

Online covering salesman problem

Huili Zhang^{1,2} · Yinfeng Xu^{1,2}

Published online: 12 January 2018

© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract Given a graph $G = (V, E, D, W)$, the generalized covering salesman problem (CSP) is to find a shortest tour in G such that each vertex $i \in D$ is either on the tour or within a predetermined distance L to an arbitrary vertex $j \in W$ on the tour, where $D \subset V, W \subset V$. In this paper, we propose the online CSP, where the salesman will encounter at most k blocked edges during the traversal. The edge blockages are real-time, meaning that the salesman knows about a blocked edge when it occurs. We present a lower bound $\frac{1}{1+(k+2)L}k + 1$ and a CoverTreeTraversal algorithm for online CSP which is proved to be $k + \alpha$ -competitive, where $\alpha = 0.5 + \frac{(4k+2)L}{OPT} + 2\gamma\rho$, γ is the approximation ratio for Steiner tree problem and ρ is the maximal number of locations that a customer can be served. When $\frac{L}{OPT} \rightarrow 0$, our algorithm is near optimal. The problem is also extended to the version with service cost, and similar results are derived.

Keywords Covering salesman problem · Realtime blockage · Competitive analysis · Service cost

1 Introduction

The Covering salesman problem(CSP) was firstly proposed by Current and Schilling (1989). It can be stated as follows: identify the minimum cost tour of a subset of n

✉ Huili Zhang
zhang.huilims@xjtu.edu.cn

Yinfeng Xu
yfxu@xjtu.edu.cn

¹ School of Management, Xi'an Jiaotong University, Xi'an 710049, China

² State Key Lab of Manufacturing and System Engineering, Xi'an 710049, China

given cities such that every city not on the tour is within some predetermined covering distance standard, L_i , of a city that is on the tour. The CSP can be viewed as a generalization of the traveling salesman problem (Garey and Johnson 1990). If $L_i = 0$ or $L_i < \min_j p_{ij}$, where p_{ij} denotes the shortest distance between vertices i and j , the CSP reduces to a TSP (thus, it is NP-hard).

Current and Schilling (1989) referred to several real world examples, such as the routing of rural health care delivery teams, where not all the cities are required to be visited. The application also includes the humanitarian aid, police patrolling, sensor location and so on (Hachicha et al. 2000; Ha et al. 2013; Flores-Garza et al. 2015; Naji-Azimi et al. 2012; Yang et al. 2014)

Recently, it has some new applications as the blooming of online shopping. Due to the promotion of e-commerce company, the amount of orders will increase dramatically, which usually leads to delay of delivery. In order to speed up the delivery, the salesman usually visits part of the customers and visit one of the several third-party convenient stores nearby in order to complete the delivery within some promised deadline, say two or three hours, as some express company *Shunfeng* and *DHL* do. In the above practical case, the salesman only visit a subset of vertices to cover the remainder customer vertices.

Traveling on the urban traffic network, the salesman suffers a lot from the uncertain travel time. It is usually induced by the uncertain traffic blockages which may be from the the traffic accident, rush hour or bad weather. Taking the uncertain traffic blockages into account, we propose the online covering salesman problem. Assuming that a blockage can be known when it occurs, we try to design a robust online routing algorithm for the salesman to complete the delivery as soon as possible.

1.1 Literature review

After Current and Schilling, some extensions of the CSP have appeared in the literature. Considering the application in concert tour planning, Golden et al. (2012) defined a generalized version of CSP, where each vertex i needs to be covered k_i times, and there is a cost associated with each visited vertex. The cost may come from the fixed cost associated with a concert.

Current and Schilling (1994) introduced two bi-criterion routing problems: the median tour problem(MTP) and the maximal covering tour problem(MCTP). In both problems the tour must visit only p of the n vertices on the network, and they both have one goal to minimize the total tour length. In MTP, the other goal is the minimization of the total distance from each demand at the vertices to their nearest stop on the tour. In MCTP, the other goal is to maximize the total demand within some given distance. Another generalization and closely related problem discussed in Gendreau et al. (1997) is the covering tour problem (CTP). Here, given an undirected graph $G = (V \cup W, E)$, the vertices in V can be on the tour, the vertices in $T \subset V$ must be visited while the vertices in W must be covered. Like the CSP, a vertex i not on the tour must be within a predefined covering distance L_i of a vertex on the tour. When $T \cup W = \phi$, the CTP reduces to a CSP, and when the subset of vertices that must be on the tour consists of the entire vertex set $T = V \cup W$, the CTP reduces to the TSP.

After these fundamental works, there are also some variations considering multiple vehicles (Hachicha et al. 2000; Ha et al. 2013; Flores-Garza et al. 2015), capacity constraints (Naji-Azimi et al. 2012), disk covering constraint (Yang et al. 2014), random demands (Tricoire et al. 2012). The application includes the humanitarian aid, police patrolling, sensor location and so on. CSP is also related to several famous routing problems, such as the prize collecting TSP(PCTSP) (Bienstock et al. 1993), selective TSP (Laporte and Martello 1990), clustered TSP (Chisman 1975). The interested readers could refer the survey by Gutin and Punnen (2002).

Most of the above works gave an exact solution or a heuristic solution for the problems. As to the approximation algorithm, Slavik (1997) focused on the tree cover problem and covering tour problem, where all the vertices in the graph are visited or covered. The algorithm was based on LP relaxation rounding operations, and the algorithm is proved to be $\frac{3}{2}\rho$ -approximation where ρ is maximum of locations from which a customer can be served. Safra and Schwartz (2006) proved various geometric covering problems related to the TSP cannot be efficiently approximated to within any constant factor unless $P = NP$. It includes the TSP with neighborhood, group steiner tree, the minimum watchman problem and so on. All the problems are defined in the Euclidean plane. For the version considering the penalty of the unvisited sites, which is actually the PCTSP. Based on Christofides' algorithm (Christofides 1976) for TSP as well as a method of rounding of linear programming(LP) relaxation to integers, PCTSP was proved to be 2.5-approximation (Bienstock et al. 1993) and the ratio was improved to be $2 - \frac{1}{n}$ by Goemans et al. (1992).

For the previous research, the online TSP with online edge blockages may be the most related. Liao and Huang (2014) studied the online TSP in complete weighted graphs, called covering Canadian traveler problem (cCTP). An algorithm with competitive ratio as $O(\sqrt{k})$ was proposed, where k is also the upper bound of number of blockages. If the salesman only visits a subset of vertices, the problem is called Steiner TSP(sTSP). In our own paper (Zhang et al. 2015), we studied the sTSP with online edge blockages, and proposed a near optimal polynomial strategy with competitive ratio $k + 4$ while the lower bound was proved to be $k + 1$. The version with advance information was also studied (Zhang et al. 2016), where the traveler can get the information a time units in advance.

The original version of cCTP(or sTSP) is the Canadian traveler problem(CTP), which is the Shortest path problem with online edge blockages. It is introduced by Papadimitriou and Yannakakis (1991). It is PSPACE-complete to find an online algorithm with a constant competitive ratio for CTP (Papadimitriou and Yannakakis 1991). When the number of blocked edges is at most k , the variation is denoted as k -CTP. The best competitive ratio for k -CTP is $2k + 1$ by Zhu et al. (2003) and Westphal (2008) independently. Zhang et al. (2013) also extended k -CTP to the case of multiple travelers.

1.2 Our results

Motivated by delivery in urban traffic network, we studied the *covering salesman problem* with online edge blockages, which has a wide application in operations society

nowdays. For the version without service cost, we present a lower bound $\frac{1}{1+(k+2)L}k+1$, the online algorithm is proved to be $k + \alpha$ competitive, $\alpha = 0.5 + \frac{4kL}{OPT} + 2\gamma\rho$. When $\frac{L}{OPT} \rightarrow 0$, our algorithm is near optimal. We also extend the analysis to the version with service cost.

The remainder of the paper is organized as follows. Section 2 gives the problem formulation and some fundamental assumptions. In Sect. 3, we present a lower bound for online covering salesman problem. In Sect. 4, the *CoverTreeTraversal* algorithm is proposed and the competitive ratio is proved. Sect. 5 extends the analysis to online covering salesman problem with service cost, where the service fee is charged by the third-party for the service per parcel. We summarize the main conclusions of this work and proposes some future research problems in Sect. 6.

2 Preliminary

Given an edge-weighted graph $G = (V, E, D, W)$, where $|V| = n$ and $|E| = m$ and $D \subset V$ is the collection of customer vertices, $W \subset V$ is the collection of the convenient stores vertices. The edge weight $w(u, v)$ represents the traversal time, or the distance, between the two vertices for edge $(u, v) \in E$. In the paper, we assume there are at most k edge blockages, and let $\delta = \langle e_1, e_2, \dots, e_k \rangle$ denote the sequence of these online blocked edges. Each of the blocked edge is revealed to the salesman once it occurs. We try to find a tour with minimum travel time to guarantee each vertex $v \in D$ is either on the tour or within a distance L to some vertex in W on the tour. Starting from a depot s , we seek a minimum cost closed tour that vertex $i \in D$ is either on the tour or within a distance L to an arbitrary vertex $j \in E$ on the tour. We call this problem online *Covering Salesman Problem*(online CSP) with edge blockages.

This way, the instance of the online can be denoted as a quintuple $I = (V, E, D, W, \delta)$ (or simplified as a couple $I = (G, \delta)$). For an algorithm \mathcal{A} , the competitive ratio is $c_{\mathcal{A}} = \sup_I \frac{\mathcal{A}(I)}{OPT(I)}$, where $\mathcal{A}(I)$ is the total travel time derived by algorithm \mathcal{A} for instance I , and $OPT(I)$ is the corresponding optimal value. The lower bound of this problem is $\beta = \inf_{\mathcal{A}} c_{\mathcal{A}}$. Note that $\beta \geq 1$ for our problem.

Our discussion is based on the assumptions that the remainder graph is connected after deleting the blocked edge and the traversed edge will not be blocked later.

3 Lower bound

Recall that δ denotes the sequence of online blocked edges that will be revealed to the salesman during the traversal. Due to the oblivious adversary, the realtime information is equivalent to the case that every blockage $e = (u, v)$ is revealed exactly when the salesman is at u or v .

Theorem 1 For online CSP, $\beta \geq \frac{1}{1+(k+2)L}k + 1$.

Proof We prove this lower bound of $k + 1$ by showing that for any online algorithm, there is an instance $I = (V, E, D, W, \delta)$ of online CSP, such that the competitive ratio of the algorithm on the instance I is almost $k + 1$. In the instance I , the graph

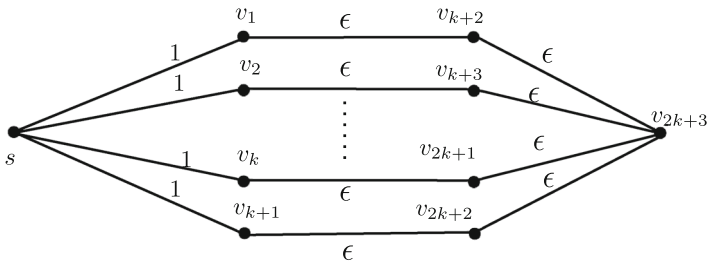


Fig. 1 Special unweighted graph

$G = (V, E, D)$ is shown in Fig. 1, with $D = \{s, v_{k+2}, v_{k+3}, \dots, v_{2k+2}\}$, $V \setminus D = \{v_1, v_2, \dots, v_{k+1}\}$, $W = V$, and $E = \{(s, v_i), \{v_i, v_{k+1+i}\}, \{v_{k+1+i}, v_{2k+3}\} | 1 \leq i \leq k + 1\}$. The edge weights are, for $1 \leq i \leq k + 1$.

$$\begin{cases} w(s, v_i) = 1 \\ w(v_i, v_{k+1+i}) = \epsilon \\ w(v_{k+1+i}, v_{2k+3}) = \epsilon \end{cases} \tag{3.1}$$

where ϵ is an arbitrary small positive number.

For any online algorithm \mathcal{A} , from depot s , if the salesman chooses to traverse edge (s, v_i) , for some i such that $1 \leq i \leq k + 1$, then edge (v_i, v_{k+1+i}) is blocked by the adversary when the salesman arrives at vertex v_i . Clearly the salesman has to go back to depot s afterward, and uses \mathcal{A} to choose another edge (s, v_j) to traverse, for some j such that $1 \leq j \leq k + 1$; if $j = i$, then edge (v_j, v_{k+1+j}) is blocked when the salesman arrives at vertex v_j . And so on for the first k distinct attempts by the salesman to traverse exactly k out of the $k + 1$ edges of form (s, v_i) , $i = 1, 2, \dots, k + 1$, the corresponding k edges of form (v_i, v_{k+1+i}) are blocked in sequence, denoted as δ .

1) If $\epsilon > L$, the salesman will succeed to arrive at v_{2k+3} in his $(k + 1)$ st try. Then he will visit every vertex with demand and return to s , resulting a closed tour. Due to Eq. 3.1, the cost of algorithm \mathcal{A} is

$$\begin{aligned} \mathcal{A}(G, \delta) &= 2k + 1 + \epsilon + 2(k + 1)\epsilon + \epsilon + 1 \\ &= 2k + 2 + 2(k + 2)\epsilon \end{aligned}$$

The optimal closed tour for adversary on the $G = (V, E - \delta, D)$ is

$$\begin{aligned} \text{OPT}(G, \delta) &= 1 + \epsilon + 2(k + 1)\epsilon + \epsilon + 1 \\ &= 2 + 2(k + 2)\epsilon \end{aligned}$$

From all above,

$$\frac{\mathcal{A}(G, \delta)}{\text{OPT}(G, \delta)} = \frac{2k + 2 + 2(k + 2)\epsilon}{2 + 2(k + 2)\epsilon} \leq 1 + \frac{k}{1 + (k + 2)L} = c_{\mathcal{A}}$$

Hence, $c_{\mathcal{A}} \geq \frac{1}{1+(k+2)L}k + 1$ holds for all \mathcal{A} .

2) If $\epsilon \leq L$, the salesman visit vertex v_i means that v_{k+1+i} has been covered. The salesman will go to s instantly, and the travel time is

$$\mathcal{A}(G, \delta) = 2k + 2;$$

The offline salesman will visit v_{2k+3} to cover D , and the optimal offline traversal time is

$$\text{OPT}(G, \delta) = 1 + \epsilon + 2\epsilon + \epsilon + 1 = 2 + 4\epsilon.$$

For this scenario,

$$\frac{\mathcal{A}(G, \delta)}{\text{OPT}(G, \delta)} \leq \frac{2k + 2}{2 + 4\epsilon} \leq k + 1 = c_{\mathcal{A}},$$

where $\epsilon \geq 0$. Hence, $c_{\mathcal{A}} \geq k + 1$ holds for all \mathcal{A} .

Hence, we have $\beta \geq \min \left\{ \frac{1}{1+(k+2)L}k + 1, k + 1 \right\} = \frac{1}{1+(k+2)L}k + 1$. This proves the theorem. □

4 CoverTreeTraversal algorithm

In this section, we will introduce the CoverTreeTraversal algorithm for the online Covering Salesman Problem (online CSP). The algorithm includes two parts. The first part is to get an approximation solution of Covering Steiner Tree(CST) w.r.t. D via the linear programming relaxation. The second part is to execute Deep-First-Search(DFS) traversal to visit all the customer vertices on the CST or the subtrees induced by the blockages.

4.1 LP relaxation for CST

In this section, we compute a tree by the LP relaxation of CST. The Steiner Tree is a special case of Covering Steiner tree with covering distance $L = 0$. Since the Steiner tree problem is NP-hard, CST is also NP-hard.

For the initial solution, we define the covering matrix A as follows: $A = (a_{ij})$, where $(a_{ij})_{i,j \in V} = \begin{cases} 1 & 0 \leq l(i, j) < L, i \in D, j \in W \\ 1 & i = j, i \in D \\ 0 & \text{otherwise} \end{cases}$, where $l(i, j)$ is the travel time between i and j . Vertex i can be served at set $a[i]$, where $a[i] = \{j \in W \mid a_{ij} = 1\} \cup \{i\}$. We also define that $\rho = \max_{i \in D} \{|a[i]|\}$, i.e. one customer can be served on at most ρ vertices. In most of practical cases, ρ is at most two or three. Now we give the integer programming of CSP, with the depot as vertex 0.

IP_1 : CSP

$$\begin{aligned}
 Z_1 = \min & \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \\
 \text{s.t. (a)} & \sum_{i \in V} x_{ij} - \sum_{k \in V} x_{jk} = 0, \quad \forall j \in V \\
 \text{(b)} & \sum_{i \in V/S} x_{ij} \geq y_j, \quad \forall j \in V, S \subset V, |S \cap \{j, 0\}| = 1 \\
 \text{(c)} & \sum_{j \in a[i]} a_{ij} y_j \geq 1, \quad \forall i \in D \\
 \text{(d)} & y_j \in \{0, 1\}, \quad \forall j \in V \\
 \text{(e)} & x_{ij} \in \{0, 1\}, \quad \forall i, j \in V
 \end{aligned} \tag{4.1}$$

Remove the constraint (a) of Eq. 4.1, the remainder forms the integer programming of CST we call it IP_2 , i.e.

$$IP_2 : \left\{ Z_2 = \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \mid (b) - (e) \right\} \tag{4.2}$$

Clearly, $Z_2 \leq Z_1$ since the lack of (a).

LP_2 is the linear programming relaxation of IP_2 as follows:

$$LP_2 : \left\{ Z_2^* = \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \mid (b), (c), 0 \leq y_i \leq 1, 0 \leq x_{ij} \leq 1 \right\} \tag{4.3}$$

We give an approximation algorithm, say LP-CST, for CST. The idea is to select a subset of vertices in $D \cup W$ to satisfy the covering constraint, and connect all the selected vertices as a tree.

Algorithm 1 LP-CST

Output: Covering Steiner Tree (CST).

Step 1: solve LP_2 to get $(y, x, Z_2^*) = ((y_j^*)_{j \in D \cup W}, (x_{ij}^*)_{i, j \in V}, Z_2^*)$;

Step 2: define $\bar{x}_{ij} = \rho x_{ij}^*, \bar{y}_j = \begin{cases} 1 & y_j^* \geq \frac{1}{\rho} \\ 0 & \text{otherwise} \end{cases}, T = \{j \in D \cup W \mid y_j^* \geq 1/\rho\}$;

Step 3: compute the Steiner tree w.r.t. T .

Lemma 1 $\forall i \in D$, there is at least one vertex in T covering it.

Proof Due to $|a[i]| \leq \rho$ and $\sum_{j=1}^n a_{ij} y_j^* = \sum_{j \in a[i]} y_j^* \geq 1$ due to constraint (c). Then there is at least one vertex, say j_0 , that $y_{j_0}^* \geq 1/\rho$, then $\bar{y}_{j_0} = 1, j_0 \in T$. It proves that at least one vertex, j_0 , covers vertex i . □

Let $T \subset V$, define y_j as follows: $y_j = 1$ if $j \in T$; otherwise, $y_j = 0$. Consider the following problem,

$$\begin{aligned}
 IP_3 : Z_3(T) = \min \sum_{i \in V} \sum_{j \in V} c_{ij}x_{ij} \\
 \text{s.t. (b) } \sum_{i \in V/S} x_{ij} \geq y_j, \quad \forall S \subset V, \forall j \in S \tag{4.4}
 \end{aligned}$$

$$\begin{aligned}
 \text{(e) } x_{ij} \in N^+, \quad \forall i, j \in V \\
 LP_3 : \left\{ Z_3^*(T) = \min \sum_{i \in V} \sum_{j \in V} c_{ij}x_{ij} \mid \text{(b), } x_{ij} \geq 0 \right\} \tag{4.5}
 \end{aligned}$$

Lemma 2 $Z_3^*(T) \leq \sum_{i,j \in V} c_{ij}\bar{x}_{ij}$.

Proof $\forall i, j \in V, \bar{x}_{ij} = \rho x_{ij}^* \geq 0$ holds. Due to this, constraint (e) is satisfied. For the definition of \bar{x}_{ij} , we have $\sum_{i \in V/S} \bar{x}_{ij} = \rho \sum_{i \in V/S} x_{ij}^* \geq \rho y_j^* \geq \rho \frac{1}{\rho} = 1$. Hence, constraint (b) is satisfied and $(\bar{x}_{ij})_{i,j \in V}$ is a feasible solution to LP_3 , and $Z_3^*(T) \leq \sum_{i,j \in V} c_{ij}\bar{x}_{ij}$. This proves the theorem. \square

Theorem 2 Algorithm LP-CST is $\gamma\rho$ approximation, where γ is the approximation ratio of Steiner tree problem.

Proof According to Theorem 2 of Wolsey (1980), there is an approximation algorithm Z with $\frac{Z}{Z_3^*(T)} \leq \gamma$ if $\frac{Z}{Z_3(T)} \leq \gamma$ holds. Thus, from $l(T) \leq \gamma Z_3(T)$, we get $l(T) \leq \gamma Z_3^*(T)$ holds.

$$l(T) \leq \gamma Z_3^*(T) \leq \gamma \sum_{i,j \in V} c_{ij}\bar{x}_{ij} = \gamma\rho Z_2^* \leq \gamma\rho Z_2 \leq \gamma\rho Z_1 \tag{4.6}$$

This proves the theorem. \square

From all above, we get a covering Steiner tree, with the approximation ratio as $\gamma\rho$, where γ is at least 1.39 in general graph (Byrka et al. 2010). We can also approximate CST by solving LP_1 directly to select a subset of vertices, computing a tour to cover the selected vertices by Christofides’ algorithm for TSP (Christofides 1976), and deleting the redundant copy of edges in the TSP tour to get a tree. Similar as the analysis of LP-CST algorithm, we can derive that $l(T) \leq \frac{3}{2}Z_3^*(T) \leq \frac{3}{2} \sum_{i,j \in V} c_{ij}\bar{x}_{ij} = \frac{3}{2}\rho Z_1^* \leq \frac{3}{2}\rho Z_1$.

4.2 CoverTreeTraversal

For the algorithm CoverTreeTraversal, we first get a CST from LP-CST, called T . Then, the salesman will try to execute a DFS traversal to visit all the customer vertices on the CST. During the traversal, the salesman will encounter at most k blockages which will cut the original CST into at most $k + 1$ subtrees. Let Ω be the collection of subtrees which are not visited. Initially $\Omega = \{T\}$. The salesman will try to execute DFS traversal on each subtree, and connect each subtree via a shortest path chain, which we call a *jump*. We treat the traversal between two jumps as an *iteration*.

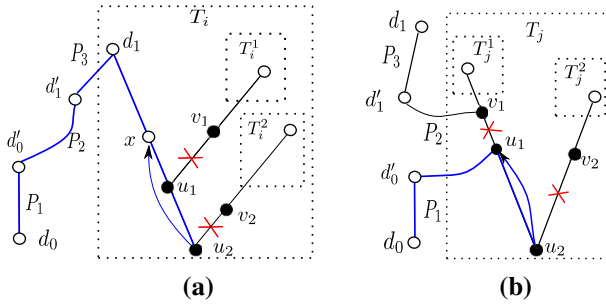


Fig. 2 LB-SubtreeTraversal 1. The salesman’s trace in Case 1 in blue line. $e_1 = (u_1, v_1)$ cuts the T_i into two subtrees. The part connected to u_1 is the new T_i on which the salesman continues the DFS traversal, and the other part T_i^1 is added into Ω after chopping the leaves which are not customer vertex, where $v_1 \notin D$ are chopped. After $e_2 = (u_2, v_2) \in T_i$ is encountered, add subtree T_i^2 in Ω and return to $x \in T_i$, which is new d_0 in the next iteration. The trace in Case 2.1.1 (blue line) includes no customer vertex on T_j , and the salesman returns to d_0 and starts a new iteration (Color figure online)

Let Q denote all the customer vertex unvisited. We remark that the service depot is always considered unvisited to force the salesman to return to the service depot, $s \in Q$.

In one typical iteration of the algorithm, the salesman stands at d_0 , and computes the shortest paths from all vertices in $a[d_0]$ to all the vertices in $a[d_1]$, where $d_1 \in Q$ and in some subtree $T_i \in \Omega$, i.e. $d_1 \in Q \cap T_i$. Denote a shortest one of all the above shortest paths as $P(d'_0, d'_1)$, where $d'_0 \in a[d_0]$ and $d'_1 \in a[d_1]$. Please note that d'_0 and d'_1 may not be on any subtree in Ω and they may already be served. We define the shortest path chain as $\mathbb{P} = P_1 : P_2 : P_3$, where $P_1(d_0, d'_0)$, $P_2(d'_0, d'_1)$ and $P_3(d'_1, d_1)$. Please note that $l(P_1), l(P_3) \leq L$.

During the traversal, once the salesman visits a customer in Q , called v , update $Q = Q - a[v], \forall v \in Q$. If the salesman arrives at d_1 , then executes the DFS traversal on the subtree T_i ; otherwise if the salesman is blocked on \mathbb{P} , the salesman travels depending on the location of the blockage on \mathbb{P} .

The details are listed as follows:

Case 1 A new edge blockage $e_1 = (u_1, v_1)$ is revealed after the salesman reaches d_1 (Fig. 2a, and the salesman is considered reaching d_1 . Note that $d_0 = d_1 = s$ in the first iteration. If $e_1 \in T_i$, then T_i is broken down into two smaller subtrees by removing the edge and subsequently updated to be one of the two smaller subtrees that contains d_1 . The other subtree, say T_i^1 is added into Ω after chopping the leaves not in D . The salesman continues his DFS traversal on the subtree T_i until all the customer vertex in T_i has been visited, and he stands on some vertex in D , say x . Note that during the DFS traversal, new edge blockages can be revealed. When a new blocked edge is in T_i , it is processed in the same manner repeatedly. The iteration ends with the salesman standing at x .

Case 2 A new edge blockage $e_1 = (u_1, v_1)$ is revealed on \mathbb{P} .

Case 2.1 If e_1 is on another subtree $T_j \in \Omega$, update T_j as the small subtree that contains u_1 , and add the other subtree T_j^1 that contains v_1 into Ω . The salesman executes DFS

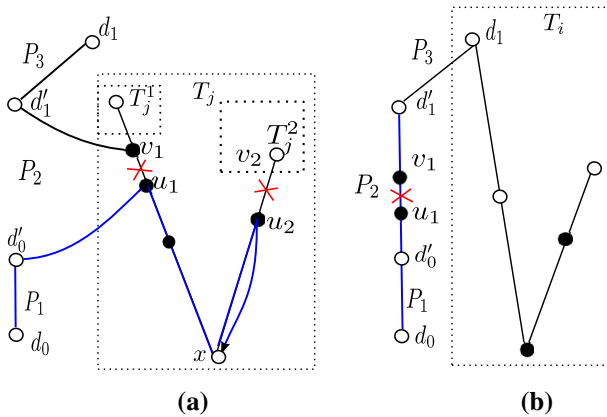


Fig. 3 LB-SubtreeTraversal 2. The salesman does DFS traversal on T_j from u_1 , and the trace in Case 2.1.2 (blue line) includes one customer vertex x on T_j . x is the d_0 in the new iteration. In Case 2.2, the salesman retraces to d_0 and starts a new iteration (Color figure online)

traversal on T_j from u_1 and returns to u_1 at the end of traversal. Note that during the DFS traversal, new edge blockages can be revealed. When a new blocked edge $e_2 = (u_2, v_2)$ is in T_j , it is processed in the same manner repeatedly as in Case 1.

Case 2.1.1 If T_j includes no destination at the end of DFS traversal (see Fig. 2b), the salesman returns to d_0 , the iteration terminates.

Case 2.1.2 When the salesman visited the last destination on T_j (see Fig. 3a), called x , update x as d_0 , the iteration terminates.

Case 2.2 If e is not in any subtree in Ω (see Fig. 3b), the salesman returns to d_0 , update $a[d_0]$, the iteration terminates.

Lemma 3 In each iteration, $l(P_2) \leq 0.5OPT$, where $P_2 = P(d'_0, d'_1)$.

Proof For an arbitrary vertex $d_0 \in D$, according to the definition of $a[d_0]$, the offline salesman has to visit one of the vertex in $a[d_0]$. Similarly, for $d_1 \in Q$, the offline salesman has to visit one of the vertex in $a[d_1]$. Since P_2 is the shortest path connected $a[d_0]$ and $a[d_1]$, $l(P_2) = l(P(d'_0, d'_1)) = \min\{l(P(i, j)) | i \in a[d_0], j \in a[d_1]\} \leq 0.5OPT$. This proves the lemma. \square

Recall that P is a shortest path chain including P_1, P_2 and P_3 , where $l(P_1), l(P_3) \leq L$ and $l(P_2) \leq 0.5OPT$. Hence, $l(P) \leq 0.5OPT + 2L$.

The travel time of the salesman including two parts, one is the traversal time on the subtrees, the other is the jumping time on P .

Lemma 4 In an iteration, the jumping time of a salesman is

$$\begin{cases} l(P) & \text{Case 1} \\ 2l(P) & \text{Case 2.1.1} \\ l(P) & \text{Case 2.1.2} \\ 2l(P) & \text{Case 2.2} \end{cases}$$

Proof The salesman travels along \mathbb{P} and arrives at d'_1 in Case 1, it is clear that the jumping time is $l(\mathbb{P})$. In Case 2.1.1, the salesman encounters a blockage on some other subtree T_j , and there is no unvisited vertex in T_j , the salesman returns to d_0 , the jumping time is at most $2l(\mathbb{P})$. In Case 2.1.2, the salesman encounters a blockage on some other subtree T_j , and there is some vertex in Q , called $x \in T_j \cap Q$, the salesman stop at x , the jumping time is at most $l(\mathbb{P})$. The salesman also returns to d_0 in Case 2.2, the jumping time is at most $2l(\mathbb{P})$.

This proves the lemma. □

Theorem 3 *For online CSP, CoverTreeTraversal is polynomial time and the competitive ratio is $k + \alpha$, where $\alpha = 0.5 + \frac{4kL}{OPT} + 2\gamma\rho$.*

Proof Solving the linear programming to get the steiner vertex set is proved polynomial. The approximation algorithm for Steiner tree is based on linear programing, and it is also polynomial. Computing the shortest path between every pair of vertex in Q is at most $O(|V|^3)$, and computing \mathbb{P} in each iteration needs $O(k|V|^3)$, the entire DFT traversal is $O(|V|^2)$. Hence, algorithm CoverTreeTraversal is polynomial time.

The traveling includes two parts, one is the DFS traversal in T and the other is the jumping on \mathbb{P} . From the two cases in algorithm CoverTreeTraversal, the salesman travels on every edge in $T_i \in \Omega$ at most twice, $t_d \leq 2l(T) \leq 2\gamma\rho OPT$. Due to Lemma 4, the jumping time per subtree is at most $2l(\mathbb{P})$. Note that there will be at most k times jumping for encountering blockages and one more to go back to s . The last jumping costs at most $l(P)$. Hence, the total jumping time is $t_j \leq 2kl(\mathbb{P}) + l(P)$.

The total cost of CoverTreeTraversal algorithm is $t = t_d + t_j \leq 2l(T) + k(4L + OPT) + 0.5OPT \leq (k + 0.5 + 2\gamma\rho)OPT + 4kL$.

Hence, the competitive ratio is

$$c_A = \frac{t}{OPT} \leq \frac{(k + 0.5 + 2\gamma\rho)OPT + 4kL}{OPT} = k + 0.5 + \frac{4kL}{OPT} + 2\gamma\rho = k + \alpha, \tag{4.7}$$

where $\alpha = 0.5 + \frac{4kL}{OPT} + 2\gamma\rho$ and $\gamma \geq 1.39$. This proves the theorem. □

When $\frac{L}{OPT} \rightarrow 0$, our algorithm is near optimal.

5 Online CSP with service cost

In this section, we will study a more practical scenario considering the service cost for the covered customers. The service cost may be charged by the third-party convenient store per parcel, or comes from the waiting time for customers who are not on the tour. Hence, the total online delivery time is the sum of the travel time and the service cost.

The service cost can be seen as the penalty charged. Once a vertex in D is visited, there is no penalty; if it is covered, it is charged w_i depends on weight(number) of orders on the vertex. It comes to prize collecting TSP that has to follow the covering constraint.

5.1 Lower bound

Theorem 4 For online CSP with service cost, $\beta \geq \min \left\{ \frac{1}{1+(k+2)L}k + 1, \frac{1}{1+0.5(k+2)w}k + 1 \right\}$.

Proof The proof is similar to Theorem 1. Consider the instance in Fig. 1. Suppose that all the vertices in D has the same amount of orders, then the penalty w is uniform. If $\epsilon > L$, the salesman has to visit every customer in D . $c_{\mathcal{A}} \geq \frac{1}{1+(k+2)L}k + 1$ holds for an arbitrary algorithm \mathcal{A} . If $\epsilon \leq L$, for $2\epsilon < w$, it is cheaper to visit the vertice in D , and the online delivery time is $\mathcal{A}(I) \leq 2k + 2 + 2\epsilon + 2\epsilon + 2(k + 1)\epsilon$ while $\text{OPT} = 2 + 2\epsilon + 2(k + 1)\epsilon$, and derive $c_{\mathcal{A}} > k + 1$ as $\epsilon \rightarrow 0$; for $2\epsilon \geq w$, the choice is to cover the vertices in D , and the online delivery time is $\mathcal{A}(I) \leq 2k + 2$ while $\text{OPT} = 2 + 4\epsilon + (k + 1)w$, and derive $c_{\mathcal{A}} > \frac{1}{1+0.5(k+2)w}k + 1$.

Hence, $\beta \geq \min \left\{ \frac{1}{1+(k+2)L}k + 1, \frac{1}{1+0.5(k+2)w}k + 1 \right\}$. This proves the theorem. \square

Please note that $\beta \geq \frac{1}{1+(k+2)L}k + 1$ for $w = 0$, which is same with the case without service cost.

In the following subsection, we will show LP-CST also woks when the service cost is considered, and then apply the CoverTreeTraversal algorithm in Sect. 4.2 to solve the problem with service cost.

5.2 Competitive analysis

For the problem with service cost, we give the integer programming formulation and do the linear programming relaxation to lead to the approximate initial solution. For each vertex $i \in D$, the penalty w_i is charged if it is covered without visiting.

Recall the covering matrix A in Sect. 4.1 with covering range L . Again let $\rho = \max_{i \in D} \{ |a[i]| \}$, $s = 0$ as the depot. The service cost for vertex $i \in D$ is $(1 - y_i)w_i$, where $y_i = 1$ means this vertex is visited, and the total service cost is $t_s = \sum_{i \in D} (1 - y_i)w_i$. IP_0 (Eq. 5.1) is the integer programming of CSP with service cost. Omitting the constraint (a) leads to integer programming of CST with service cost IP'_1 (Eq. 5.2). The linear relaxation of IP'_1 induces LP'_1 . We get LP_2 (Eq. 4.3) from LP'_1 by leaving the service cost. Then we can use the LP-CST in Sect. 4.1 to get a CST.

$$IP_0 : \left\{ Z_0 = \min \sum_{i \in V} \sum_{j \in V} c_{ij}x_{ij} + \sum_{i \in D} (1 - y_i)w_i \mid (a) - (e) \right\} \tag{5.1}$$

$$IP'_1 : \left\{ Z'_1 = \min \sum_{i \in V} \sum_{j \in V} c_{ij}x_{ij} + \sum_{i \in D} (1 - y_i)w_i \mid (b) - (e) \right\} \tag{5.2}$$

$$LP'_1 : \left\{ Z^*_1 = \min \sum_{i \in V} \sum_{j \in V} c_{ij}x_{ij} + \sum_{i \in D} (1 - y_i)w_i \mid (b) - (c), 0 \leq y_i \leq 1, 0 \leq x_{ij} \right\} \tag{5.3}$$

Clearly, $Z'_1 \leq Z_0$ for the lack of (a). $Z_1^* < Z'_1$ since LP_1' is the linear programming relaxation of IP_1' . Omitting the service cost, we get LP_2 in Eq. 4.3, with $Z_2^* = Z_2^* = \min \sum_{i \in V} \sum_{j \in V} c_{ij}x_{ij}$.

LP-CST algorithm will select a subset of vertices, say T , to construct a Steiner tree by a γ -approximation algorithm. Recalling LP_3 and IP_3 (Eq. 4.4 and 4.5), the total cost of the CST is at most $\gamma Z_3^*(T) + \alpha \sum_{i \in D} (1 - y_i)w_i$.

Theorem 5 *The LP-CST algorithm is $\gamma\rho$ approximation, where γ is an approximation ratio of Steiner tree.*

Proof Recall Eq. 4.6, we have

$$\begin{aligned} t &\leq \gamma Z_3^*(T) + \sum_{i \in D} (1 - \bar{y}_i)w_i \\ &\leq \gamma \sum_{i,j \in V} c_{ij}\bar{x}_{ij} + \sum_{i \in D} (1 - \bar{y}_i)w_i \\ &\leq \gamma \sum_{i,j \in V} \rho c_{ij}x_{ij}^* + \frac{\rho}{\rho - 1} \sum_{i \in D} (1 - y_i^*)w_i \\ &\leq \gamma\rho Z_1^* \leq \gamma\rho Z_1 \leq \gamma\rho Z_0 \end{aligned}$$

This proves the theorem. □

We apply the CoverTreeTraversal algorithm to the online CSP with service cost. With similar analysis, we can get the following theorem.

Theorem 6 *For Covering sTSP with service cost, the CoverTreeTraversal algorithm is polynomial time and $k + \alpha$ -competitive, where $\alpha = \frac{4L}{OPT} + 2\gamma\rho$.*

6 Conclusion

We studied the *covering salesman problem* with online edge blockages, which a good match to the instant delivery routing problem in urban traffic network. For the version without service cost, we present a lower bound $\frac{1}{1+(k+2)L}k + 1$, the online algorithm is proved to be $k + \alpha$ competitive, where $\alpha = 0.5 + \frac{4kL}{OPT} + 2\gamma\rho$. We also extend the analysis to the version with service cost. The competitive ratio is still at most $k + \alpha$, and the lower bound is $\min \left\{ \frac{1}{1+(k+2)L}k + 1, \frac{1}{1+0.5(k+2)w}k + 1 \right\}$, where w is the uniform penalty for each vertex in D . When $\frac{L}{OPT} \rightarrow 0$, our algorithm is near optimal.

A more careful analysis for the competitive ratio of CoverTreeTraversal can considered in the future research. We can also assume that there is no blockage between the customer vertex and the third-party vertex since they are always in a small neighborhood. With these assumption, the algorithm will have a better performance. Another interesting work is to consider a more general service cost function, which includes two parts. One part is that the fixed cost to active a third-party service site and the other part is the changeable cost charged per parcel. In this case, the service cost is the sub modular function, which may be more interesting.

Acknowledgements Zhang and Xu would like to acknowledge the financial support of NSFC Grants Nos. 71601152 and 71732006. Zhang is also supported by Grant Nos. 2016M592811 and 2015T81040 from China Postdoctoral Science Foundation.

References

- Bienstock D, Goemans MX, Simchi-Levi D, Williamson D (1993) A note on the prize collecting traveling salesman problem. *Math Program* 59(1–3):413–420
- Byrka J, Grandoni F, Rothvo T, Sanit L (2010) An improved ip-based approximation for steiner tree. In: *ACM Symposium on theory of computing*, pp 583–592
- Chisman JA (1975) The clustered traveling salesman problem. *Comput Oper Res* 2(2):115–119
- Christofides N (1976) Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document
- Current JR, Schilling DA (1989) The covering salesman problem. *Transp Sci* 23(3):208–213
- Current JR, Schilling DA (1994) The median tour and maximal covering tour problems: formulations and heuristics. *Eur J Oper Res* 73(94):114–126
- Flores-Garza DA, Salazar-Aguilar MA, Ngueveu SU, Laporte G (2015) The multi-vehicle cumulative covering tour problem. *Ann Oper Res* 258:1–20
- Garey MR, Johnson DS (1990) *Computers and intractability; a guide to the theory of NP-completeness*. W. H. Freeman & Co., New York
- Gendreau M, Laporte G, Semet F (1997) The covering tour problem. *Oper Res* 45(4):568–576
- Goemans X, Williamson Michel, David P (1992) A general approximation technique for constrained forest problems. *SIAM J Comput* 24(2):296–317
- Golden B, Naji-Azimi Z, Raghavan S, Salari M, Toth P (2012) The generalized covering salesman problem. *Inf J Comput* 24(4):534–553
- Gutin G, Punnen AP (eds) (2002) *The traveling salesman problem and its variations, volume 2 of Combinatorial optimization*. Kluwer Academic Publishers, Dordrecht
- Ha MH, Bostel N, Langevin A, Rousseau LM (2013) An exact algorithm and a metaheuristic for the multi-vehicle covering tour problem with a constraint on the number of vertices. *Eur J Oper Res* 226(2):211–220
- Hachicha M, Hodgson MJ, Laporte G, Semet F (2000) Heuristics for the multi-vehicle covering tour problem. *Comput Oper Res* 27(1):29–42
- Laporte G, Martello S (1990) The selective traveling salesman problem. *Discret Appl Math* 26(2–3):193–207
- Liao C-S, Huang Y (2014) The covering canadian traveller problem. *Theor Comput Sci* 530:80–88
- Naji-Azimi Z, Renaud J, Ruizbc A (2012) A covering tour approach to the location of satellite distribution centers to supply humanitarian aid. *Eur J Oper Res* 222(3):596–605
- Papadimitriou CH, Yannakakis M (1991) Shortest paths without a map. *Theor Comput Sci* 84(1):127–150
- Safra S, Schwartz O (2006) On the complexity of approximating tsp with neighborhoods and related problems. *Comput Complex* 14(4):281–307
- Slavik P (1997) The errand scheduling problem. CSE technical report 97-02
- Tricoire F, Graf A, Gutjahr WJ (2012) The bi-objective stochastic covering tour problem. *Comput Oper Res* 39(7):1582–1592
- Westphal S (2008) A note on the k-canadian traveller problem. *Inf Process Lett* 106(3):87–89
- Wolsey LA (1980) Heuristic analysis, linear programming and branch and bound. *Math Program Study* 13(13):121–134
- Yang JJ, Jiang JR, Lai YL (2014) A decreasing k-means algorithm for the disk covering tour problem in wireless sensor networks. In: *IEEE international conference on parallel and distributed systems*, pp 906–910
- Zhang H, Tong W, Xu Y, Lin G (2015) The steiner traveling salesman problem with online edge blockages. *Eur J Oper Res* 243(1):30–40
- Zhang H, Tong W, Xu Y, Lin G (2016) The steiner traveling salesman problem with online advanced edge blockages. *Comput Oper Res* 70:26–38
- Zhang H, Xu Y, Qin L (2013) The k-canadian travelers problem with communication. *J Comb Optim* 26(2):251–265
- Zhu Z, Xu Y, Liu C (2003) The covering canadian traveller problem. *J Syst Eng* 18:261–270 (in Chinese)