6}}(v_6) = 35$. As $v_7 = \arg\max_{v \in N(v_6)} D_{T_v^{v_6}}(v_6)$, we search along the subtree $T_{v_7}^{v_6}$.

Step 2:

Iter 1. Consider the tree T rooted at $s = v_7$. As $N(v_7) = \{v_6, v_8\}$, we relabel $\widetilde{w}(v_7) := 16$, $Dis(v_7) := 38$, and compute $D_{T_{v_6}^{v_7}}(v_7) := 13$, and $D_{T_{v_8}^{v_7}}(v_7) := 25$. As $v_8 = \arg\max_{v \in N(v_7)} D_{T_v^{v_7}}(v_7) \neq r$, we further set $r = v_7$ and $s = v_8$.

Iter 2. Similarly, we relabel $\widetilde{w}(v_8) := 16$, $Dis(v_8) = 38$, and compute $D_{T_v^{v8}}(v_8) = 21, 7, 10$ for $v = v_7, v_9, v_{11}$, respectively. As $v_7 = \arg\max_{v \in N(v_8)} D_{T_v^{v8}}(v_8)$, we know $B(T) \subset \{v_7, v_8\}$. Furthermore, applying Condition 2. of Theorem 2, we check that v_8 is the unique balance vertex of the tree with $bal(v_8) = 4$.

3. Conclusions

We considered in this paper a generalization of the balance vertices on trees; see [6]. We shortened and simplified the proof for the main theorem, which states that the set of balance vertices consists of exactly one or two adjacent vertices. Furthermore, a linear time algorithm that finds the set of all balance vertices on the underlying tree is also developed.

Acknowledgments

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under Grant Number 101.01-2016.08 and partially funded by Can Tho University under project number T2018-80. We are grateful to the anonymous referees for their comments which really help to improve the construction of the paper.

References

[1] O. Kariv, S.L. Hakimi, An algoithmic approach to network location problems, I. The p-centers, SIAM J. Appl. Math. 37 (1979) 513–538.
[2] O. Kariv, S.L. Hakimi, An algoithmic approach to network location problems, II. The p-medians, SIAM J. Appl. Math. 37 (1979) 536–560.
[3] A.J. Goldman, Optimal center location in simple networks, Transp. Sci. 5 (1971) 539–560.
[4] L.K. Hua, Application of mathematical models to wheat harvesting, Chin. Math. 2 (1962) 539–560.
[5] S. Nickel, J. Puerto, Location Theory - A Unified Approach, Springer, 2005.
[6] K.B. Reid, Balance vertices in trees, Networks 34 (1999) 264–271.
[7] E. Shan, L. Kang, A note on balance vertices in trees, Discrete Math. 280 (2004) 265–269.
[8] K.B. Reid, E. DePalma, Balance in trees, Discrete Math. 304 (2005) 34–44.
[9] J. Puerto, F. Ricca, A. Scozzari, Extensive facility location problems on networks: An updated review, TOP 26 (2018) 187–226.