Information Processing Letters 127 (2017) 54–57

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

A note on approximation algorithms of the clustered traveling salesman problem

Xiaoguang Bao [a,*], Zhaohui Liu [b], Wei Yu [b], Ganggang Li [c]

[a] College of Information Technology, Shanghai Ocean University, Shanghai 201306, China
[b] Department of Mathematics, East China University of Science and Technology, Shanghai 200237, China
[c] School of Information Technology, Jiangxi University of Finance and Economics, Nanchang 330013, China

ARTICLE INFO

Article history:
Received 22 August 2016
Received in revised form 6 July 2017
Accepted 6 July 2017
Available online 13 July 2017
Communicated by Tsan-sheng Hsu

Keywords:
Traveling salesman problem
Clustered traveling salesman problem
Approximation algorithms

ABSTRACT

In an earlier paper (Bao and Liu [1]), we considered a version of the *clustered traveling salesman problem* (CTSP), in which both the starting and ending vertex of each cluster are free to be selected, and proposed a 2.167-approximation algorithm. In this note, we first improve this approximation ratio to 1.9 by introducing a new method to define the inter-node lengths for all the nodes in Step 2 of Algorithm A of Bao and Liu [1]. Based on the above method, we then provide a 2.5-approximation algorithm for another version of CTSP where the starting vertex of each cluster is given while the ending vertex is free to be selected, which improves the previous approximation ratio of 2.643 of Guttmann-Beck et al. [5].

1. Introduction

Given a complete undirected graph $G = (V, E)$, where V is the vertex set partitioned into K clusters $V_1, V_2, \cdots, V_K$ and E is the edge set with edge lengths satisfying triangle inequality, the *clustered traveling salesman problem* (CTSP) is to compute a shortest Hamiltonian tour so that all vertices are visited and the vertices of each cluster are visited consecutively. Guttmann-Beck et al. [5] introduced several versions of CTSP and pointed out that they are all NP-hard. In an earlier paper (Bao and Liu [1]), we considered one of the above versions of CTSP, in which no starting and ending vertices of any cluster are specified, i.e., the two vertices of each cluster are free to be selected, and proposed a 2.167-approximation algorithm. The algorithm generates two candidate tours and then outputs the better one. When constructing the second candidate tour in Step 2 of Algorithm A given in [1], we first determine the order of the clusters by regarding each cluster as a node and creating a tour for all the nodes, and then determine the order of the vertices in each cluster by computing a Hamiltonian path with two specified end vertices. One key to implement this process is to construct an auxiliary weighted graph for all the nodes, such that the edge lengths are as small as possible and satisfy the triangle inequality.

In this note, we first introduce a new method to define the inter-node lengths for all the nodes in Step 2 of Algorithm A of Bao and Liu [1] and present a 1.9-approximation algorithm, for the version of CTSP in which both the starting and ending vertex of each cluster are free to be selected. This algorithm improves the approximation ratio of 2.167 of Bao and Liu [1]. Based on the above method, we then present a 2.5-approximation algorithm, for another version of CTSP in which the starting vertex of each cluster is given while the ending vertex is free to be selected. This algorithm improves the approximation ratio of 2.643 of Guttmann-Beck et al. [5].

The remainder of this note is organized as follows. Following preliminaries in Section 2, we consider the version

* Corresponding author.
E-mail addresses: xgbao@shou.edu.cn (X. Bao), zhliu@ecust.edu.cn (Z. Liu), yuwei@ecust.edu.cn (W. Yu), lgg198505@126.com (G. Li).

http://dx.doi.org/10.1016/j.ipl.2017.07.003
0020-0190/

of CTSP in which no starting and ending vertices of any cluster are specified in Section 3, and address the version of CTSP in which the starting vertex of each cluster is given while the ending vertex is free to be selected in Section 4.

2. Preliminaries

For the sake of convenience, we use the same notations as in Bao and Liu [1]. For a graph $G = (V, E)$, we denote by $t_{i,j}$ the length of an edge $[i, j] \in E$. The edge lengths are symmetric and satisfy the triangle inequality. For an edge subset $E' \subseteq E$, we denote by $t(E') = \sum_{[i,j]\in E'} t_{i,j}$ the total length of the edges. In particular, for a Hamiltonian tour S, its length is given by $t(S)$.

In the following, we review several related problems which are mentioned in the algorithms of Sections 3 and 4. Notice that the edge lengths mentioned in the problems are symmetric and satisfy the triangle inequality.

The Traveling Salesman Path Problem **(TSPP).** Here we only consider a special case of TSPP. Let s and t be two specified vertices in G. The problem is to compute a shortest Hamiltonian path with end vertices s and t. Hoogeveen [6] provided an algorithm, say Algorithm TSPP1, with 5/3-approximation ratio. Guttmann-Beck et al. [5] first presented a new algorithm, say Algorithm TSPP2, and then selected the best of the two paths generated by Algorithms TSPP1 and TSPP2 as approximate solution. This method makes proving 5/3-approximation easier, but more important, it leads to improvements in the analysis of their algorithms. We briefly review these two algorithms (for details, please see [6] and [5]).

Algorithm TSPP1 proceeds as follows: step 1. find a minimum-length spanning tree (MST) of the graph G; step 2. create a connected Eulerian path by adding to the MST only edges of a minimum-length perfect matching on the vertex set consisting of odd degree vertices of $V\setminus\{s, t\}$ and even degree vertices of $\{s, t\}$ in the MST; step 3. transform the Eulerian path into a Hamiltonian path from s to t by applying shortcuts. Algorithms TSPP2 and TSPP1 differ in step 2 where Algorithm TSPP2 construct a connected Eulerian path by duplicating all edges of the MST except those on the unique path from s to t.

The Rural Postman Problem **(RPP).** Consider a graph $G = (V, E)$. Let $E' = \{[s_i, t_i] : i = 1, 2, \cdots, K\} \subseteq E$ be a given subset of E. The problem is to compute a shortest tour that visits all the edges in E'. Frederickson [3] presented a 3/2-approximation algorithm for RPP.

The Stacker Crane Problem **(SCP).** Let $G = (V, E)$ be a graph. Let $D = \{(s_i, t_i) : i = 1, 2, \cdots, K\}$ be a specified directed arc set, where the length of arc (s_i, t_i) equals the length of the corresponding edge $[s_i, t_i]$. The problem is to compute a shortest tour which covers each arc (s_i, t_i) in D according to the specified direction (from s_i to t_i). Frederickson et al. [4] presented a 9/5-approximation algorithm for SCP.

3. Unspecified end vertices

In this section we deal with the version of CTSP in which both the starting and ending vertex of each cluster are free to be selected. We give an algorithm with an approximation ratio of 1.9. It follows the algorithm of Bao and Liu [1] with a modification to one candidate. More specifically, we introduce a new method to define the inter-node lengths for all the nodes in Step 2 of Bao and Liu's algorithm.

Algorithm UEV.

Step 1. (See [1].) For each cluster V_i, $1 \le i \le K$, find vertices a_i and b_i such that $t_{a_i,b_i} = \max\{t_{x,y} : x, y \in V_i\}$. Apply Algorithm TSPP2 to $G(V_i)$ to obtain a path, say $path_i$, with end vertices a_i and b_i. Apply the algorithm of Frederickson [3] to the edge set $E' = \{[a_i, b_i], i = 1, \cdots, K\}$ to generate a tour S'_1. Replacing each edge $[a_i, b_i]$ by $path_i$ in S'_1. We obtain a candidate tour S_1.

Step 2. Represent each cluster V_i by a node n_i, $1 \le i \le K$. For each pair of nodes n_i and n_j, let $t'_{n_i,n_j} = \min\{t_{x,y} : x \in V_i, y \in V_j\}$. Define the inter-node length t_{n_i,n_j} as the length of the shortest path between nodes n_i and n_j according to t'_{n_i,n_j}s. It is easy to see that t_{n_i,n_j}s satisfy the triangle inequality. For the node set $\{n_i : i = 1, 2, \cdots, K\}$, apply Christofides' algorithm [2] to generate a tour S'_2. Without loss of generality, suppose that $S'_2 = (n_1, n_2, \cdots, n_K) = ([c_1, d_2], \cdots, [c_{K-1}, d_K], [c_K, d_1])$, where $c_i, d_i \in V_i$. For each cluster V_i, apply Algorithm TSPP1 to $G(V_i)$ to generate a path, say $path_i$, with end vertices c_i and d_i. Finally, $S'_2 \cup (\bigcup_{i=1}^{K} path_i)$ forms a candidate tour S_2.

Step 3. Choose the better one of S_1 and S_2 as the approximate solution.

We introduce several notations to analyze the algorithm. Let S_{UEV} be an optimal solution of the referred version of CTSP. Let L_i denote the path which goes through cluster V_i in S_{UEV}. Let $L = \bigcup_{i=1}^{K} L_i$ and $A = S_{UEV} \setminus L$. Then, we have $t(S_{UEV}) = t(L) + t(A)$.

Lemma 1. $t(S_1) \le \frac{7}{2}t(S_{UEV}) - 2t(E')$.

Proof. See Lemma 1 in Bao and Liu [1]. □

Lemma 2. $t(S'_2) \le \frac{3}{2}t(A)$.

Proof. Note that the edges in A form a tour on the node set $\{n_i : i = 1, 2, \cdots, K\}$ and Christofides' algorithm guarantees a 3/2 approximation solution for TSP [2]. Thus, the conclusion holds. □

Lemma 3. $\sum_{i=1}^{K} t(path_i) \le \frac{3}{2}t(L) + \frac{1}{2}t(E')$.

Proof. See Lemma 2 in Bao and Liu [1]. □