An LP-based approximation algorithm for the generalized traveling salesman path problem

Jian Sun[a], Gregory Gutin[b], Ping Li[c], Peihao Shi[d], Xiaoyan Zhang[e,*]

[a] *Department of Operations Research and Information Engineering, Beijing University of Technology, Beijing 100124, P.R. China*
[b] *Department of Computer Science Royal Holloway, University of London Egham, Surrey, TW200EX, UK*
[c] *Huawei Technologies Co. Ltd. Theory Lab, Central Research Institute, 2012 Labs, Hongkong 9990777, P. R. China*
[d] *Nanjing Kinghua Operations Research and Artificial Intelligence Industrial Technology Research Institute, Jiangsu 210035, P. R. China*
[e] *School of Mathematical Science and Institute of Mathematics, Nanjing Normal University, Jiangsu 210023, P.R. China*

Abstract

The traveling salesman problem (TSP) is one of the classic research topics in the field of operations research, graph theory and computer science. In this paper, we propose a generalized model of traveling salesman problem, denoted by generalized traveling salesman path problem. Let $G = (V, E, c)$ be a weighted complete graph, in which c is a nonnegative metric cost function on edge set E, i.e., $c : E \to \mathbb{R}^+$. The traveling salesman path problem aims to find a Hamiltonian path in G with minimum cost. Compared to the traveling salesman path problem, we are given extra vertex subset V' and edge subset E' in the problem proposed in this paper; its goal is to construct a path which traverses all the edges in E' while only needs to visit each vertex in V' exactly once. Based on integer programming, we give a mathematical model of the problem, and design a $\frac{1+\sqrt{5}}{2}$-approximation algorithm for the problem by combining linear programming rounding strategy and a special graph structure.

Keywords: TSP, Generalized traveling salesman path problem, Approximation algorithm, LP rounding

*Corresponding author Xiaoyan Zhang

Email addresses: B201806011@emails.bjut.edu.cn (Jian Sun), g.gutin@rhul.ac.uk (Gregory Gutin), liping129@huawei.com (Ping Li), sph@kinghua.com.cn (Peihao Shi), zhangxiaoyan@njnu.edu.cn (Xiaoyan Zhang)

A preliminary version of this paper appeared in Proceedings of the 15^{th} Annual International Conference on Combinatorial Optimization and Applications (COCOA), pp. 641-652, 2021.

1. Introduction

The traveling salesman problem (abbreviated as TSP), as well as its related generalized model, is one of the hot research topics in many fields, such as operations research, graph theory, and computer science [1, 3, 13, 19, 23, 26]. In the TSP, a salesman must visit several cities exactly once and return to the city where he starts. Traveling between different cities requires a certain cost, the salesman wants the total traveling cost to be minimized. The traveling salesman problem can be defined by graph theory as follows: given a complete graph $G = (V, E)$ with an edge-weight (or edge-cost) function $c : E \to \mathbb{R}_+$, the object is to find a Hamiltonian cycle with minimum cost in G. That is, TSP is equivalent to finding the minimum-weighted Hamiltonian cycle on the corresponding weighted undirected graph, i.e., the cycle which passes through all vertices once and has the smallest sum of weights. Dantzig et al. [8] proposed the earliest mathematical programming formulation of TSP and designed a feasible solution for the instance consisting of 49 cities in the America.

Unfortunately, the traveling salesman problem is NP-hard even if the cost function satisfies the metric property [23]. Heuristic algorithms and approximation algorithms are commonly used to solve this kind of problem. Heuristic algorithms can usually construct a feasible solution, but cannot theoretically characterize the quality of the solution. In this paper, we mainly focus on designing approximation algorithm, whose performance is measured by the approximation ratio.

For metric TSP (i.e., the edge-cost function is metric), the most well-known algorithm is the 1.5-approximation algorithm independently designed by Christofides and Serdjukov [6, 35]. This algorithm has not been improved for more than forty years, until Karlin et al. proposed a $(1.5 - \epsilon)$-approximation algorithm in which ϵ is a fairly small constant greater than 10^{-36} [22]. For TSP under Euclidean [4], low-genus [9, 10] or planar metrics [5, 17, 24], polynomial-time approximation schemes (PTAS) have been found. With regard to negative results, it has been proved that there is no polynomial-time algorithm with an approximation ratio better than $\frac{220}{219}$ for this problem unless unless P=NP [29]. Furthermore there is no polynomial-time approximation algorithm with constant ratio if the edge-cost function doesn't satisfy the triangle inequality, under the same assumption [30].

The traveling salesman problem with the special metric cost function has also received extensive attention, such as the graph-TSP. In the graph-TSP, the cost of an edge is defined by the minimum number of edges along the path of its two endpoints in the underlying graph. Gharan, Saberi and Singh [15] designed an approximation algorithm with ratio $(1.5 - \epsilon)$ in which $\epsilon > 0$. Their algorithm framework is quite similar to Christofides' algorithm, but they used different strategies for selecting spanning trees. Specifically, they first proposed the linear programming relaxation of the problem, and then selected the spanning tree randomly by sampling from the maximum entropy distribution defined by linear programming relaxation, except in some cases where the solution of LP is almost integral. Mömke and Svensson [27] proposed a 1.461-approximation algorithm

based on a new use of matching that allows certain edges to be removed from the matching. Mucha [28] proposed an improved analysis of the method proposed by Mömke and Svensson, then the bound of the approximation ratio is proved to be $\frac{13}{9}$. Besides, for the travelling salesman path problem in graphic metrics, he proposed an upper bound of $\frac{19}{12}+\epsilon$ in which ϵ is an arbitrary constant greater than 0. For the graph-TSP and other related problems, Sebö and Vygen [34] obtained several new results. Particularly, they designed a better algorithm for graph-TSP with an approximation ratio of 1.4.

The traveling salesman path problem (TSPP) is another important generalization model of TSP, many scholars have studied this problem and obtained a series of achievements [14, 20, 21, 25, 31, 33, 37]. Compared with TSP, in this problem we are given an additional origin and destination, the goal is to find a path from the starting point to the destination which passes through all the other cities exactly once with minimum cost. Hoogeveen [21] presented the first constant factor approximation algorithm for TSPP, where the approximation ratio of this algorithm is $\frac{5}{3}$. AN et al. [2] improved the algorithm for the first time in nearly two decades, and combining the liner programming relaxation with a special kind of graph structure (called narrow cut), they designed a $\frac{1+\sqrt{5}}{2}$-approximation algorithm for the TSPP. A series of exciting progress has been made in approximation algorithms for TSPP based on the critical chain structure of narrow cuts [16, 31, 32, 37, 38]. These developments culminated in Zenklusen's recent breakthrough results [39]. Combining the idea of dynamic programming introduced by Traub and Vygen [37] with Karger's breakthrough in approaching the minimum number of cuts, Zenklusen [39] broke through the limitations of the concept of narrow cuts and designed a better algorithm based on this result, with an approximation ratio of 1.5.

Let $G=(V,E,c)$ be the weighted complete graph defined above, V' and E' represent the subset of vertices and edges, respectively. In this paper, we consider the problem defined as follows: given two distinct vertices s (source) and t (terminal), the goal is to construct an s-t path with minimum sum of weights that traverses all the edges in E' and visits each vertex of V' exactly once. We refer to this problem as the generalized traveling salesman path problem, which is abbreviated as GTSPP for convenience. Based on the linear programming relaxation of this problem, we propose a spanning tree selection strategy. After making appropriate modifications to the spanning tree, we can construct a feasible solution to the original problem. Based on the above strategies, we design an approximation algorithm for GTSPP. In the process of analyzing the approximation ratio of the algorithm, we further discuss the related characteristics of graph structure, especially the narrow cut. At the beginning, we prove that the approximation ratio of this algorithm is 1.6577, and we subsequently improved the approximation ratio to $\frac{1+\sqrt{5}}{2}$ by further observation and using more rigorous analysis which improves our previous work [36].

The remainder of this paper is organized as follows. In Section 2, we present some preliminary conclusions and LP model of GTSPP. We design an algorithm

to solve GTSPP and analyze it in Section 3. In Section 4, we propose a tighter analysis on the approximation ratio. We present our conclusions in Section 5.

2. Preliminaries

By investigating the structure characteristics of this problem, we characterize the sufficient and necessary conditions for the existence of feasible solutions: feasible solutions exist if and only if E' consists of disjoint paths and the degree of s and t in E' is at most 1. Therefore, edges in $E \setminus E'$ associated with an internal vertex in one of these paths can never be used, thus all such edges can be removed, and each path in E' can be treated as an edge from the starting point to the end point. In the presence of feasible solutions, we may wish to assume that the edges in E' are pairwise disjoint.

Since the edge-weight function satisfies triangle inequality, then vertices that are neither in V', nor s or t, or endpoints in E' can be deleted. Otherwise we can shortcut any tour that visits such vertices without increasing the total cost. Based on the above observations, we construct a new graph $G_1 = (V_1, E_1)$ and consider solving the original problem on it. $G_1 = G[V_1]$ is the induced subgraph of G on V_1 in which $V_1 = V' \cup V(E') \cup \{s, t\}$. In order to facilitate the analysis of the feasibility of the algorithm in the following, it can be assumed that $\{s, t\} \cap V(E') = \emptyset$. This assumption is reasonable. According to the symmetry, we might as well take s as an example to explain, if s is the endpoint of one edge in E', that is, there is an edge $e \in E'$, and its endpoints are s and s_1, then any feasible solution of GTSPP can be divided into two parts: s-s_1 and an s_1-t path that visits each vertex of V' exactly once and traverses $E' \setminus \{e_1\}$. Thus, we just need to find the s_1-t path above in this case.

Before presenting the model and algorithm of TSPP, we first introduce some notations and terminology used in this paper. For a partition $(U, \bar{U})$ in which $U \subset V$, the corresponding set of cut edges is denoted by $\delta(U) := \{\{u, v\} | u \in U, v \in \bar{U}\}$. The partition $(U, \bar{U})$ is called an s-t cut if $|U \cap \{s, t\}| = 1$; otherwise it is called the non-separating cut.

Inspired by the Held-Karp relaxation of TSP and TSPP, we propose the

LP-relaxation of GTSPP as follows:

$$
\begin{aligned}
\min \quad & \sum_{e \in E_1} c_e x_e & & \\
\text{s.t.} \quad & \sum_{e \in \delta(S)} x_e \geq 1 & & \forall S \subsetneq V_1, |S \cap \{s,t\}| = 1 \\
& \sum_{e \in \delta(S)} x_e \geq 2 & & \forall S \subsetneq V_1, |S \cap \{s,t\}| \neq 1, S \neq \emptyset \\
& \sum_{e \in \delta(\{v\})} x_e = 2 & & \forall v \in V_1 \setminus \{s,t\} \\
& \sum_{e \in \delta(\{s\})} x_e = \sum_{e \in \delta(\{t\})} x_e = 1 & & \\
& x_e = 1 & & \forall e \in E' \\
& x_e \geq 0 & & \forall e \in E_1 \setminus E',
\end{aligned}
\tag{1}
$$

The first and second constraints ensure that there are no sub-path and sub-tour. Though this LP-relaxation contains exponential multiple constraints, we can still solve it in polynomial time by using the ellipsoid method in which the separation oracle is a minimum cut problem [18].

Definition 1. *Let Q and J be the vertex subset and edge subset, respectively. For the spanning subgraph $G' = (V, J)$, if its set of odd-degree vertices happens to be Q, then J is a Q-join.*

In fact, the matching with vertex set Q is a special kind of Q-join (all vertices have degree 1). To illustrate the concept of Q-join more intuitively, we present a simple instance in Figure 1:

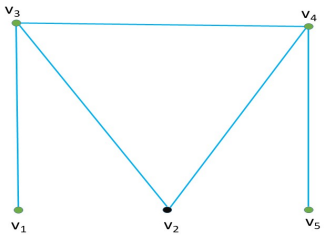

Figure 1: An instance of Q-join, $Q = \{v_1, v_3, v_4, v_5\}$.

Definition 2. *(wrong-degree set) For a spanning tree T and two specified distinct vertices s and t, the wrong-degree set is defined as $odd(T) \triangle \{s,t\} := (odd(T) \setminus \{s,t\}) \cup (\{s,t\} \setminus odd(T))$, where $odd(T)$ denotes the set of vertices with odd degree in T.*

During the analysis of the algorithm, we need to analyze the relationship between the weight of Q-join and the optimal value of (1), for which we give the following terms in advance:

Definition 3. *[12] (Q-join polytope)*

$$P^{\uparrow}_{Q-join} := \{y \in \mathbb{R}^{E}_{\geq 0} | y(\delta(C)) \geq 1 \; \forall C \subset V \; and \; |C \cap Q| \; is \; odd\}.$$

Given a vertex subset Q, J is assumed to be the minimum Q-join, then $\forall y \in P^{\uparrow}_{Q-join}$, $c(J) \leq c(y)$ holds.

Definition 4. *(spanning tree polytope [11]) The spanning tree polytope of $G = (V, E)$, denoted by $P_{ST}(G)$, can be defined as follows:*

$$\begin{cases} x(E) = |V| - 1, \\ x(E[S]) \leq |S| - 1, \forall S \subset V, S \neq \emptyset \\ x \in \mathbb{R}^{E}_{\geq 0}, \end{cases} \tag{2}$$

where $E[S]$ represents the set of edges whose end-vertices are both in S.

Lemma 5. *Let $\mathcal{D}$ represent the feasible region of (1), then we can obtain that $\mathcal{D} \subset P_{ST}(G_1)$.*

Proof. Let x be a feasible solution of (1), i.e., $x \in \mathcal{D}$, thus it can be derived that

$$\begin{aligned} x(E_1) &= \frac{1}{2} \sum_{v \in V_1} d_{G_1}(v) \\ &= \frac{1 + 1 + 2(|V_1| - 2)}{2} \\ &= |V_1| - 1 \end{aligned}$$

and

$$x(E_1[S]) = \frac{1}{2} \left(\sum_{v \in S} d_{G_1}(v) - \sum_{e \in \delta(S)} x_e \right), \forall S \subset V_1, S \neq \emptyset.$$

If $|S \cap \{s, t\}| = 1$, then

$$\begin{aligned} x(E_1[S]) &= \frac{1}{2} \left[\sum_{v \in S} d_{G_1}(v) - \sum_{e \in \delta(S)} x_e) \right] \\ &\leq \frac{1}{2} \left[1 + 2(|S| - 1) - 1 \right] \\ &= |S| - 1; \end{aligned}$$

else , we have that

$$\begin{aligned} x(E_1[S]) &= \frac{1}{2}\left[\sum_{v\in S} d_{G_1}(v) - \sum_{e\in\delta(S)} x_e)\right] \\ &\leq \frac{1}{2}(2|S|-2) \\ &= |S|-1. \end{aligned}$$

□

Let x be a given element of $\mathcal{D}$, it can be decomposed into a convex combination of several spanning trees of G_1, i.e., $x := \sum_{i=1}^{k} \gamma_i \chi_{T_i}$ in which $\{T_1, T_2, \cdots, T_k\}$ is the set of spanning tree of G_1, k is bounded by a polynomial, $\sum_{i=1}^{k} \gamma_i = 1$, and $\gamma_i \geq 0\ \forall i \in [k]$. Based on the relevant conclusions in the literature [7, 18], it can be derived that such a convex combination can be found in polynomial time.

In addition, according to the characteristics of the problem, we have the following conclusion for the spanning trees in the above convex combination of x:

Lemma 6. *$\forall e \in E'$ and $1 \leq i \leq k$, $\chi_{T_i}(e) = 1$ holds.*

Proof. Let $x := \sum_{i=1}^{k} \gamma_i \chi_{T_i}$, since $\forall x \in \mathcal{D}$ and for every edge e in E', $x_e = 1$ holds. Thus for each $e \in E'$ we have $\sum_{i=1}^{k} \gamma_i \chi_{T_i}(e) = 1$. Besides, $0 \leq \chi_{T_i}(e) \leq 1\ \forall e \in E'$, and hence

$$\sum_{i=1}^{k} \gamma_i \chi_{T_i}(e) \leq \sum_{i}^{k} \gamma_i = 1,$$

implying that for each $e \in E'$ and $i \in [k]$, $\chi_{T_i}(e) = 1$. □

3. Algorithm and analysis

The framework of algorithms to solve TSP or TSPP is usually divided into three steps: in the first step, the goal is to construct a cost-bounded spanning tree. In the second step, edges are added to the spanning tree so as to construct an Euler tour or trail (the same edge appearing more than once is denoted as a multiple edge). Some vertices may be visited multiple times in the Euler tour or trail obtained in the second step, so the third step uses the shortcut strategy to convert Euler tour or trail into a Hamiltonian cycle or path. Notice that in the framework above, the operation of adding edges only occurs in the first and second steps, and the third step only deletes edges. The main difference between GTSPP and classical TSPP is that in GTSPP, feasible solutions are not free to select edges that go through, but must traverse the edges in E'. Therefore, when designing the algorithm to solve GTSPP based on the above framework, it is necessary to ensure that after completing the first and second steps, the edges in E' have been included in the Euler tour or trail. It is required that the edges

in E' are not removed after performing the shortcut operation in the third step (this can be done by using the strategy shown in Figure 2).

A natural doubt is whether Zenklusen's algorithm [39] can be called upon to tackle this generalized problem and get the same approximation ratio. Unfortunately, the answer is no due to the graphically structured nature of the problem itself. Specifically, in the Zenklusen algorithm, a special feasible solution y is constructed according to x^*, the optimal solution of TSPP's linear programming relaxation, then the minimum spanning tree on the support of y (denoted by T) is constructed. If the algorithm is called directly, there is no guarantee that the above feasible solution y can be found in polynomial time, nor that the edges in E' will be selected by T.

Therefore, in this section we design an algorithm that is guaranteed to output a feasible solution for GTSPP but with an approximation ratio greater than 1.5.

3.1. Approximation algorithm for GTSPP

The algorithm we designed still uses the algorithm framework for TSP and TSPP: firstly, we construct a spanning tree, and then add some edges to modify the spanning tree to make it become an Euler trail. Finally, we use the shortcut strategy to obtain a Hamilton paths satisfying the constraints. The details are as follows:

Algorithm 1

Input:

An undirected graph $G = (V, E)$ with metric edge-cost function c.
Source s and terminal t.
Required vertex subset $V' \subseteq V$ and edge subset $E' \subseteq E$.

Output: A path satisfying the constraints mentioned-above.

begin:
Construct an induced subgraph of G, $G_1 := G[V_1]$ in which $V_1 = V(E') \cup \{s, t\} \cup V'$.
Solve the LP-relaxation (1) and let x^* denote the optimal solution.
Decompose x^* into a convex combination of polynomial number of spanning trees $\{T_1, T_2, \cdots, T_k\}$, that is, $x^* = \sum_{i=1}^{k} \gamma_i \chi_{T_i}$.
Sample the spanning tree T_i with probability γ_i and let T represent the final selected spanning tree.
Find the wrong-degree set of T, denote by Q.
Compute a minimum Q-join J.
Perform the shortcut operation on the Eulerian s-t trail in $T \cup J$, denote the resulting path by P.
output P.
end

Lemma 7. *Algorithm 1 will produce a feasible solution of GTSPP.*

Proof. In GTSPP, a feasible solution will traverse the edges in E' and visit each vertex in V' exactly once. Since P is an s-t Hamiltonian path of G_1, thus it

must visit each vertex of G_1 exactly once. Furthermore, V' is a subset of $V(G_1)$, so P must visit each vertex in V_1 exactly once.

The next step is to prove that P traverse the edges in E'. Note that, the Eulerian s-t trail in $T \cup J$ goes through each edge in E' exactly once. So the key is to prove that there is an appropriate shortcut operation which guarantees that the edges in E' will not be deleted in Step 11 of Algorithm 1.

If no edge is removed after performing the Step 11 of Algorithm 1, that is, there is no cycle in $T \cup J$, then P is exactly the Eulerian s-t trail, so all edges in E' are traversed by it once. Otherwise, suppose there are several cycles in $T \cup J$. Based on the reasonable assumptions we we made in Section 2: edges in E' are pairwise disjoint and $\{s, t\} \cap V(E') = \emptyset$, the cycles in $T \cup J$ can exist only in the two forms shown in Figure 2:

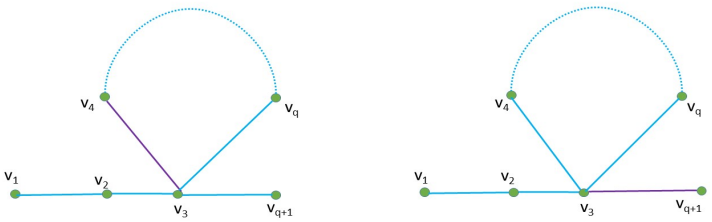

Figure 2: An example illustrating the case when the cycle in $T \cup J$ exists, in which the edges in E' are marked in purple.

In the left instance, we preform the shortcut and get a subpath v_1-v_2-v_3-v_4-$\cdots$-v_q-v_{q+1}; in the right instance, we preform the shortcut and get a subpath v_1-v_2-v_4-$\cdots$-v_q-v_3-v_{q+1}. For both two instances, we construct a feasible subpath via shortcut. □

3.2. Analysis of approximation ratio

In this subsection, we conduct a comprehensive analysis of the performance of Algorithm 1 and finally prove that its approximation ratio is no more than $\frac{9-\sqrt{33}}{2}$. Since the edge-cost function c is metric, we have that $c(P) \leq c(T \cup J)$. Therefore, we consider the upper bounds of the cost of T and J to analyze the relationship between the cost of P and the optimal value of (1). According to the selection strategy of T, it can be derived that

$$E[c(T)] = \sum_{i=1}^{k} \gamma_i c(T_i) = c(x^*).$$

Thus the next task is to analyze the cost of the minimum Q-join J, while it is difficult to directly analyze the cost of J. Motivated by the work of An et al. [2], we give the upper bound of the cost of J by analyzing the cost of a special element in Q-join polytope, which is composed of x^* and χ_T. In order to find an element in Q-join polytope with desired properties, we need to use the structural characteristics of narrow cut, which is defined as follows:

Definition 8. *[2] Let $(U, \bar{U})$ be an s-t cut, if $\sum_{e\in\delta(U)} x_e^* < 1+\tau$ for $0 < \tau \leq 1$, then $(U, \bar{U})$ is called a τ-narrow cut.*

In analyzing the cost of J, we take advantage of the concept of Q-join polytope. But in defining this concept, we use only those sets whose subsets that intersect Q contains odd number of vertices. For these sets, the following property holds:

Lemma 9. *[2] Let $(U, \bar{U})$ be an s-t cut and $|U \cap Q| \equiv 1(\mod 2)$, then the inequality $|\delta(U) \cap T| \geq 2$ holds.*

For an *s-t* cut $(U, \bar{U})$, the probability that $|U \cap Q| \equiv 1(\mod 2)$ can be formulated by the following lemma.

Lemma 10.

$$Pr[|U \cap Q| \equiv 1(\mod 2)] \leq \sum_{e\in\delta(U)} x_e^* - 1.$$

Proof. Let $|T|$ represent the number of edges in T, $|\delta(U) \cap T| = X$ and $p(x)$ denote the probability function of the random variable X, then we have that

$$\begin{aligned} E[X] - 1 &= \sum_{K=0}^{|T|} K \cdot p(X = K) - \sum_{K=0}^{|T|} p(X = K) \\ &= \sum_{K=0}^{|T|} (K-1) \cdot p(X = K), \end{aligned}$$

$|\delta(U) \cap T| \geq 1$ always holds since T is a spanning tree of G_1, thus

$$\begin{aligned} E[X] - 1 &= \sum_{K=1}^{|T|} (K-1) \cdot p(X = K) \\ &= \sum_{K=2}^{|T|} (K-1) \cdot p(X = K) \\ &\geq \sum_{K=2}^{|T|} p(X = K) \\ &= Pr[X \geq 2]. \end{aligned}$$

As Lemma 9 states, if $|U \cap Q| \equiv 1(\mod 2)$ then $|\delta(U) \cap T| \geq 2$, implying

$$\begin{aligned} Pr[|U \cap Q| \equiv 1(\mod 2)] &\leq Pr[X \geq 2] \\ &\leq E[X] - 1 \\ &= \sum_{e \in \delta(U)} x_e^* - 1. \end{aligned}$$

Since for any edge e, the probability that it belongs to T equals to x_e^*, thus the last equation holds. □

Initially, we set $y := \alpha x^* + \beta \chi_T$. Let $(U, \bar{U})$ be a cut such that $|U \cap Q|$ is odd, then $y(\delta(U)) \geq 2\alpha + \beta$ if $(U, \bar{U})$ is the non-separating cut; otherwise, we have that $y(\delta(U)) \geq \alpha \sum_{e \in \delta(U)} x_e^* + 2\beta$. In order to prove that y is in the Q-join polytope, it is necessary to ensure that $y(\delta(U)) \geq 1$ in the above two cases which can be achieved if both $2\alpha + \beta \geq 1$ and $\alpha \sum_{e \in \delta(U)} x_e^* + 2\beta \geq 1$ hold. If we set $\alpha = \beta = \frac{1}{3}$, then neither $2\alpha + \beta$ nor $\sum_{e \in \delta(U)} x_e^* + 2\beta$ is less than 1, this is because $\sum_{e \in \delta(U)} x_e^* \geq 1$. However, the approximation ratio of the algorithm is not good enough at this time. In the above analysis, when partition $(U, \bar{U})$ separating s and t, we consider the worst case, that is, $\sum_{e \in \delta(U)} x_e^* = 1$. But in fact this case is not necessary to consider, otherwise $E[|\delta(U) \cap T|] = \sum_{e \in \delta(U)} x_e^* = 1$, which implies $|U \cap Q|$ is even (Lemma 9), a contradiction.

A natural thought is whether it is possible to conduct a more detailed analysis of the assignment criteria of α and β to ensure that the value of $2\alpha + \beta$ is still not less than 1. Also, make $\alpha \sum_{e \in \delta(U)} x_e^* + 2\beta$ can be greater than or equal to 1 after modification (may not need to be modified).

Now we perform a specific analysis according to the value range of $\sum_{e \in \delta(U)} x_e^*$. If $(U, \bar{U})$ is a large-weighted odd s-t cut, then $\alpha \sum_{e \in \delta(U)} x_e^* + 2\beta \geq 1$ still holds after appropriately increasing α and decreasing β. For the odd s-t cut $(U, \bar{U})$ with low weights, we may need to add small fractions of it so as to ensure $y \in P^{\uparrow}_{Q-join}$. It should be noted that an edge may be contained by different s-t cuts, we need to handle this carefully. Using the concept of τ-narrow cut, these low-weighted s-t cuts are shown to be "almost" disjoint. This provides great convenience for us to modify y.

An, Kleinberg, and Shmoys [2] proved that the τ-narrow cuts form a chain, and there exists a partition $\{M_i\}_{i=1}^k$ of V_1 satisfying:

1. $M_1 = \{s\},\ M_k = \{t\}$;
2. $\{U | (U, \bar{U})$ is τ-narrow, $s \in U\} = \{U_i | 1 \leq i < k\}$ in which $U_i := \cup_{j=1}^{i} M_j$.

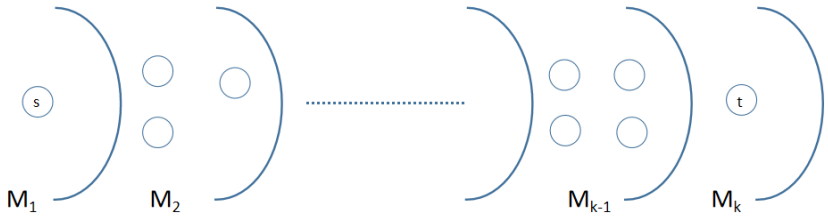

Figure 3: An example illustrating the chain structure of τ-narrow cuts.

Let $M_{\leq i} := \cup_{j=1}^{i} M_j$, $M_{\geq i} = \cup_{j=i}^{k} M_j$ and $L_i := E(M_i, M_{\geq i+1})$, it is obvious that L_i's are disjoint and $L_i \subseteq \delta(U_i)$ for all i. For each τ-narrow cut $(U_i, \bar{U}_i)$, they propose a lower bound on $\sum_{e \in L_i} x_e^*$ as follows:

$$\sum_{e \in L_i} x_e^* > \frac{1 - \tau + \sum_{e \in \delta(U_i)} x_e^*}{2} \geq 1 - \frac{\tau}{2}.$$

For each τ-narrow cut, let $h_{U_i}^*$ denote the indicator vector:

$$(h_{U_i}^*)_e = \begin{cases} x_e^*, & \text{if } e \in L_i; \\ 0, & \text{otherwise.} \end{cases}$$

We present the first result for the approximation ratio analysis of Algorithm 1.

Theorem 11. *The expectation of the cost of P is no more than* 1.6577 *times the optimal value of (1), i.e, $E[c(P)] \leq 1.6577 c(x^*)$.*

Proof. Set

$$y := \alpha x^* + \beta \chi_T + \sum_{i:|U_i \cap Q| \text{ is odd},\ 1 \leq i < k} \frac{1 - (\alpha + 2\beta)}{1 - 0.5\tau} h_{U_i}^*,$$

where $\alpha = 0.35$, $\beta = 0.3$ and $\tau = \frac{1-2\beta}{\alpha} - 1$. Then we show that $y \in P_{Q-join}^{\uparrow}$. It is obvious that $y \geq 0$, besides $y(\delta(U))$ is not less than 1 for non-separating cut $(U, \bar{U})$ since $2\alpha + \beta = 1$ still holds.

Suppose $(U, \bar{U})$ is an s-t cut such that $|U \cap Q|$ is odd, if $\sum_{e \in \delta(U)} x_e^* \geq 1 + \tau$, then

$$\begin{aligned} y(\delta(U)) &\geq \alpha \sum_{e \in \delta(U)} x_e^* + \beta |\delta(U) \cap Q| \\ &\geq \alpha(1 + \tau) + 2\beta \\ &= 1. \end{aligned}$$

Else, $\sum_{e\in\delta(U)} x_e^* < 1+\tau$, then

$$\begin{aligned} y(\delta(U)) &\geq \alpha \sum_{e\in\delta(U)} x_e^* + \beta|\delta(U)\cap Q| + \frac{1-(\alpha+2\beta)}{1-0.5\tau} h_U^*(\delta(U)) \\ &\geq \alpha + 2\beta + \frac{1-(\alpha+2\beta)}{1-0.5\tau}(1-0.5\tau) \\ &= 1. \end{aligned}$$

So we have proved that $y \in P_{Q-join}^{\uparrow}$. The next task is to analyze the cost of P.

$$\begin{aligned} E[c(P)] &\leq E(c(T)) + E[c(J)] \\ &\leq E[c(T)] + E[c(y)] \\ &= E[c(T)] + \alpha E[c(x^*)] + \beta E[c(\chi_T)] + E\left[c\left(\sum_{i:|U_i\cap Q| \text{ is odd},\ 1\leq i<k} A\cdot h_{U_i}^*\right)\right] \\ &= (1+\alpha+\beta)c(x^*) + c\left(\sum_{i=1}^{k-1} Pr[|U_i\cap Q| \text{ is odd}]\cdot A\cdot h_{U_i}^*\right) \\ &\leq (1+\alpha+\beta)c(x^*) + \tau\cdot A\cdot c\left(\sum_{i=1}^{k-1} h_{U_i}^*\right) \\ &\leq (1+\alpha+\beta+\tau\cdot A)c(x^*), \end{aligned}$$

where $A := \frac{1-(\alpha+2\beta)}{1-0.5\tau}$.

The third inequality holds based on Lemma 10, this is because for every τ-narrow cut $(U,\bar{U})$, the probability that $|U\cap Q|$ is odd is less than τ. The last inequality is obvious from the disjointness of L_i's.

Substitute the values of α, β and τ into the above formula, we get that:

$$E[c(P)] \leq 1.6577c(x^*).$$

□

Next, our goal is to analyze the algorithm more carefully to get an improved ratio. Different from the previous analysis, we use $\frac{1-\tau+\sum_{e\in\delta(U_i)} x_e^*}{2}$ to represent the lower bound of $\sum_{e\in L_i} x_e^*$, rather than $1-\frac{\tau}{2}$. Then we can improve the approximation ratio as follows:

Theorem 12. $E[c(P)] \leq \frac{9-\sqrt{33}}{2} c(x^*)$.

Proof. Set

$$y := \alpha x^* + \beta\chi_T + \sum_{i:|U_i\cap Q| \text{ is odd},\ 1\leq i<k} \frac{1-(\alpha\sum_{e\in\delta(U_i)} x_e^* + 2\beta)}{b_i} h_{U_i}^*,$$

where $\alpha = \frac{33-\sqrt{33}}{66}$, $\beta = \frac{\sqrt{33}}{33}$ and $b_i := \frac{1-\tau+\sum_{e\in\delta(U_i)} x_e^*}{2}$.

We can prove that $y \in P_{Q-join}^{\uparrow}$ in a similar way to that in Theorem 11, while there is no fundamental change in the proof, so we omit it here.

For the cost of P, it can be derived that

$$\begin{aligned}
E[c(P)] &\leq (1+\alpha+\beta)c(x^*) + c\left(\sum_{i=1}^{k-1} Pr[|U_i \cap Q| \text{ is odd}] \frac{1-(\alpha\sum_{e\in\delta(U_i)} x_e^* + 2\beta)}{b_i} h_{U_i}^*\right) \\
&\leq (1+\alpha+\beta)c(x^*) + c\left(\sum_{i=1}^{k-1} (x^*(\delta(U_i))-1) \frac{1-(\alpha\sum_{e\in\delta(U_i)} x_e^* + 2\beta)}{b_i} h_{U_i}^*\right) \\
&\leq (1+\alpha+\beta)c(x^*) + \left[\max_{0\leq\eta\leq\tau}\left(\eta\frac{1-(2\beta+\alpha(1+\eta))}{1-0.5\tau+0.5\eta}\right)\right] c\left(\sum_{i=1}^{k-1} h_{U_i}^*\right) \\
&\leq \left(1+\alpha+\beta+\max_{0\leq\eta\leq\tau}\left(\eta\frac{1-(2\beta+\alpha(1+\eta))}{1-1-0.5\tau+0.5\eta}\right)\right) c(x^*).
\end{aligned}$$

Let $R(\eta) := \eta\frac{1-(2\beta+\alpha(1+\eta))}{1-1-0.5\tau+0.5\eta}$; then utilizing relevant knowledge in mathematical analysis, $R(\eta)$ takes its maximum value at

$$\eta_0 = \frac{1}{\alpha}(1-3\alpha-2\beta+\sqrt{(-2\alpha)(1-3\alpha-2\beta)}),$$

hence

$$\begin{aligned}
E[c(P)] &\leq (11\alpha+5\beta-1-4\sqrt{(-2\alpha)(1-3\alpha-2\beta)})c(x^*) \\
&= \frac{9-\sqrt{33}}{2}c(x^*).
\end{aligned}$$

□

4. A tighter analysis of approximation ratio

In subsection 3.2, in order to ensure that $y \in P_{Q-join}^{\uparrow}$, L_i's are selected to supplement the odd s-t cut $(U, \bar{U})$ with small weights. There are two criteria for the selection of edge set in L_i's, one is that its weight needs to be large enough, that is, not less than a threshold; and the other is that any two edge sets do not intersect.

In this section, by appropriately relaxing the requirements of the second criterion, we are able to select $\{\hat{h}_{U_i}^*\}$ with higher weights.

Lemma 13. *There exists a set of indicator function $\{\hat{g}_{U_i}^*\}_{i=1}^{k-1}$ such that*

- $\hat{h}_{U_i}^* \in \mathbb{R}_{E_1}^+ \ \forall i \in \{1, 2, \cdots, k-1\}$;
- $\sum_{i=1}^{k-1} \hat{h}_{U_i}^* \leq x^*$;

- $\hat{h}^*_{U_i}(\delta(U_i)) \geq 1 \ \forall i \in \{1, 2, \cdots, k-1\}$.

The proof is similar to the Lemma 3.12 of [2], so we omit it here. Then based on Lemma 13, we can obtain the desired approximation ratio.

Theorem 14. $E[c(P)] \leq \frac{1+\sqrt{5}}{2} c(x^*)$.

Proof. Set

$$y := \alpha x^* + \beta \chi_T + \sum_{i:|U_i \cap Q| \text{ is odd},\ 1 \leq i < k} [1 - (\alpha x^*(\delta(U_i)) + 2\beta)] \hat{g}^*_{U_i},$$

where $\alpha = \frac{\sqrt{5}}{5}$ and $\beta = \frac{5-2\sqrt{5}}{5}$.

By the same argument as Theorem 12, we can prove that y is in Q-join polytope. For a τ-narrow cut $(U, \bar{U})$ with $|U \cap Q|$ odd, we have that:

$$\begin{aligned} y\left(\delta(U)\right) &\geq \alpha \sum_{e \in \delta(U_i)} x^*_e + \beta |\delta(U) \cap Q| + \left[1 - \left(\alpha \sum_{e \in \delta(U_i)} x^*_e + 2\beta\right)\right] \hat{h}^*_{U_i}(\delta(U)) \\ &\geq \alpha \sum_{e \in \delta(U_i)} x^*_e + 2\beta + \left[1 - \left(\alpha \sum_{e \in \delta(U_i)} x^*_e + 2\beta\right)\right] \cdot 1 \\ &= 1. \end{aligned}$$

Now, we aim to analyze the cost of P.

$$\begin{aligned} E[c(P)] &\leq (1 + \alpha + \beta) c(x^*) + c \left(\sum_{i=1}^{k-1} Pr\left[|U_i \cap Q| \text{ is odd}\right] \left[1 - \left(\alpha \sum_{e \in \delta(U_i)} x^*_e + 2\beta\right)\right] \hat{h}^*_{U_i} \right) \\ &\leq (1 + \alpha + \beta) c(x^*) + c \left(\sum_{i=1}^{k-1} \left(\sum_{e \in \delta(U_i)} x^*_e - 1 \right) \left[1 - \left(\alpha \sum_{e \in \delta(U_i)} x^*_e + 2\beta\right)\right] \hat{h}^*_{U_i} \right) \\ &\leq (1 + \alpha + \beta) c(x^*) + \left\{ \max_{0 \leq \eta \leq \tau} \eta \left[1 - (2\beta + \alpha(1 + \eta))\right] \right\} c \left(\sum_{i=1}^{k-1} \hat{h}^*_{U_i} \right) \\ &\leq (1 + \alpha + \beta) c(x^*) + \left\{ \max_{0 \leq \eta \leq \tau} \eta [1 - (2\beta + \alpha(1 + \eta))] \right\} c(x^*). \end{aligned}$$

According to the calculation,

$$\begin{aligned} \max_{0 \leq \eta \leq \tau} \eta [1 - (2\beta + \alpha(1 + \eta))] &= \max_{0 \leq \eta \leq \tau} \alpha \eta (\tau - \eta) \\ &= \frac{(1 - \alpha - 2\beta)^2}{4\alpha}. \end{aligned}$$

Then we obtain

$$\begin{aligned} E[c(P)] &\leq \left(1 + \alpha + \beta + \frac{(1 - \alpha - 2\beta)^2}{4\alpha}\right) c(x^*) \\ &= \frac{1 + \sqrt{5}}{2} c(x^*). \end{aligned}$$

□

5. Conclusion

In this paper, we introduce the generalized traveling salesman path problem and propose its linear programming relaxation. Based on the LP-relaxation and the narrow cuts, we design an algorithm for GTSPP and prove that its approximation ratio is $\frac{1+\sqrt{5}}{2}$. Compared with the traveling salesman path problem, there is still some gap in the approximation ratio, and it is our future work to reduce or even eliminate this gap.

Acknowledgements

The work is supported by National Natural Science Foundation of China (Nos.11871081 and 11871280) and Qinglan Project.

References

[1] Alfandari L, Toulouse S. Approximation of the double traveling salesman problem with multiple stacks. Theoretical Computer Science, 877: 74-89, 2021.

[2] AN H-C, Kleinberg R and Shmoys D-B. Improving Christofides' algorithm for the s-t path TSP. In: Proceedings of the 44th annual ACM Symposium on Theory of computing, pp. 875-886, 2012.

[3] Angelelli E, Bazgan C, Speranza M-G, and Tuza Z. Complexity and approximation for traveling salesman problems with profits. Theoretical Computer Science, 531: 54-65, 2014.

[4] Arora S. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. Journal of the ACM, 45(5): 753-782, 1998.

[5] Arora S, Grigni M, Karger D, Klein P, and Woloszyn A. A polynomial-time approximation scheme for weighted planar graph TSP. In: Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 33–41, 1998.

[6] Christofides N. Worst-case analysis of a new heuristic for the traveling salesman problem. Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[7] Cunningham W-H. Testing membership in matroid polyhedra. Journal of Combinatorial Theory, Series B, 36(2): 161-188, 1984.

[8] Dantzig G, Fulkerson R and Johnson S. Solution of a large-scale traveling salesman problem. Journal of the Operations Research Society of America, 2(4): 393-410, 1954.

[9] Demaine E-D, Hajiaghayi M-T and Kawarabayashi K. Contraction decomposition in H-minor-free graphs and algorithmic applications. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, pp. 441-450, 2011.

[10] Demaine E-D, Hajiaghayi M-T and Mohar B. Approximation algorithms via contraction decomposition. Combinatorica, 30(5): 533-552, 2010.

[11] Edmonds J. Matroids and the greedy algorithm. Mathematical Programming, 1(1): 127-136, 1971.

[12] Edmonds J and Johnson E-L. Matching, Euler tours and the Chinese postman. Mathematical Programming, 5(1): 88-124, 1973.

[13] Frederickson G-N. Approximation algorithms for some postman problems. Journal of the ACM, 26: 538-554, 1979.

[14] Fumei L and Alantha N. Traveling salesman path problems. Mathematical Progromming, 113(1): 39-59, 2008.

[15] Gharan S-O, Saberi A and Singh M. A randomized rounding approach to the traveling salesman problem. In: Proceedings of 52nd Annual Symposium on Foundations of Computer Science, pp. 550-559, 2011.

[16] Gottschalk C and Vygen J. Better s-t-tours by Gao trees. In: Proceedings of 18th International Conference on Integer Programming and Combinatorial Optimization, pp. 126-137, 2016.

[17] Grigni M, Koutsoupias E and Papadimitriou C. An approximation scheme for planar graph TSP. In: Proceedings of IEEE 36th Annual Foundations of Computer Science, pp. 640-645, 1995.

[18] Grötschel M, Lovász L and Schrijver A. The ellipsoid method and its consequences in combinatorial optimization. Combinatorica, 1(2): 169-197, 1981.

[19] Gutin G and Punnen A. The traveling salesman problem and its variations. Kluwer, Dordrecht, 2002.

[20] Guttmann-Beck N, Hassin R, Khuller S, and Raghavachari B. Approximation algorithms with bounded performance guarantees for the clustered traveling salesman problem. Algorithmica, 28(4): 422-437, 2000.

[21] Hoogeveen J-A. Analysis of Christofides' heuristic: some paths are more difficult than cycles. Operations Research Letters, 10(5): 291-295, 1991.

[22] Karlin A-R, Klein N and Gharan S-O. A (slightly) improved approximation algorithm for metric TSP. In: Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing, pp. 32-45, 2021.

[23] Karp R-M. Reducibility among combinatorial problems. Complexity of Computer Computations, 2: 85-103, 1972.

[24] Klein P-N. A linear-time approximation scheme for planar weighted TSP. In: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, pp. 647-656, 2005.

[25] Lam F, Newman A. Traveling salesman path problems. Mathematical Programming, 113(1): 39-59, 2008.

[26] Michail O, Spirakis P G. Traveling salesman problems in temporal graphs. Theoretical Computer Science, 634: 1-23, 2016.

[27] Mömke T and Svensson O. Removing and adding edges for the traveling salesman problem. Journal of the ACM, 63(1): 2, 2016.

[28] Mucha M. 13/9-approximation for graphic TSP. Theory of Computing Systems, 55(4): 640-657, 2014.

[29] Papadimitriou C-H and Vempala S. On the approximability of the traveling salesman problem. Combinatorica, 26(1): 101-120, 2006.

[30] Sahni S and Gonzales T. P-complete approximation problems. Journal of the ACM, 23(3): 555-565, 1976.

[31] Sebö A. Eight fifth approximation for TSP paths. In: Goemans M., Correa J. (eds) Integer Programming and Combinatorial Optimization 2013, LNCS, vol 7801, pp. 362-374. Springer, Berlin, Heidelberg, 2013.

[32] Sebö A, Van Zuylen A. The salesman's improved paths: a 3/2+1/34 approximation. In: Proceedings of the 57th Annual Symposium on Foundations of Computer Science, pp. 118-127, 2016.

[33] Sebö A and Van Zuylen A. The salesman's improved paths through forests. Journal of the ACM, 66(4): 28, 2019.

[34] Sebö A and Vygen J. Shorter tours by nicer ears: 7/5-approximation for the graph-TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. Combinatorica, 34(5): 597-629, 2014.

[35] Serdjukov A. Some extremal bypasses in graphs. Upravlyaemye Sistemy, 1978, 17: 76-79. (in Russian)

[36] Sun J, Gutin G and Zhang X. A LP-based approximation algorithm for generalized traveling salesperson path problem. In: Proceedings of the 15th International Conference on Combinatorial Optimization and Applications, pp. 641-652, 2021.

[37] Traub V and Vygen J. Approaching $\frac{3}{2}$ for the s-t path TSP. Journal of the ACM, 66(2): 14, 2019.

[38] Vygen J. Reassembling trees for the traveling salesman. SIAM Journal on Discrete Mathematics, 30(2):875-894, 2016.

[39] Zenklusen R-A. 1.5-approximation for path TSP. In: Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1539-1549, 2019.