

最小化碳排放的共享单车迁移问题

苏兵, Wyatt Carlson, 范佳彬, GAO Arthur, 邵艳君, 林国辉

引用本文:

苏兵, Wyatt Carlson, 范佳彬, GAO Arthur, 邵艳君, 林国辉. 最小化碳排放的共享单车迁移问题[J]. 运筹学学报, 2022, 26(3): 75-91.

SU Bing, CARLSON Wyatt, FAN Jiabin, GAO Arthur, SHAO Yanjun, LIN Guohui. Sharing bicycle relocating with minimum carbon emission[J]. Operations Research Transactions, 2022, 26(3): 75-91.

相似文章推荐 (请使用火狐或IE浏览器查看文章)

Similar articles recommended (Please use Firefox or IE to view the article)

以商圈为中心的O2O动态外卖配送路径优化模型与算法

Optimization model and algorithm for Online to Offline dynamic take-out delivery routing problem centered on business districts
运筹学学报. 2022, 26(3): 17-30 <https://doi.org/10.15960/j.cnki.issn.1007-6093.2022.03.002>

集装箱码头桥机调度问题基于完工时间下界的算法

An algorithm based on makespan lower bound for quay crane schedule problem in container terminals
运筹学学报. 2020, 24(3): 67-76 <https://doi.org/10.15960/j.cnki.issn.1007-6093.2020.03.005>

工件具有子工件工期的排序问题

Scheduling with sub-jobs' due dates
运筹学学报. 2019, 23(2): 67-74 <https://doi.org/10.15960/j.cnki.issn.1007-6093.2019.02.006>

移动传感器网络中的最大价值路径扫描覆盖算法

Algorithms for max-value path sweep coverage in mobile sensor networks
运筹学学报. 2019, 23(4): 155-164 <https://doi.org/10.15960/j.cnki.issn.1007-6093.2019.04.014>

具有机器适用限制的分布式置换流水车间问题的模型与算法

Model and algorithms of the distributed permutation flow shop scheduling problem with machine eligibility constraints
运筹学学报. 2018, 22(4): 17-30 <https://doi.org/10.15960/j.cnki.issn.1007-6093.2018.04.002>

Sharing bicycle relocating with minimum carbon emission*

SU Bing¹ CARLSON Wyatt² FAN Jiabin²
GAO Arthur² SHAO Yanjun¹ LIN Guohui^{2,†}

Abstract We formulate the sharing bicycle relocating practice as a novel optimization problem, which can be regarded as a variant of the classic TSP problem while its objective function is no longer the length of the Hamiltonian tour but the carbon emission. A well-adopted carbon emission formula that is the product of the load of the vehicle and the travel distance is employed and we propose two heuristic algorithms Greedy and TSP-based, inside both of which we set the priority to reduce the load of the vehicle for minimizing carbon emission. The feasibility of both algorithms is proven and numerical experiments are conducted to validate their performance empirically. The promise of Greedy over TSP-based algorithm is shown to the sharing bicycle companies for their daily dispatching practice.

Keywords sharing bicycle, carbon emission, integer quadratic program, traveling salesman problem, heuristics, Cplex

Chinese Library Classification O221.2

2010 Mathematics Subject Classification 90C47, 90C26, 90C30

最小化碳排放的共享单车迁移问题*

苏 兵¹ Wyatt Carlson² 范佳彬²
GAO Arthur² 邵艳君¹ 林国辉^{2,†}

摘要 本文考虑共享单车迁移问题, 它可看作是经典旅行售货商问题的一个新颖变形, 不同的是其目标函数为最小化碳排放。其中, 碳排放利用单车负载与其行驶路程的乘积进行刻画。我们提出了两个启发式算法: 贪心和基于 TSP 的算法, 每个算法的核心思想均是优先减少单车负载。从理论上证明算法的可行性并给出数据实验以验证算法的实际性能。数据实验结果表明贪心算法优于基于 TSP 的算法, 这为共享单车企业进行日常单车分配提供了理论依据。

关键词 共享单车, 碳排放, 整数二次规划, 旅行售货商问题, 启发式算法, Cplex

中图分类号 O221.2

Received: 2022-01-08

* Funding project: National Social Science Foundation of China (No. 20XGL023)

1. School of Economics and Management, Xi'an Technological University, Xi'an 710021, Shaanxi, China; 西安工业大学经济管理学院, 陕西西安 710021

2. Department of Computing Science, University of Alberta, Edmonton, T6G 2E8 Alberta, Canada; 阿尔伯塔大学计算机科学系, 加拿大阿尔伯塔埃德蒙顿 T6G 2E8

† Corresponding author E-mail: guohui@ualberta.ca

2010 数学分类号 90C47, 90C26, 90C30

China promises to reduce its carbon emission up to 40–50% by 2020, in comparison with the total emission in 2005. This is one of the worldwide efforts in resolving global warming and climate change. In China, the transportation industry alone is estimated to release about 9% of the overall carbon emission, for which initiatives including carbon emission tax and re-developing operation strategies are required.

Sharing bicycle, as a new shared economic form, is in line with the sustainable development concept of “innovation, green, open, sharing, coordination”. Sharing bicycles are conveniently used by public for a ride of one to several kilometers, from a hotspot such as a transit station, a shopping mall entrance or a residential district entrance, to another. Similar to rush hours, bicycle riding also shows a one-way pattern, that between two stations, there are many more rides from one to the other during certain periods of time, while the other periods tend to have more rides going the opposite way. Such a riding pattern causes the phenomenon of sharing bicycle overflow or shortage at many stations, despite the continuous efforts in selecting sites as stations to balance the needs^[1, 2].

To resolve the subsequent problems of “no space to return the bicycle” and “no bicycle to borrow” at a station, it is necessary for the companies to re-locate the bicycles, that is, to collect the sharing bicycles from overflowing stations and re-distribute them to shortage stations. A usual practice process is first to calculate for each station the number of bicycles in surplus or in shortage, using the current number and the historical needs at the station, then to dispatch a service vehicle of certain capacity and to plan a tour for the vehicle to visit each station once, such that if the station is overflowing, then the extra bicycles are collected, or otherwise the shortage is filled up. Since the service vehicles are fuel-powered, both their selection and the service tour over all the stations are decided based on the principle of minimum carbon emission, to achieve the most “green and efficient” dispatching. We remark that in the literature, there are existing research on other objectives besides the traditional vehicle travel time, travel distance^[3], and carbon emission, such as to minimize the deviation from the target numbers of bicycles for the stations^[4, 5].

One sees that the above dispatching practice process, after the service vehicle is chosen, becomes a vehicle routing problem (VRP). Furthermore, since it is generally true that the vehicle leaves from the sharing bicycle depot and later comes back to the depot, and the planned tour is Hamiltonian in that every station is visited exactly once, the dispatching practice is actually a variant of the traveling salesman problem (TSP). Note that we do not have to visit a station that is neither overflowing nor lacking bicycles, and thus every station under consideration either has some positive number of bicycles to be collected by the service vehicle, or has a negative number of bicycles indicating the shortage to be filled up. Our problem is thus a generalization of the delivery TSP^[6, 7, 8], in which each station is associated with either +1 or −1. Moreover, recall that our objective function is to minimize the carbon emission, which is determined by both the distance traveled and the load of the vehicle (including a fixed portion representing the weight of the vehicle, if necessary); we will present some more details on the formula for calculating carbon emission in the next

section. Therefore, our problem is not really any TSP variants that have been studied in the literature to minimize (traditionally only) the length of the tour.

In this paper, we attempt to formulate such a sharing bicycle relocating practice as an optimization problem, propose heuristic algorithms, and validate them empirically. Our goal is to show the promise of these heuristics to the sharing bicycle companies for their daily dispatching practice.

The rest of the paper is organized as follows. In the next section, we first present our optimization problem that models the sharing bicycle relocating practice, review briefly the formula for calculating carbon emission in terms of the traveled distance and the vehicle load, and provide a brief literature review on the past research related to our optimization problem. We propose two heuristic algorithms, namely Greedy and TSP-based, for our optimization problem in Section 3. In Section 4, we conduct the empirical experiments based on two real datasets of sharing bicycle stations in Xi'an, China, summarize the results, and discuss important discoveries that could be helpful for the real dispatching practice. Lastly in Section 5, we conclude the paper to provide some possible broader implications of the work and identify potential areas for future work.

1 Preliminaries

1.1 Problem description

We first formulate the optimization problem, denoted as SBR, from the Sharing Bicycle Relocating practice. In the SBR problem, we are given an edge-weighted complete graph $G = (V, d(\cdot))$ representing the underlying traffic network, where a vertex of V represents a station, the edge weight $d(\cdot)$ measures the distance (or the travel time) between two stations, and the edge weights satisfy the triangle inequality.

Among the vertex set $V = \{v_0, v_1, v_2, \dots, v_n\}$, where $n \geq 2$, v_0 represents the sharing bicycle depot which is assumed to have unlimited storage and an unlimited supply of sharing bicycles, and each v_i , $i = 1, 2, \dots, n$, represents a sharing bicycle station associated with an integer q_i stating the known surplus number of bicycles, when $q_i > 0$, or the known shortage, when $q_i < 0$.

There is a service vehicle of capacity Q and it is obliged to collect the surplus bicycles from all the overflowing stations, re-distribute them to shortage stations to fully satisfy their needs, and then come back to the depot. To this purpose, the vehicle has to carry extra $q_0 = \max\{0, -\sum_{i=1}^n q_i\}$ bicycles out of the depot, and bring back to the depot $\max\{0, \sum_{i=1}^n q_i\}$ bicycles. It is assumed without loss of generality that $|q_i| \leq Q/2$ for each i and $|\sum_{i=1}^n q_i| \leq Q$, for the sake of feasibility, that is, there exists a feasible Hamiltonian tour for the service vehicle*. The cost for the vehicle traveling from v_i to v_j is measured by carbon emission,

*One sees that this is necessary by looking at a small instance of three stations, two of which each has a surplus $Q/2$ while the third has a shortage of $1 + Q/2$. Then a service vehicle of capacity strictly less than Q is not able to fulfill the re-locating purpose, under the Hamiltonian constraint that every station is visited exactly once and the constraint that the service vehicle leaves the depot with 0 bicycles and comes back to the depot with $Q/2 - 1$ bicycles.

which is the product of the load of the vehicle (in the number of bicycles) and the travel distance $d(v_i, v_j)^{[9, 10]}$ (see also discussion in Subsection 2.2). In general, the vehicle itself has weight that is equivalent to a fixed number a_0 of bicycles, and thus the load is $a_0 + c$, with c being the actually number of bicycles at the back of the vehicle.

The goal of the SBR problem is to plan a Hamiltonian tour for the service vehicle such that 1) each station is visited exactly once, 2) its need is satisfied in full (that is, either the surplus bicycles are fully collected or the shortage is fully filled), and 3) the total carbon emission of the service vehicle is minimized. A Hamiltonian tour satisfying the first two conditions is said to be feasible. Given the easily seen NP-hardness of the SBR problem (see also discussion in Subsection 2.3), we propose and empirically validate two heuristic algorithms to compute feasible and low carbon emission solutions to the problem.

For simplicity we denote $d_{ij} = d(v_i, v_j)$. We next formulate an integer quadratic program (IQP) for the SBR problem, borrowing the ideas for the classic TSP problem. To this purpose, we define a binary variable $x_{ij} = 1$ if and only if the arc (i.e., directed edge) from v_i to v_j is included in the Hamiltonian tour. We note from $n \geq 2$ that $x_{ii} = 0$ (i.e., no loop), and $x_{ij} = 1$ implies $x_{ji} = 0$ (i.e., each edge is used at most once), for all $i, j \in \{0, 1, 2, \dots, n\}$. Let c_i denote the to-be-determined number of bicycles at the back of the vehicle leaving the station v_i , which is a non-negative integer less than or equal to Q ; specifically, $c_0 = q_0$. If the vehicle travels along the edge from v_i to v_j (that is, $x_{ij} = 1$) and either picks up the surplus or fills the shortage at v_j , then its number of bicycles at the station v_j is $c_j = c_i + q_j$, which must also be a non-negative integer less than or equal to Q , and the traveling along this arc (v_i, v_j) incurs an amount $(a_0 + c_i)d_{ij}$ of carbon emission.

$$\min \sum_{i=0}^n \sum_{j=0}^n (a_0 + c_i) \cdot d_{ij} x_{ij} \quad (1)$$

$$\text{s.t.} \quad \sum_{j=0}^n x_{ij} = 1, \quad \forall i \in \{0, 1, 2, \dots, n\}, \quad (2)$$

$$\sum_{i=0}^n x_{ij} = 1, \quad \forall j \in \{0, 1, 2, \dots, n\}, \quad (3)$$

$$\sum_{i,j \in U} x_{ij} \leq |U| - 1, \quad \forall U \subset \{0, 1, 2, \dots, n\}, \quad (4)$$

$$c_0 = \max\{0, -\sum_{i=1}^n q_i\}, \quad (5)$$

$$c_j = \sum_{i=0}^n x_{ij} (c_i + q_j), \quad \forall j \in \{1, 2, \dots, n\}, \quad (6)$$

$$0 \leq c_j \leq Q, \quad \forall j \in \{1, 2, \dots, n\}, \quad (7)$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j \in \{0, 1, 2, \dots, n\}. \quad (8)$$

In the above IQP formulation, (1) is the objective to minimize the total carbon emission, where the load of the vehicle is the sum of the vehicle weight a_0 and the number of bicycles c_i it carries when leaving v_i ; the constraints (2) and (3) state that every vertex v_i is visited by the service vehicle exactly once; the constraint (4) forces, together with (2) and (3), the tour to be Hamiltonian; the constraint (5) says that the service vehicle starts from the depot, carrying with it q_0 bicycles, and the constraints (6) and (7) express how the load changes along the tour and, the number of bicycles carried by the vehicle should not exceed

the vehicle capacity at any point, nor go below 0.

1.2 Carbon emission formula

Carbon emission, among several other emissions from fuel-powered engines, is estimated to be directly proportional to fuel consumption^[11]. Many elements influence fuel consumption, including travel related factors such as driving style, vehicle characteristics such as engine size, road geometry such as gradients, and meteorological conditions such as ambient temperature. After averaging out most factors and regressing on the load of the vehicle, fuel consumption, and thus the subsequent carbon emission per unit distance, often takes an affine form of $a_0 + a_1 \cdot c$ ^[9, 10], where a_0 is the amount of fuel consumption by the vehicle itself, a_1 is the additional amount of fuel consumption with unit goods on the vehicle, and c is the total number of units of goods.

We adopt this affine emission per unit distance in our formulation, but scale it (i.e., dividing it by a_1) to use c to denote the number of bicycles at the back of the service vehicle and a_0 to denote the weight of the service vehicle itself as an integral number of bicycles. That is, the carbon emission per unit distance for the service vehicle is its load of $a_0 + c$.

In practice, there are at least three different capacity vehicles that are available for sharing bicycle companies in Xi'an, China and Chengdu, China, one of which has capacity of $Q = 15$ and itself is so light that the weight is negligible, that is, $a_0 = 0$. The other two have capacities of 25 and 50, and their own weights are equivalent to the weights of 10 and 20 bicycles, respectively. In the empirical experiments, all three of these vehicles are examined.

1.3 Related work

Our SBR problem is a variant of VRP, which has numerous applications in operations research and has been studied extensively for more than six decades. For most VRP variants, one major objective is to minimize the total distance traveled by the service vehicle (or by a group of multiple vehicles). This includes the classic TSP problem, for which Christofides' algorithm is a 1.5-approximation^[12], that is, it returns a Hamiltonian tour of length within 1.5 times of the minimum. When the service vehicle needs to pick up goods from some customers (e.g., extra bicycles at the overflowing stations in our case) and drop them off to the other customers (e.g., the shortage stations in our case), the problem is referred to as delivery TSP^[13]. A special case of the delivery TSP, also a special case of our SBR problem, in which each q_i is either +1 or -1, was studied by Anily and Hassin^[6], who presented a 2.5-approximation when the service vehicle has capacity $Q = 1$; Chalasani and Motwani^[7] continued the study and proposed a better 2-approximation when the service vehicle has capacity $Q = 1$, a 9.5-approximation when the service vehicle has an arbitrary but finite capacity Q , and a 2-approximation when the service vehicle has infinitely large capacity $Q = +\infty$. The 9.5-approximation is improved to a 5-approximation by Charikar et al.^[8].

The SBR problem we formulated seems novel and we found no existing work on it in the literature (to the best of our knowledge). When carbon emission has a weak dependence

on the number of bicycles carried by the vehicle but is proportional to the traveled distance (that is, $a_0 \gg Q$), our SBR problem reduces to the classic TSP problem. Therefore, the SBR problem is NP-hard and likely more difficult to approximate.

2 Two heuristic algorithms

Given that carbon emission along an edge is measured as the product of the load of the vehicle and the weight of the edge, intuitively one should seek to reduce the vehicle load as much as possible and only pick up surplus bicycles from an overflowing station as the last choice; one should also seek to drive to the “closest” station from the current station, for either pick-up or drop-off purposes. In the following two algorithms, reducing the vehicle load is common to both and has a higher priority, while the “closest” station is defined differently, one globally in the greedy algorithm and the other guided in the TSP-based algorithm.

2.1 Greedy algorithm

Recall that the number of bicycles at the back of the service vehicle leaving the station v_i is c_i , satisfying $0 \leq c_i \leq Q$. The greedy algorithm decides the next station to drive to as follows.

Let S^+ denote the set of unvisited stations with surplus, and S^- denote the set of unvisited stations v_j with shortage $q_j \geq -c_i$ (that is, its shortage can be filled if chosen so in this iteration).

Recall that we have the priority to reduce the vehicle load. If $S^- \neq \emptyset$, then the vehicle moves to the closest station $v_j \in S^-$ (that is, $v_j = \arg \min\{d(v_i, v_j) \mid v_j \in S^-\}$), drops down exactly $-q_j$ bicycles, and updates the number of bicycles at the back as $c_j = c_i + q_j$; otherwise, the vehicle moves to the closest station $v_j \in S^+$ (that is, $v_j = \arg \min\{d(v_i, v_j) \mid v_j \in S^+\}$), collects the surplus q_j , and updates the number of bicycles at the back as $c_j = c_i + q_j$. A high level description of an iteration of the algorithm is depicted in the following.

Algorithm Greedy:

Each iteration (the vehicle is at v_i , with $c_i \leq Q$ bicycles at the back): set $S^+ = \{v_j \mid v_j \text{ is not visited, and } q_j > 0\}$; set $S^- = \{v_j \mid v_j \text{ is not visited, and } -c_i \leq q_j < 0\}$; if $S^- \neq \emptyset$, then let $v_j = \arg \min\{d(v_i, v_j) \mid v_j \in S^-\}$; otherwise, let $v_j = \arg \min\{d(v_i, v_j) \mid v_j \in S^+\}$; the vehicle moves to v_j , and updates the number of bicycles as $c_j = c_i + q_j$.

Theorem 1 *The algorithm Greedy runs in $O(n^2)$ time and outputs a feasible tour, where n is the number of stations.*

Proof Note that when $S^- = \emptyset$, if there is no more unvisited shortage station, then the vehicle is able to pick up the surplus bicycles from all the remaining unvisited stations and thus $c_i + q_j \leq Q$ by the assumption $|\sum_{i=1}^n q_i| \leq Q$; if there are unvisited shortage stations, then we conclude from $|q_h| \leq Q/2$ for any $h \in \{1, 2, \dots, n\}$ that $c_i < Q/2$ and thus $c_i + q_j < Q$. In other words, the vehicle is able to collect all the surplus bicycles from the

station $v_j \in S^+$. This proves that the algorithm Greedy terminates and outputs a feasible tour.

Since each iteration takes $O(n)$ time, the overall running time for the algorithm Greedy is in $O(n^2)$. \square

2.2 TSP-based algorithm

Assume an approximate TSP tour has been computed (in our implementation, by Christofides' 1.5-approximation), which takes only the edge weights into consideration with the objective to minimize the total travel distance, and the tour is $C = \langle v_0, v_1, v_2, \dots, v_n, v_0 \rangle$. Note that the depot v_0 is regarded as both a surplus and shortage station, and at the start, the vehicle carries $c_0 = \max\{0, -\sum_{j=1}^n q_j\}$ bicycles.

Starting at v_0 , we fix a direction on C for the vehicle, such that the stations are visited by the vehicle in a sequential order. That is, C is the guide tour in our algorithm. The vehicle maintains v^+ and v^- to be the first unvisited surplus station and the first unvisited shortage station, respectively, and they are initialized to be the first ones in the sequential order from v_0 . At the beginning of an iteration (a high level description of an iteration in the TSP-BASED algorithm is depicted in the following), the vehicle stands at a station, has c bicycles at the back, decides the next station to drive to and updates v^+ and v^- as follows.

Again recall that we have the priority to reduce the vehicle load. There are three possible situations below with decreasing preference:

1) The vehicle can fill the shortage of v^- . From its current location, the vehicle moves to v^- , and subsequently updates the number c of bicycles and updates the station v^- to be the next unvisited shortage station in the sequential order from the current location.

2) The vehicle cannot fill the shortage of v^- . If any, let v_j be the unvisited first shortage station on the way from the current station to v^+ , of which the shortage can be filled by the vehicle; the vehicle moves to v_j , and subsequently updates the number c of bicycles.

(Comment: In this case, v_j comes after v^- in the sequential order, but is visited before v^- ; therefore, v^- doesn't need to be updated.)

3) The vehicle cannot fill the shortage of v^- , neither can it fill the shortage of any shortage station on the way from the current station to v^+ , or there is no shortage station on the way from the current station to v^+ at all. The vehicle collects the surplus of the first unvisited surplus station in the sequential order from the current location, and subsequently updates the number c of bicycles; furthermore, if this first surplus station is v^+ , then it updates the station v^+ to be the next unvisited surplus station in the sequential order from the current location.

(Comment: If this first surplus station is not v^+ , then it comes after v^+ in the sequential order; therefore, v^+ doesn't need to be updated.)

Algorithm TSP-based:

Each iteration: (stations v^+ and v^- have been defined): if the vehicle can fill the shortage of v^- , it moves to v^- , updates c and v^- ; if the vehicle can fill the shortage of v_j on the way to v^+ , it fills the shortage of v_j , and updates c ; if the vehicle cannot fill the shortage

of any station on the way to v^+ , it moves to the first surplus station v_j , and updates c ; if $v_j = v^+$, then it updates v^+ .

Theorem 2 *The algorithm TSP-based runs in $O(n^{2.5}/\log n)$ time and outputs a feasible tour, where n is the number of stations.*

Proof When v^- becomes undefined (or $v^- = v_0$, if the depot v_0 is viewed also as a shortage station), the vehicle is able to pick up the surplus bicycles from all the unvisited stations, including v^+ , by the assumption $|\sum_{i=1}^n q_i| \leq Q$; otherwise, we conclude from $|q_h| \leq Q/2$ for any $h \in \{1, 2, \dots, n\}$ that $c < Q/2$ and thus the vehicle is able to collect all the surplus bicycles from the first unvisited surplus station on the way to v^+ . This proves the feasibility.

Christofides' 1.5-approximation algorithm involves an $O(n^{2.5}/\log n)$ -time phase for computing a maximum weight matching^[14], which is dominant. Afterwards, each iteration takes $O(n)$ time, and thus the overall running time for the algorithm TSP-based is in $O(n^{2.5}/\log n)$. \square

3 Numerical experiments

We first describe the datasets and the settings we used in the empirical experiments, then summarize the main statistics on the results, followed by additional experiments and points of discussion.

3.1 Datasets and settings

We have obtained two real datasets of 25 and 110 sharing bicycle stations in Xi'an, China. The smaller one (referred to as real 25-station dataset) is from a northern small district of Xi'an city, and it doesn't have its own depot. The larger one (referred to as real 110-station dataset) is a superset of the former, it covers a much larger northern area of Xi'an city, and it has its own depot, which sits roughly at the geometric center. In our experiments, we add a pseudo-depot to the 25-station dataset at the average latitude and longitude of the 25 stations (this pseudo-depot is very close to the 110-station dataset depot, within a few tenths of a degree both latitude and longitude).

We note that the sharing bicycle stations are quite well chosen, so that the surplus and the shortage (collected at one time for re-locating purpose) are mostly a single digit, with the maximum being slightly over a dozen. In both datasets, we have the latitude and the longitude for each station, which are used to calculate the inter-station approximate Manhattan distances between stations by using a factor of 110.574 km for one latitude degree and $111.320 \times \cos(\text{latitude})$ km for one longitude degree. We remark that the Manhattan distance matches well with the grid-like street layout in Xi'an city.

Besides the above two, we simulated datasets with $n \in [16, 20]$ stations using the first n stations from the real 25-station dataset plus the depot, which are referred to as the n -station datasets, respectively.

For each n -station dataset, $n \in [16, 20]$, we independently randomly generate 100 simulated instances on the same set of stations, by only simulating the surplus/shortage value for every station within the closed interval $[-Q/2, Q/2]$ and ensuring the total is within $[-Q, Q]$. Recall that we have three different capacities $Q = 15, 25, 50$, for which the weights of the service vehicles are equivalent to $a_0 = 0, 10, 20$ bicycles, respectively. For the 25-station dataset and the 110-station dataset, we also independently randomly generate 100 simulated instances for each capacity $Q = 15, 25, 50$. We note that the real instances of the n -station datasets for $n \in [16, 20]$, that is, instances with the real surplus/shortage values, correspond to $Q = 15$; and the real instances of the 25-station dataset and the 110-station dataset correspond to $Q = 25$. Each real instance replaces one corresponding simulated instance in the experiments.

3.2 Algorithms and their implementation

We examine the performance of the two proposed heuristic re-locating algorithms Greedy and TSP-based, both were implemented in a Python (version 3.7.0) program. We employed the “`math`”, “`random`”, “`csv`” and “`sys`” libraries. Both algorithms can run on all the instances as their time complexities are low.

For each instance of the n -station datasets with $n \in [16, 20]$, we computed the optimal solution in CPLEX, where the IQP formulation was implemented using the CPLEX Studio IDE 12.10.0 (the implementation is denoted as IQP-Cplex from here on) and the instances were run from the prompt using the `oplrun` command. In more details, the IQP formulation is written in the OPL language with some execution blocks written in JavaScript for pre-processing (calculating distances and problem setup) and post-processing (writing results into a file). Due to the high time and space complexity in the size of the instance, two of the CPLEX parameters were changed to more efficiently use memory while solving each instance: the memory reduction switch was set to “`conserve memory`” wherever possible, and the variable selection strategy was set to “`strong branch`” (partially solving subproblems to see which branch is the most promising).

3.3 Results

For every instance, we collect the following values whenever applicable:

- IQP-Cplex: the service path, the total carbon emission, the travel distance by the vehicle, and the running time;
- Greedy: the service path, the total carbon emission, and the travel distance by the vehicle;
- TSP-based: the length of the guide TSP tour by the component Christofides’ algorithm, the service path, the total carbon emission, and the travel distance by the vehicle.

We remark that running either the Greedy or the TSP-based algorithm on a collection of 100 instances associated with a couple (n, Q) took within seconds on a computer with

a Ryzen 5 2600 CPU @3.4GHz and 16Gb of 3000MHz DDR4 Ram; we therefore did not record their detailed running time for the experiments.

Table 1 summarizes the experimental results on the n -station datasets with $n \in [16, 20]$ (and the service vehicle has capacity $Q = 15$). The second row lists the lengths of the TSP tours by Christofides' algorithm; we note that for all the 100 simulated instances associated with $(n, 15)$, their TSP tours are the same since the simulation is on the surplus/shortage values only. The next three rows contain the average emissions of the optimal solutions, the solutions by Greedy, and the solutions by TSP-based, and their standard deviations.

Table 1 The performance of all three algorithms IQP-Cplex, Greedy and TSP-based on the 100 simulated instances for the n -station datasets, where $n \in \{16, 17, \dots, 20\}$. In this experiment, a light truck is dispatched with its capacity $Q = 15$ and its own weight ignored. Rows 3–5 (6–8, respectively) contain the average carbon emissions (average travel distances, respectively) and their standard deviations over the 100 instances, in the solutions by the three algorithms. The last row records the average running times of IQP-Cplex and the standard deviations

#Stations	16	17	18	19	20
Christofides' distance	46.76	46.84	44.26	45.47	46.69
IQP-Cplex emission	110.75±18.31	110.87±21.79	121.26±22.83	119.07±19.61	121.63±19.01
Greedy emission	188.37±44.09	189.04±43.75	213.50±51.24	220.65±55.84	218.74±51.07
TSP-based emission	229.75±52.15	255.32±69.64	258.47±64.29	252.58±60.50	256.84±58.52
IQP-Cplex distance	50.81±5.41	52.43±4.94	59.23±5.79	59.22±6.57	61.04±6.20
Greedy distance	49.91±6.15	51.40±5.63	57.25±6.46	58.75±6.77	59.97±6.52
TSP-based distance	58.43±7.09	63.92±7.61	66.59±9.05	66.50±9.82	68.54±8.82
IQP-Cplex time (seconds)	12.68±36.90	81.15±176.65	1 673.73±5 331.30	2 028.63±5 685.02	3 756.00±8 362.41

For $n = 20$, the detailed emission values by the three algorithms are plotted in Figure 1(a), where the instances are sorted in their increasing optimal emission order. One sees that Greedy and TSP-based solutions emit on average about 1.76 and 2.15 times, respectively, that of the optimal solutions. Interestingly, in terms of travel distance, Greedy tours are often shorter, and in fact about 2% shorter than the tour in the optimal solutions on average; while TSP-based tours are on average about 15% longer than the tour in the optimal solutions. The detailed distance values for these 100 instances by the three algorithms are plotted in Figure 1(b), where the instances are sorted in their increasing Greedy distance order.

From Fig. 1(a), one also sees that, for these 100 instances, the emissions by Greedy and TSP-based fluctuate up and down, but interestingly the general tendencies are increasing. That is, in general, the emissions by Greedy and TSP-based correlate positively to the minimum emissions, with their correlation coefficients 0.686 and 0.622 respectively. For distances, one sees from Fig. 1(b) that 1) there doesn't seem to be any correlation between any two groups of the distances (their pairwise correlation coefficients are 0.272, -0.017 , 0.386, respectively), 2) TSP-based looks inferior to both Greedy and IQP-Cplex on majority of the instances, and 3) IQP-Cplex wins over Greedy on 42 instances and thus they are competitive to each other.

The last row of Table 1 records the average running time of the IQP-Cplex solver, roughly seconds to minutes on a 16/17-station instance, half an hour on an 18/19-station instance, and one hour on a 20-station instance. Figure 2 plots the detailed running times on the 100 simulated instances of the 20-station dataset by the IQP-Cplex algorithm, sorted in their running time order. One sees that 1) there are huge differences in the solution spaces

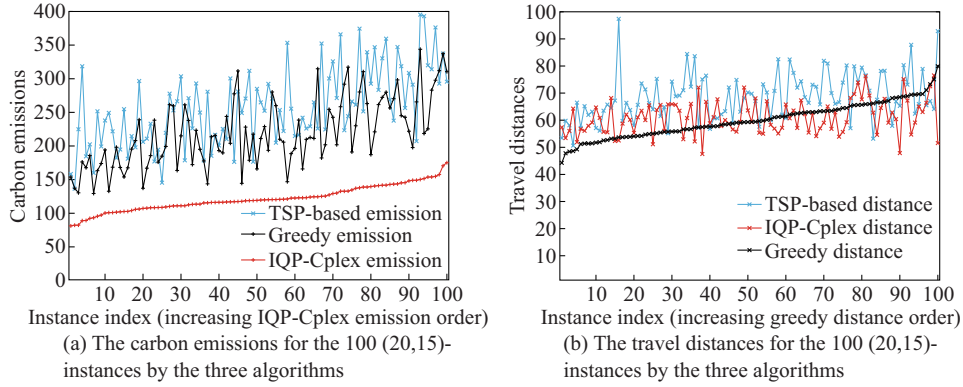


Fig. 1 The performances of the three algorithms IQP-Cplex, Greedy and TSP-based on the 100 simulated instances of the 20-station dataset, when a light service vehicle of capacity $Q = 15$ is deployed: (a) The carbon emissions and (b) the travel distances. Note that in the two plots, the instances are sorted in different orders for better viewing

explored by IQP-Cplex, 2) 79 out of the 100 instances were solved by IQP-Cplex under an hour, while 13 instances required longer than two hours. The standard deviations in the last row of Table 1 are very large, suggesting the huge running time variation across different instances; indeed, one sees from Fig. 2 that the shortest and longest times for a 20-instance are 86 seconds and 14 hours, respectively.

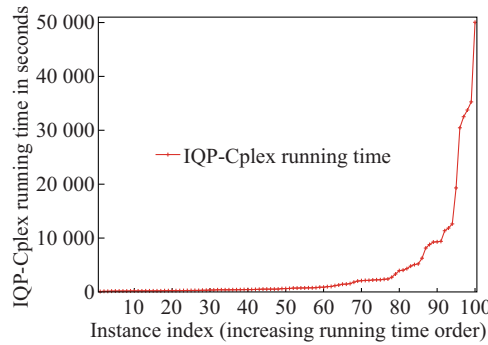


Fig. 2 The running time of IQP-Cplex on the 100 simulated instances of the 20-station dataset, where the instances are sorted in increasing running time order

From the 100 instances associated with the couple (20, 15), we pick two of them to illustrate the service paths by the three algorithms in Fig. 3. On one instance Greedy significantly outperforms TSP-based in carbon emission, and the carbon emissions for these three paths are 121.13, 146.90, 353.48 by IQP-Cplex, Greedy, TSP-based, respectively; on the other instance TSP-based significantly outperforms Greedy in carbon emission, and the carbon emissions for these three paths are 117.22, 176.41, 277.57 by IQP-Cplex, Greedy, TSP-based, respectively. A general observation from these two instances is that, a good service path does not seem to cross over itself too often.

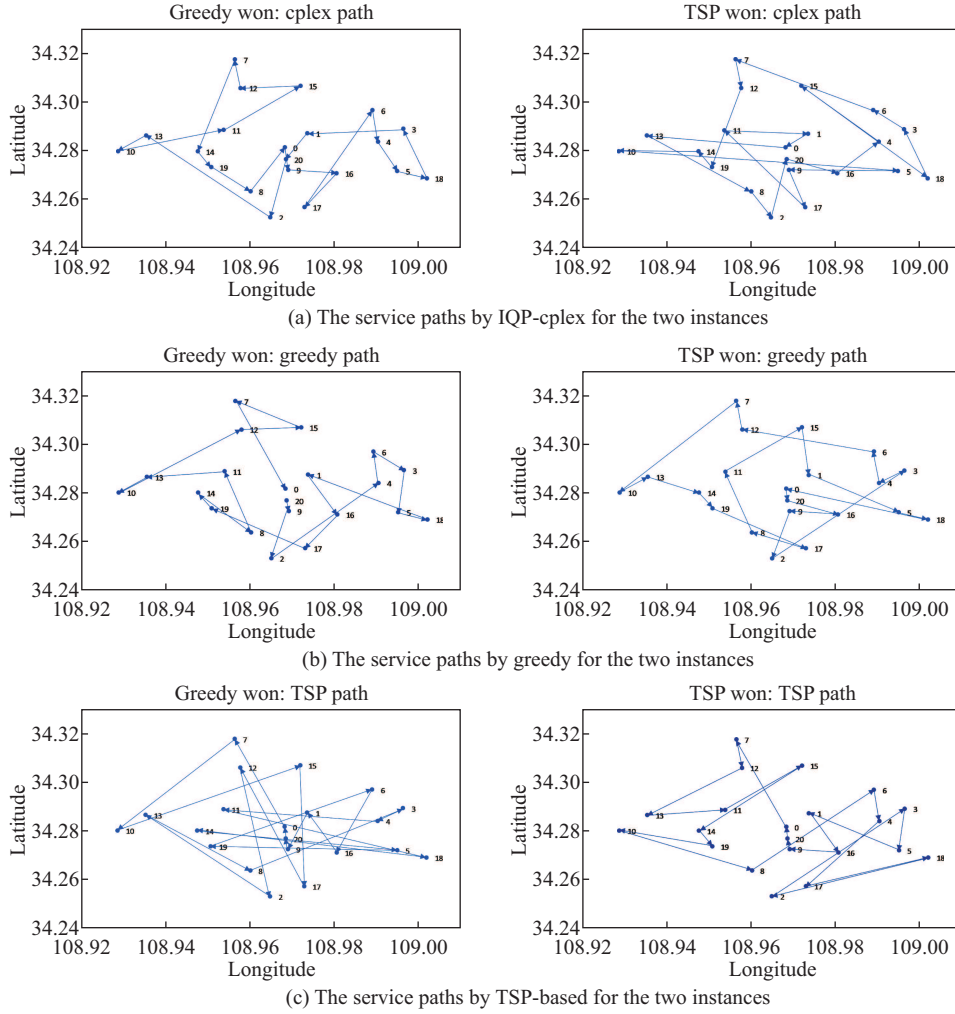


Fig. 3 The service paths generated by the three algorithms on two simulated instances of the 20-station dataset, on the left of which Greedy significantly outperforms TSP-based in carbon emission and on the right TSP-based significantly outperforms Greedy. (a) the paths by IQP-Cplex, with emission 121.13 and travel distance 52.52 (left) and emission 117.22 and travel distance 65.68 (right); (b) the paths by Greedy, with emission 146.90 and travel distance 53.67 (left) and emission 277.57 and travel distance 55.87 (right); (c) the paths by TSP-based, with emission 353.48 and travel distance 97.41 (left) and emission 176.41 and travel distance 68.82 (right). In the figures, the dots represent labelled stations with the depot labelled 0; a directed edge tells from which station to another the service vehicle moves along the streets

Since IQP-Cplex did not finish any instance associated with (25, 15) within the 24-hour time limit, we collected statistics for only the Greedy and TSP-based algorithms on the instances for the 25- and 110-station datasets. Table 2 summarizes the experimental results. The second row lists the lengths of the TSP tours by Christofides' algorithm. The next three groups of four rows each contain the results for deploying a light, a medium, and a heavy

Table 2 The performance of the two algorithms Greedy and TSP-based on the 100 instances for each of the 25-station and 110-station datasets. In the rows 3–6 (7–10, 11–14, respectively), a light (medium, large, respectively) truck is dispatched with its capacity $Q = 15$ (25, 50, respectively) and its weight equivalent to $a_0 = 0$ (10, 20, respectively) bicycles

	Q/a_0	25-station	won	110-station	won
Christofides' distance		54.99		175.52	
Greedy emission	15/0	272.03±56.89	82	686.39±98.45	99
TSP-based emission		342.90±72.79	18	1 099.99±206.38	1
Greedy distance		73.23±7.38	87	192.13±20.45	99
TSP-based distance		88.29±11.81	13	291.31±38.37	1
Greedy emission	25/10	1 199.28±161.65	82	3 129.66±295.90	100
TSP-based emission		1 428.43±222.73	18	4 719.92±578.61	0
Greedy distance		74.42±8.11	84	194.47±15.73	100
TSP-based distance		86.43±12.26	16	287.97±31.36	0
Greedy emission	50/20	2 383.08±326.58	89	6 367.98±555.97	100
TSP-based emission		2 911.67±419.40	11	9 568.11±1 327.58	0
Greedy distance		73.18±8.26	89	196.91±15.80	100
TSP-based distance		86.51±10.80	11	285.32±36.45	0

truck, respectively, with capacity $Q = 15, 25$, and 50 , respectively. These results include the average emissions and the average travel distances in the solutions by Greedy and the solutions by TSP-based, and the standard deviations. For both carbon emission and travel distance, one sees that the solutions by Greedy are mostly better than the solutions by TSP-based, across all three different capacity values; on the 25-station instances, Greedy performs better on more than 80% of the instances, and on the 110-station instances, Greedy performs better on all but one instance. Interestingly, one sees that the capacity Q does not seem to affect the travel distance much, for both algorithms. However, the capacity Q apparently affects emission. From this data (and the data on the n -station datasets for $n \in \{16, 17, \dots, 20\}$), we might be able to safely make recommendations to the sharing bicycle companies to adopt Greedy for their dispatching practice.

3.4 Discussion

3.4.1 Carrying extra bicycles doesn't really help

One might argue that starting with $q_0 = \max\{0, -\sum_{i=1}^n q_i\}$ bicycles is too restricted for the service truck, and perhaps carrying a certain number of extra bicycles could be helpful. For example, in the illustration 3-station instance in which two stations each has a surplus $Q/2$ while the third has a shortage of $1 + Q/2$, if the service vehicle is allowed an extra bicycle then one could pick up $Q/2$ from one surplus station, drop off the $1 + Q/2$ bicycles to the shortage station, and move to the last station to pick up $Q/2$ bicycles before returning to the depot.

We experimented with carrying extra 5 and 10 (but no more than $Q - q_0$, to ensure that the service vehicle capacity is never exceeded) bicycles when leaving the depot, for all three different capacities, and collected the same statistics in Table 3.

From Table 3, and comparing against Table 2, one sees that carrying extra bicycles out

Table 3 The performance of the two algorithms Greedy and TSP-based on the 100 instances for each of the 25-station and 110-station datasets, using a combination of a light (medium, large, respectively) truck and extra 5 (10, respectively) bicycles

	$Q(a_0 + \#bicycles)$	25-station	won	110-station	won
Greedy emission	15(0+5)	303.65±68.93	83	730.16±108.21	100
TSP-based emission		394.12±78.14	17	1 180.03±213.49	0
Greedy distance		72.28±7.24	93	192.07±20.12	100
TSP-based distance		91.31±11.73	7	292.48±41.24	0
Greedy emission	15(0+10)	355.52±90.61	81	763.26±118.17	99
TSP-based emission		453.38±73.05	19	1 255.51±214.03	1
Greedy distance		71.91±7.55	94	190.68±19.45	99
TSP-based distance		93.09±11.56	6	293.69±39.23	1
Greedy emission	25(10+5)	1 231.78±175.21	82	3 162.01±309.60	100
TSP-based emission		1 501.64±220.49	18	4 835.77±569.09	0
Greedy distance		73.99±8.32	84	194.57±16.52	100
TSP-based distance		87.63±11.87	16	289.05±30.53	0
Greedy emission	25(10+10)	1 245.23±165.22	86	3 199.52±277.91	100
TSP-based emission		1 541.14±225.77	14	4 936.85±623.24	0
Greedy distance		72.46±7.61	90	193.88±14.94	100
TSP-based distance		88.13±11.31	10	290.35±33.19	0
Greedy emission	50(20+5)	2 411.84±333.12	91	6 409.37±588.91	100
TSP-based emission		2 993.68±435.39	9	9 808.92±1382.58	0
Greedy distance		73.33±8.81	89	196.74±16.60	100
TSP-based distance		87.54±11.34	11	287.73±38.43	0
Greedy emission	50(20+10)	2 476.63±356.19	92	6 417.28±614.07	100
TSP-based emission		3 057.94±442.25	8	9 794.56±1386.81	0
Greedy distance		74.01±9.34	88	195.71±16.47	100
TSP-based distance		87.56±12.13	12	285.04±37.42	0

of the depot does not have a significant impact on travel distance, for both Greedy and TSP-based algorithms. This is perhaps reasonable since the instances are randomly generated so that there are many service paths having similar distances. One could expect to see that on average carbon emission increases by an amount equal to the product of the number of extra bicycles and the travel distance. Nevertheless, for $Q = 15$, on average carbon emission increases by only 10% of the expected amount on the 25-station instances, and by only 5% of the expected amount on the 110-station instances; for $Q = 25$ and 50, the average carbon emission increases are similar.

We note that by the design idea in both Greedy and TSP-based algorithms, when carrying extra bicycles leaving the depot, the service vehicle first tries to drop off bicycles to the shortage stations, while at the end it needs to keep collecting the surplus bicycles from overflowing stations and brings them back to the depot. That is, extra bicycles allow for some extent of freedom at both ends of the service path, but they also incur higher emission; therefore, it is difficult to argue whether or not it is always helpful. Another interesting observation is that Greedy consistently performs better than TSP-based, and even slightly better than in the default setting of no extra bicycles.

3.4.2 Greedy performs well for delivery TSP

Recall that in the delivery TSP problem each station is associated with $+1$ or -1 , that is, in our problem this translates to one bicycle in surplus or in shortage. We also experimented with this case on the 110-station dataset, by dispatching a service vehicle with a capacity $Q \in \{1, 2, 4, 8\}$; the vehicle weight is set to 0 since the capacity is lower than 15.

Table 4 contains the collected statistics. Recall that we take the value of the capacity Q into consideration during dataset simulation, that is, the absolute value of the sum of all the surpluses and shortages should be no greater than Q . For the 100 instances simulated with $Q = 1$, if we deploy a vehicle with capacity $Q > 1$, then both Greedy and TSP-based yield the same route as deploying a vehicle with capacity $Q = 1$; this is not surprising since both algorithms set the higher priority to drop off the bicycles on the back. We have the following observations from Table 4:

(1) The service paths by Greedy are much shorter than those by TSP-based (and in fact about half as long), compared to earlier results on the regularly simulated instances. On average, the service paths by Greedy are about 15% shorter than their counterparts in Tables 2 and 3; but the service paths by TSP-based are about 8% longer than their counterparts in Tables 2 and 3. In this sense, the delivery TSP problem is a quite special case in our SBR problem.

(2) The carbon emission is about half of the travel distance for both Greedy and TSP-based algorithms. One might be able to imagine that after the service vehicle picks up a

Table 4 The performance of the two algorithms Greedy and TSP-based on a collection of 100 simulated instances for the 110-station dataset, where the simulated surplus or shortage is limited to 1 and the vehicle capacity $Q \in \{1, 2, 4, 8\}$ (the weight of the vehicle is 0)

	Q	110-station	won
Greedy emission	1	83.86±11.63	100
TSP-based emission		149.53±25.99	0
Greedy distance		169.80±21.78	100
TSP-based distance		311.11±46.25	0
Greedy emission	2	82.15±12.62	100
TSP-based emission		154.12±27.63	0
Greedy distance		166.53±24.24	100
TSP-based distance		319.41±50.31	0
Greedy emission	4	85.99±12.44	100
TSP-based emission		155.01±26.10	0
Greedy distance		166.15±21.44	100
TSP-based distance		313.98±42.46	0
Greedy emission	8	101.12±29.84	95
TSP-based emission		176.22±41.87	5
Greedy distance		162.42±20.13	100
TSP-based distance		313.76±48.69	0

surplus bicycle, it prefers to find a shortage station to drop the bicycle down, as this is designed into both algorithms; therefore, the expected travel distance without carrying any bicycle is about half of the total distance.

(3) Greedy outperformed the TSP-based algorithm, winning on all 400 instances in travel distance, and winning on all 300 instances associated with capacities 1, 2 and 4 and on 95 out of 100 instances associated with capacity 8 in carbon emission.

We thus may once again safely recommend Greedy over TSP-based in such a special case.

4 Conclusions

In this paper, we formulated the sharing bicycle relocating practice as a novel optimization problem, which can be regarded as a variant of the classic TSP problem with its objective function no longer the length of the Hamiltonian tour but the total carbon emission by the service vehicle. We adopted the affine carbon emission formula and scaled it as the product of the load of the vehicle and the travel distance, and proposed two heuristic algorithms Greedy and TSP-based. The priority in both algorithms is to drop off bicycles to reduce the load of the vehicle, and Greedy seeks for the closest station globally while TSP-based also seeks for the closest station but guided by an approximate TSP tour. We designed numerical experiments to validate their performance empirically, showing the settings where each of Greedy and TSP-based outperforms the other to make recommendations to the sharing bicycle companies for their daily dispatching practice. We also formulated an integer quadratic program for the re-locating problem, solved it to the optimum inside CPLEX for small instances, and benchmarked the performances of Greedy and TSP-based against these optimal solutions.

References

- [1] Benchimol M, Benchimol P, Chappert B, de la Taille A, Laroche F, Meunier F, and Robinet L. Balancing the stations of a self service bike hire system [J]. *RAIRO Operations Research*, 2011, **45**: 37-61.
- [2] Toro E M, Franco J F, Echeverri M G, et al. Green open location-routing problem considering economic and environmental costs [J]. *International Journal of Industrial Engineering Computations*, 2017, **8**(2): 203-216.
- [3] Dell'Amico M, Hadjicostantinou E, Iori M, et al. The bike sharing rebalancing problem: Mathematical formulations and benchmark instances [J]. *Omega*, 2014, **45**: 7-19.
- [4] Rainer-Harbach M, Papazek P, Hu B, and Raidl G R. Balancing bicycle sharing systems: A variable neighborhood search approach [C]//*Proceedings of the 2013 Evolutionary Computation in Combinatorial Optimization*, LNCS 7832, 2013: 121-132.
- [5] Gaspero L D, Rendl A, Urli T. Balancing bike sharing systems with constraint programming [J]. *Constraints*, 2016, **21**: 318-348.
- [6] Anily S, Hassin R. The swapping problem [J]. *Networks*, 1992, **22**: 419-433.
- [7] Chalasani P, Motwani R. Approximating capacitated routing and delivery problems [J]. *SIAM Journal on Computing*, 1999, **28**: 2133-2149.

-
- [8] Charikar M, Khuller S, and Raghavachari B. Algorithms for capacitated vehicle routing [C]// *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC'98)*, 1998: 349-358.
 - [9] Peng Y, Wang X. Research on a vehicle routing schedule to reduce fuel consumption [C]// *Proceedings of the 2009 IEEE International Conference on Measuring Technology and Mechatronics Automation*, 2009: 825-827.
 - [10] Scott C, Urquhart N, Hart E. Influence of topology and payload on CO₂ optimised vehicle routing [C]// *Proceedings of EvoApplications 2010, Part II*, LNCS 6025, 2010, 141-150.
 - [11] Palmer A. The development of an integrated routing and carbon dioxide emissions model for goods vehicles [D]. Bedford: Cranfield University, 2007.
 - [12] Christofides N. Worst-case analysis of a new heuristic for the travelling salesman problem [R]. Graduate School of Industrial Administration, Carnegie Mellon University, 1976.
 - [13] Min H. The multiple vehicle routing problems with simultaneous delivery and pickup points [J]. *Transportation Research*, 1989, **23**: 377-386.
 - [14] Goldberg A V, Karzanov A V. Maximum skew-symmetric flows and matchings [J]. *Mathematical Programming*, 2004, **100**: 537-568.