




Approximation algorithms with constant ratio for general cluster routing problems

Xiaoyan Zhang^{1,5} · Donglei Du² · Gregory Gutin³ · Qiaoxia Ming¹ · Jian Sun⁴ 

Accepted: 9 June 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

Due to their ubiquity and extensive applications, graph routing problems have been widely studied in the fields of operations research, computer science and engineering. An important example of a routing problem is the traveling salesman problem. In this paper, we consider two variants of the general cluster routing problem. In these variants, we are given a complete undirected graph $G = (V, E)$ with a metric cost function c and a partition $V = C_1 \cup C_2 \cdots \cup C_k$ of the vertex set. For a given subset V' of V and subset E' of E , the task is to find a walk T with minimum cost such that it visits each vertex in V' exactly once and covers each edge in E' at least once. Besides, for every $i \in [k]$, all the vertices in $T \cap C_i$ are required to be visited consecutively in T . We design two combinatorial approximation algorithms with ratios $21/11$ and 2.75 for the two variants, respectively; both ratios match the approximation ratios for

A preliminary version of this paper was published in the proceedings of 26th International Computing and Combinatorics Conference (COCOON 2020) (Zhang et al. 2020).

✉ Jian Sun
B201806011@emails.bjut.edu.cn

Xiaoyan Zhang
zhangxiaoyan@nynu.edu.cn

Donglei Du
ddu@unb.ca

Gregory Gutin
g.gutin@rhul.ac.uk

- ¹ School of Mathematical Science and Institute of Mathematics, Nanjing Normal University, Jiangsu 210023, People's Republic of China
- ² Faculty of Management, University of New Brunswick, Fredericton, Canada
- ³ Department of Computer Science Royal Holloway, University of London, Egham, UK
- ⁴ Department of Operations Research and Information Engineering, Beijing University of Technology, Beijing 100124, People's Republic of China
- ⁵ Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Beijing, China

the corresponding variants of the cluster traveling salesman problem, a special case of general cluster routing problem.

Keywords Graph routing problem · Traveling salesman problem · Combinatorial approximation algorithm · General cluster routing problem

1 Introduction

Graph routing problems are classical problems in combinatorial optimization. Since the early 1970s, scholars have conducted extensive researches on them (Frederickson 1979; Karp 1972). *Traveling salesman problem (TSP)* is arguably the most well-known routing problem (see e.g. Gutin and Punnen 2002 for a summary of the results on this problem). In this problem, there is an edge-weighted graph $G = (V, E)$ (directed or undirected) with a cost function $c : E \rightarrow \mathbb{R}_+$. The aim is to search a minimum-cost Hamiltonian cycle, i.e., a cycle that goes through each vertex only once at minimum cost. G is assumed to be a complete graph.

TSP is an NP-hard problem (Karp 1972) even if when the cost function c is metric. Heuristic and approximation algorithms are often used to solve NP-hard problems. Heuristic algorithms are algorithms based on intuition or experience, which construct a feasible solution for each instance of a combinatorial optimization problem that needs to be solved at an acceptable cost (e.g. computation time and storage space). Many scholars have done a great deal of excellent work in this area, see e.g. Lin and Kernighan (1973), Gutin et al. (2002), Punnen et al. (2003), Golden (2010). However, it is generally impossible to predict the degree of deviation between the feasible solution and the optimal solution, that is, the quality of the output solution cannot be guaranteed.

Unlike for heuristic algorithms, for approximation algorithms we have the quality of the output solution and the time complexity to obtain the solution, which means that approximation algorithms can obtain a solution of guaranteed quality. The approximation ratio is usually used to measure the quality of output solutions of approximation algorithms. When the cost function is metric, Christofides (1976) proposed a 1.5-approximation algorithm whose running time is $O(n^3)$ for TSP. This is still the best polynomial-time approximation algorithm available for metric TSP. Sahni and Gonzales (1976) have proved that, there is no constant factor approximation algorithm with polynomial-time complexity for TSP, in the case of general cost function, unless $P = NP$. It is worth noting that approximation algorithms can be used not only to obtain approximate solutions of NP-hard problems, but also to such solutions of polynomial-time solvable problems with high runtime complexity.

Before presenting the problem studied in this paper, we first introduce two generalizations of TSP which will be the subroutine of our considered problems: *general routing problem (GRP)* and *cluster traveling salesman problem (CTSP)*. In fact, the problem which we consider is the extension of these two problems, as shown later in this paper.

In GRP, we are given a specified vertex subset V' and edge subset E' in addition to graph G and the metric cost function c . The main difference between TSP and GRP is that the feasible solution of GRP does not need to traverse the vertices in V , but

only needs to traverse the vertices in V' . Besides, the feasible solution of GRP needs to pass through each edge of E' at least once. The goal is to find a solution that meets the above constraints with minimum cost. Based on the algorithm of Christofides for metric TSP, Jansen (1992) designed an approximation algorithm with ratio 1.5 for GRP. Since GRP is a generalization of TSP, it is clear that the above algorithm is also best known for GRP. Any further progress on the approximation ratio of GRP will also lead to an improvement for TSP.

In CTSP, there is a partition of the vertex set V , denoted by C_1, \dots, C_k (called clusters). The only difference between CTSP and TSP is that CTSP requires a Hamiltonian cycle whose vertices satisfy certain sequence requirements, that is, the cycle visits all vertices of every cluster consecutively. For the case that there is a pre-specified starting vertex in each cluster, Arkin et al. (1997) proposed a 3.5-approximation algorithm. For the special case that the order of clusters visited by the tour is given as a part of input, Anily et al. (1999) presented an approximation algorithm with ratio 1.5. Guttmann-Beck et al. (2000) designed a series of approximation algorithms based on whether the starting vertex or the ending vertex of each cluster is given. For instance, they proposed a $\frac{21}{11}$ -approximation algorithm when the two end vertices of each cluster are specified; they also presented a 1.8-approximation algorithm for each cluster with two given end vertices, one of which is the starting vertex and the other is the end vertex.

In this paper, we consider another generalization of TSP: the *general cluster routing problem* (GCRP) and design combinatorial approximation algorithms for this problem. In GCRP, we are given a complete undirected graph $G = (V, E)$ with the metric cost function $c : E \rightarrow \mathbb{R}_+$, a partition of V denoted by C_1, \dots, C_k (i.e., $V = C_1 \cup C_2 \cup \dots \cup C_k$), a vertex subset V' and an edge subset E' . The task is to find a walk T with minimum cost such that it visits each vertex in V' exactly once and covers each edge in E' at least once. Besides, for every $i \in [k]$, all the vertices in $T \cap C_i$ are required to be visited consecutively in T .

In the following, we consider two variants of GCRP: in the first variant, both starting and ending vertices of each cluster are given as part of the input, we design a $\frac{21}{11}$ -approximation algorithm for this case; while in the second variant, there are no pre-specified end vertices in each cluster, and we propose a combinatorial approximation algorithm with ratio 2.75.

When files are stored on computer disks, space is allocated in “storage clusters”. A file can be split, for example, across several storage clusters scattered on the same disk. The defragmentation process involves reallocating storage clusters so that all clusters allocated to the same file become contiguous. To maintain data integrity, only one file reassignment is performed at a time. It is natural to aim for minimizing the time required for defragmentation. The distance that read and write heads travel between files is to be minimized. The program used for defragmenter is part of the Microsoft Windows operating system. Storage clusters belonging to the same file form a cluster in the GCRP, and the problem of minimizing defragmentation time translates straight to GCRP.

Note that Algorithm 1 of this paper is an improvement of Algorithm 1 of Zhang et al. (2020), the COCOON 2020 version of this paper. This allows us to improve bounds in Theorems 2, 4 and 7. As a result, the bounds $\frac{21}{11}$ and 2.75 above are improvements

of the bounds for the cases with pre-specified end vertices and without pre-specified end vertices, respectively in Zhang et al. (2020).

The paper is organized as follows. In Sect. 2 we provide some preliminaries and review three algorithms. We present our main results and the corresponding proofs in Sects. 3 and 4. We summarize this paper and discuss the possible research directions in Sect. 5.

2 Preliminaries

In this section, we review some existing algorithms for the traveling salesman path problem, stacker crane problem and rural postman problem, as well as some preliminary results that will be used as subroutines in our algorithms.

2.1 The traveling salesman path problem

Compared to TSP, the *traveling salesman path problem (TSPP)* is a more general model and has also received much attention (Fumei and Alantha 2008; Traub and Vygen 2019; Hoogeveen 1991; Guttmann-Beck et al. 2000; Sebö and Van Zuylen 2019; Sebö 2013). The only difference between TSP and TSPP is that the TSP's goal is to find the Hamilton cycle with minimum cost, while the TSPP's goal is to search for the minimum-cost Hamilton path.

Depending on whether the end vertices are given, Hoogeveen (1991) considered three subproblems of TSPP as follows and presented a modification of Christofides' heuristic.

- (1) Neither the starting vertices nor ending vertices is pre-specified;
- (2) One of the starting vertices and ending vertices is pre-specified;
- (3) Both the starting vertices and ending vertices are pre-specified.

Property 1 It was proved in Hoogeveen (1991) that for the cases (1) and (2), the approximation ratio of the modified algorithm is $\frac{3}{2}$.

While case (3) is more difficult than cases (1) and (2), the modified algorithm in Hoogeveen (1991) is a $\frac{5}{3}$ -approximation algorithm. This result has not been improved in nearly two decades. In 2012, An et al. (2015) presented a deterministic approximation algorithm with ratio $\frac{1+\sqrt{5}}{2}$ for the metric s - t path TSP. Traub and Vygen (2019) obtained an improvement and proposed a $1.5 + \epsilon$ -approximation algorithm for any fix $\epsilon > 0$. Zenklusen (2019) designed an approximation algorithm with ratio 1.5 which matches the ratio of cases (1) and (2).

The following notations will be used in this paper:

- OPT denotes the optimal solution;
- L represents the sum of the paths' cost in OPT within each cluster;
- A denotes the costs of the other edges of OPT that are not in L ;
- D is the set of arcs (s_i, t_i) , $i = 1, \dots, k$, thus $c(D) = \sum_{i=1}^k c(s_i, t_i)$.

We present the following result of Guttmann-Beck et al. (2000), which will be used later in the design and analysis of the algorithms.

Theorem 1 (Guttmann-Beck et al. 2000) *There exists a polynomial-time algorithm for TSPP with given end vertices s and t , which finds two solutions S_1 and S_2 for the TSP satisfying the following inequalities:*

$$\begin{aligned} c(S_1) &\leq 2MST(G) - c(s, t) \leq 2OPT - c(s, t), \\ c(S_2) &\leq MST(G) + \frac{1}{2}(OPT + c(s, t)) \leq \frac{3}{2}OPT + \frac{1}{2}c(s, t). \end{aligned}$$

Corollary 1 $\min\{c(S_1), c(S_2)\} \leq \frac{5}{3}OPT$.

Recently, there has been some progress on a special case of TSPP: TSPP on the unit weight (undirected) graphs called the s - t path graph TSP. For this problem we know that the exact integrality ratio is 1.5, and there is an approximation algorithm that matches that ratio. The approximation ratio of the s - t path graph TSP has been improved four times over a nine-month period in 2011–2012. First, Mömke and Svensson (2016) presented a 1.586-approximation algorithm. Then Mucha (2014) designed a 1.584-approximation algorithm. An et al. (2015) improved the ratio to 1.578, and Sebö and Vygen (2014) to 1.5. More recently, Traub and Vygen (2018) designed a further improved approximation algorithm with ratio 1.497.

We consider a generalization of TSPP—the *general traveling path problem (GTPP)*: given an edge-weighted connected graph $G = (V, E, c)$ with two pre-specified end vertices $s, t \in V$, a metric weight (cost) function c , a vertex subset V' and an edge subset E' , the object is to find a path between s and t which traverse the vertices in V' and pass through each edge of E' at least once with minimum cost. If s and t are the identical vertex, this problem turns out to be the general routing problem introduced in Bienstock et al. (1991). In this paper, we mainly consider the case that s and t are distinct vertices.

Since the cost function c is metric and our goal is to find a feasible path with minimum cost, we can remove the vertices that are not in V' or $V(E')$ by using shortcutting to get a modified feasible path without increasing the total cost. Thus we define a new graph as follows:

$$G' = (\{v | v \in e, e \in E'\} \cup \{s\} \cup \{t\} \cup V', E').$$

Based on the new graph G' , we design an algorithm, Algorithm 1, for solving the GTPP. Before presenting the algorithm, we first briefly introduce the overall framework of the algorithm. Firstly, use depth-first search to obtain the connected components of G' . Secondly, contract each component to a single vertex. Thirdly, construct a new weighted complete graph G^* : the vertex set consists of these contracted components; the weight (cost) function of edge set is defined by the link with minimum cost between each pair of component. Note that we only consider such kind of edges that are adjacent with the 0-degree and 1-degree vertices in G' . Thirdly, construct a new connected graph $\hat{G}' := G' \cup T^*$, where T^* is the minimum spanning tree of G^* . Finally, we construct a feasible solution from graph $\hat{G}' \cup J$.

In Algorithm 1 and a related result, Theorem 2, we will use the following terminology.

Definition 1 (*S-join*) Let S be a subset of V . Then an edge set J with odd degrees precisely for the vertices in S is called an S -join, i.e., $\text{odd}(J) = S$ where $\text{odd}(J)$ denotes the set of odd-degree vertices in J .

Definition 2 (*wrong-degree set*) Let T be a spanning tree of G and $s \neq t \in V$. Then the vertex subset $\text{odd}(T) \triangle \{s, t\} := (\text{odd}(T) \setminus \{s, t\}) \cup (\{s, t\} \setminus \text{odd}(T))$ is called the wrong-degree set.

Algorithm 1 Algorithm of GTPP with specified vertex

Input:

- 1: An undirected graph $G = (V, E)$.
- 2: A metric cost function $c : E \rightarrow \mathbb{R}_+$.
- 3: Two end vertices s and t in G .
- 4: The required vertex subset V' and edge subset E' .

Output: A general traveling salesman path.

begin:

- 5: Construct the subgraph $G' = (\{v|v \in e, e \in E'\} \cup \{s\} \cup \{t\} \cup V', E')$.
 - 6: If there is a vertex $v \in V'$ with $d_{G'}(v) > 2$, stop: **output** there is no feasible solution.
 - 7: Compute the connected components K_1, \dots, K_r of G' .
 - 8: If there exists a component K_i such that $V(K_i) \subseteq V'$ and $d_{G'}(v) = 2 \forall v \in V(K_i)$, stop: **output** there is no feasible solution.
 - 9: Let U be the set of vertices v with degree $d_{G'}(v) \in \{0, 1\}$ or $v \notin V'$. Define a complete graph $G^* = ([r], E_r)$, the cost $c(e)$ of edge $e = (i, j)$ with $i \neq j$ is defined to the shortest link between a vertex in $K_i \cap U$ and a vertex in $K_j \cap U$.
 - 10: Compute the minimum spanning tree T^* of G^* and add the edges of T^* to G' , denote the resulting graph by \hat{G}' .
 - 11: Determine the wrong degree vertex set \hat{S} of \hat{G}' , and then find a minimum \hat{S} -join J .
 - 12: Find an Eulerian walk between s and t on $\hat{G}' \cup J$.
 - 13: Based on the Eulerian walk, construct an s - t Hamilton path P via shortcut method.
 - 14: **output** P .
- end**
-

Remark 1 In Step 6, if there is a vertex $v \in V'$ with $d_{G'}(v) > 2$, i.e., v is incident to at least three edges in E' , thus the final tour must pass through v at least twice. While a feasible tour can only pass through v exactly once for each $v \in V'$.

Remark 2 In Step 8, suppose that K_i be the componet such that $V(K_i) \subseteq V'$ and $d_{G'}(v) = 2 \forall v \in V(K_i)$. Since the final tour will connect all the components of G' , then there exists a vertex v of $V(K_i)$ which will be incident to an edge not in K_i in the final tour. This means the final tour will visit v at least twice, while a feasible tour can only pass through v exactly once for each $v \in V'$.

Lemma 1 The time complexity of Algorithm 1 is at most $O(|V|^4)$.

Proof In Step 5, the time complexity of constructing G' is $O(|V| + |E|)$. The time complexity of Step 6 and Step 8 is at most $O(|V|^2)$. In Step 7, DFS (Deep-First-Search) technique can be used to compute all the components with $O(|V|^2)$ time. In

Step 9, the time complexity of determine i and j is polynomial, since the size of vertex pair is $O(|V|^2)$. In Step 10, Algorithm 1 construct a minimum spanning tree, whose time complexity is obvious polynomial (Kruskal Algorithm with time complexity at most $O(|E| \log |E|)$). In Step 11, we construct a minimum \bar{S} -join, this operation can be done with at most $O(|V|^4)$ time via classical procedures (Sebö 1986). \square

Theorem 2 *Let P be the output path of Algorithm 1, then its cost is bounded as follows:*

$$c(P) \leq \min \left\{ 2OPT - c(s, t), \frac{3}{2}OPT + \frac{1}{2}c(s, t) \right\}.$$

Proof Let T^* denote the minimum spanning tree of G^* . By adding the edges of T^* to G' , we construct a new connected graph denoted by \hat{G}' . We make a copy of the edges of $MST(\hat{G}')$ except for the edge (s, t) and denote the resulting graph by S , which is a connected multigraph and all of its vertices are of even degree except for s and t . In this multigraph, we find an Eulerian walk connecting s and t . By the standard Short-cut procedure (Lovász 1976), it can be converted to an s - t Hamiltonian path without increasing the cost. Since the edge costs satisfy triangle inequality, its length is at most $2c(MST(\hat{G}')) - c(s, t) \leq 2OPT - c(s, t)$.

Moreover, adding the edge (s, t) to OPT to obtain a cycle with length $OPT + c(s, t)$, then decomposing the cycle into two matchings. The length of the smaller matching is at most $\frac{1}{2}(OPT + c(s, t))$. Let \bar{S} be the set of wrong-degree vertices in $MST(\hat{G}')$, construct a \bar{S} -join J with minimum cost then add it to $MST(\hat{G}')$, we obtain an upper bound for any feasible solution. By applying the Shortcut procedure (Lovász 1976) again, we have that $c(P) \leq c(MST(\hat{G}')) + \frac{1}{2}(OPT + c(s, t))$ due to the triangle inequality. So the value of the feasible solution output by Algorithm 1 is at most $\frac{3}{2}OPT + \frac{1}{2}c(s, t)$. \square

Corollary 2 *The length of the tour output by Algorithm 1 is at most $\frac{5}{3}OPT$.*

Proof By Theorem 2, if $c(s, t) \geq \frac{1}{3}OPT$, then $c(P) \leq 2OPT - c(s, t) \leq \frac{5}{3}OPT$. Otherwise, if $c(s, t) \leq \frac{1}{3}OPT$, we have $c(P) \leq \frac{3}{2}OPT + \frac{1}{2}c(s, t) \leq \frac{5}{3}OPT$. \square

Now we present a simple example that illustrates the process of Algorithm 1 (Fig. 1).

The Euler walk is $s - v_4 - v_2 - t - v_3 - v_1 - v_5 - s - t$; using the shortcut method, we can obtain a Hamiltonian s - t path: $s - v_4 - v_2 - v_3 - v_1 - v_5 - t$ in the original graph G without increasing cost.

2.2 The stacker crane problem

Let $G = (V, E, c)$ be a weighted graph in which the weight (cost) function c is metric. Let $D = \{(\overrightarrow{s_i, t_i}) : i = 1, \dots, k\}$ be a given set of special directed arcs. Let $l_i = c(\overrightarrow{s_i, t_i})$ denote the cost of each arc. The existence of arc $\overrightarrow{s_i, t_i}$ indicates that there is an object needed to be shipped from vertex s_i to t_i via a vehicle (called a

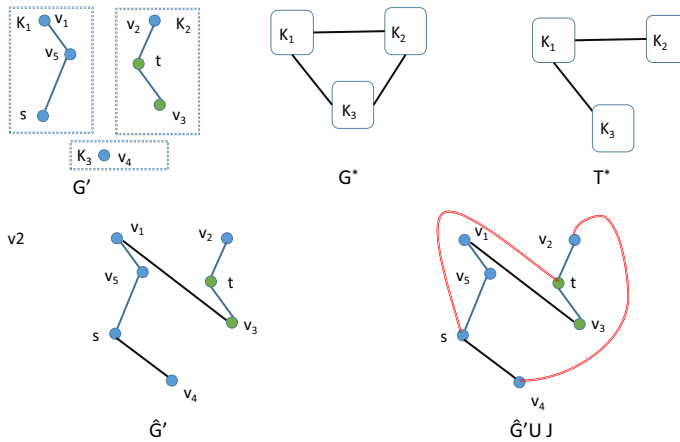


Fig. 1 An example illustrating the process of Algorithm 1, where every blue vertex belongs to V' , every blue edge belongs to E' , and every green vertex belongs to $V(E') \setminus V'$. The length of the edge connecting different vertex in G^* is calculated as Step 9 of Algorithm 1

stacking crane). The object of Stacker Crane Problem (SCP) is to compute a minimum cost walk which traverses every directed arc $\overrightarrow{(s_i, t_i)}$ at least once.

Based on the following two algorithms, Frederickson (1979) presented an algorithm for SCP, denoted by Alg-SCP in this paper for convenience, they proved that the approximation ratio of Alg-SCP is 1.8. The frameworks of these two algorithms are presented in the following (see Frederickson 1979; Guttmann-Beck et al. 2000 for more details):

- Alg-SA_1: Contract the directed arcs to simplify the problem to a TSP instance. Use the Christofides' heuristic to solve the resulted TSP instance, then construct a solution for the original problem based on the solution of TSP instance.
- Alg-LA_1: First, construct an Eulerian graph by performing the following steps:
 1. Complete the set of directed arcs into a directed cycle cover;
 2. Find a set of undirected edges that connect the cycles at the minimum cost;
 3. Add two copies for each of these edges, and orient the copies such that they are in opposite directions.

Then compute an Euler walk of the resulted graph.

In order to facilitate the analysis of approximation ratios in the following sections, we derive the theorem from Frederickson (1979) as follows:

Theorem 3 (Frederickson 1979) *Consider an instance of the Stacker Crane Problem and define following notations: Let OPT be the optimal solution, and $A = OPT - c(D)$ where $c(D)$ denotes the sum of the costs of the given directed arcs. The Alg-SA_1 returns a walk with the cost up to $\frac{3}{2}A + 2c(D)$. The Alg-LA_1 outputs a walk with the cost at most $3A + c(D)$.*

2.3 The rural postman problem

Let $E' \subseteq E$ denote the subset of specified edges, the aim of the Rural Postman Problem (RPP) is to find a walk visiting all the edges in E' with minimum cost. If $E' = E$, i.e., the walk has to traverse all the edges, this problem becomes the well-known Chinese Postman Problem. It is worth noting that RPP is an NP-hard problem, while the Chinese Postman Problem is polynomial solvable. Let $c(E') = \sum_{e_i \in E'} l(e_i)$ be the sum of cost of the edges in the specified subset. Let us review the algorithms in (Frederickson 1979; Guttmann-Beck et al. 2000).

- Alg-SA_2: Shrink the required edges to simplify the problem to a TSP instance. Use the Christofides' heuristic to solve the resulted TSP instance, and then construct a solution for the original problem based on the solution of TSP instance.
- Alg-LA_2: First, construct an Eulerian graph by performing the following steps:
 1. Complete the set of edges into an undirected cycle cover;
 2. Find a set of undirected edges that connect the cycles together at the minimum cost;
 3. Add two copies for all the edges in the set founded in the last step.

Then compute an Euler walk in the resulted graph.

Comparing Alg-LA_2 and Alg-LA_1, we find that they are very similar. The main difference is that in Alg-LA_2, E' is a set of undirected edges; while in Alg-LA_1, D is the set of special directed arcs.

Remark 3 Frederickson (1979) has proven that the above algorithms produce an approximation algorithm for RPP with ratio $\frac{3}{2}$.

For convenience, the induced algorithm mentioned in Remark 3 will be denoted as Alg-RPP.

Obviously, the desired tour in GCRP may not exist, for instance, there exists a required edge $e \in E'$ connecting two different clusters C_i and C_j , $i \neq j$, in addition the starting and ending vertices of C_i and C_j have been given. Then at least one of the clusters C_i and C_j must be visited more than once when e is not a (t_i, s_j) edge. In this paper, we only consider the situation that feasible solutions exist.

3 The general cluster routing problem with pre-specified end vertices

Note that there are two possible scenarios in this GCRP variant. First, the end-vertices of each edge in E' are within a cluster. Second, there are some edges in E' that connect two different clusters.

Let s_i and t_i denote the pre-specified end vertices of cluster C_i , $G[C_i]$ be the induced subgraph of G for $i \in [k]$, the vertices and edges that are not in V' and not in E' can be deleted from graph G , as the goal is to find a walk with minimum cost and c is a metric cost function, i.e., it satisfies the triangle inequality. Specially, for $i \in [k]$,

define \bar{C}_i as follows:

$$\bar{C}_i = (\bar{V}_i, \bar{E}_i) = \left(\{v | v \in e, e \in E'_i\} \cup V'_i \cup \{s_i\} \cup \{t_i\}, E'_i \right).$$

where $V'_i := V' \cap C_i$, $E'_i = E' \cap E(G[C_i])$.

The framework of our algorithm is as follows: First, we call Algorithm 1 to find an s_i - t_i path p_i traversing the vertices in \bar{V}_i and edges in \bar{E}_i within each cluster \bar{C}_i . Second, in order to construct a feasible solution, we add some edges to connect the paths p_i such that the resulted graph is a cycle.

Algorithm 2 Algorithm of given starting and ending vertices

Input:

- 1: A weighted complete graph $G = (V, E)$.
- 2: A metric cost function $c : E \rightarrow \mathbb{R}_+$
- 3: A partition of the vertex set $\{C_1, \dots, C_k\}$ with a pair of pre-specified end vertices s_i and t_i for each vertex subset C_i , $i = 1, \dots, k$.

Output: A general cluster routing tour.

begin:

- 4: Construct an induced subgraph $G[\cup_{i=1}^k \{s_i, t_i\}]$ of G .
- 5: For $i = 1, \dots, k$, apply Algorithm 1 on \bar{C}_i to get a path p_i and orient the (s_i, t_i) edge a direction, from s_i to t_i , to obtain the arc $\overrightarrow{(s_i, t_i)}$.
- 6: Apply Alg-SCP on the SCP instance with induced subgraph $G[\cup_{i=1}^k \{s_i, t_i\}]$ and the directed arc sets $\{\overrightarrow{(s_i, t_i)}, i = 1, \dots, k\}$, and output the solution T .
- 7: For $i = 1, \dots, k$, make a replacement on the special directed arc (s_i, t_i) in T with the path p_i , denote the resulting tour by T_s .
- 8: **Output** T_s .

end

We present a simple schematic diagram of the execution of Steps 5 and 6 in Algorithm 2 in the following (Fig. 2):

The main result of this section is described in the following form:

Theorem 4 *The cost of T_s outputted by the Algorithm 2 can be bounded as follows:*

$$c(T_s) \leq \frac{21}{11} OPT.$$

Proof In line 5, we generate a path p_i in each cluster \bar{C}_i via Algorithm 1. According to the definitions of L and D and Theorem 2, the lengths of the two solutions to the traveling salesman path problem with given starting and ending vertices are at most $2L - c(D)$ and $\frac{3}{2}L + c(D)$, respectively. Let P be the solution output by Algorithm 1. Since the minimum of a set of quantities is no more than their convex combination,

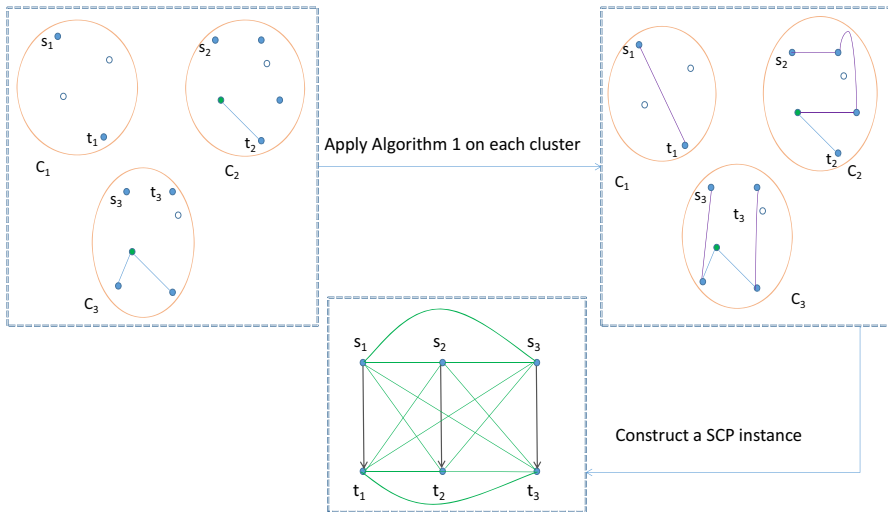


Fig. 2 In Step 6, we construct a SCP instance and call Alg-SCP to solve this instance. Then in Step 7, we recover a feasible solution of the original problem

we have

$$\begin{aligned}
 c(P) &\leq \min \left(2L - c(D), \frac{3}{2}L + \frac{1}{2}c(D) \right) \\
 &\leq \frac{9}{11}(2L - c(D)) + \frac{2}{11} \left(\frac{3}{2}L + \frac{1}{2}c(D) \right) \\
 &= \frac{21}{11}L - \frac{8}{11}c(D).
 \end{aligned}$$

Since A denotes the length of the other edges of OPT that are not in L and $c(D)$ is the total length of the directed arcs $\overrightarrow{(s_i, t_i)}, i = 1, \dots, k$, it follows that there exists a solution to the Stack Crane Problem of length at most $A + c(D)$. By Theorem 3, Alg-LA_1 and Alg-SA_1 construct two solutions for the Stack Crane Problem with cost at most $\frac{3}{2}A + 2c(D)$ and $3A + c(D)$, respectively. Let S be the solution output by the algorithm for the Stack Crane Problem. Since the minimum value of a set of quantities is no more than their convex combination, we have

$$\begin{aligned}
 c(S) &\leq \min \left(\frac{3}{2}A + 2c(D), 3A + c(D) \right) \\
 &\leq \frac{8}{11} \left(\frac{3}{2}A + 2c(D) \right) + \frac{3}{11}(3A + c(D)) \\
 &= \frac{21}{11}A + \frac{19}{11}c(D).
 \end{aligned}$$

Finally, we combine the two solutions by replacing arcs of length $c(D)$ in the solution of TSPP and the solution output by Alg-SCP for the Stack Crane Problem. We obtain an upper bound on the cost of T_s via the above inequalities.

$$\begin{aligned} c(T_s) &= c(P) - c(D) + c(S) \\ &\leq \frac{21}{11}L - \frac{8}{11}c(D) - c(D) + \frac{21}{11}A + \frac{19}{11}c(D) \\ &= \frac{21}{11}(L + A) = \frac{21}{11}OPT. \end{aligned}$$

The proof is completed. \square

Next, we consider the second case, i.e., some edges in E' connects different clusters. Such edges must be (t_i, s_j) edges. We will prove that this case can be boiled down to the Case 1. The existence of such edges has no effect on the formation of Hamiltonian path inside C_i for $i \in [k]$, thus we can temporarily remove these edges and merge C_i and C_j after constructing s_i - t_i Hamiltonian path and s_j - t_j Hamiltonian path respectively, finally we obtain an arc (s_i, t_j) .

Based on the above induction and the result of Theorem 4, we conclude that Algorithm 2 is an approximation algorithm with ratio $\frac{21}{11}$ for the general cluster routing problem with specified end vertices.

4 The general cluster routing problem without specified end vertices

In this section, we mainly focus on the case that there is no specified end vertices in each cluster C_i .

Instead of the original graph G , we consider GCRP on the subgraph $\bar{G} = \cup \bar{C}_i$, which is defined as follows:

$$\bar{C}_i = (\bar{V}_i, \bar{E}_i) = (\{v | v \in e \in E'_i\} \cup V'_i, E'_i),$$

where $V'_i := V' \cap C_i$, $E'_i = E' \cap E(G[C_i])$.

It should be noted that the tour may not exist. For instance, if $d_{\bar{G}}(v) > 2$ for some vertex $v \in \bar{V}_i$, then at least one of the clusters must be visited more than once in such case. While in this paper, we only consider solving the problem when a feasible solution exists. Therefore, in the following text, we default to the existence of a feasible solution.

We present the algorithm for the case in which there is no specified vertices as follows:

Algorithm 3 Algorithm of unspecified ending vertices**Input:**

- 1: An edge-weighted graph $G = (V, E)$.
- 2: A metric cost function $c : E \rightarrow \mathbb{R}_+$
- 3: A partition of the vertex set $\{C_1, \dots, C_k\}$.

Output: A general cluster routing tour.**begin:**

- 4: Consider the subgraphs \bar{C}_i defined as above, for $i \in [k]$.
- 5: Apply the algorithm in Guttman-Beck et al. (2000) on the induced subgraph $G[\bar{V}_i]$ to get a path p_i , denote its end vertices by a_i and b_i .
- 6: Apply Alg-RPP on the RPP instance with induced subgraph $G[\cup_{i=1}^k \{a_i, b_i\}]$ and the edge set $\{(a_i, b_i), i = 1, \dots, k\}$ to construct a solution for RPP.
- 7: Replace special edge (a_i, b_i) with path p_i for $i \in [k]$ and denoted the resulting tour by T_1 .
- 8: For $i \in [k]$, find a pair of vertices $\{s_i, t_i\}$ with maximum cost in \bar{C}_i , i.e., $c(s_i, t_i) = \max_{u, v \in \bar{V}_i} c(u, v)$. Apply Algorithm 2 with the set of end vertices pairs $\{\{s_i, t_i\}, i = 1, 2, \dots, k\}$ to obtain a tour T_2 .
- 9: **Output** the lower cost tour of T_1 and T_2 .

end

The approximation ratio of Algorithm 3 is analyzed as follows:

Theorem 5 *The upper bound on the cost of T_1 calculated in Step 5 of Algorithm 3 is as follows:*

$$c(T_1) \leq \frac{3}{2}OPT + \frac{1}{2}L + 2c(D).$$

Proof We first consider an optimal solution OPT and orientate its edges arbitrarily. In OPT , let u_i and v_i be the first and last vertices of cluster \bar{C}_i . W.l.o.g., suppose that a_i and b_i to be a pair of vertices in OPT and u_i, a_i, b_i, v_i are in this order. Then for all k clusters, by summing them up, we can get

$$L \geq \sum_{i=1}^k \left(c(u_i, a_i) + c(a_i, b_i) + c(b_i, v_i) \right).$$

Since A denotes the edge links between clusters of OPT , thus there exists a rural postman tour with special edges (a_i, b_i) of length at most

$$A + \sum_{i=1}^k \left(c(u_i, a_i) + c(a_i, b_i) + c(b_i, v_i) \right).$$

The approximation ratio of Alg-RPP is 1.5 from Remark 3 earlier. Then after applying the algorithm, the length of the tour is at most

$$1.5 \left(A + \sum_{i=1}^k \left(c(u_i, a_i) + c(a_i, b_i) + c(b_i, v_i) \right) \right).$$

Since $\sum_{i=1}^k c(p_i) \leq \frac{3}{2}L$, we replace the special edge (a_i, b_i) by the path p_i to obtain

$$\begin{aligned} c(T_1) &\leq \frac{3}{2} \left(A + \sum_{i=1}^k \left(c(u_i, a_i) + c(a_i, b_i) + c(b_i, v_i) \right) \right) - \sum_{i=1}^k c(a_i, b_i) + \frac{3}{2}L \\ &\leq \frac{3}{2}OPT + \frac{1}{2} \sum_{i=1}^k \left(c(u_i, a_i) + c(a_i, b_i) + c(b_i, v_i) \right) \\ &\quad + \sum_{i=1}^k \left(c(u_i, a_i) + c(b_i, v_i) \right) \\ &\leq \frac{3}{2}OPT + \frac{1}{2} \sum_{i=1}^k \left(c(u_i, a_i) + c(a_i, b_i) + c(b_i, v_i) \right) + 2c(D) \\ &\leq \frac{3}{2}OPT + \frac{1}{2}L + 2c(D). \end{aligned}$$

Since $c(s_i, t_i) = \max_{u,v \in \bar{V}_i} c(u, v)$, besides a_i, b_i, u_i, v_i all belong to the vertex set \bar{V}_i , thus the third inequality above holds. \square

Theorem 6 *The upper bound on the cost of T_2 computed in Step 6 of Algorithm 3 is as follows:*

$$c(T_2) \leq \frac{3}{2}OPT + 2L - 2c(D).$$

Proof For T_2 , we first determine a pair of vertices $\{s_i, t_i\}$ with maximum cost, which can be done in polynomial time. This process can be viewed as a Rural Postman Problem instance. We can operate the shortcutting on the OPT to construct a feasible solution to the corresponding RPP and therefore there is a solution for RPP whose cost is at most $\frac{3}{2}OPT$. Then, we make a replacement on each special edge (s_i, t_i) with a path connecting s_i and t_i , that includes all required vertices in \bar{C}_i . The total length of these paths is bounded from above as follows:

$$2 \sum_{i=1}^k MST(\bar{C}_i) - c(D) \leq 2L - c(D).$$

Hence the length of the tour is at most

$$\frac{3}{2}OPT - c(D) + (2L - c(D)) = \frac{3}{2}OPT + 2L - 2c(D).$$

\square

We can get the main result of this section as follows:

Theorem 7 Let T denote the tour outputted by Algorithm 3, thus

$$c(T) \leq \frac{11}{4} OPT.$$

Proof Note that $L \leq OPT$. If $2c(D) \leq \frac{3}{4}L$, Theorem 5 implies that

$$c(T_1) \leq \frac{3}{2} OPT + \frac{5}{4} L \leq \frac{11}{4} OPT.$$

Otherwise, when $2c(D) \geq \frac{3}{4}L$, Theorem 6 implies that

$$c(T_2) \leq \frac{11}{4} OPT.$$

Since the algorithm always outputs the shorter one of T_1 and T_2 , thus we complete the proof. \square

Based on the results of Theorem 7, we can conclude that the approximation ratio of the algorithm proposed in this section is 2.75.

5 Conclusion

In this paper, we present constant combinatorial approximation algorithms for two variations of the general cluster routing problem. Considering the progress made in recent years in the traveling salesman path problem based on Held–Karp relaxation, a natural question is whether a better approximation algorithm can be designed based on similar technique and other methods for the problem studied in this paper, which will be the direction of our future research.

Acknowledgements This research is supported or partially supported by the National Natural Science Foundation of China (Grant Nos. 11871280, 11771386, 11728104 and 11871081), the Natural Sciences and Engineering Research Council of Canada (NSERC) Grant 06446 and Qinglan Project.

References

- An H-C, Kleinberg R, Shmoys D-B (2015) Improving Christofides' algorithm for the s - t path TSP. J ACM 62(5):34
- Anily S, Bramel J-D, Hertz A (1999) A $5/3$ -approximation algorithm for the clustered traveling salesman tour and path problems. Oper Res Lett 24(1–2):29–35
- Arkin E, Hassin R, Klein L (1997) Restricted delivery problems on a network. Networks 29:205–216
- Bienstock D, Goemans M-X, Simchi D, Williamson D-P (1991) A note on the prize-collecting traveling salesman problem. Math Program 59:413–420
- Christofides N (1976) Worst-case analysis of a new heuristic for the traveling salesman problem. Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group
- Frederickson G-N (1979) Approximation algorithms for some postman problems. J ACM 26:538–554
- Fumei L, Alantha N (2008) Traveling salesman path problems. Math Program 13:39–59
- Golden B-L (2010) A statistical approach to the TSP. Networks 7(3):209–225

- Gutin G, Punnen A (2002) The traveling salesman problem and its variations. Kluwer, Dordrecht
- Gutin G, Yeo A, Zverovich A (2002) Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP. *Discrete Appl Math* 117(1–3):81–86
- Guttmann-Beck N, Hassin R, Khuller S, Raghavachari B (2000) Approximation algorithms with bounded performance guarantees for the clustered traveling salesman problem. *Algorithmica* 28:422–437
- Hoogeveen J-A (1991) Analysis of Christofides' heuristic: some paths are more difficult than cycles. *Oper Res Lett* 10:291–295
- Jansen K (1992) An approximation algorithm for the general routing problem. *Inf Process Lett* 41:333–339
- Karp R-M (1972) Reducibility among combinatorial problems. *Complex Comput Comput* 2:85–103
- Lin S, Kernighan B-W (1973) An effective heuristic algorithm for the traveling salesman problem. *Oper Res* 21(2):498–516
- Lovász L (1976) On some connectivity properties of Eulerian multigraphs. *Atca Math Acad Sci Hung* 28:129–138
- Mömke T, Svensson O (2016) Removing and adding edges for the traveling salesman problem. *J ACM* 63(1):2
- Mucha M (2014) 13/9-approximation for graphic TSP. *Theory Comput Syst* 55:640–657
- Punnen A, Kabadi M-S (2003) TSP heuristics: domination analysis and complexity. *Algorithmica* 35(2):111–127
- Sahni S, Gonzales T (1976) P -complete approximation problems. *J ACM* 23(3):555–565
- Sebö A (1986) Finding the join structure of graphs. *Math Program* 36:123C134
- Sebö A (2013) Eight fifth approximation for TSP paths. In: Goemans M, Correa J (eds) *Integer Programming and Combinatorial Optimization 2013*, LNCS, vol 7801. Springer, Berlin, Heidelberg, pp 362–374
- Sebö A, Van Zuylen A (2019) The salesman's improved paths through forests. *J ACM* 66(4):28
- Sebö A, Vygen J (2014) 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
- Traub V, Vygen J (2018) Beating the integrality ratio for s - t -tours in graphs. In: *Proceedings of 59th annual symposium on foundations of computer science*, pp 766–777
- Traub V, Vygen J (2019) Approaching $\frac{3}{2}$ for the s - t path TSP. *J ACM* 66(2):14
- Zenklusen R-A (2019) 1.5-Approximation for path TSP. In: *Proceedings of the 30th annual ACM-SIAM symposium on discrete algorithms*, pp 1539–1549
- Zhang X, Du D, Gutin G, Ming Q, Sun J (2020) Approximation algorithms for general cluster routing problem. In: Kim D., Uma R., Cai Z., Lee D. (eds) *Computing and combinatorics. COCOON 2020. Lecture notes in computer science*, vol 12273. Springer, Cham. https://doi.org/10.1007/978-3-030-58150-3_38

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.