Iran University of Science &Technology

INTERNATIONAL CONFERENCE ON COMBINATORICS, CRYPTOGRAPHY AND COMPUTATION

Proceedings of the 2nd International Conference on Combinatorics, Cryptography and Computation (I4C2017)

An efficient algorithm for the inverse eccentric problem under the Chebyshev distance

Javad Tayyebi, Seyed Mohammad Reza Kazemi
Department of Industrial Engineering, Birjand University of Technology,
javadtayyebi@birjandut.ac.ir, kazemi@birjandut.ac.ir

ABSTRACT

For a given network G(V, A, c) and two specified nodes s and t, the inverse eccentric problem is to modify the edge length vector c as little as possible so that t becomes the furthest node from s. In this paper, we study the inverse eccentric problem when the underlying network is a tree and the length modifications are measured by the Chebyshev distance. We present a polynomial-time algorithm based on the binary search technique to solve the problem.

KEYWORDS: Eccentric problem; Inverse problem; Chebyshev distance.

Mathematics Subject Classification (2010): 90C27, 90C35, 11Y16.

1 INTRODUCTION

Suppose that $G(V,A,c)$ is a connected and undirected network in which V is the set of n nodes, $A = \{e_1, e_2, \dots, e_m\}$ is the set of edges. Each edge e_j is associated with a nonnegative length c_j . We designate a specified node s as the origin and another node t as the destination. For every two nodes v and v', we denote by $d_c(v, v')$ the length of the shortest path from v to v' with respect to the length vector c. We say that t is an eccentric node of s if t is the furthest node from s, i.e., $t = argmax\{d_c(s, v) : v \in V\}$. The problem of finding the eccentric node of a node has some applications [3]. As an example, consider the location problem where one is interested in positioning a facility in a location such that maximum distance travelled to the facility is minimized. The problem can be solved by finding an eccentric node. Such a node is a possible candidate for the location of the facility.

The inverse eccentric problem is to adjust minimally the edge length vector c in such a way that t becomes an eccentric node of s. Various types of inverse combinatorial optimization problems are studied in the literature. We refer the reader to Demange and Monnot (2010) and Heuberger (2004) for a survey. To the best of our knowledge, the inverse eccentric problem is considered only when the modifications are measured by the Manhattan distance (see Nguyen and Chassein (2014)). In this paper, we consider the problem on trees under the Chebyshev distance. We design a polynomial-time algorithm to solve the problem. Since there exists a unique path P_{sv} from s to each node v in tree, the inverse eccentric problem under the Chebyshev distance can be formulated as follows:

$$\min z = \max_{e_j \in A} \left| \hat{c}_j - c_j \right| \tag{1a}$$

$$\sum_{e_j \in P_{sv}} \hat{c}_j \leq \sum_{e_j \in P_{st}} \hat{c}_j \quad \forall v \in V, \tag{1b}$$

$$\max\{0, c_j - l_j\} \leq \hat{c}_j \leq c_j + u_j \quad \forall e_j \in A, \tag{1c}$$

where $\hat{c}$ is the new nonnegative length vector to be determined, l_j and u_j are respectively the lower and upper bounds on length modifications of each edge e_j. It is remarkable that the bound constraints (1c) guarantee the nonnegativity of $\hat{c}$.

2 AN EFFICIENT ALGORITHM

To obtain a feasible solution of the problem (1), one has to increase the length of some edges $e_j \in P_{st}$ and decrease the length of some edges $e_j \notin P_{st}$ to satisfy the constraints (1b). Suppose that $\alpha_j = |\hat{c}_j - c_j|$ for every $e_j \in A$. Therefore, we have

- $\alpha_j = \hat{c}_j - c_j$ for every $e_j \in P_{st}$.
- $\alpha_j = c_j - \hat{c}_j$ for every $e_j \in A \backslash P_{st}$.

By assuming $\lambda = \max_{e_j \in A}\{|\hat{c}_j - c_j\}$, we can convert the problem (1) into the following linear programming problem:

$$\min z = \lambda \tag{2a}$$

$$\sum_{e_j \in P_{sv} \Delta P_{st}} \alpha_j \geq \sum_{e_j \in P_{sv}} c_j - \sum_{e_j \in P_{st}} c_j \quad \forall v \in V, \tag{2b}$$

$$0 \leq \alpha_j \leq u_j \quad \forall e_j \in P_{st}, \tag{2c}$$

$$0 \leq \alpha_j \leq l_j \quad \forall e_j \in A \backslash P_{st}, \tag{2d}$$

$$0 \leq \alpha_j \leq \lambda \quad \forall e_j \in A, \tag{2e}$$

where $P_{sv} \Delta P_{st} = (P_{sv} \cup P_{st}) - (P_{sv} \cap P_{st})$. We define a special solution α^λ of the problem (2) as follows:

$$\alpha_j^\lambda = \begin{cases} \min\{\lambda, u_j\} & e_j \in P_{st}, \\ \min\{\lambda, l_j\} & e_j \in A \backslash P_{st}. \end{cases} \tag{3}$$

It is notable that α^λ satisfies the constraints (2c), (2d), (2e) and its objective value is λ. Thus, α^λ is feasible only if it satisfies the constraints (2b).

Lemma 2.1. *If the problem (2) contains a feasible solution with objective value less than or equal to λ, then the solution α^λ defined by (3) is feasible to the problem.*

Proof. Suppose that the problem (2) contains a feasible solution whose objective value is at most λ. Based on the definition of α^λ and the bound constraints of the problem, $\alpha_j^\lambda \geq \alpha_j$ for every $e_j \in A$.This implies that

$$\sum_{e_j \in P_{sv} \Delta P_{st}} \alpha_j^\lambda \geq \sum_{e_j \in P_{sv} \Delta P_{st}} \alpha_j \geq \sum_{e_j \in P_{sv}} c_j - \sum_{e_j \in P_{st}} c_j$$

for each $v \in V$. Therefore, α^λ is feasible to the problem (2). □

Based on Lemma 2.1, we can restrict our attention to solutions α^λ and look for an optimal solution among such solutions. The following corollary states formally this result.

Corollary 2.2. *If the optimal value of the problem (2) is λ^*, then α^{λ^*} is an optimal solution to it.*

Corollary 2.3. *The problem (2) is feasible if $\alpha^{\lambda_{max}}$ satisfies the constraints (2b) where $\lambda_{max} = max\{max\{l_j: e_j \in A\backslash P_{st}\}, max\{u_j: e_j \in P_{st}\}\}$.*

Proof. The result is immediate based on Lemma 2.1 and the fact that $\alpha^{\lambda_{max}} = \alpha^{\lambda}$ for each $\lambda > \lambda_{max}$. □

Note that if $\lambda = 0$ is the optimal value of the problem (2), then t is an eccentric node of s with respect to the initial length vector c. As an immediate result of Corollary 2.2, the problem (2) is reduced to finding the least value $\lambda \in [0, \lambda_{max}]$ so that the solution α^{λ} satisfies the constraints (2b). Suppose that $\lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_m$ is a sorted list of elements of the set $\{l_j : e_j \in A\backslash P_{st}\} \cup \{u_j : e_j \in P_{st}\} \cup \{0\}$. Obviously, $\lambda_0 = 0$ and $\lambda_m = \lambda_{max}$. Our proposed algorithm contains two phases. In the first phase, the algorithm finds an interval $(\lambda_{i-1}, \lambda_i]$ for some $i \in \{1,2, \ldots, m\}$ so that the optimal value belongs to it. In the second phase, the optimal objective value is computed by using the result obtained from the first phase. For finding an interval $(\lambda_{i-1}, \lambda_i]$ containing the optimal objective value, it is sufficient to look for an index $i \in \{1,2, \ldots, m\}$ so that $\alpha^{\lambda_{i-1}}$ is not feasible while α^{λ_i} is feasible. This index i is identified by using the binary search technique. Suppose that the algorithm has identified such the index i. In the second phase, the algorithm computes the optimal objective value $\lambda^* \in (\lambda_{i-1}, \lambda_i]$. Based on Corollary 2.2, $\alpha^{\lambda^*}$is an optimal solution of the problem. By Substituting α^{λ^*} in the constraints (2b), for each $v \in V$, we have

$$\sum_{e_j \in P_{sv}} c_j - \sum_{e_j \in P_{st}} c_j \leq \sum_{e_j \in P_{sv} \Delta P_{st}} \alpha^{\lambda^*}$$
$$= \sum_{e_j \in P_{sv} \backslash P_{st}} \min\{\lambda^*, l_j\} + \sum_{e_j \in P_{st} \backslash P_{sv}} \min\{\lambda^*, u_j\}$$
$$= \lambda^* \left|(P_{sv} \backslash P_{st}) \backslash A^{\lambda^*}\right| + \sum_{e_j \in (P_{sv} \backslash P_{st}) \cap A^{\lambda^*}} l_j$$
$$+ \lambda^* \left|(P_{st} \backslash P_{sv}) \backslash A^{\lambda^*}\right| + \sum_{e_j \in (P_{st} \backslash P_{sv}) \cap A^{\lambda^*}} u_j$$
$$= \lambda^* \left|(P_{sv} \Delta P_{st}) \backslash A^{\lambda^*}\right| + \sum_{e_j \in (P_{sv} \Delta P_{st}) \cap A^{\lambda^*}} \lambda_j$$

where $A^{\lambda^*} = \{e_j \in A: \lambda_j < \lambda^*\} = \{e_j \in A: \lambda_j < \lambda_i\}$. Consequently,

$$\lambda^* \geq \frac{1}{|(P_{sv} \Delta P_{st}) \backslash A^{\lambda^*}|} \left(\sum_{e_j \in P_{sv}} c_j - \sum_{e_j \in P_{st}} c_j - \sum_{e_j \in (P_{sv} \Delta P_{st}) \cap A^{\lambda^*}} \lambda_j \right).$$

Note that the right handed side of the last inequality is dependent on $v \in V$. Therefore, the optimal objective value can be computed as follows:

$$\lambda^* = \max_{v \in V} \left\{ \frac{1}{|(P_{sv} \Delta P_{st}) \backslash A^{\lambda^*}|} \left(\sum_{e_j \in P_{sv}} c_j - \sum_{e_j \in P_{st}} c_j - \sum_{e_j \in (P_{sv} \Delta P_{st}) \cap A^{\lambda^*}} \lambda_j \right) \right\} \tag{4}$$

The fact that $\alpha^{\lambda_{i-1}}$ is not feasible together with the feasibility of α^{λ_i} guarantee that $\lambda^* \in (\lambda_{i-1}, \lambda_i]$. The second phase computes the optimal objective value of the problem (2) by using (4). We are ready to state formally our proposed algorithm (see Algorithm 1).

Algorithm 1. An efficient algorithm to solve the problem (2)

Input: A tree $G(V, A)$ with the edge length vector c; the lower bound l_j and the upper bound u_j for each $e_j \in A$.

Step 1: Sort elements of $\{l_j : e_j \in A \backslash P_{st}\} \cup \{u_j : e_j \in P_{st}\} \cup \{0\}$ in increasing order. Suppose that $\lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_m$ is the sorted list.

Step 2: If α^{λ_m} is not feasible, then the problem (2) is infeasible (see Corollary 2.3) and stop.

Step 3: If α^{λ_0} is feasible, then this solution is an optimal solution to the problem (2) and stop.

Step 4: Set $l = 0$ and $u = m$.

Step 5: Set $i = \left[\frac{l+u}{2}\right]$. If the solution α^{λ_i} is feasible, then set $u = i$ else set $l = i$. Repeat this step until $u - l > 1$.

Step 6: Set $i = u$. Compute $\lambda^* \in (\lambda_{i-1}, \lambda_i]$ by using (4) and stop.

Output: If the problem (2) is feasible then, the solution α^{λ^*} is an optimal solution to the problem with the optimal value λ^*.

We now analyze the complexity of the algorithm. Obviously, the bottleneck operation is Step 5 which uses the binary search technique. The number of iterations of this step is $O(\log m) = O(\log n)$. On the other hand, the feasibility of α^{λ_i} can be checked in $O(n^2)$ time. We thus establish the following result.

Theorem 2.4. *Algorithm 1 solves the problem (2) in $O(n^2 \log n)$.*

3 CONCLUDING REMARKS

In this paper, we considered the inverse eccentric problem on trees under the Chebyshev distance and presented an efficient algorithm to solve the problem. Our proposed algorithm cannot be extended in the case that the network is not a tree. Therefore, it is meaningful to consider the inverse eccentric problem under the Chebyshev distance in the general case.

REFERENCES

Demange M. and Monnot J. (2010) An introduction to inverse combinatorial problems, In: Vangelis Th. Paschos, Paradigms of Combinatorial Optimization (Problems and New approaches), Wiley,London-Hoboken.

Heuberger C. (2004) Inverse combinatorial optimization: a survey on problems, methods, and results. Journal of Combinatorial Optimization 8, 329-361.

Mneimneh M. and Sakallah K. (2003) Computing vertex eccentricity in exponentially large graphs: QBF formulation and solution. In: Proceedings of 6th international conference SAT03. Volume 2919 of LNCS.

Nguyen K.T. and Chassein A. (2014). Inverse eccentric vertex problem on networks. Central European Journal of Operations Research, DOI 10.1007/s10100-014-0367-2.