A 3/4 Differential Approximation Algorithm for Traveling Salesman Problem

Yuki Amano[1] Kazuhisa Makino[1]

Abstract: In this paper, we consider differential approximability of the traveling salesman problem (TSP). The differential approximation ratio was proposed by Demange and Paschos in 1996 as an approximation criterion that is invariant under affine transformation of the objective function. We show that TSP is 3/4-differential approximable, which improves the currently best known bound $3/4 - O(1/n)$ due to Escoffier and Monnot in 2008, where n denotes the number of vertices in the given graph.

1. Introduction

The traveling salesman problem (TSP) finds a shortest *Hamiltonian cycle* in a given complete graph with edge length, when a cycle is called *Hamiltonian* (also called a *tour*) if it visits every vertex exactly once. TSP is one of the most fundamental NP-hard optimization problems in operations research and computer science, and has been intensively studied from both practical and theoretical view points [7], [19], [21], [22]. It has a number of applications such as planning, logistics, and the manufacture of microchips [4], [11]. Because of these importance, many heuristics and exact algorithms have been proposed [3], [13], [14], [15]. From a view point of computational complexity, TSP is NP-hard, even in the Euclidean case, which includes the metric case. It is known that metric TSP is approximable with factor 1.5 [6], and inapproximable with factor 117/116 [5]. Euclidean TSP admits a polynomial-time approximation scheme (PTAS), if the dimension of the Euclidean space is bounded by a constant [1]. We note that the approximation factors (i.e., ratios) above are widely used to analyze approximation algorithms.

Let Π be an optimization problem, and let I be an instance of Π. We denote by $\mathrm{opt}(I)$ the value of an optimal solution to I. For an approximation algorithm A for Π, we denote by $\mathrm{apx}_A(I)$ the value of the approximate solution computed by A for the instance I. Let

$$r_A(I) = \mathrm{apx}_A(I)/\,\mathrm{opt}(I),$$

and define the *standard approximation ratio* of A by $\sup_{I\in\Pi} r_A(I)$, where we assume that Π is a minimization problem. Although the standard approximation ratio is well-studied and an important concept in algorithm theory, it is not invariant under affine transformation of the objective function. Namely, if the objective function $f(x)$ is replaced by $a + bf(x)$ for some constant a and b, which might depend on the instance I, the standard ratio is not preserved. For example, the vertex cover problem and the independent set problem have affinely dependent objective functions. However they have different characteristics in the standard approximation ratio. The vertex cover problem is 2-approximable [20], while the independent set problem is inapproximable within $O(n^{1-\epsilon})$ for any $\epsilon > 0$ [9], where n denotes the number of vertices in a given graph. In order to remedy to this phenomenon, Demange and Paschos [8] proposed the *differential approximation ratio* defined by $\sup_{I\in\Pi} \rho_A(I)$, where

$$\rho_A(I) = \frac{\mathrm{wor}(I) - \mathrm{apx}_A(I)}{\mathrm{wor}(I) - \mathrm{opt}(I)}$$

and $\mathrm{wor}(I)$ denotes the value of a worst solution to I. Note that for any instance I of Π

$$\mathrm{apx}_A(I) = \rho_A(I)\,\mathrm{opt}(I) + (1 - \rho_A(I))\,\mathrm{wor}(I).$$

Thus we have $0 \le \rho_A(I) \le 1$ and the larger $\rho_A(I)$ implies the better approximation for the instance I. Moreover, by definition, the differential approximation ratio remains invariant under affine transformation of the objective function. For this, it has been recently attracted much attention in approximation algorithm [2]. It is known [17] that TSP, metric TSP, max TSP, and max metric TSP are affinely equivalent, i.e., their objective functions are transferred to each other by affine transformations, where max TSP is the problem to find a longest Hamiltonian cycle and max metric TSP is max TSP, in which the input weighted graph satisfies the metric condition. Therefore, these problems have the identical differential approximation ratio.

Hassin and Khuller [12] first studied differential approximability of TSP, and showed that it is 2/3-differential approximable. Escoffier and Monnot [10] improved it to

[1] Research Institute for Mathematical Sciences, Kyoto University, Kyoto, Japan. {ukiamano,makino}@kurims.kyoto-u.ac.jp

$3/4 - O(1/n)$, where n denotes the number of vertices of a given graph. Monnot et al. [16], [18] showed that TSP is 3/4-differential approximable if each edge length is restricted to one or two.

In this paper, we show that TSP is 3/4-differential approximable, which improves the currently best known results [10], [16], [18]. Our algorithm is based on an idea in [10] for the case in which a given graph G has an even number of vertices and a triangle (i.e., cycle with 3 edges) is contained in a minimum weighted 2-factor of G. Their algorithm first computes minimum weighted 1- and 2-factors of a given graph, modify them to four path covers P_i (for $i = 1, \ldots, 4$), and then extend each path cover P_i to a tour by adding edge set F_i to it in such a way that at least one of the tours guarantees 3/4-differential approximation ratio. Here the definitions of factor and path cover can be found in Section 2. We generalize their idea to the general even case. Note that $\bigcup_{i=1}^{4} F_i$ in their algorithm always forms a tour, where in general it does not. We show that there exists a way to construct path covers such that the length of $\bigcup_{i=1}^{4} F_i$ is at most the worst tour length. Our algorithm for odd case is much more involved. For each path with three edges, we first construct a 2-factor and two path covers of a given graph which has minimum length among all these which completely and partially contains the path, modify them to eight path covers, and then extend each path cover to a tour, in such a way that at least one of the eight tours guarantees 3/4-differential approximation ratio.

The rest of the paper is organized as follows. In Section 2, we define basic concepts of graphs and discuss some properties on 2-matchings, which will be used in the subsequent sections. In Sections 3 and 4, we provide an approximation algorithms for TSP in which a given graph G has even and odd numbers of vertices, respectively.

2. Preliminary

Let $G = (V, E)$ be an undirected graph, where n and m denote the number of vertices and edges in G, respectively. In this paper, we assume that a given graph G of TSP is complete, i.e., $E = \binom{V}{2}$, and it has an edge length function $\ell : E \to \mathbb{R}_+$, where $\mathbb{R}_+$ denotes the set of nonnegative reals. For a set $F \subseteq E$, let $V(F)$ denote the set of vertices with incident edges in F, i.e., $V(F) = \{v \in V \mid \exists (v, w) \in F\}$. A set $F \subseteq E$ is called *spanning* if $V(F) = V$, and *acyclic* if F contains no cycle. For a positive integer k, a set $F \subseteq E$ is called a *k-matching* (resp., *k-factor*) if each vertex has at most (resp., exactly) k incident edges in F. Here 1-matching is simply called a *matching*. Note that an acyclic 2-matching F corresponds to a family of vertex-disjoint paths denoted by $\mathcal{P}(F) \subseteq 2^E$. A 2-matching is called a *path cover* if it is spanning and acyclic. For a set $F \subseteq E$, $V_1(F)$ and $V_2(F)$ respectively denote the sets of vertices with one and two incident edges in F. For a set $F \subseteq E$ and a vertex $v \in V$, let $\delta_F(v) = \{e \in F \mid e \text{ is incident to } v\}$.

Definition 1. *A pair of spanning 2-matchings (S, T) is called* valid *if it satisfies the following three conditions:*

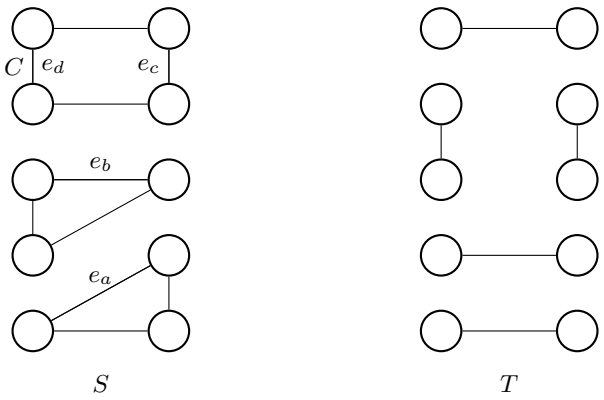


Fig. 1: A valid pair (S, T) of spanning 2-matchings.

(i) *T is acyclic.*

(ii) $\delta_S(v) = \delta_T(v)$ *for any* $v \in V_2(S) \cap V_2(T)$. (1)

(iii) $V(C) \neq V(P)$ *for any cycle* $C \subseteq S$ *and any path* $P \subseteq \mathcal{P}(T)$. (2)

Lemma 2. *Let (S, T) be a valid pair of spanning 2-matchings. If S contains a cycle C, then C contains two edges e_i for $i = 1, 2$ such that $S_i = S \setminus \{e_i\}$ and $T_i = S \cup \{e_i\}$ satisfy the following three conditions:*

(S_i, T_i) *is valid for* $i = 1, 2$. (3)

$V_1(S_i) \cup V_1(T_i) = V_1(S) \cup V_1(T)$ *and* $V_1(S_i) \cap V_1(T_i) = V_1(S) \cap V_1(T)$ *for* $i = 1, 2$. (4)

$\mathcal{P}(T)$ *contains a path P such that $P \cup \{e_1\}$ and $P \cup \{e_2\}$ are both paths.* (5)

Note that (S_1, T_1) and (S_2, T_2) in Lemma 2 satisfy

$$S_i \cup T_i = S \cup T \text{ and } S_i \cap T_i = S \cap T \text{ for } i = 1, 2, \quad (6)$$

which immediately implies

$$\ell(S_i) + \ell(T_i) = \ell(S) + \ell(T) \text{ for } i = 1, 2, \quad (7)$$

where $\ell(F) = \sum_{e \in F} \ell(e)$ for a set $F \subseteq E$.
Figure 1 shows a valid pair of spanning 2-matchings and two edges e_c and e_d in C that satisfy the three conditions of Lemma 2.

3. Approximation for even instances

In this section, we construct an approximation algorithm for TSP in which a given graph has an even number of vertices. Our algorithm first construct four path covers from minimum weighted 1- and 2-factors of a given graph G, and then extend each path cover to a tour in such a way that at least one of the tours guarantees 3/4-differential approximation ratio.

Let us first explain the procedure `FourPathCovers` where it can be found in Fig. 2. Let (S, T) be a valid pair of spanning 2-matchings of (G, ℓ) such that S is a 2-factor. The procedure computes from (S, T) four path covers S_1, S_2, T_1, and T_2 that satisfies (4), (6), $V_1(S_i)$ and $V_1(T_i)$ is a partition of $V_1(T)$ for $i = 1, 2$, i.e.,

$$V_1(S_i) \cup V_1(T_i) = V_1(T) \text{ and } V_1(S_i) \cap V_1(T_i) = \emptyset \text{ for } i = 1, 2, \quad (8)$$

and

$$\text{there exist } e_1, e_2 \in E \text{ and } P \in \mathcal{P}(T_1 \cap T_2) \text{ such that } T_1 \setminus T_2 = \{e_1\},\ T_2 \setminus T_1 = \{e_2\}, \text{ and } P \cup \{e_1\} \text{ and } P \cup \{e_2\} \text{ are both paths.} \quad (9)$$

In Fig. 2 we apply **Procedure** `FourPathCovers` to (S, T) in Fig. 1.

Lemma 3. *For a graph $G = (V, E)$, let (S, T) be a valid pair of spanning 2-matchings such that S has a cycle. Then* ***Procedure*** `FourPathCovers` *returns four path covers S_1, S_2, T_1, and T_2 that satisfy* (4), (6), *and* (9). *Furthermore, if S is addition a 2-factor of G, then the four path covers satisfy* (8).

Note that (S, T) is a valid and $V_1(T) = V$, if S and T are 2- and 1-factors of G, respectively.

Let S and T be 2- and 1-factors of G, respectively. Note that our algorithm explain later makes use of minimum weighted 2-factor S and 1-factor T of (G, ℓ) which can be computed from (G, ℓ) in polynomial time. We assume that S is not a tour of G, i.e., S contains at least two cycles, since otherwise, S itself is an optimal tour. Let S_1, S_2, T_1, and T_2 be path covers returned by **Procedure** `FourPathCover`(S, T).

Let us then show how to construct edge sets A_1, A_2, B_1, and B_2, such that

$$S_i \cup A_i \text{ is tour (for } i = 1, 2), \quad (10)$$

$$T_i \cup B_i \text{ is tour (for } i = 1, 2), \text{ and} \quad (11)$$

$$\ell(A_1) + \ell(A_2) + \ell(B_1) + \ell(B_2) \leq \mathrm{wor}(G, \ell), \quad (12)$$

where $\mathrm{wor}(G, \ell)$ denotes the length of a longest tour of (G, ℓ).

Let $e_1 = (p_1, p_2)$ and $e_2 = (p_3, p_4)$ be edges in Lemma 3. Since e_1 and e_2 are chosen from a cycle C and satisfy (9), we can assume that $p_1 \neq p_3, p_4$ and $p_4 \neq p_1, p_2$, where $p_2 = p_3$ might hold. We note that $\mathcal{P}(S_1) \setminus \mathcal{P}(S_2)$ consists of a (p_1, p_2)-path $P_1 = C \setminus \{e_1\}$, and $\mathcal{P}(S_2) \setminus \mathcal{P}(S_1)$ consists of a (p_3, p_4)-path $P_2 = C \setminus \{e_2\}$.

Let $Q_i\,(i = 1, \ldots, k)$ denote vertex-disjoint (x_i, y_i)-paths such that $\{Q_1, \ldots, Q_k\} = \mathcal{P}(S_1) \cap \mathcal{P}(S_2)$ and x_1 and y_1 satisfy

$$\ell(p_2, x_1) + \ell(p_3, y_1) \leq \ell(p_2, y_1) + \ell(p_3, x_1). \quad (13)$$

Define A_1 and A_2 by

$$\begin{aligned} A_1 &= \{(p_2, x_1)\} \cup \{(y_k, p_1)\} \\ &\quad \cup \{(y_i, x_{i+1}) \mid i = 1, \ldots, k-1\} \\ A_2 &= \{(p_3, y_1)\} \cup \{(x_k, p_4)\} \\ &\quad \cup \{(x_i, y_{i+1}) \mid i = 1, \ldots, k-1\}. \end{aligned} \quad (14)$$

Figure 3 shows two edge sets A_1 and A_2 for S_1 and S_2 in Fig. 2.

Lemma 4. *Two edge sets A_1 and A_2 defined in* (14) *satisfy* (10) *and $A_1 \cap A_2 = \emptyset$, and $A_1 \cup A_2$ consists of* (i) *a (p_1, p_4)-path if $p_2 = p_3$ and either* (ii) *vertex-disjoint (p_1, p_3)- and (p_2, p_4)-paths or* (iii) *vertex-disjoint (p_1, p_2)- and (p_3, p_4)-paths if $p_2 \neq p_3$.*

Let us next construct B_1 and B_2. Let $O_i\,(i = 1, \ldots, d)$ denote vertex-disjoint (z_i, w_i)-paths such that $\{O_1, \ldots, O_d\} = \mathcal{P}(T_1) \cap \mathcal{P}(T_2)$. Note that $\mathcal{P}(T_1) \cap \mathcal{P}(T_2) = \emptyset$ (i.e., $d = 0$) might hold. We separately consider the following four cases.

(1) $p_2 = p_3$, $p_1 \neq p_4$, and $\mathcal{P}(T_1 \cap T_2)$ contains a (p_1, p_4)-path.
(2) $p_2 = p_3$, $p_1 \neq p_4$, and $\mathcal{P}(T_1 \cap T_2)$ contain no (p_1, p_4)-path.
(3) $p_2 \neq p_3$, $p_1 \neq p_4$, and $\mathcal{P}(T_1 \cap T_2)$ contains (p_1, p_4)- and (p_2, p_3)-paths.
(4) $p_2 \neq p_3$, $p_1 \neq p_4$, and $\mathcal{P}(T_1 \cap T_2)$ contains a (p_2, p_3)-path and no (p_1, p_4)-path.

Here we recall that $e_1 = (p_1, p_2)$ and $e_2 = (p_3, p_4)$ satisfy Lemma 3.

Case 1: Let R_1 denote a (p_1, p_4)-path in $\mathcal{P}(T_1 \cap T_2)$, and for some vertex q_2, let R_2 denote (p_2, q_2)-path in $\mathcal{P}(T_1 \cap T_2)$. Then, we have

$$\begin{aligned} \mathcal{P}(T_1) &= \{O_1, \ldots, O_d\} \cup \{R_1 \cup \{e_1\} \cup R_2\} \\ \mathcal{P}(T_2) &= \{O_1, \ldots, O_d\} \cup \{R_1 \cup \{e_2\} \cup R_2\}, \end{aligned}$$

where $R_1 \cup \{e_1\} \cup R_2$ and $R_1 \cup \{e_2\} \cup R_2$ are (p_4, q_2)- and (p_1, q_2)-paths, respectively. Define B_1 and B_2 by

$$\begin{aligned} B_1 &= \begin{cases} \{(q_2, p_4)\} & \text{if } d = 0 \\ \{(q_2, z_1)\} \cup \{(w_d, p_4)\} \cup \{(w_i, z_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1 \end{cases} \\ B_2 &= \begin{cases} \{(q_2, p_1)\} & \text{if } d = 0 \\ \{(q_2, w_1)\} \cup \{(z_d, p_1)\} \cup \{(z_i, w_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1. \end{cases} \end{aligned} \quad (15)$$

By definition, two edge sets B_1 and B_2 satisfy (11) and $B_1 \cap B_2 = \emptyset$, and $B_1 \cup B_2$ consists of a (p_1, p_4)-path

Case 2: For some vertices q_1, q_2 and q_4, let R_1, R_2, and R_4 respectively denote (p_1, q_1)-, (p_2, q_2)-, and (p_4, q_4)-paths in $\mathcal{P}(T_1 \cap T_2)$. Then, we have

$$\begin{aligned} \mathcal{P}(T_1) &= \{O_1, \ldots, O_d\} \cup \{R_4, R_1 \cup \{e_1\} \cup R_2\} \\ \mathcal{P}(T_2) &= \{O_1, \ldots, O_d\} \cup \{R_1, R_4 \cup \{e_2\} \cup R_2\}, \end{aligned}$$

where $R_1 \cup \{e_1\} \cup R_2$ and $R_4 \cup \{e_2\} \cup R_2$ are (q_1, q_2)- and (q_4, q_2)-paths, respectively. Define B_1 and B_2 by

$$\begin{aligned} B_1 &= \begin{cases} \{(q_2, q_4), (p_4, q_1)\} & \text{if } d = 0 \\ \{(q_2, z_1)\} \cup \{(w_d, q_4), (p_4, q_1)\} \cup \{(w_i, z_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1 \end{cases} \\ B_2 &= \begin{cases} \{(q_2, q_1), (p_1, q_4)\} & \text{if } d = 0 \\ \{(q_2, w_1)\} \cup \{(z_d, q_1), (p_1, q_4)\} \cup \{(z_i, w_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1. \end{cases} \end{aligned} \quad (16)$$

Similarly to Case 1, we have (11), $B_1 \cap B_2 = \emptyset$, and $B_1 \cup B_2$

Procedure `FourPathCovers`(S, T)
/*(S, T) is a valid pair of spanning 2-matchings such that S has a cycle. The procedure returns 4 path covers S_1, S_2, T_1, and T_2 that satisfies (4), (6), and (9).*/

if S has exactly one cycle **then**
  Take two edges e_1 and e_2 in Lemma 2.
  return $S_1 = S \setminus \{e_1\}$, $T_1 = T \cup \{e_1\}$, $S_2 = S \setminus \{e_2\}$, and $T_2 = T \cup \{e_2\}$
else /* *S has at least two cycles.* */
  Take an edge e_1 in Lemma 2.
  return `FourPathCovers`$(S \setminus \{e_1\}, T \cup \{e_1\})$
end if

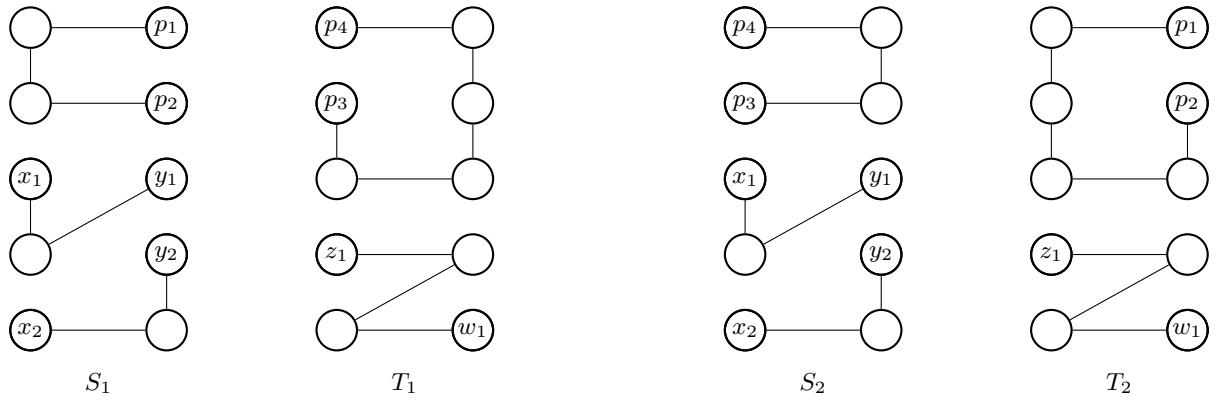


Fig. 2: Two pairs (S_1, T_1) and (S_2, T_2) computed by **Procedure** `FourPathCovers` for a valid pair (S, T), $e_1^{(1)} = e_a$, $e_1^{(2)} = e_b$, $e_1^{(3)} = e_c$ and $e_2^{(3)} = e_d$ in Fig. 1, where $e_i^{(j)}$ denotes the edge chosen as e_i in the j-th round of the procedure.

consists of a (p_1, p_4)-path.

Case 3: Let R_1 and R_2 respectively denote (p_1, p_4)- and (p_2, p_3)-paths in $\mathcal{P}(T_1 \cap T_2)$. Then, we have

$$\mathcal{P}(T_1) = \{O_1, \ldots, O_d\} \cup \{R_1 \cup \{e_1\} \cup R_2\}$$
$$\mathcal{P}(T_2) = \{O_1, \ldots, O_d\} \cup \{R_1 \cup \{e_2\} \cup R_2\},$$

where $R_1 \cup \{e_1\} \cup R_2$ and $R_1 \cup \{e_2\} \cup R_2$ are (p_3, p_4)- and (p_1, p_2)-paths, respectively. Define B_1 and B_2 by

$$B_1 = \begin{cases} \{(p_3, p_4)\} & \text{if } d = 0 \\ \{(p_3, z_1)\} \cup \{(w_d, p_4)\} \cup \{(w_i, z_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1 \end{cases}$$
$$B_2 = \begin{cases} \{(p_2, p_1)\} & \text{if } d = 0 \\ \{(p_2, w_1)\} \cup \{(z_d, p_1)\} \cup \{(z_i, w_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1. \end{cases} \quad (17)$$

Similarly to the previous cases, we have (11), $B_1 \cap B_2 = \emptyset$, and $B_1 \cup B_2$ consists of either vertex-disjoint (p_1, p_2)- and (p_3, p_4)-paths or vertex-disjoint (p_1, p_3)- and (p_2, p_4)-paths.

Case 4: Let R_2 denote (p_2, p_3)-path in $\mathcal{P}(T_1 \cap T_2)$, and for some vertices q_1 and q_4, let R_1 and R_4 respectively denote (p_1, q_1)- and (p_4, q_4)-paths in $\mathcal{P}(T_1 \cap T_2)$. Then, we have

$$\mathcal{P}(T_1) = \{O_1, \ldots, O_d\} \cup \{R_4, R_1 \cup \{e_1\} \cup R_2\}$$
$$\mathcal{P}(T_2) = \{O_1, \ldots, O_d\} \cup \{R_1, R_4 \cup \{e_2\} \cup R_2\},$$

where $R_1 \cup \{e_1\} \cup R_2$ and $R_4 \cup \{e_2\} \cup R_2$ are (q_1, p_3)- and (q_4, p_2)-paths, respectively. Define B_1 and B_2 by

$$B_1 = \begin{cases} \{(p_3, q_4), (p_4, q_1)\} & \text{if } d = 0 \\ \{(p_3, z_1)\} \cup \{(w_d, q_4), (p_4, q_1)\} \cup \{(w_i, z_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1 \end{cases}$$
$$B_2 = \begin{cases} \{(p_2, q_1), (p_1, q_4)\} & \text{if } d = 0 \\ \{(p_2, w_1)\} \cup \{(z_d, q_1), (p_1, q_4)\} \cup \{(z_i, w_{i+1}) \mid i = 1, \ldots, d-1\} & \text{if } d \geq 1. \end{cases} \quad (18)$$

Similarly to the previous cases, we have (11), $B_1 \cap B_2 = \emptyset$, and $B_1 \cup B_2$ consists of either vertex-disjoint (p_1, p_2)- and (p_3, p_4)-paths or vertex-disjoint (p_1, p_3)- and (p_2, p_4)-paths.

In summary, we have the following lemma.

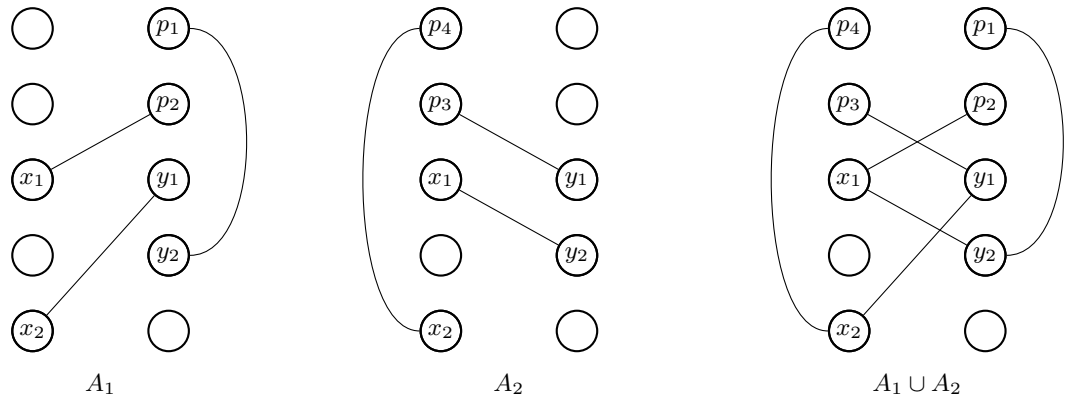
Lemma 5. *Two edge sets B_1 and B_2 defined as above satisfy* (11) *and $B_1 \cap B_2 = \emptyset$, and $B_1 \cup B_2$ consists of* (i) *a (p_1, p_4)-path if $p_2 = p_3$ and either* (ii) *vertex-disjoint (p_1, p_2)- and (p_3, p_4)-paths or* (iii) *vertex-disjoint (p_1, p_3)- and (p_2, p_4)-paths if $p_2 \neq p_3$.*

Figure 4 shows two edge sets B_1 and B_2 for path covers T_1 and T_2 in Fig. 2. Furthermore, A_i and B_i $(i = 1, 2)$ satisfy the following properties.

Lemma 6. *Let A_1, A_2, B_1, and B_2 be defined as above. Then they are all pairwise disjoint, and $C = A_1 \cup A_2 \cup B_1 \cup B_2$ is a 2-factor, consisting of either one or two cycles. Furthermore, there exists a tour H of G such that $\ell(H) \geq \ell(C)$.*

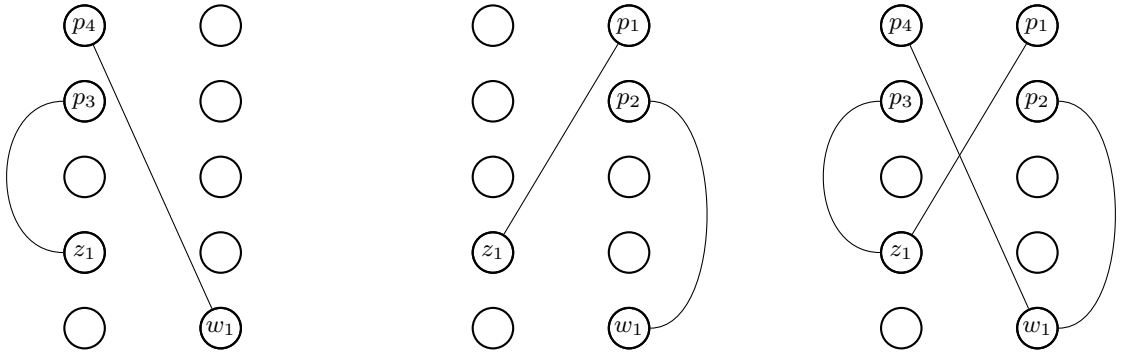
We are now ready to describe our approximation algorithm.

Theorem 7. *For a complete graph $G = (V, E)$ with*



A_1 A_2 $A_1 \cup A_2$

Fig. 3: Two edge sets A_1 and A_2 for path covers S_1 and S_2 in Fig. 2.



B_1 B_2 $B_1 \cup B_2$

Fig. 4: Two edge sets B_1 and B_2 for path covers T_1 and T_2 in Fig. 2.

an even number of vertices and an edge length function $\ell : E \to R_+$, ***Algorithm*** `TourEven` *computes a* 3/4*-differential approximate tour of* (G, ℓ) *in polynomial time.*

Proof. We show that **Algorithm** `TourEven` outputs a 3/4-differential approximate tour T_{apx} in polynomial time. If a minimum weighted 2-factor S of (G, ℓ) computed in the algorithm is a tour, then clearly $T_{\mathrm{apx}} = S$ is an optimal tour. On the other hand, if S is not a tour, then we have

$$\begin{aligned} &4\ell(T_{\mathrm{apx}}) \\ &\quad \le \ell(S_1 \cup A_1) + \ell(S_2 \cup A_2) + \ell(T_1 \cup B_1) + \ell(T_2 \cup B_2) \\ &\quad = 2(\ell(S) + \ell(T)) + \ell(A_1) + \ell(A_2) + \ell(B_1) + \ell(B_2) \\ &\quad \le 3\,\mathrm{opt}(G, \ell) + \mathrm{wor}(G, \ell), \end{aligned}$$

where the first equality follows from Lemmas 4, 5, and 6, and the last inequality follows from Lemma 6, and $\ell(S) \le \mathrm{opt}(G, \ell)$, and $2\ell(T) \le \mathrm{opt}(G, \ell)$. Thus T_{apx} is a 3/4-differential approximate tour. Note that minimum weighted 1- and 2-factors can be computed in polynomial time, and A_i and B_i $(i = 1, 2)$ can be computed in polynomial time. Thus **Algorithm** `TourEven` is polynomial, which completes the proof. □

Before concluding the section, let us remark that 3/4-differential approximability is known for graph with an even number of vertices [10]. Different from the algorithm in [10], ours is constructed in a uniform framework, which can further be extended to the odd case.

4. Approximation for odd instances

In this section, we construct an approximation algorithm for TSP with an odd number of vertices. Our algorithm is much more involved than the even case. It first guesses a path P with three edges in an optimal tour, constructs eight path covers based on P, and extend each path cover to a tour in such a way that at least one of the eight tours guarantees 3/4-differential approximation ratio.

More precisely, for each path P with three edges, say, $P = \{(v_1, v_2), (v_2, v_3), (v_3, v_4)\}$ with all v_i's distinct, let S be a minimum weighted 2-factor among those containing P, let T be a minimum weighted path cover among those satisfying $(v_1, v_2), (v_2, v_3) \in T$ and $V_1(T) = V \setminus \{v_2\}$, and let T' be a minimum weighted path cover among those satisfying $(v_2, v_3), (v_3, v_4) \in T'$ and $V_1(T') = V \setminus \{v_3\}$. Assume that S is not a tour, i.e., it contains at least two cycles, since otherwise, is optimal, and hence ensures 3/4-differential approximability if some optimal tour contains P. We note that (S, T) and (S, T') are both valid pairs of spanning 2-matchings. We apply **Procedure** `FourPathCovers` to them, but not arbitrarily. Let us specify two cycles C^* and C^{**} in S such that $P \subseteq C^*$ and $P \cap C^{**} = \emptyset$. We define two vertices v_0 and v_5 in $V(C^*)$ such that $v_0 \ne v_2$, $v_5 \ne v_3$, and $(v_0, v_1), (v_4, v_5) \in C^*$. By definition $v_0 = v_4$ and $v_5 = v_1$ hold if $|C^*| = 4$. Furthermore, we define two edges f and f' in C^{**} that satisfy the properties in the next

Algorithm `TourEven`

Input: A complete graph $G = (V, E)$ with even $|V|$, and an edge length function $\ell : E \to \mathbb{R}_+$.
Output: A tour T_{apx} in G.

Compute minimum weighted 2-factor S and 1-factor T of (G, ℓ).
if S is a tour **then**
  $T_{\mathrm{apx}} := S$.
else
  $S_1, T_1, S_2, T_2 := \texttt{FourPathCovers}(S, T)$.
  Compute edge sets A_1, A_2, B_1, B_2 defined in (14), (15), (16), (17), and (18).
  $\mathcal{T} := \{S_1 \cup A_1, S_2 \cup A_2, T_1 \cup B_1, T_2 \cup B_2\}$.
  $T_{\mathrm{apx}} := \operatorname{argmin}_{T \in \mathcal{T}} \ell(T)$.
end if
Outputs T_{apx} and halt.

lemma.

Lemma 8. *Let C^{**}, T and T' be defined as above. Then there exist two edges $f \in C^{**} \setminus T$ and $f' \in C^{**} \setminus T'$ such that*

(i) *they have a common endpoint q, and*
(ii) *$T \cup \{f\}$ and $T' \cup \{f'\}$ are path covers.*

We note that f and f' in Lemma 8 might be identical, and (ii) in Lemma 8 implies that two pairs $(S \setminus \{f\}, T \cup \{f\})$ and $(S \setminus \{f'\}, T' \cup \{f'\})$ are valid.

Our algorithm uses **Procedure** `FourPathCovers` for (S, T) defined as above in such a way that edge $e_1 = f$ is chosen in the first round and two edges $e_1 = (v_3, v_4)$ and $e_2 = (v_0, v_1)$ are chosen in the last round. Similarly, our algorithm uses **Procedure** `FourPathCovers` for (S, T') defined as above in such a way that edge $e_1 = f'$ is chosen in the first round and two edges $e_1 = (v_1, v_2)$ and $e_2 = (v_4, v_5)$ are chosen in the last round. Let S_1, T_1, S_2, and T_2 be four path covers obtained by **Procedure** `FourPathCover`(S, T), and let S'_1, T'_1, S'_2, and T'_2 be four path covers returned by **Procedure** `FourPathCover`(S, T').

Lemma 9. *Let S, T, S_i, and T_i $(i = 1, 2)$ be defined as above. Then S_1, S_2, T_1, and T_2 are path covers such that*

$$S_i \cup T_i = S \cup T \text{ and } S_i \cap T_i = S \cap T \text{ for } i = 1, 2, \tag{19}$$

$$V_1(S_i) \cup V_1(T_i) = V \setminus \{v_2\} \text{ and } V_1(S_i) \cap V_1(T_i) = \emptyset \text{ for } i = 1, 2, \tag{20}$$

$$T_1 \setminus T_2 = \{(v_3, v_4)\}, T_2 \setminus T_1 = \{(v_0, v_1)\}, \text{and } \{(v_1, v_2), (v_2, v_3)\} \in \mathcal{P}(T_1 \cap T_2), \text{and} \tag{21}$$

$$q \in V_1(S_1) \cap V_1(S_2), \tag{22}$$

where $v_i \in V(C^*)$ $(i = 0, \ldots, 4)$ are defined as above and q is a common endpoint of f and f' in Lemma 8.

Similarly, we have the following lemma.

Lemma 10. *Let S, T', S'_i, and T'_i $(i = 1, 2)$ be defined as above. Then S'_1, S'_2, T'_1, and T'_2 are path covers such that*

$$S'_i \cup T'_i = S \cup T' \text{ and } S'_i \cap T'_i = S \cap T' \text{ for } i = 1, 2, \tag{23}$$

$$V_1(S'_i) \cup V_1(T'_i) = V \setminus \{v_3\} \text{ and } V_1(S'_i) \cap V_1(T'_i) = \emptyset \text{ for } i = 1, 2, \tag{24}$$

$$T'_1 \setminus T'_2 = \{(v_1, v_2)\}, T'_2 \setminus T'_1 = \{(v_4, v_5)\}, \text{and } \{(v_2, v_3), (v_3, v_4)\} \in \mathcal{P}(T'_1 \cap T'_2), \text{and} \tag{25}$$

$$q \in V_1(S'_1) \cap V_1(S'_2), \tag{26}$$

where $v_i \in V(C^*)$ $(i = 1, \ldots, 5)$ are defined as above and q is a common endpoint of f and f' in Lemma 8.

We can construct edge sets $A_i^{(\prime)}$ and $B_i^{(\prime)}$ (for $i = 1, 2$), such that

$$S_i \cup A_i \text{ is tour (for } i = 1, 2), \tag{27}$$

$$T_i \cup B_i \text{ is tour (for } i = 1, 2), \tag{28}$$

$$S'_i \cup A'_i \text{ is tour (for } i = 1, 2), \tag{29}$$

$$T'_i \cup B'_i \text{ is tour (for } i = 1, 2), \text{ and} \tag{30}$$

$$\ell(A_1) + \ell(A_2) + \ell(B_1) + \ell(B_2) + \ell(A'_1) + \ell(A'_2) + \ell(B'_1) + \ell(B'_2) \le 2\,\mathrm{wor}(G, \ell) - 2\ell(v_2, v_3), \tag{31}$$

where $\mathrm{wor}(G, \ell)$ denotes the length of a longest tour of (G, ℓ).

We are now ready to describe our approximation algorithm, called `TourOdd`.

Before analyzing **Algorithm** `TourOdd`, let us evaluate $\ell(S)$, $\ell(T)$ and $\ell(T')$.

Lemma 11. *For a path $P = \{(v_1, v_2), (v_2, v_3), (v_3, v_4)\}$, let S, T and T' be defined as above. If there exists an optimal tour that contains P, then*

$$2\ell(S) + \ell(T) + \ell(T') \le 3\,\mathrm{opt}(G, \ell) + \ell(v_2, v_3). \tag{32}$$

Theorem 12. *For a complete graph $G = (V, E)$ with an odd number of vertices and an edge length function $\ell : E \to \mathbb{R}_+$, **Algorithm** `TourOdd` computes a 3/4-differential approximate tour of (G, ℓ) in polynomial time.*

Acknowledgments This work was partially supported by the joint project of Kyoto University and Toyota Motor Corporation, titled "Advanced Mathematical Science for Mobility Society" and by KAKENHI.

References

[1] Arora, Sanjeev. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems.

Algorithm `TourOdd`

Input: A complete graph $G = (V, E)$ with odd $|V|$, and an edge length function $\ell : E \to \mathbb{R}_+$.
Output: A tour T_{apx} in G.

if $n < 17$ **then**
  Compute an optimal tour T_{opt} of (G, ℓ) by exhaustive search.
  Output T_{opt} and halt.
else
  $\mathcal{T} := \emptyset$.
  for v_1,v_2,v_3 and v_4 in the 4-permutations of V **do**
    Compute a minimum weighted 2-factor S among those containing $\{(v_1, v_2), (v_2, v_3), (v_3, v_4)\}$.
    Compute a minimum weighted path cover T among those satisfying $(v_1, v_2), (v_2, v_3) \in T$ and $V_1(T) = V \setminus \{v_2\}$.
    Compute a minimum weighted path cover T' among those satisfying $(v_2, v_3), (v_3, v_4) \in T'$ and $V_1(T') = V \setminus \{v_3\}$.
    if S is a tour **then**
      $\mathcal{T} := \mathcal{T} \cup \{S\}$.
    else
      $S_1, T_1, S_2, T_2 :=$ `FourPathCovers`(S, T).
      $S'_1, T'_1, S'_2, T'_2 :=$ `FourPathCovers`(S, T').
      Compute edge sets A_1, A_2, B_1, B_2, A'_1, A'_2, B'_1, and B'_2, which satisfy (27), (28), (29), (30), and (31).
      $\mathcal{T} := \mathcal{T} \cup \{S_1 \cup A_1, S_2 \cup A_2, T_1 \cup B_1, T_2 \cup B_2, S'_1 \cup A'_1, S'_2 \cup A'_2, T'_1 \cup B'_1, T'_2 \cup B'_2\}$.
    end if
  end for
  $T_{\mathrm{apx}} := \operatorname*{argmin}_{T \in \mathcal{T}} \ell(T)$.
  Output T_{apx} and halt.
end if

Journal of the ACM, 45(5):753–782, 1998.

[2] Ausiello, Giorgio and Bazgan, Cristina and Demange, Marc and Paschos, Vangelis Th. Completeness in differential approximation classes. *International Journal of Foundations of Computer Science*, 16(06):1267–1295, 2005.

[3] Bellman, Richard. Dynamic programming treatment of the travelling salesman problem. *Journal of the ACM (JACM)*, 9(1):61–63, 1962.

[4] Bland, Robert G and Shallcross, David F. Large travelling salesman problems arising from experiments in X-ray crystallography: a preliminary report on computation. *Operations Research Letters*, 8(3):125–128, 1989.

[5] Chlebík, Miroslav and Chlebíková", Janka. Approximation hardness of Travelling Salesman via weighted amplifiers. In *International Computing and Combinatorics Conference*, pages 115–127. Springer, 2019.

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

[7] Cook, William J. *In pursuit of the traveling salesman: mathematics at the limits of computation*. Princeton University Press, 2011.

[8] Demange, Marc and Paschos, Vangelis Th. On an approximation measure founded on the links between optimization and polynomial approximation theory. *Theoretical Computer Science*, 158(1-2):117–141, 1996.

[9] Engebretsen, Lars and Holmerin, Jonas. Clique is hard to approximate within $n^{1-O(1)}$. In *International Colloquium on Automata, Languages, and Programming*, pages 2–12. Springer, 2000.

[10] Escoffier, Bruno and Monnot, Jérôme. A better differential approximation ratio for symmetric TSP. *Theoretical Computer Science*, 396(1-3):63–70, 2008.

[11] Grötschel, Martin and Jünger, Michael and Reinelt, Gerhard. Optimal control of plotting and drilling machines: a case study. *Zeitschrift für Operations Research*, 35(1):61–84, 1991.

[12] Hassin, Refael and Khuller, Samir. z-Approximations. *Journal of Algorithms*, 41(2):429–442, 2001.

[13] Held, Michael and Karp, Richard M. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied mathematics*, 10(1):196–210, 1962.

[14] Lin, Shen and Kernighan, Brian W. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, 21(2):498–516, 1973.

[15] Little, John DC and Murty, Katta G and Sweeney, Dura W and Karel, Caroline. An algorithm for the traveling salesman problem. *Operations research*, 11(6):972–989, 1963.

[16] Monnot, Jérôme. Differential approximation results for the traveling salesman and related problems. *Information Processing Letters*, 82(5):229–235, 2002.

[17] Monnot, Jérôme and Paschos, Vangelis Th and Toulouse, Sophie. Approximation algorithms for the traveling salesman problem. *Mathematical methods of operations research*, 56(3):387–405, 2003.

[18] Monnot, Jérôme and Paschos, Vangelis Th and Toulouse, Sophie. Differential approximation results for the traveling salesman problem with distances 1 and 2. *European Journal of Operational Research*, 145(3):557–568, 2003.

[19] Monnot, Jérôme and Toulouse, Sophie. The traveling salesman problem and its variations. *Paradigms of Combinatorial Optimization: Problems and New Approaches*, pages 173–214, 2014.

[20] Papadimitriou, Christos H and Steiglitz, Kenneth. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1998.

[21] Punnen, Abraham P. The traveling salesman problem: Applications, formulations and variations. In *The traveling salesman problem and its variations*, pages 1–28. Springer, 2007.

[22] Shmoys, DB and Lenstra, JK and Kan, AHG Rinnooy and Lawler, EL. *The traveling salesman problem*, volume 12. Wiley, 1985.