J. Appl. Math. & Informatics Vol. **28***(2010), No. 1 - 2, pp. 283 - 293*
Website: http://www.kcam.biz

INVERSE CONSTRAINED MINIMUM SPANNING TREE PROBLEM UNDER HAMMING DISTANCE

LI JIAO* AND HENGYOUNG TANG

ABSTRACT. In this paper, inverse constrained minimum spanning tree problem under Hamming distance. Such an inverse problem is to modify the weights with bound constrains so that a given feasible solution becomes an optimal solution, and the deviation of the weights, measured by the weighted Hamming distance, is minimum. We present a strongly polynomial time algorithm to solve the inverse constrained minimum spanning tree problem under Hamming distance.

AMS Mathematics Subject Classification : 90D10, 90C27
Key words and phrases : Spanning tree, inverse problem, polynomial algorithm.

1. Introduction

The inverse optimization problems have attracted increasing interest in recent years. The research was motivated by its background in traffic planning, and people have found more and more applications, such as high speed communication, computerized tomography, conjoint analysis, behavioral decision making, geophysical science, performance evaluation, etc. (For example, see [5,6,7]). In an inverse optimization problem, a candidate solution is given and goal is to modify parameters of the original problem so that the given solution becomes an optimal one under the new parameters and simultaneously minimize the costs incurred by the modification of parameters[4].

Before we introduce the inverse problem, we first present constrained minimum spanning tree problem. Let $G = (V, E)$ be a connected undirected network consisting of the node set V and the edge set E. Let $n = |V|$ and $m = |E|$. We assume that $V = \{1, 2, \cdots, n\}$, $E = \{e_1, e_2, \cdots, e_m\}$. Each edge e_i has an associated edge cost $c_i \geq 0$ and associated weight $\widetilde{w}_i \geq 0$, let $c = (c_1, c_2, \cdots, c_m)$ denote the edge cost vector and $\widetilde{w} = (\widetilde{w}_1, \widetilde{w}_2, \cdots, \widetilde{w}_m)$ denote the edge weight vector. We call spanning tree T a feasible solation and let $\mathcal{T}$ be the set of all

Received September 29. 2008. October 14, 2009. Accepted October 28, 2009. *Corresponding authors.

feasible solutions. For any a spanning tree $T \in \mathcal{T}$, let $g(F, w) = \max_{e_i \in T} \widetilde{w}(e)$ and $f(F, w) = \sum_{e_i \in T} \widetilde{w}(e)$. The constrained minimum spanning tree problem is to find a feasible solution that minimizes $f(F, w)$ subject to a bound constraint on $g(F, w)$, which is mathematically formulated as follows:

$$\min_{T \in \mathcal{T}} \quad f(F, w) \ s.t \ \ g(F, w) \leq B.$$

If G becomes disconnected after deleting the edges whose weights are not less than B, then the constrained minimum spanning tree problem is infeasible. If G becomes connected after deleting the edges whose weights are not less than B, then we can obtain the minimum spanning tree by Kruskal's algorithm[9]. It is clear that its time complexity is $O(mn)$.

The constrained minimum spanning tree problem have a big application potential. For example, in traffic network, the distance between the gas stations should not surpass the fixed distance, otherwise, the automobiles cannot be refueled and which make them could not drive; in the information network, the distance between the signal towers should not surpass the fixed distance, otherwise, the signal cannot cover the entire area.

In this paper, firstly, we consider inverse constrained minimum spanning tree problem under Hamming distance, which can be described as follows: Let $G = (V, E)$ be a connected undirected network consisting of the node set V and the edge set E. Let $n = |V|$ and $m = |E|$. We assume that $V = \{1, 2, \cdots, n\}$, $E = \{e_1, e_2, \cdots, e_m\}$. Each edge e_i has an associated edge cost $c_i \geq 0$ and associated weight $\widetilde{w}_i \geq 0$. Let $b^-, b^+ \geq 0$ be two bound vectors and c be cost vector defined on E. Let $\widetilde{T}$ be a spanning tree of G. We look for an edge weight vector $w = (w_1, w_2, \cdots w_m)$ such that

(1) $\widetilde{T}$ ia the minimum weight spanning tree with respect to w;

(2) For each $e_i \in E$, $\widetilde{w}_i - b_i^- \leq w_i \leq \widetilde{w}_i + b_i^+$;

(3) For each $e_i \in \widetilde{T}$, $g(F, w) \leq B$;

(4) $\sum_{i=1}^m c_i H(\widetilde{w}_i, w_i)$ is minimized, where $H(\widetilde{w}_i, w_i)$ is the Hamming distance between $\widetilde{w}_i$ and w_i, i.e.,$H(\widetilde{w}_i, w_i) = 0$ if $\widetilde{w}_i = w_i$ and 1 otherwise. Mathematically, it can be formulated as the the following problem:

$$\begin{array}{lll} \min & \sum_{i=1}^{m} c_i H(\widetilde{w}_i, w_i) & \\ s.t & \sum_{e_i \in \widetilde{T}} w_i \leq \sum_{e_i \in T} w_i \quad \forall \ T \in \mathcal{T} & \\ & \max_{e_i \in \widetilde{T}} w_i \leq B & (1) \\ & \widetilde{w}_i - b_i^- \leq w_i \leq \widetilde{w}_i + b_i^+ \quad \forall \ e_i \in E. & \end{array}$$

2. Solving the inverse constrained minimum spanning tree problem under Hamming distance.

In this section, we propose a general method to solve the inverse constrained minimum spanning tree problem and give a strongly polynomial time algorithm.

Before we start the main part, we list some useful notations: for given spanning tree $\widetilde{T}$, we refer to the edge set consisting of edges in $\widetilde{T}$ as tree edge set, and the set of other edges as nontree edge set. For each $e_j \in E\backslash\widetilde{T}$, $\widetilde{T} \cup \{e_j\}$ contains a unique cycle, denote by P_j the edge set consisting of all edges in this cycle except e_j.

Firstly, we should consider B and the bounds on the modifications of weights, let $b_{max} = \max\{\widetilde{w}_i - b_i^- | e_i \in \widetilde{T}\}$,

Case 1: $B < b_{max}$; Case 2: $B \geq b_{max}$;

If $B < b_{max}$, then the modification of the weights aren't in the interval $[\widetilde{w}_i - b_i^-, \widetilde{w}_i + b_i^+]$ for $e_i \in \widetilde{T}$. So the inverse constrained minimum spanning tree problem is infeasible.

If $B \geq b_{max}$, then let $\widetilde{T_B} := \{e_i \in \widetilde{T} | \widetilde{w}_i > B\}$. Clearly, for each edge $e_i \in \widetilde{T_B}$, we need to reduce the weight $\widetilde{w}_i$ to B in order to make the maximum weight on $\widetilde{T}$ equal to B. If $\widetilde{T}$ is a minimum spanning tree under weight vector w^B also holds, then

$$\begin{cases} w_i^B = B, & \forall \ e_i \in \widetilde{T_B} \\ w_i^B = \widetilde{w}_i, & \forall \ e_i \in E\backslash\widetilde{T_B} \end{cases} \tag{2}$$

is obviously the optimal solution to the inverse constrained minimum spanning tree problem. Then the cost incurred by modifications is

$$C(B) := \sum_{e_i \in \widetilde{T_B}} c_i H(\widetilde{w}_i, w_i^B).$$

Otherwise, let $B_i = \min\{B, \widetilde{w}_i + b_i^+\}$, (1) can be reformulated as follows:

$$\begin{array}{lll} \min & \sum_{i=1}^{m} c_i H(\widetilde{w}_i, w_i) & \\ s.t & \sum_{e_i \in \widetilde{T}} w_i \leq \sum_{e_i \in T} w_i & \forall \ T \in \mathcal{T} \\ & \widetilde{w}_i - b_i^- \leq w_i \leq B_i & \forall \ e_i \in \widetilde{T} \\ & \widetilde{w}_i - b_i^- \leq w_i \leq \widetilde{w}_i + b_i^+ & \forall \ e_i \in E \setminus \widetilde{T} \end{array} \tag{3}$$

It is well known [1] that $\widetilde{T}$ is a minimum spanning tree with respect to the edge weight vector w if and only if it satisfies the following optimality conditions: $w_i \leq w_j$ for each $e_j \in E \setminus \widetilde{T}$ and $e_i \in P_j$. It implies that the new weight vector w should satisfy that $\alpha_i = |w_i - \widetilde{w}_i|$ for $i = 1, 2, \cdots, m$, where $w_j = \widetilde{w}_j + \alpha_j$ for each $e_j \in E \setminus \widetilde{T}$, and $w_i = \widetilde{w}_i - \alpha_i$ if $e_i \in P_j$ for at least one $e_j \in E \setminus \widetilde{T}$.

Taking the bounds on the modifications of weights into consideration, because $\widetilde{w}_i - b_i^- \leq w_i \leq B_i$ $(e_i \in \widetilde{T})$, $-b_i^- \leq w_i - \widetilde{w}_i \leq B_i - \widetilde{w}_i$ $(e_i \in \widetilde{T})$, then $|w_i - \widetilde{w}_i| \leq \min(b_i^-, B_i - \widetilde{w}_i) = l_i$ $(e_i \in \widetilde{T})$ and because $\widetilde{w}_i - b_i^- \leq w_i \leq \widetilde{w}_i + b_i^+$ $(e_i \in E\backslash\widetilde{T})$,

$-b_i^- \leq w_i - \widetilde{w}_i \leq b_i^+$ $(e_i \in E \backslash \widetilde{T})$, then $|w_i - \widetilde{w}_i| \leq \min(b_i^-, b_i^+) = u_i$ $(e_i \in E \backslash \widetilde{T})$. Hence (2) can be reformulated as follows:

$$\begin{aligned}
\min \quad & \sum_{i=1}^{m} c_i H(\alpha_i, 0) & & \\
s.t \quad & \widetilde{w}_i - \alpha_i \leq \widetilde{w}_j + \alpha_j & & \text{for each } e_j \in E \setminus \widetilde{T} \text{ and } e_i \in P_j \\
& 0 \leq \alpha_i \leq l_i & & \forall \; e_i \in \widetilde{T} \\
& 0 \leq \alpha_j \leq u_j & & \forall \; e_j \in E \setminus \widetilde{T}
\end{aligned} \tag{4}$$

If $\widetilde{w}_i \leq \widetilde{w}_j$ is valid for some $e_i \in P_j$, then the constraint $\widetilde{w}_i - \alpha_i \leq \widetilde{w}_j + \alpha_j$ in (4) is also satisfied for any $0 \leq \alpha_i \leq l_i, 0 \leq \alpha_j \leq u_j$. Hence we can delete this inequality in constraints of (4). By setting $P_j^{'} = \{e_i \mid e_i \in P_j \text{ and } \widetilde{w}_i > \widetilde{w}_j\}$,(3) is equivalent to the problem as follows:

$$\begin{aligned}
\min \quad & \sum_{i=1}^{m} c_i H(\alpha_i, 0) & & \\
s.t \quad & \widetilde{w}_i - \alpha_i \leq \widetilde{w}_j + \alpha_j & & \text{for each } e_j \in E \setminus \widetilde{T} \text{ and } e_i \in P_j^{'} \\
& 0 \leq \alpha_i \leq l_i & & \forall \; e_i \in \widetilde{T} \\
& 0 \leq \alpha_j \leq u_j & & \forall \; e_j \in E \setminus \widetilde{T}
\end{aligned} \tag{5}$$

To get an polynomial algorithm for solving (5), we construct a bipartite graph $G^{'}$ with respect to the tree $\widetilde{T}$ and find the minimum-cost node cover problem for the bipartite graph. In fact, the bipartite graph $G^{'} = (N^{'}, A) = (N_1^{'} \cup N_2^{'}, A)$ can be obtained as follows: The node set $N^{'} = N_1^{'} \cup N_2^{'}$ satisfies $N_1^{'} = \widetilde{T}$ and $N_2^{'} = E \setminus \widetilde{T}$, i.e. each edge of E corresponds to a node of $G^{'}$, and the edge set A is obtained by considering each nontree edge e_j one by one and adding the edge (e_i, e_j) to A for each $e_i \in P_j^{'}$; that is, $A = \{(e_i, e_j) | e_j \in E \setminus \widetilde{T} \text{ and } e_i \in P_j^{'}\}$. We further define the cost of each node e_i of $G^{'}$ as c_i.

We now describe the approach for finding a minimum-cost node cover in the bipartite graph $G^{'}$. The argument of the minimum-cost node cover problem is similar to the argument of the minimum-weight node cover problem. The minimum-weight node cover problem restricted to a bipartite graph is strongly polynomially solvable via a reduction to the maximum flow problem [2,3]. The reduction works by constructing a directed network from $G^{'}$ as follows: Introduce a source s into $G^{'}$ with an arc going to each node in $N_1^{'}$ of capacity equal to the cost of that node, introduce a sink t with an arc coming in from each node in $N_2^{'}$ of capacity equal to the cost of that node. Similarly, direct each arc in A from $N_1^{'}$ to $N_2^{'}$ and make its capacity $+\infty$. Denote $\widetilde{G}$ the resulting network. The minimum (s,t)-cut in $\widetilde{G}$ can be obtained via a maximum flow computation. Moreover, the minimum cut must be finite because the net flow out of the source is finite. Hence no edge in A belong to that cut. Let

$K = \{e_j | e_j \neq s, t,$ and e_j is a node of the edge in the minimum(s,t) − cut, $\}$

then K is a node cover for G' and the capacity of the cut is exactly equal to the cost of this node cover. On the other hand, it is not difficult to see that any node cover implies a cut of capacity equal to the cost of the node cover. Therefore, the node cover determined by the minimum (s,t)-cut must be a minimum-cost node cover.

Theorem 1. *Problem (5) has a feasible solution if and only if for each* $(e_i, e_j) \in A$*, we have* $\widetilde{w}_i - \widetilde{w}_j \leq l_i + u_j$.

Proof. (Necessity) Suppose $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ is feasible solution for (5) and there exists an edge $(e_{i_1}, e_{j_1}) \in A$, such that $\widetilde{w}_{i_1} - \widetilde{w}_{j_1} > l_{i_1} + u_{j_1}$. However, from the feasibility of α, we have $\widetilde{w}_{i_1} - \alpha_{i_1} \leq \widetilde{w}_{j_1} + \alpha_{j_1}$, $0 \leq \alpha_{i_1} \leq l_{i_1}$ and $0 \leq \alpha_{j_1} \leq u_{j_1}$. Hence we get $\widetilde{w}_{i_1} - \widetilde{w}_{j_1} \leq \alpha_{i_1} + \alpha_{j_1} \leq l_{i_1} + u_{j_1}$. A contradiction.

(Sufficiency) Suppose for each $(e_i, e_j) \in A$, we have $\widetilde{w}_i - \widetilde{w}_j \leq l_i + u_j$. Then set $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ satisfying:

$$\alpha_i = \begin{cases} l_i & \text{for } e_i \in \widetilde{T} \\ u_i & \text{for } e_i \in E \setminus \widetilde{T} \end{cases}$$

It is easy to check that α is a feasible solution for (5)

According to Theorem 1, we first find all edges in G which must be changed in any feasible solution. Let

$$D = \{e_i \mid \alpha_i \neq 0 \text{ in every feasible solution } \alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)\}.$$

We then solve (4) by computing the minimum-cost node cover of the bipartite graph G'' which is obtained by deleting D and all edges incident to the nodes of D.

Firstly, we consider how to determine D. Three cases are considered as follows:

Case 1*: If there exists an edge $(e_{i_1}, e_{j_1}) \in A$ such that $\max\{l_{i_1}, u_{j_1}\} < \widetilde{w}_{i_1} - \widetilde{w}_{j_1} \leq l_{i_1} + u_{j_1}$, then we conclude that $e_{i_1} \in D$ and $e_{j_1} \in D$. In fact, assume that $e_{i_1} \notin D$, then there must exist a feasible solution $\alpha' = (\alpha'_1, \alpha'_2, \cdots, \alpha'_m)$ for (5) such that $\alpha'_{i_1} = 0$. It follows that $\alpha'_{i_1} + \alpha'_{j_1} = \alpha'_{j_1} \leq u_{j_1} < \widetilde{w}_{i_1} - \widetilde{w}_{j_1}$, which contradicts the feasibility of α'. The same argument can show $e_{j_1} \in D$.

Case 2*: If there exists an edge $(e_{i_2}, e_{j_2}) \in A$ such that $\min\{l_{i_2}, u_{j_2}\} < \widetilde{w}_{i_2} - \widetilde{w}_{j_2} \leq \max\{l_{i_2}, u_{j_2}\}$, then we have that $e_{i_2} \in D$ if $l_{i_2} > u_{j_2}$ and $e_{j_2} \in D$. Otherwise, it can be shown in the same way as Case 1*.

Case 3*: If there exists an edge $(e_{i_3}, e_{j_3}) \in A$ such that $0 < \widetilde{w}_{i_3} - \widetilde{w}_{j_3} \leq \min\{l_{i_3}, u_{j_3}\}$, then by setting

$$\alpha_i = \begin{cases} l_i & \text{if } e_i \neq e_{i_3} \text{ and } e_i \in \widetilde{T} \\ u_i & \text{if } e_i \in E \setminus \widetilde{T} \\ 0 & \text{if } e_i = e_{i_3} \end{cases}$$

We can show that $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ is a feasible solution of (5). For this solution, we have $\alpha_{i_3} = 0$, hence we cannot deduce that $e_{i_3} \in D$ at this moment. Similarly, by setting

$$\alpha_i^{'} = \begin{cases} l_i & \text{if } e_i \in \widetilde{T} \\ u_i & \text{if } e_i \neq e_{j_3} \text{ and } e_i \in E \setminus \widetilde{T} \\ 0 & \text{if } e_i = e_{j_3} \end{cases}$$

We know that $\alpha^{'} = (\alpha_1^{'}, \alpha_2^{'}, \cdots, \alpha_m^{'})$ is a feasible solution of (5), too. For this solution, we have $\alpha_{j_3}^{'} = 0$, hence we cannot deduce that $e_{j_3} \in D$.

We further normalize the bipartite graph $G^{'}$ in the following way: check every edge (e_i, e_j) in $G^{'}$ whether the above Cases 1* and 2* occur. If yes, modify D (add e_i or e_j or both to D) and delete the edge (e_i, e_j) from $G^{'}$. After this, for each $e_i \in D$, delete all edges in $G^{'}$ which are incident to the node e_i. Denote by $G^{''} = (N_1^{'} \cup N_2^{'}, A^{'})$ the resulting bipartite network, where $A^{'} = \{(e_i, e_j) \mid (e_i, e_j) \in A,\ e_i \notin D \text{ and } e_j \notin D\}$, $N_1^{'}$ and $N_2^{'}$ are unchanged. With the above analysis, each node in D is isolated in $G^{''}$ and every minimum-cost node cover for $G^{''}$ dose not contain any node in D.

Theorem 2. *Suppose the problem (5) has feasible solutions, Let $K^{'}$ be the minimum-cost node cover for $G^{''}$, and $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ satisfy*

$$\alpha_i = \begin{cases} l_i & \textit{if } e_i \in (K^{'} \cup D) \cap \widetilde{T} \\ u_i & \textit{if } e_i \in (K^{'} \cup D) \cap (E \setminus \widetilde{T}) \\ 0 & \textit{otherwise} \end{cases}$$

then α is an optimal solution for (5) with value $C(K^{'} \cup D)$.

Proof. We first show that α is a feasible solution. The last two groups of constraints in (5) hold obviously. We now prove that the first group of constrains holds for each $e_j \in E \setminus \widetilde{T}$ and $e_i \in P_j^{'}$ by the following four mutually exclusive and collectively exhaustive cases.

Case 1. $e_i \in K^{'} \cup D$, $e_j \in K^{'} \cup D$. In this case, we have that $\widetilde{w}_i - \alpha_i = \widetilde{w}_i - l_i$ and $\widetilde{w}_j + \alpha_j = \widetilde{w}_j + u_j$. From Theorem 1, we know $\widetilde{w}_i - \widetilde{w}_j \leq l_i + u_j$, i.e., $\widetilde{w}_i - \alpha_i \leq \widetilde{w}_j + \alpha_j$.

Case 2. $e_i \in K^{'} \cup D$, $e_j \notin K^{'} \cup D$. In this case, we have that $\widetilde{w}_i - \alpha_i = \widetilde{w}_i - l_i$ and $\widetilde{w}_j + \alpha_j = \widetilde{w}_j$. If further $e_i \in D$, then $\widetilde{w}_i - \widetilde{w}_j \leq \max\{l_i, u_j\} = l_i$ which follows $\widetilde{w}_i - \alpha_i \leq \widetilde{w}_j + \alpha_j$. Otherwise $e_i \in K^{'}$, we know $\widetilde{w}_i - \widetilde{w}_j \leq \min\{l_i, u_j\} \leq l_i$. It follows $\widetilde{w}_i - \alpha_i \leq \widetilde{w}_j + \alpha_j$, too.

Case 3. $e_j \in K^{'} \cup D$, $e_i \notin K^{'} \cup D$. The arguments are similar to *Case 2.*.

Case 4. $e_i \notin K^{'} \cup D$, $e_j \notin K^{'} \cup D$. We show that this case is impossible. If this case occurs, from $e_i \notin D$ and $e_j \notin D$, we know that $(e_i, e_j) \in A^{'}$. But $e_i \notin K^{'}$ and $e_j \notin K^{'}$ imply that (e_i, e_j) can not be covered by $K^{'}$. A contradiction.

Therefore we conclude that α defined above is a feasible solution for problem(5) and the objective value yielded by α is equal to $\sum_{e_i \in D} c_i + \sum_{e_j \in K'} c_j$.

Let $\alpha' = (\alpha'_1, \alpha'_2, \cdots, \alpha'_m)$ be an arbitrary feasible solution for (5). From the construction of D, we know that if $e_i \in D$, then $\alpha'_i \neq 0$. Let $Y = \{e_r \mid \alpha'_r \neq 0$ and $e_r \notin D\}$, we claim that Y is a node cover for G''. In fact, if there exists $(e_i, e_j) \in A'$ such that $e_i \notin Y$ and $e_j \notin Y$, then $\widetilde{w}_i - \widetilde{w}_j > 0$. On the other hand, the constraint $\widetilde{w}_i - \alpha'_i \leq \widetilde{w}_j + \alpha'_j$ is satisfied because $\alpha' = (\alpha'_1, \alpha'_2, \cdots, \alpha'_m)$ is a feasible solution for (4). $e_i \notin Y$ and $e_j \notin Y$ imply $\alpha'_i = 0$ and $\alpha'_j = 0$. Hence we have $\widetilde{w}_i \leq \widetilde{w}_j$, which contradicts to $\widetilde{w}_i - \widetilde{w}_j > 0$. Therefore Y is a node cover for G'', and the objective value yielded by α' is equal to $\sum_{e_i \in D} c_i + \sum_{e_j \in Y} c_j$. Because K' is the minimum-cost node cover for G'', we have

$$\sum_{e_i \in D \cup K'} c_i = \sum_{e_i \in D} c_i + \sum_{e_j \in K'} c_j \leq \sum_{e_i \in D} c_i + \sum_{e_j \in Y} c_j.$$

Therefore we conclude that α defined above is an optimal solution for (5) and the objective value yielded by α is equal to $C(K' \cup D)$.

According to the above discuss, we get the following algorithm to solve the discussed problem.

Algorithm A_1:

Step 1. For each $e_i \in \widetilde{T_B}$, reduce the weight $\widetilde{w}_i$ to B. If $\widetilde{T}$ is a minimum spanning tree under weight vector w^B, then stop; (2) is the optimal solution, the modified cost is $C(B)$. Otherwise, go to Step 2.

Step 2. Construct a bipartite graph $G' = (\widetilde{T} \cup (E \setminus \widetilde{T}), A)$ where $A = \{(e_i, e_j) | e_j \in E \setminus \widetilde{T}, \text{ and } e_i \in P'_j\}$. and define the cost of each node e_i to be c_i.

Step 3. If there exists some $(e_i, e_j) \in A$, such that $\widetilde{w}_i - \widetilde{w}_j > l_i + u_j$, then the problem has no feasible solution and stop. Otherwise go to Step 4.

Step 4. Compute G'' and D by normalizing G'.

Step 5. Construct a directed network $\widetilde{G}$ from G'', find its minimum (s,t)-cut, and transform it to a minimum-cost node cover K' for G''.

Step 6. Output an optimal solution $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ where α is specified is Theorem 2, the minimum modified cost is: $C(B) + C(K' \cup D)$.

Theorem 3. *The inverse constrained minimum spanning tree problem under Hamming distance can be solved, its time complexity is* $O(n^3 m)$.

Proof. From Theorem 2, we know that the algorithm is right. So, the inverse constrained minimum spanning tree problem under Hamming distance can be solved. We next study the time complexity of algorithm, it is clear that Steps 1,2,3,4 and 6 all take $O(mn)$ time. The main task is Step 5 (construct a directed network $\widetilde{G}$), Step 5 is to compute the maximum flow in the bipartite network $\widetilde{G}$, since $|N'_1| = n - 1$ and $|N'_2| = m - n + 1$, this step takes $O(n^3 m)$ time[1]. Hence the algorithm runs in $O(n^3 m)$ in the worst-cast and is a strongly polynomial algorithm.

3. Solving the inverse constrained minimum spanning tree problem under the bottleneck-type Hamming distance

Nextly, we consider inverse constrained minimum spanning tree problem under the bottleneck-type Hamming distance, which can be described as follows: Let $G = (V, E)$ be a connected undirected network consisting of the node set $V = \{1, 2, \cdots, n\}$ and the edge set $E = \{e_1, e_2, \cdots, e_m\}$. Each edge e_i has is associated with a weight $\widetilde{w}_i \geq 0$ and a cost $c_i \geq 0$ for modifying the weight. Let $\widetilde{w} = (\widetilde{w}_1, \widetilde{w}_2, \cdots, \widetilde{w}_m)$ denote the weight vector and $c = (c_1, c_2, \cdots, c_m)$ denote the cost vector. Let $b^-, b^+ \geq 0$ be two bound vectors defined on E. Let $\widetilde{T}$ be a spanning tree of G. We look for an edge weight vector $w = (w_1, w_2, \cdots w_m)$ such that

(1) $\widetilde{T}$ ia the minimum weight spanning tree with respect to w;

(2) For each $e_i \in E$, $\widetilde{w}_i - b_i^- \leq w_i \leq \widetilde{w}_i + b_i^+$;

(3) For each $e_i \in \widetilde{T}$, $g(F, w) \leq B$;

(4) The maximum modification cost among all edges, i.e., $\max\{c_i H(\widetilde{w}_i, w_i)\ i = 1, 2, \cdots, m\}$ is minimized.

The model can be formulated as followings :

$$\min \quad \max_{i=1,2,\cdots,m} c_i H(\widetilde{w}_i, w_i)$$

$$\begin{aligned} s.t \quad & \textstyle\sum_{e_i \in \widetilde{T}} w_i \leq \sum_{e_i \in T} w_i \quad \forall\ T \in \mathcal{T} \\ & \max_{e_i \in \widetilde{T}} w_i \leq B \\ & \widetilde{w}_i - b_i^- \leq w_i \leq \widetilde{w}_i + b_i^+ \quad \forall\ e_i \in E. \end{aligned} \tag{6}$$

Similarly, we should consider B and the bounds on the modifications of weights,too. Let $b_{max} = \max\{\widetilde{w}_i - b_i^- | e_i \in \widetilde{T}\}$,

Case 1: $B < b_{max}$; Case 2: $B \geq b_{max}$;

If $B < b_{max}$, then the modification of the weights aren't in the interval $[\widetilde{w}_i - b_i^-, \widetilde{w}_i + b_i^+]$ for $e_i \in \widetilde{T}$. So the inverse constrained minimum spanning tree problem is infeasible.

If $B \geq b_{max}$, then let $\widetilde{T_B} := \{e_i \in \widetilde{T} | \widetilde{w}_i > B\}$. Clearly, for each edge $e_i \in \widetilde{T_B}$, we need to reduce the weight $\widetilde{w}_i$ to B in order to make the maximum weight on $\widetilde{T}$ equal to B. If $\widetilde{T}$ is a minimum spanning tree under weight vector w^B also holds, then

$$\begin{cases} w_i^B = B, & \forall\ e_i \in \widetilde{T_B} \\ w_i^B = \widetilde{w}_i, & \forall\ e_i \in E \backslash \widetilde{T_B} \end{cases} \tag{7}$$

is obviously the optimal solution to the inverse constrained minimum spanning tree problem. Then the cost incurred by modifications is

$$C'(B) := \max_{i=1,2,\cdots,m} c_i H(\widetilde{w}_i, w_i^B).$$

Otherwise, let $B_i = \min\{B, \widetilde{w}_i + b_i^+\}$, (6) can be reformulated as follows:

$$\begin{array}{lll} \min & \max\limits_{i=1,2,\cdots,m} c_i H(\widetilde{w}_i, w_i) & \\ s.t & \sum_{e_i \in \widetilde{T}} w_i \leq \sum_{e_i \in T} w_i & \forall \ T \in \mathcal{T} \\ & \widetilde{w}_i - b_i^- \leq w_i \leq B_i & \forall \ e_i \in \widetilde{T} \\ & \widetilde{w}_i - b_i^- \leq w_i \leq \widetilde{w}_i + b_i^+ & \forall \ e_i \in E \setminus \widetilde{T} \end{array} \tag{8}$$

According to the discussion of the section 2, (8) is equivalent to the problem as follows:

$$\begin{array}{lll} \min & \max\limits_{i=1,2,\cdots,m} c_i H(\alpha_i, 0) & \\ s.t & \widetilde{w}_i - \alpha_i \leq \widetilde{w}_j + \alpha_j & \text{for each } \ e_j \in E \setminus \widetilde{T} \ \text{ and } \ e_i \in P_j' \\ & 0 \leq \alpha_i \leq l_i & \forall \ e_i \in \widetilde{T} \\ & 0 \leq \alpha_j \leq u_j & \forall \ e_j \in E \setminus \widetilde{T} \end{array} \tag{9}$$

To obtain an polynomial algorithm for solving (9), we still construct a bipartite graph $G' = (N, A) = (\widetilde{T} \cup \ (E \backslash \widetilde{T}), A)$ with respect to the tree $\widetilde{T}$ as follows: The node set $N = \widetilde{T} \cup \ (E \backslash \widetilde{T})$, i.e. each edge of E corresponds to a node of G', on the left side of G' if the edge is in $\widetilde{T}$, on the right side otherwise, and the edge set $A = \{(e_i, e_j) | e_j \in (E \backslash \widetilde{T})$ and $e_i \in P_j'\}$. We further define the cost of each node e_i of G' as c_i.

Noting that problems (5) and (9) have the same constraints, and as shown in section 2,we conclude that they have a feasible solution if and only if for each $(e_i, e_j) \in A$, we have $\widetilde{w}_i - \widetilde{w}_j \leq l_i + u_j$.

In order to solve (9), we first need to find the nodes in G' whose weights must be changed in every feasible solution.

Denote by $D = \{e_i \mid \alpha_i \neq 0$ in every feasible solution $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)\}$ the nodes which weights must be changed in every feasible solution. Section 2 tells us how to determine D: Suppose problem (9) is feasible, (i) For each $e_i \in \widetilde{T}$, $e_i \in D$ if and only if there exists $e_j \in (E \backslash \widetilde{T})$ such that $(e_i, e_j) \in A$ and $\widetilde{w}_i - \widetilde{w}_j > u_j$. (ii) For each $e_j \in (E \backslash \widetilde{T})$, $e_j \in D$ if and only if there exists $e_i \in \widetilde{T}$ such that $(e_i, e_j) \in A$ and $\widetilde{w}_i - \widetilde{w}_j > l_i$.

With the above analysis, we further normalize the bipartite graph G' in the following way: We start with $D = \emptyset$. Check every edge (e_i, e_j) in G' to see whether the condition in one of the above Cases 1 and 2 satisfies. If yes, modify D(add e_i or e_j or both to D) and delete the edge (e_i, e_j) from G'. After this process, for each $e_i \in D$, delete all edges in G' which are incident to the node e_i as well as the node e_i itself. In this way, we reduce G' to G'', where $G'' = ((\widetilde{T} \backslash D) \cup (\overline{\widetilde{T}} \backslash D), A')$ and $A' = \{(e_i, e_j) | (e_i, e_j) \in A, \ e_i \notin D$ and $e_j \notin D\}$.

Theorem 4. *Suppose the problem (9) has a feasible solution. Let K^* be a minimum bottleneck-cost node cover of G'', i.e., K^* is a node cover of G'' such* *that* $C_b(K^*) = \min\{C_b(K)|K$ *is a node cover of* G'' $\}$*, where* $C_b(K)$ *is the maximum cost of the element in* K*. Define* $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ *as*

$$\alpha_i = \begin{cases} l_i & \text{if } e_i \in (K^* \cup D) \cap \widetilde{T}, \\ u_i & \text{if } e_i \in (K^* \cup D) \cap (E \setminus \widetilde{T}), \\ 0 & \text{if } e_i \notin (K^* \cup D), \end{cases}$$

then α *is an optimal solution of problem (9) with the minimum objective value* $C(K^* \cup D)$.

Proof. As shown in section 2, if K^* is a minimum sum-cost node cover of G'', then α defined in the theorem is the optimal solution of problem (9). Noting again that problems (5) and (9) have the same feasible solutions, hence by a similar argument we can obtain the result.

We now describe an approach for obtaining K^*. As we do not see any reference giving explicitly an algorithm for the purpose, here we present an algorithm in detail.

Algorithm A_2:

Step 1. Let $K = \emptyset$, sort the nodes of the current graph(it is G'' initially) according to the non-decreasing order of node costs.

Step 2. Find the node e_i with the minimum cost and a positive degree in the current graph, set $K := K \cup \{e_i\}$, and delete the node e_i and all edges incident to it in the current graph.

Step 3. Repeat the process in Step 2 until the edge set of the current graph is empty. Tale $K^* = K$ and stop.

Next we show that the set K^* resulted from algorithm A_2 is indeed a minimum bottleneck-cost node cover of G''. In fact, it is easy to know that K^* is a node cover of G''. We further show that K^* has the minimum bottleneck-cost as follows. Denote $C_{i_k} = \max\{C_i|e_i \in K^*\}$, and let K' be another node set of G'' such that $\max\{C_i|e_i \in K'\} < C_{i_k}$. Then we know that $e_{i_k} \notin K'$. On the other hand, according to algorithm A_2, there is a node e_{i_r} which is adjacent to e_{i_k} such that $C_{i_r} \geq C_{i_k}$ implies that $e_{i_r} \notin K'$ and thus both end nodes of the edge (e_{i_k}, e_{i_r}) are not covered by K'. So, K' cannot be a node cover of G'' and we have finished the proof.

According to the above discuss, we get the following algorithm to solve the discussed problem.

Algorithm A_3:

Step 1. For each $e_i \in \widetilde{T_B}$, reduce the weight $\widetilde{w}_i$ to B. If $\widetilde{T}$ is a minimum spanning tree under weight vector w^B, then stop; (2) is the optimal solution, the modified cost is $C'(B)$. Otherwise, go to Step 2.

Step 2. Construct a bipartite graph $G' = (\widetilde{T} \cup (E \setminus \widetilde{T}), A)$ where $A = \{(e_i, e_j)|e_j \in E \setminus \widetilde{T},$ and $e_i \in P'_j\}$. and define the cost of each node e_i to be c_i.

If there exists some $(e_i, e_j) \in A$, such that $\widetilde{w}_i - \widetilde{w}_j > l_i + u_j$, then the problem has no feasible solution and stop. Otherwise go to Step 3.

Step 3. Compute G'' and D by normalizing G'.

Step 4. Run algorithm A_2 to get a minimum bottleneck-cost node cover K^* of G''.

Step 5. Output an optimal solution $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_m)$ where α is specified is Theorem 4, the minimum modified cost is: $C(B) + C(K^* \cup D)$.

References

1. R. K. Ahuja, T.L. Magnanti, and J. B. Orlin , *Network Flows: Theory, Algorithms, and Application,* Prentice Hall: Englewood Cliffs, NJ, 1993.
2. E. L. Lawler, *Combinatorial Optimization: Networks and Matroids,* Holt, Rinehart and Winston, 1976.
3. R. Motwani, *Approximation Algorithms,* Lecture Notes, Stanford University, Stanford, 2000.
4. Hochbaum DS. *Efficient algorithms for the inverse spanning tree problem.* Operations Research 2003; 51: 785-97.
5. Ahuja RK, Orlin JB, *Inverse optimization, part i: Linear programming and general problem,* Oper Res 2001; 35: 771-783.
6. Heuburger C, *nverse optimization, a survey on problems, methods, and results,* J Comb Optim, 2004, 361.
7. Orlin JB, *Inverse optimization and partial inverse optimization,* PPT presentation on Optimization Day Columbia University November 3 2003.
8. He Y, Zhang B, Yao E, *Weighted inverse minimum spanning tree problems under Hamming distance,* J Comb Optim 2005; 9: 91-100.
9. Kruskal J.B., *On the shortest spanning subtree of a graph and the traveling salesman problem,* Proceedings of the AMS 1956; 7: 48-50.

Department of Mathematics Shenyang Normal University, Shenyang, Liaoning, 110034
e-mail : jiaoli82@yahoo.cn