Journal of Signal Processing, Vol.17, No.4, pp.115-118, July 2013

SELECTED PAPER

Heuristic Methods for the Bike-Sharing System Routing Problem

Satomi Wada, Takafumi Matsuura and Kazumiti Numata

Graduate School of Engineering, Tokyo University of Science
1-3 Kagurazaka, Shinjuku-ku, Tokyo 162-8601, Japan
Phone/FAX: +81-3-3260-4271
E-mail: wada@ms.kagu.tus.ac.jp, matsuura@ms.kagu.tus.ac.jp, numata@ms.kagu.tus.ac.jp

Abstract

In recent years, a bike-sharing system (BSS) has been introduced as a means of inner city transportation in cities of various countries. The BSS consists of one control center and many dispersed stations where rental bicycles are parked. After many people use the bicycles, the distribution of bicycles among stations changes from the managed initial distribution to a disarranged one. Hence, it is necessary to restore the disarranged distribution to the initial one at the certain time intervals to sustain the good performance of the BSS. This work is done by vehicles that transport bicycles from stations with excess bicycles to ones with a shortage. How to accomplish this task as efficiently as possible is an interesting issue. We have presented a mathematical optimization model on this issue named the BSS routing problem (BSSRP) to determine the shortest tour of the transport vehicle, and proposed heuristic solution methods to solve it. In this paper, we propose two new solution methods for the BSSRP. One is based on the idea of searching for solutions including the infeasible space, and the other is based on the idea of dividing the search process into two phases. The results of numerical experiments show that both newly proposed methods are superior to previous methods.

1. Introduction

In recent years, to reduce heavy traffic and exhaust gas pollution in town streets, and to promote people's good health, the bicycle has received considerable attention as a means of transportation. In fact, a new bike-sharing system (BSS), where bicycle rental is flexibly shared by users, has been introduced in several cities, for example, Barcelona, Paris, and New York [1, 2]. The BSS consists of many dispersed stations where bicycles are parked, and one control center that monitors the number of bicycles at each station. A user can rent a bicycle at a convenient station and return it to any station after use. Thus, after many people use this system, in general, the distribution of bicycles among stations changes from the initial one. To restore the number of available bicycles to the initially assigned number, excess bicycles at some stations must be transported to understocked stations. For this purpose, a single transport vehicle starts from the center, loads and unloads bicycles at all necessary stations, and then returns to the center. One of the most interesting issues in the BSS is to determine the shortest tour for this vehicle to visit each of the stations only once.

The problem of determining the shortest tour of the vehicle is very similar to the one-commodity pickup-and-delivery traveling salesman problem (1-PDTSP) proposed by Pérez and González [3], but the 1-PDTSP involves a vehicle that starts from the depot (center) with a preloaded commodity (bicycles), whereas the vehicle in the BSS must start from the center with no load. Independently of the 1-PDTSP, we presented the optimization model of the BSS, called the bike sharing system routing problem (BSSRP), and proposed basic heuristic methods to solve it [4, 5].

In this study, we develop new solution methods that generate more precise solutions than those in [4, 5]. In the followings, we define the BSSRP as an integer programming problem, and investigate some feasibility conditions of the problem. Then, we propose two heuristic methods: Method 1 based on the idea to search the solution space including the infeasible domain, and Method 2 based on the idea to divide the search space into two subspaces. We show the results of numerical experiments to evaluate these proposed methods.

2. Bike-Sharing System Routing Problem (BSSRP)

2.1 Problem

The BSS consists of one control center (depot) and many stations at which bicycles are parked. The stations are classified into two sets: the first is the set of overstocked stations where the current number of parked bicycles is larger than the number of initially assigned bicycles, and the second is the set of understocked stations where the current number of parked bicycles is smaller than the number of initially assigned bicycles. The stations with unchanged numbers are not considered. At certain intervals, the excess bicycles at overstocked stations must be transported to understocked stations to restore the bicycle distribution. This restoration is executed by a single vehicle that starts from the depot with empty, loads and unloads bicycles at necessary stations, and returns to the depot. Each station must be visited exactly once, and the number of bicycles loaded on the vehicle cannot exceed its capacity during the tour. The aim of the BSSRP is to determine the shortest tour of the vehicle.

2.2 Formulation

We formulate the BSSRP as an integer linear programming problem. This formulation gives exact solutions for small instances applied to Gurobi [6], a general-purpose MIP solver. First, let $G = (V, A)$ be a complete directed graph where $V = \{0, 1, ..., m, m{+}1, ..., m{+}n\}$ is the set of all of nodes: the depot and the stations under consideration. $\{0\}$ is the depot and $\{1, ..., m+n\}$ are the stations. $P = \{1, ..., m\}$ is the set of all increased (loading) stations, and $D = \{m+1, ..., m+n\}$ is the set of all decreased (unloading) stations. A is the set of edges that represent the potential movement of the vehicle. Let c_{ij} be the distance between stations i and j ($c_{ij} = c_{ji}$). Second, let $b_i(b_i > 0)$ be the increased number of bicycles at loading station $i(\in P)$, and $-b_k(b_k < 0)$ be the decreased number of bicycles at unloading station $k(\in D)$. Obviously, $\sum_{i\in P} b_i + \sum_{k\in D} b_k = 0$. $b_i(i \in P)$ bicycles are loaded to the vehicle at station i, then, $-b_k(k \in D)$ bicycles on the vehicle are unloaded at station k. Let q be the capacity of the vehicle.

Lastly, we introduce decision variables. Let y_{ij} be a non-negative integer variable that denotes the number of bicycles on the vehicle when it moves from station i to station j. Let x_{ij} be the 0-1 variable that denotes whether the vehicle moves from station i to station j directly (1) or not (0). The non-negative integer variable f_{ij}, used for subtour elimination, represents the number of stations visited by the vehicle before the vehicle visits station j.

Using above notations, we can describe BSSRP as fellow:

$$\text{minimize} \quad \sum_{i\in V}\sum_{j\in V} c_{ij}x_{ij} \tag{1}$$

$$\text{subject to} \quad \sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \tag{2}$$

$$\sum_{j\in V} x_{ij} = 1 \qquad \forall i \in V \tag{3}$$

$$y_{ij} \le qx_{ij},\ y_{ij} \ge 0 \qquad \forall i, j \in V \tag{4}$$

$$y_{0j} = 0 \qquad \forall j \in V \tag{5}$$

$$y_{i0} = 0 \qquad \forall i \in V \tag{6}$$

$$\sum_{j\in V} y_{ij} - \sum_{k\in V} y_{ki} = b_i \qquad \forall i \in V \tag{7}$$

$$\sum_{j\in V} f_{ij} - \sum_{k\in V} f_{ki} = 1 \qquad \forall i \in V \tag{8}$$

$$f_{ij} \le nx_{ij},\ f_{ij} \ge 0 \qquad \forall i, j \in V \tag{9}$$

$$f_{0j} = 0 \qquad \forall j \in V \tag{10}$$

$$f_{i0} = n \qquad \forall j \in V \tag{11}$$

Equation (1) is the objective function of the BSSRP. Equations (2) and (3) describe that every station should be visited exactly once. Equation (4) shows that when the vehicle moves from station i to station j, the number of bicycles on the vehicle is less than the capacity of the vehicle and also at least zero during the tour. Equations (5) and (6) are constraints at the depot. When the vehicle starts from the depot and returns to the depot, it is empty. Equation (7) describes that after visiting station i, the number of bicycles on the vehicle increases (or decreases). Equations (8) to (11) are subtour elimination constraints.

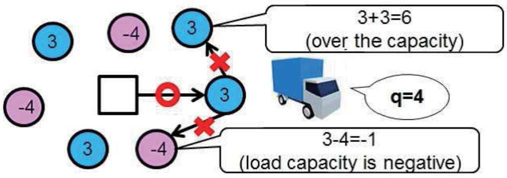
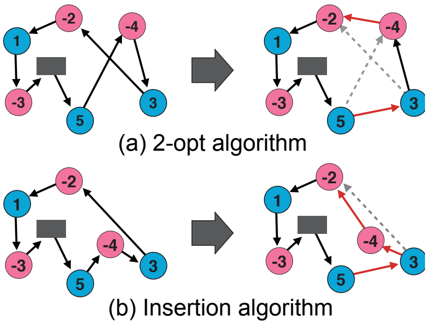


Figure 1: Example of infeasible solution



(a) 2-opt algorithm

(b) Insertion algorithm

Figure 2: 2-opt algorithm and the insertion algorithm

2.3 Feasibility conditions of BSSRP

Does any instance of the BSSRP satisfying $|b_i| \le q$ have a feasible solution? The answer is "no" (Fig. 1). Concerning the feasibility of the BSSRP, we obtained two propositions.

Prop. 1: If $b_i > q/2 + \varepsilon$ for all $i \in P$ and $b_k < -q/2 - 2\varepsilon$ for all $k \in D$, then no feasible solution exists.

Prop. 2: If $\max_{i\in P\cup D} |b_i| \le q/2$ then a feasible solution exists.

Considering Prop. 2, we generate BSSRP instances with $|b_i| \le q/2$ in the following numerical experiments.

3. Proposed Heuristic Methods

To solve the BSSRP, we have already proposed heuristic methods [4, 5]. The methods are based on the simplest constructive and improved methods for the TSP: the nearest neighbor method, the 2-opt algorithm, and the insertion algorithm (Fig. 2). In the 2-opt algorithm, two edges are deleted, and two new edges are added to construct a shorter tour (Fig. 2(a)). On the other hand, in the insertion algorithm, a consecutive partial tour whose length is between one and $m + n - 2$ is inserted into an edge (Fig. 2(b)). In this section, we propose two heuristic methods: Method 1 and Method 2. In both methods, first, an infeasible solution (tour) is generated. Second, the infeasible tour is modified to a feasible solution.

3.1 Method 1

In Method 1, first, an infeasible tour that does not satisfy the capacity constraint of the vehicle is constructed by the nearest neighbor method. Then, the infeasible tour is modified to a feasible tour. Finally, by using the 2-opt algorithm and the insertion algorithm, the feasible tour is improved to obtain a shorter tour. Method 1 is described as follows.

1. A loading station i is randomly selected. The vehicle goes to loading station i to pick up b_i bicycles and station i becomes the new starting point.
2. The vehicle moves to the nearest loading or unloading station from the starting point. The nearest station is selected so that the condition $\alpha \leq y_{ij} \leq q + \alpha$ is satisfied.
3. Step 2 is repeated until all stations are visited. As a result, an infeasible (or a feasible) tour $\boldsymbol{\sigma} = \{\sigma(0) = 0, \sigma(1), \sigma(2), ..., \sigma(m+n)\}$ is constructed. $\sigma(i)$ describes the ith station visited by the vehicle.
4. The infeasible tour $\boldsymbol{\sigma}$ is modified to a feasible tour. $z(t) = \sum_{i=0}^{t} b_{\sigma(i)}$ is the number of bicycles on the vehicle at station $\sigma(i)$. Figure 3 shows an example of an infeasible tour and the time-series of $z(t)$. To make a feasible tour, the station at which the value of $|z(t)|$ is maximum is moved. If, at the t'th visited station ($\sigma(t')$), the value of $|z(t)|$ is maximum, the station $\sigma(t')$ and a station in $\{\sigma(t'+1), ..., \sigma(m+n)\}$ are exchanged. If $\sigma(t')$ is the loading station, $\sigma(t')$ and the unloading station whose visited order is the nearest to t' are exchanged. If $\sigma(t')$ is the unloading station, $\sigma(t')$ and the loading station whose visited order is the nearest to t' are exchanged. Two stations are exchanged, until the capacity constraint is satisfied ($0 \leq z(t) \leq q$),
5. The 2-opt algorithm and the insertion algorithm are applied to the feasible tour until no further improvement can be obtained.

3.2 Method 2

In Method 2, first, a subtour $\boldsymbol{\tau} = \{\tau(0), \tau(1), ..., \tau(m)\}$ consisting of the loading stations and the depot is constructed by the nearest neighbor method. Then, to obtain a feasible tour, all unloading stations are inserted into edges in subtour. To insert unloading stations into edges in the subtour $\boldsymbol{\tau}$, $\Delta_{\tau(i),k} = d_{\tau(i),k} + d_{k,\tau(i+1)} - d_{\tau(i),\tau(i+1)}$ ($i = 0, ..., m,\ k \in D$) is calculated (Fig. 4). Here, d_{ij} is the distance between stations i and j, and $\tau(i)$ is the ith visited loading station in the subtour. Then, to construct a feasible tour, the unloading stations are inserted so that the sum of $\Delta_{\sigma(i),k}$ is minimum and the capacity constraints of the vehicle are satisfied. Finally, the feasible tour is improved by the 2-opt algorithm and the insertion algorithm. To insert all unloading stations into edges in the subtour, the following mathematical model is solved by the general-purpose MIP solver.

$$\text{min.} \quad \sum_{i=0}^{m} \sum_{k=m+1}^{n} \Delta_{\tau(i),k} u_{\tau(i),k} \tag{12}$$

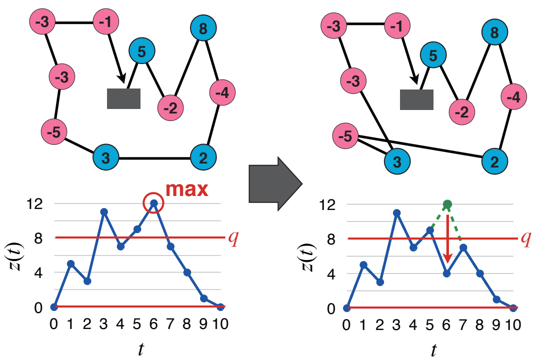


Figure 3: Relationship between a tour and a time series of $z(t)$

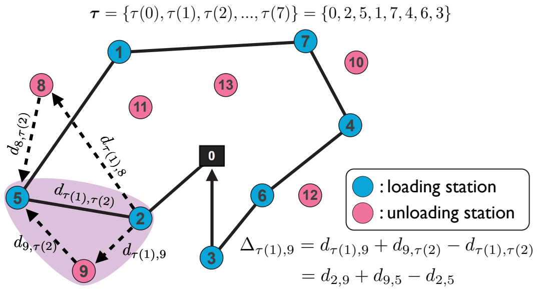

$\boldsymbol{\tau} = \{\tau(0), \tau(1), \tau(2), ..., \tau(7)\} = \{0, 2, 5, 1, 7, 4, 6, 3\}$
1
7
10
8
13
11
4
0
5
2
6
12
9
3
$d_{8,\tau(2)}$
$d_{\tau(1),8}$
$d_{\tau(1),\tau(2)}$
$d_{9,\tau(2)}$
$d_{\tau(1),9}$
: loading station
: unloading station
$\Delta_{\tau(1),9} = d_{\tau(1),9} + d_{9,\tau(2)} - d_{\tau(1),\tau(2)}$
$= d_{2,9} + d_{9,5} - d_{2,5}$

Figure 4: Definition of $\Delta_{\sigma(i),k}$

$$\text{s.t.} \quad g_{\tau(0)} = 0 \tag{13}$$

$$f_{\tau(1)} = 0 \tag{14}$$

$$g_{\tau(i)} = f_{\tau(i)} + b_{\tau(i)} \qquad i = 1, ..., m \tag{15}$$

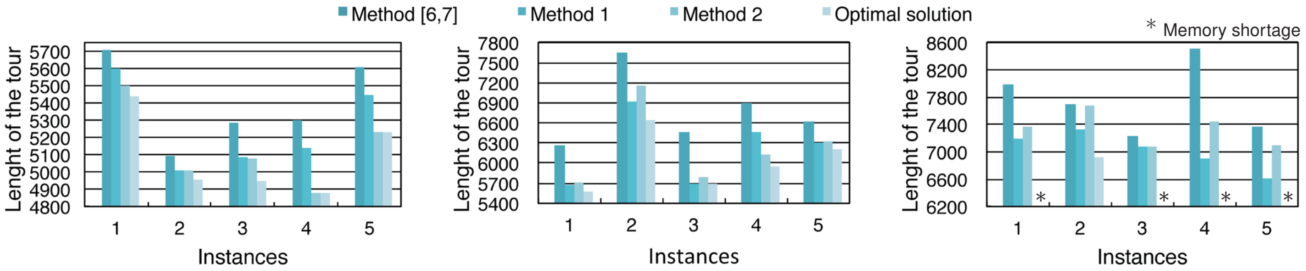
$$f_{\tau(i)} = g_{\tau(i-1)} - \sum_{k=m+1}^{m+n} b_k u_{\tau(i-1),k} \quad i = 2, ..., m \tag{16}$$

$$f_{\tau(i)} \geq 0 \qquad i = 0, ..., m \tag{17}$$

$$g_{\tau(i)} \leq q \qquad i = 1, ..., m \tag{18}$$

$$\sum_{i=0}^{m} u_{\tau(i),k} = 1 \qquad k = m+1, ..., m+n \tag{19}$$

Here, $g_{\tau(i)}$ is the number of bicycles on the vehicle when the vehicle leaves the ith visited loading station for the next loading station $\tau(i+1)$ or unloading station. $f_{\tau(i)}$ is the number of bicycles on the vehicle when the vehicle arrives at the ith loading station. If unloading station k is inserted into the edge between $\tau(i)$ and $\tau(i+1)$, $u_{\tau(i),k}$ is (1), otherwise, (0). Equation (12) is the objective function that minimizes the sum of $\Delta_{\tau(i),k}$. Equations (13) and (14) are capacity constraints at the depot. When the vehicle arrives at the first loading station ($\tau(1)$), no bicycles are loaded on the vehicle ($f_{\tau(1)} = 0$) because the vehicle starts from the depot empty ($g_{\tau(0)} = 0$). Equation (15) shows that $b_{\tau(i)}$ bicycles are loaded on the vehicle after visiting loading station $\tau(i)$. Equation (16) describes that if unloading station k is inserted between $\tau(i-1)$



(a) $n = 30$ (b) $n = 40$ (c) $n = 50$

Figure 5: Length of tours obtained by the proposed methods

and $\tau(i)$, b_k bicycles are unloaded before arriving at the station $\tau(i)$. Equations (17) and (18) are capacity constraints of the vehicle. Equation (19) shows that each unloading station is inserted into an edge in the subtour.

4. Numerical Experiments

To investigate the performances of Methods 1 and 2, we solved the problem for various sizes of instances ($n = 30, 40, 50$). All stations are generated randomly with uniform probability in a square region [0,1000]×[0,1000]. A depot is positioned at the average point of all generated stations. The capacity of the vehicle is set to $q = 20$. Then, the number of bicycles b_i is set to a random number between -10 and 10. We adopted 5 instances for each size. All instances satisfy the conditions described in Sect. 2.3. The parameter α in Method 1 is set to 10. We found an optimal solution for each instance by using general-purpose mixed integer program solver. In the simulation, we used the Gurobi 4.0.1 solver [6].

Figure 5 shows the length of an optimal tour and that obtained using Methods 1 and 2. From Fig. 5, although optimal solutions are found using the Gurobi solver for small-sized instances ($n = 30, 40$), in the case of large-size instances ($n = 50$), the optimal solution cannot be obtained owing to memory shortage. The performances of Methods 1 and 2 are better than that of our previously proposed method [4, 5] for all instances. Method 2 can yield the optimal solution for small-size instances (Fig. 5(a)).

Table 1 shows the average calculation time of the Gurobi solver, and Methods 1 and 2. From Table 1, the calculation time of the Gurobi solver is seen to exponentially increase. On the other hand, for large-size instances, Methods 1 and 2 can quickly find better solutions. Method 2 requires a longer calculation time than Method 1.

Table 1: Average computational time [s]

Method	$n = 30$	$n = 40$	$n = 50$
Gurobi	995	5685	63555
Method 1	14	18	31
Method 2	39	68	646

5. Conclusion

In this paper, we proposed two solution methods for the BSSRP. Method 1 first constructs a relatively short but infeasible tour by relaxing the capacity constraint of the vehicle, then modifies it into a feasible tour by changing the visiting order of stations with the smallest possible deterioration of tour length. On the other hand, Method 2 first constructs many subtours of the depot and all loading stations, and then, for each of them, inserts each unloading station into an interval of consecutive loading stations so that the shortest possible feasible tour is constructed.

The results of numerical experiments show that both methods outperform our previous methods reported in [4, 5]. We conclude that the proposed search strategies work well to a certain extent. Method 1 also makes use of local search procedures in [4, 5], so it shows stable performance for a wide range of instances. Since the current version of Method 2 lacks sufficient ability to search for a good subtour, it finds near-optimum solutions for small-size instances, but for large-size instances, it is inferior to Method 1.

However, Method 2 with a good subtour generates near-optimum solutions. A subject of future work will be to equip Method 2 with the ability to search for good subtours and to combine the searching process with the insertion process.

References

[1] P. DeMario and J. Gifford: Will Smart Bikes succeed as public transportation in the United States?, J. of Public Transportation, Vol. 7, No. 2, pp. 1–15, 2004.

[2] Velib Official Site: http://www.velib.paris.fr

[3] H.H. Pérez and J.S. Gonzáles: A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery, Discrete Appl. Mat., Vol. 145, pp. 126–139, 2004.

[4] T. Matsuura, S. Wada and K. Numata: Optimum restoration of the shared pallets distribution among factories and warehouses, Enabling Manufacturing Competitiveness and Economic Sustainability, Springer, Berlin-Heidelberg, pp. 574–579, 2011.

[5] S. Wada, T. Matsuura and K. Numata: Proc. of SCI'12, pp. 290–291, 2012 (in Japanese).

[6] Gurobi Optimization: http://www.gurobi.com