INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 31 (2024) 2309–2337
DOI: 10.1111/itor.13256

Flexible vehicle scheduling with precedence constraints for tourists

Zhujun Liu[a], Ilkyeong Moon[b] and Ruiyou Zhang[a,*]

[a] *College of Information Science and Engineering, Northeastern University, Shenyang 110819, China*
[b] *Department of Industrial Engineering and Institute for Industrial Systems Innovation, Seoul National University, Seoul 08826, Republic of Korea*
E-mail: 1910326@stu.neu.edu.cn [Liu]; ikmoon@snu.ac.kr [Moon]; zhangruiyou@ise.neu.edu.cn [Zhang]

Received 29 April 2022; received in revised form 2 October 2022; accepted 17 December 2022

Abstract

This research addresses a flexible vehicle scheduling problem considering precedence constraints (FVS-P problem), which is prevalent in many scenic areas worldwide. Each group of tourists in the FVS-P problem comprises a set of visit requests that must be served by shuttle vehicles in a predefined order. A three-indexed integer linear programming model is introduced. Furthermore, an index-reduction strategy is proposed to strengthen the model. A math-heuristic route-segment generation algorithm embedded in a mathematical model is designed to solve the FVS-P problem, particularly in the case of large-sized instances. Experiments on different sizes of near-practical instances validate the mathematical models and the algorithm. The proposed math-heuristic can provide better solutions for large-sized instances in a much shorter time than the models. The trade-off between tourists' feelings and the cost of the area is investigated to provide further insights for managers.

Keywords: transportation; tourism; vehicle scheduling problem; precedence constraint; math-heuristic

1. Introduction

Efficient operation of shuttle vehicles in scenic areas is attracting increasing attention in both academic and industrial fields (Kotiloglu et al., 2017). Several popular tourism areas comprise many scenic spots. These spots are usually so far away from each other that it is almost impossible to travel between the spots on foot. Furthermore, the outer transportation modes that do not belong to the scenic area are usually prohibited considering public order issues. Therefore, shuttle vehicles and their effective management are necessary and critical for many tourism areas.

Vehicle scheduling for tourists in scenic areas is more characteristic when compared to the existing vehicle scheduling problems for some other freights (e.g., containers). One main reason is

*Corresponding author.

that a group of tourists usually visits more than one scenic spot in the scenic area involving a series of transportation requests. Furthermore, the sequence of delivering a tourist group to different scenic spots is usually precedence-constrained by a visit route. On the other hand, the visit of a group of tourists at a scenic spot contains both a former delivery service and a latter pickup service. The starting time of the pickup service is limited by both the starting time of the delivery service and the duration of the tourists' visit at this spot. These are also types of precedence constraints.

A flexible operational mode of shuttle vehicles tries to balance the utilization ratio of vehicles and the satisfaction degree of tourists, in which the transportation of a group of tourists to different locations could be satisfied by different vehicles, and a vehicle need not wait for tourists during their visit at a scenic spot. Generally, there are two typical operational modes of shuttle vehicles for tourists in both real-life scenarios and the literature. The utilization ratio of vehicles in the *charter* mode (Rajendran and Srinivas, 2020), in which a vehicle serves only one group of tourists during its entire visiting duration, is usually not high. However, tourists may need to wait at some locations under the *bus* mode (Melis and Sorensen, 2022), in which vehicles travel along fixed routes and pick up and drop off tourists at certain stops, similar to the buses in cities.

Introducing the flexible management mode into the vehicle scheduling for tourists setup brings significant challenges. First of all, the routes in the flexible mode are interconnected because the delivery and pickup activities of the same group of tourists at a scenic spot can be served by different vehicles. Consequently, the precedence constraints should be considered for the activities not only in the same route but also in different routes. Furthermore, a vehicle can serve more than one group of tourists in a segment. The combination of different tourist groups on a route segment is limited by a given riding time of tourists, which influences both the utilization ratio of vehicles and the satisfaction degree of tourists. As a result, the flexible vehicle scheduling (FVS) for tourists is more complicated than the operational mode in similar existing problems, for example, the pickup and delivery (P&D) problem (Lucci et al., 2021), the dial-a-ride problem (Melis and Sorensen, 2022), or the routing problem (Archetti et al., 2021).

The contributions made in this study are fourfold. First, we formally propose a FVS problem with precedence constraints (for short, the FVS-P problem) considering the longest riding time of transportation requests as well as the loading capacity of vehicles while also minimizing the total operating cost of vehicles. This problem is novel and complicated because of the characteristics of the scheduling for tourists and the challenges of introducing the flexible operation mode of vehicles. Second, an integer linear programming model and a stronger version of the model are presented based on a graphical formulation of the FVS-P problem. Third, we design a math-heuristic route-segment generation (MRSG) algorithm for the problem, consisting of a heuristic construction procedure and a mathematical evaluation model of route segments. Fourth, both the models and the algorithm were evaluated using some near-practical instances with several operational insights presented.

The sections of this paper are arranged as follows. Section 2 summarizes the related literature. Section 3 defines the FVS-P problem. Section 4 presents a graphical description and a mathematical model for the FVS-P problem. Section 5 proposes a strengthened version of this model. In Section 6, we describe the MRSG algorithm. Validation and evaluation of the models and the algorithm are implemented in Section 7. Finally, the conclusions are presented in Section 8.

2. Literature review

The related literature is surveyed from two perspectives. Specifically, Section 2.1 presents the scheduling problems related to the problem in this research. Section 2.2 reviews the algorithms for vehicle routing problems (VRPs) with precedence constraints.

2.1. The related scheduling problems

The vehicle scheduling problem and its variants have been popular in academic literature over the past decades (Braekers et al., 2016). The application of the vehicle scheduling problem exists in various industries. For example, Wu et al. (2022a) handled an electric vehicle scheduling problem that considered the effects of both time-of-use pricing and peak load risk. Wu et al. (2021) solved a vehicle scheduling problem in a two-echelon supply chain network, where the center locating and the inventory plans in each period are considered simultaneously. Similarly, Wu et al. (2022b) also studied a problem deciding the locations, inventory, and routes at the same time in a multi-echelon supply chain, which was motivated by the network design for the transportation and inventory of multi-class hazmat.

The scheduling of shuttle vehicles in scenic areas has not attracted enough attention from academic fields (Zhang et al., 2021a). However, planning itineraries is common in the tourism industry (Kotiloglu et al., 2017), which is similar to the orienteering problem (Gunawan et al., 2016) in terms of mathematical description. For example, a personalized itinerary problem that considered hotel selection for multi-day urban tourists was designed by Zheng et al. (2020a). Furthermore, an itinerary recommendation problem that focused on the choice of transportation modes in city tourism was proposed by Zheng et al. (2020b). Some researchers have focused on designing efficient frameworks for trip planning. For instance, Trachanatzi et al. (2020) proposed an interactive optimization framework to recommend tour trips. Persia et al. (2020) enhanced a tourist trip planning framework based on social sensing, in which the interests of users were analyzed. The scheduling of vehicles for tourists in scenic areas addressed in this research has been seldom studied.

The dial-a-ride problem is a type of P&D problem that is somehow similar to the problem in this research (Parragh et al., 2015). Each transportation request in the dial-a-ride problem has both a pickup origin and a delivery destination. Furthermore, the P&D tasks of any request must be handled considering the temporal precedence. Several features including multi-trips are usually considered in a dial-a-ride problem (Liu et al., 2015; Luo et al., 2019). Interested readers can refer to Ho et al. (2018) for a detailed review of the dial-a-ride problem. The vehicle scheduling problem with precedence constraints in scenic areas is more difficult than the dial-a-ride problem because each tourist usually has more than one transportation request in the former problem. Specifically, a time gap exists between any two neighboring requests of a tourist. The delivery destination of the former request is the same as the pickup origin of the latter request. Furthermore, in the vehicle scheduling for tourists, the precedence constraints are considered not only in each transportation request but also in different transportation requests of each tourist's predefined visit route.

The precedence constraint is a type of synchronization constraint, implying that the starting times of some tasks temporally depend on each other. Drexl (2012) reviewed different types of synchronization constraints for the VRPs in many industries such as logistics and transportation. In logistic research, a container drayage problem was studied, in which a tractor might leave to cater to other orders when a container was being loaded or unloaded (Zhang et al., 2021b). Therefore, for each order in the container drayage problem, a container must be delivered before it is collected, which is the operation synchronization with precedence constraint. Fazi et al. (2020) studied an inland container shipping problem in which the P&D of containers were simultaneous. In a team orienteering problem, every customer task must be served under a given sequence (Hanafi et al., 2020), which is also a type of synchronization constraint. Furthermore, more than one service for one patient is required in the home health-care problem usually, such that the services are synchronized (Cappanera et al., 2020). Although the synchronization constraint has been explored widely, research on scheduling problems with synchronization constraints in scenic areas is quite scarce.

Specifically, Zhang et al. (2021a) considered a similar vehicle scheduling problem in a tourism area, supposing that the capacity of a vehicle is just one tourist group and ignoring the difference in the number of tourists in groups. Furthermore, the precedence constraints among all transportation requests of a tourist group are ignored in Zhang et al. (2021a) because the visit sequence of a tourist group is not considered. As a comparison, one major contribution of this research is that one vehicle can load more than one tourist group within the longest riding time of tourists. The combination of different tourist groups on a trip would increase the complexity of the problem and influence both the utilization ratio of vehicles and the satisfaction degree of tourists.

2.2. *Algorithms to handle synchronization constraints*

Algorithms have been carefully designed to handle synchronization constraints in literature owing to interconnected vehicle routes in such scheduling problems. Transportation requests with precedence constraints may be served in different vehicle routes such that the update of one route might interfere with that of the other routes. Consequently, VRPs with synchronization constraints cannot be solved directly using the algorithms for the VRPs without such constraints.

A few exact algorithms have been applied to solve VRPs with synchronization constraints. A difficulty to handle the synchronization constraints using an exact algorithm such as the branch-and-price is that the pricing problems would become more complicated if the precedence constraints are maintained in the restricted master problem of a branch-and-price (Dohn et al., 2011). To overcome this difficulty, Rasmussen et al. (2012) proposed a branch-and-price algorithm to solve a health-care scheduling problem, in which the precedence constraints were relaxed in the restricted master problem and were satisfied through branching. Recently, a branch-and-price-and-cut algorithm with a novel branching strategy to satisfy the precedence constraints has been designed to handle synchronization constraints in the VRP (Li et al., 2020).

Heuristic algorithms are usually designed for VRPs with synchronization constraints. For example, a VRP with synchronization constraints is solved using an adaptive large neighborhood search (LNS). Furthermore, several linear programming models have been used to check the feasibility of customers with synchronization constraints (Hà et al., 2020). Similarly, a VRP with synchronization constraints at delivery locations has also been addressed by a LNS, in which the self-imposed time windows are adopted to limit the delivery locations with synchronization constraints

(Sarasola and Doerner, 2019). Afifi et al. (2015) presented a simulated annealing algorithm with several local search procedures for a VRP that considered synchronization constraints. Moreover, several metaheuristics have been designed to solve a VRP with time windows and precedence constraints by Ait Haddadene et al. (2016).

In summary, although vehicle scheduling problems with different types of synchronization constraints in other fields have been explored widely, research addressing scheduling problems with synchronization constraints in scenic areas has not been reported so far. Therefore, we present an FVS-P problem considering the precedence constraints, the longest riding time of transportation requests, and the vehicle loading capacity simultaneously. A problem-driven algorithm, namely, the MRSG algorithm, was proposed to solve the FVS-P problem efficiently.

3. The FVS-P problem

We consider a large scenic area consisting of a set of scenic spots, S. Tourists can enter or leave the scenic area through the gate, e. The tourist trips between any two locations in the scenic area must be served by a set of homogeneous vehicles, K. The vehicles initially start from the gate and finally return to the gate in each planning horizon. It is assumed that all vehicles have the same driving speed. The driving time between locations i and j for vehicles is $\tau(i, j)$, $i \in S \cup \{e\}$, $j \in S \cup \{e\}$. We have $\tau(i, j) \geq 0$, $\tau(i, j) = \tau(j, i)$, and $\tau(i, i) = 0$.

The tourists are assumed to be divided into groups in advance, recorded as the set C. Each tourist group $c \in C$ consists of q_c tourists that is supposed to be less than the loading capacity of a vehicle. All tourists can visit the scenic area during a time window $[T^A, T^B](T^A \leq T^B)$ but cannot enter the scenic area later than the time point T^c $(T^A \leq T^C \leq T^B)$. Each group $c \in C$ arrives at the gate of the scenic area at time t_c, where t_c is supposed to be the arrival time of the last tourist in group c. The length of visit time at spot $o \in O_c$ for a tourist group $c \in C$ is $T_{co}(\geq 0)$, where O_c is the set of scenic spots for group c. The scenic spots in O_c must be visited in a given order, say $\vec{R_c}$, which limits the precedence of transportation requests.

The vehicle's loading capacity, Q (≥ 0), is the maximum number of tourists that a vehicle can carry during each trip. The duration of any transportation request cannot be longer than a given value, say L (≥ 0), which is the longest riding time when a tourist group is picked up at a scenic spot until it is dropped off at the next scenic spot. We ignored the time for tourists to get on and off a vehicle. A vehicle is waiting at a scenic spot when the tourist group finishes visiting the spot.

The set of vehicles in the FVS-P problem is scheduled to serve the precedence-constrained requests of tourists with their loading capacity and the longest riding time of transportation requests, minimizing the total serving cost. The total cost includes two parts in which the fixed cost of involving one vehicle is $u_1(\geq 0)$, and the cost of unit serving time of vehicles is $u_2(\geq 0)$.

4. Graphical formulation and mathematical model

The FVS-P problem is described on a graph. Based on the graphical formulation, a three-indexed integer linear programming model of the problem is presented.

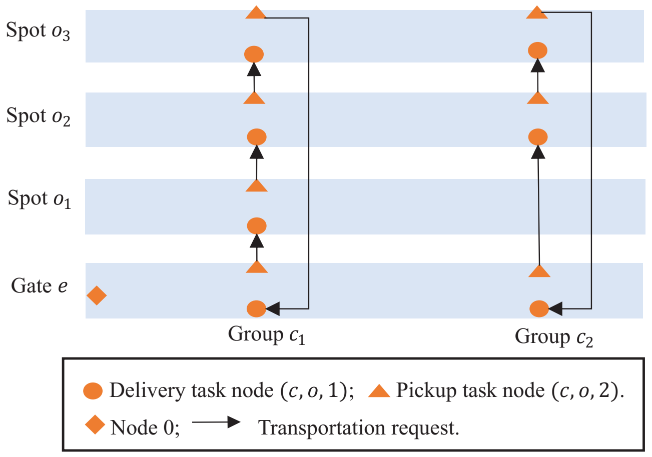
Spot o_3
Spot o_2
Spot o_1
Gate e
Group c_1
Group c_2
● Delivery task node $(c, o, 1)$; ▲ Pickup task node $(c, o, 2)$.
◆ Node 0; ⟶ Transportation request.

Fig. 1. The graphical formulation of Example $\mathcal{A}$.

4.1. Graphical formulation

A graphical formulation of the problem is introduced because the transportation requests of tourists involve both delivery and pickup tasks for vehicles. The delivery task $(c, o, 1)$ is a vehicle that delivers group c to location o. The pickup task $(c, o, 2)$ is a vehicle that picks up group c at location o. The number of transportation requests in the problem is $n = \sum_{c\in C} (|O_c| + 1)$, where $|\cdot|$ is the number of elements in the set. Therefore, the FVS-P problem is described on a graph $G = (V, A)$, in which each delivery or pickup task is a node. In G,

$$V = P \cup D \cup \{0\}, \tag{1}$$

is the node set, where $P = \{1, \cdots, n\}$, $D = \{n + 1, \cdots, 2n\}$, and Node 0 represents the pickup node set, the delivery node set, and the virtual start and return node, respectively. $N = P \cup D$ is called the task node set, compared to Node 0. Transport from node $i \in P$ to the corresponding node $(i + n) \in D$ is a transportation request. Node 0 has the same location as the gate.

To further describe the graph, $\alpha(i) = c$, $\beta(i) = o$, and $\gamma(i) = q_{\alpha(i)}$ are introduced as the attributes of *group*, *location*, and *group size* for a node $i = (c, o, 1) \in D$, or $i = (c, o, 2) \in P$, respectively. In particular, $q_{\alpha(i)}$ is the number of tourists in group $\alpha(i)$ if $i \in P$, while $q_{\alpha(i)}$ is the negative number of tourists in group $\alpha(i)$ if $i \in D$. Furthermore, let

$$\theta(i) = \begin{cases} (c, o, 2), \forall i = (c, o, 1) \in D, \\ (c, o, 1), \forall i = (c, o, 2) \in P \end{cases} \tag{2}$$

represent the relationship between the pickup node $(c, o, 2) \in P$ and the delivery node $(c, o, 1) \in D$ of tourist group c at location o (Zhang et al., 2021a).

An example (Example $\mathcal{A}$) with Groups c_1 and c_2 is given here. The visit routes of Groups c_1 and c_2 are (e, o_1, o_2, o_3, e) and (e, o_2, o_3, e), respectively, where e is the gate and o_1, o_2, o_3 are the scenic spots in the scenic area. Figure 1 shows the delivery task nodes, pickup task nodes, the virtual node, and the transportation requests of groups in the graphical formulation. A group has both a delivery and a pickup task at a visit location, and the two tasks are involved in two transportation requests, separately. As a result, the visit route of Group c_1 is

$$(c_1,\ e,\ 2) \rightarrow (c_1,\ o_1,\ 1),\ (c_1,\ o_1,\ 2) \rightarrow\ (c_1,\ o_2,\ 1),\ (c_1,\ o_2,\ 2)$$
$$\rightarrow\ (c_1,\ o_3,\ 1),\ (c_1,\ o_3,\ 2) \rightarrow\ (c_1,\ e,\ 1).$$

The visit route of Group c_2 is similar and thus omitted.
In Graph G,

$$A\ = A_1 \cup\ A_2 \cup A_3 \cup A_4 \cup A_5 \tag{3}$$

is the arc set. An arc $(i, j) \in A$ is the transfer from node i to node j. The travel time of arc $(i, j) \in A$ is t_{ij}, denoting the time that a vehicle travels from location $\beta(i)$ to location $\beta(j)$. Consequently, $t_{ij} =\ \tau(\beta(i), \beta(j))$. Note that $t_{ij} =\ 0$ if $\beta\ (i) =\ \beta(j)$.
Specifically,

$$A_1 =\ \{(i, j)\mid i\ =\ 0, j \in P; \text{or } i \in D,\ j\ =\ 0\}. \tag{4}$$

For an arc $(i, j) \in A_1$, if its starting node is Node 0, the ending node belongs to set P. If $i \in D$, its destination is Node 0. Note that arc (0, 0) is infeasible.
Furthermore,

$$A_2 = \left\{(i, j)\,|i \in P, j \in P,\ \alpha\,(j) \neq \alpha\,(i),\ \gamma\,(i) + \gamma\,(j) < Q,\ t_{ij} + t_{j,i+n} \leq L\right\}. \tag{5}$$

Both the starting and ending nodes of an arc $(i, j) \in A_2$ are the pickup nodes in P. A vehicle cannot be arranged for two pickup nodes of the same tourist group continuously because of the precedence constraints of transportation requests. Delivering a tourist group at a former scenic spot must be done before picking up the group at the same scenic spot. Moreover, the total number of tourists in the two tourist groups in $(i, j) \in A_2$ cannot exceed the vehicle's loading capacity. Note that the definition $\gamma(i) + \gamma(j) < Q$ on arc set A_2 is only to remove the infeasible arcs (see the mathematical model for a more precise constraint regarding Q). The longest riding time of group $\alpha(i)$ in arc $(i, j) \in A_2$ should also be satisfied.

$$A_3 = \left\{\,(i, j)\,|i \in P, j \in D,\ \alpha\,(j) \neq \alpha\,(i),\ t_{ij} + t_{j,i+n} \leq L;\ \text{ or } i \in P, j = i + n\right\}. \tag{6}$$

On an arc $(i, j) \in A_3$, the starting node is a pickup node in P, and the ending node is a delivery node in D. If the tourist attributes of nodes i and j are different, the longest riding time of group $\alpha(i)$ should be satisfied; otherwise, only arc $(i, i + n)$ for each node $i \in P$ can be feasible in A_3.

$$A_4 = \Big\{\,(i, j)\,|i \in D, j \in P,\ \alpha\,(j) \neq \alpha\,(i);\ \text{ or } i \in D, j \in P, \alpha\,(j) = \alpha\,(i),$$
$$\beta\,(j)\ \text{ is latter than } \beta\,(i)\ \text{ in } \vec{R}_{\alpha(i)}\ \Big\}. \tag{7}$$

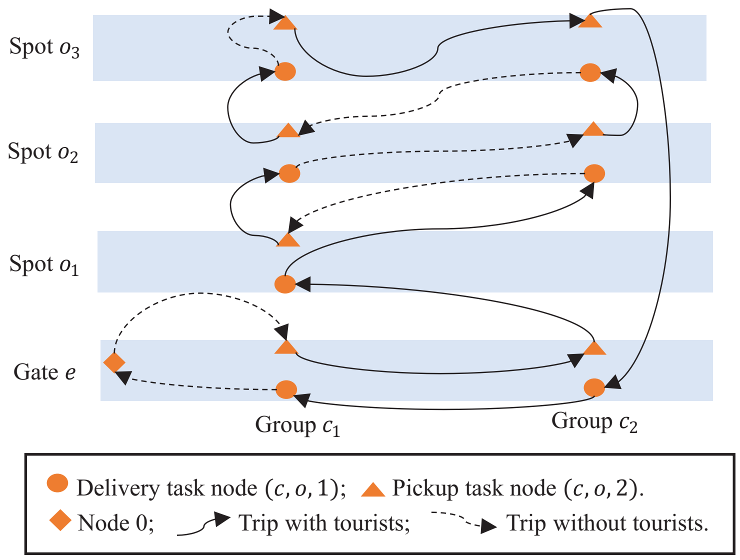
Spot o_3
Spot o_2
Spot o_1
Gate e
Group c_1
Group c_2
● Delivery task node $(c, o, 1)$; ▲ Pickup task node $(c, o, 2)$.
◆ Node 0; Trip with tourists; Trip without tourists.

Fig. 2. The vehicle route of Example $\mathcal{A}$ based on the graphical description.

In an arc $(i, j) \in A_4$, the starting node is a delivery node in D, and the ending node is a pickup node in P. Specifically, if the tourist attributes of nodes i and j are the same, the scenic spot $\beta(j)$ comes later than the spot $\beta(i)$ according to the given visit sequence of group $\alpha(i)$.

$$A_5 = \left\{(i, j)\,|\,i \in D, j \in D,\ \alpha(j) \neq \alpha(i),\ \gamma(i) + \gamma(j)\rangle - Q,\ t_{j-n,i} + t_{ij} \leq L\right\}. \tag{8}$$

Both the starting and ending nodes on arc $(i, j) \in A_5$ are the delivery nodes. A vehicle cannot be arranged for two delivery nodes of the same tourist group continuously because a tourist group must be picked up at a scenic spot before delivering the group to the latter spot. The total number of tourists in the two groups in an arc $(i, j) \in A_5$ cannot exceed the vehicle's loading capacity. The longest riding time of group $\alpha(i)$ must also be satisfied.

In one word, the FVS-P problem is described in Graph G as follows. A group of vehicle routes with the minimal cost is scheduled to serve the task nodes in N. Each task node is served by a vehicle exactly once. The starting times of the task nodes are precedence-constrained. Each route travels along a feasible arc in A considering the loading capacity and longest riding time of requests.

A solution of Example $\mathcal{A}$ on the graph can be described as a vehicle route as

$$0, (c_1, e, 2) \to (c_2, e, 2) \to (c_1, o_1, 1) \to (c_2, o_2, 1), (c_1, o_1, 2) \to (c_1, o_2, 1), (c_2, o_2, 2) \to$$
$$(c_2, o_3, 1), (c_1, o_2, 2) \to (c_1, o_3, 1), (c_1, o_3, 2) \to (c_2, o_3, 2) \to (c_2, e, 1) \to (c_1, e, 1), 0.$$

Figure 2 gives the vehicle route of Example $\mathcal{A}$ based on the graphical description as follows. An empty vehicle starts from Node 0 and picks up Group c_1 first and Group c_2 next at Gate e. It first delivers Group c_1 to Spot o_1 and delivers Group c_2 to Spot o_2 next. The emptied vehicle then leaves Spot o_2 to pick up Group c_1 at Spot o_1 after Group c_1 has finished visiting Spot o_1 and delivers it to Spot o_2. The vehicle stays there to pick up Group c_2 until it has finished visiting Spot o_2 and delivers it to Spot o_3. The emptied vehicle returns to Spot o_2 to pick up Group c_1 until it has finished visiting Spot o_2 and delivers it to Spot o_3. The vehicle stays there to pick up Groups c_1 and c_2 until they have finished visiting Spot o_3 and delivers them to Gate e (reaching at Node 0).

4.2. Three-indexed integer linear programming model

The FVS-P problem can be formulated as a mathematical model according to the description in Graph G. The decision variables were as follows:

$$x_{ijk} = \begin{cases} 1, & \text{if a vehicle } k \in K \text{ serves an arc } (i, j) \in A, \\ 0, & \text{otherwise,} \end{cases}$$

y_i : the time when node $i \in N$ is served,
z_i : the number of tourists on a vehicle after it finishes serving node $i \in V$.

A three-indexed integer linear programming model (Model ILP$_3$) of the problem is presented as follows:

$$\min\ u_1 \sum_{k\in K}\sum_{j\in P} x_{0jk} + u_2 \sum_{k\in K}\sum_{(i,j)\in A} t_{ij} x_{ijk}, \tag{9}$$

subject to

$$\sum_{j\in P} x_{0jk} = \sum_{j\in D} x_{j0k} \le 1, \quad \forall k \in K, \tag{10}$$

$$\sum_{k\in K}\ \sum_{j\in N \text{ and } (i,j)\in A} x_{ijk} = 1, \quad \forall i \in N, \tag{11}$$

$$\sum_{j\in N \text{ and } (i,j)\in A} x_{ijk} - \sum_{j\in N \text{ and } (j,i)\in A} x_{jik} = 0, \quad \forall k \in K,\ i \in N, \tag{12}$$

$$y_i + t_{ij} - y_j \le M_{ij1}\left(1 - x_{ijk}\right), \quad \forall\ (i, j) \in A \setminus A_1,\ k \in K, \tag{13}$$

$$\sum_{j\in N \text{ and } (i,j)\in A} x_{ijk} = \sum_{j\in N \text{ and } (j,i+n)\in A} x_{j,i+n,k}\ , \quad \forall i \in P,\ k \in K, \tag{14}$$

$$t_{i,i+n} \le y_{i+n} - y_i \le L, \quad \forall\ i \in P, \tag{15}$$

$$y_i = t_{\alpha(i)}\ , \quad \forall i \in P \text{ and } \beta\ (i) =\ e, \tag{16}$$

$$y_i = y_{\theta(i)}\ + T_{\alpha(i)\beta(i)}, \quad \forall i \in P \text{ and } \beta\,(i) \ne e, \tag{17}$$

$$y_i + t_{i0} - T^B \le M_{i02}\,(1 - x_{i0k}), \quad \forall i \in D,\ k \in K, \tag{18}$$

$$-2Q\left(1 - x_{ijk}\right) \le z_i + \gamma\,(j) - z_j \le 2Q\left(1 - x_{ijk}\right), \quad \forall\ i \in V, j \in N, (i, j) \in A,\ k \in K, \tag{19}$$

$$z_i \leq Q, \quad \forall i \in P, \tag{20}$$

$$z_i \leq Q\left(1 - \sum_{k \in K} x_{i0k}\right), \quad \forall i \in D, \tag{21}$$

$$x_{ijk} \in \{0, 1\}, \quad \forall (i, j) \in A,\ k \in K, \tag{22}$$

$$z_0 = 0, \tag{23}$$

$$z_i \in \mathbf{N}^*, \quad \forall\ i \in N, \tag{24}$$

$$y_i \in \mathbf{N}^*, \quad \forall i \in N. \tag{25}$$

In Model ILP$_3$, Objective Function (9) as the optimization objective minimizes the total serving cost of vehicles. Constraint (10) guarantees that the places of initial departure as well as the final return for each route are Node 0. Each route corresponds to a vehicle. Constraints (11) and (12) ensure that every task node is carried out just once. Specifically, Constraint (11) limits that the number of outgoing flows of each task node is one. Constraint (12) guarantees the match of incoming and outgoing flows of each task node.

Constraint (13) defines the relationship between the starting times for any two continuous nodes in a route. Sub-tours among the task nodes can also be removed using Constraint (13), where

$$M_{ij1} = T^C + t_{ij}, \quad \forall\ (i, j) \in A \setminus A_1.$$

If $x_{ijk} = 0$, Constraint (13) is automatically relaxed. If $x_{ijk} = 1$, then Constraint (13) becomes

$$y_i + t_{ij} \leq y_j, \quad \forall (i, j) \in A \setminus A_1.$$

Constraint (14) denotes that if a route visits node $i \in P$, it must visit the corresponding delivery node $(i + n) \in D$. The longest riding time of any request $(i, i + n)$ is ensured by Constraint (15).

Constraints (16) and (17) limit the starting time of every pickup node $i \in P$ at the gate and a scenic spot, respectively. Specifically, $t_{\alpha(i)}$ in Constraint (16) is the time when group $\alpha(i)$ arrives at the scenic area. $y_{\theta(i)} + T_{\alpha(i)\beta(i)}$ in Constraint (17) is the ending time at which group $\alpha(i)$ finishes visiting spot $\beta(i)$. The gap between the starting times of the delivery node $(c,\ o,\ 1)$ and its dual pickup node $(c, \text{o}, 2)$ is the visiting duration at Spot o. Constraints (16) and (17) also ensure the precedence constraints of all transportation requests in the visit routes of tourist groups. Constraint (18) guarantees that both the vehicles and tourist groups must finish their activities within the scenic area's opening time, wherein

$$M_{i02} = t_{i0}, \quad \forall\ i \in D.$$

If $x_{i0k} = 0$, Constraint (18) is relaxed automatically because it becomes

$$y_i \leq T^B, \quad \forall i \in D.$$

If $x_{i0k} = 1$, Constraint (18) becomes

$$y_i + t_{i0} \leq T^B, \quad \forall i \in D.$$

Constraint (19) denotes the change in the number of tourists for a vehicle when serving the two nodes of an arc. If $x_{ijk} = 0$, Constraint (19) is automatically relaxed. If $x_{ijk} = 1$, then Constraint (19) becomes

$$z_i + \gamma\ (j) = z_j, \quad \forall\ i \in V, j \in N, (i, j) \in A.$$

The loading capacity of a vehicle is limited by Constraint (20). Constraint (21) ensures that any vehicle is unloaded when it returns to Node 0. If $\sum_{k \in K} x_{i0k} = 0$, Constraint (21) is automatically relaxed. If $\sum_{k \in K} x_{i0k} = 1$, then Constraint (21) becomes:

$$z_i = 0, \quad \forall i \in D.$$

Finally, Constraints (22)–(25) define the types of decision variables.

5. Index-reduction of Model ILP$_3$

This section proposes the index-reduction of Model ILP$_3$ as its strengthened version to improve its solving efficiency. The number of variables x_{ijk} in Model ILP$_3$ is quite large because there are $|K|$ variables for an arc $(i,\ j) \in A$. However, the index k in variable x_{ijk} can be eliminated from Model ILP$_3$, except in Constraint (14), because the vehicles are homogeneous. Therefore, to eliminate index k from Constraint (14), some additional constraints must be introduced. The additional constraints ensure that the pickup and corresponding delivery nodes of each transportation request are served by the same vehicle.

The method proposed by Furtado et al. (2017) is used to represent the pairing relations in Constraint (14). We modify the three-indexed variable x_{ijk} to a two-indexed variable as follows:

$$x_{ij} = \begin{cases} 1, & \text{if } a \text{ vehicle serves } an\ arc\ (i, j) \in A, \\ 0, & \text{otherwise.} \end{cases}$$

A new variable is introduced as

v_i: index of the first node in the route that visits task node $i \in N$.

The FVS-P problem can be reformulated as a two-indexed integer linear programming model, recorded as Model ILP$_3$_IR.

$$\min\ u_1 \sum_{j \in P} x_{0j} + u_2 \sum_{(i,j) \in A} t_{ij} x_{ij}, \tag{26}$$

subject to

$$\sum_{j\in P} x_{0j} = \sum_{j\in D} x_{j0}\ , \tag{27}$$

$$\sum_{j\in N \text{ and } (i,j)\in A} x_{ij} = 1, \quad \forall i \in N, \tag{28}$$

$$\sum_{j\in N \ and\ (i,j)\in A} x_{ij} - \sum_{j\in N \text{ and } (j,i)\in A} x_{ji} = 0, \quad \forall\ i \in N, \tag{29}$$

$$y_i + t_{ij} - y_j \le M_{ij1}\left(1 - x_{ij}\right), \quad \forall\ (i,j) \in A \setminus A_1, \tag{30}$$

$$v_i = v_{i+n}\ , \quad \forall i \in P, \tag{31}$$

$$jx_{0j} \le v_j \le jx_{0j} - |P|\left(x_{0j} - 1\right), \quad \forall j \in P, \tag{32}$$

$$v_i + |P|\left(x_{ij} - 1\right) \le v_j \le v_i + |P|\left(1 - x_{ij}\right), \quad \forall\ (i,j) \in A \setminus A_1, \tag{33}$$

$$t_{i,i+n} \le y_{i+n} - y_i \le L, \quad \forall\ i \in P, \tag{34}$$

$$y_i = t_{\alpha(i)}\ , \quad \forall i \in P \text{ and } \beta\ (i) =\ e, \tag{35}$$

$$y_i = y_{\theta(i)}\ + T_{\alpha(i)\beta(i)}, \quad \forall i \in P \text{ and } \beta\,(i) \neq e, \tag{36}$$

$$y_i + t_{i0} - T^B \le M_{i02}\,(1 - x_{i0})\,, \quad \forall i \in D, \tag{37}$$

$$-2Q\left(1 - x_{ij}\right) \le z_i + \gamma\,(j) - z_j \le 2Q\left(1 - x_{ij}\right), \quad \forall\ i \in V, j \in N, (i,j) \in A, \tag{38}$$

$$z_i \le Q, \quad \forall i \in P, \tag{39}$$

$$z_i \le Q\,(1 - x_{i0})\,, \quad \forall i \in D, \tag{40}$$

$$x_{ij} \in \{0,1\}\,, \quad \forall\,(i,j) \in A, \tag{41}$$

$$z_0 =\ 0 \tag{42}$$

$$z_i \in \mathrm{N}^*, \quad \forall\ i \in N, \tag{43}$$

$$y_i \in \mathrm{N}^*, \quad \forall i \in N, \tag{44}$$

$$v_i \in \mathrm{N}^*, \quad \forall\ i \in N. \tag{45}$$

In Model ILP$_3$_IR, Objective Function (26) refers to Objective Function (9). Constraints (27)–(30) refer to Constraints (10)–(13), respectively. Based on the introduction of variable v_i, Constraints (31)–(33) are used to reinforce Constraint (14). Note that Constraint (31) guarantees that the two tasks in a transportation request are served by the same vehicle. Constraint (32) ensures that the index of a route is recorded as the value of the first node. If $x_{0j} = 0$, Constraint (32) is automatically relaxed. If $x_{0j} = 1$, Constraint (32) becomes

$$v_j = j, \quad \forall j \in P.$$

Similarly, Constraint (33) guarantees that all nodes in the same route have the same index as the first node. If $x_{ij} = 0$, Constraint (33) is automatically relaxed. If $x_{ij} = 1$, then Constraint (33) becomes

$$v_j = v_i, \quad \forall i \in N, j \in N.$$

Constraints (34)–(44) refer to Constraints (15)–(25), respectively. Constraint (45) defines the type of variable v_i for any node $i \in N$.

6. MRSG algorithm

An MRSG algorithm embedded in a mathematical model was designed to efficiently solve the FVS-P problem. The reason for introducing this algorithm is that the mathematical models including Models ILP$_3$ and ILP$_3$_IR are both difficult to solve using general software, for example, the CPLEX software, especially for large-sized instances. Moreover, the FVS-P problem cannot be solved directly using the algorithms designed for existing problems because it differs significantly from the existing ones (e.g., the P&D problems). The framework of math-heuristic has gained popularity because of its efficiency in solving various problems including the production routing problem (Qiu et al., 2021) and the inventory routing problem (Bertazzi et al., 2019).

Definition 1. *A* route segment *is defined as the process in which an empty vehicle serves a transportation request, or several requests from different groups, and the vehicle becomes empty again after finishing all the requests.*

Definition 2. *A* solution seed *is defined as a tuple* (S^{Rem}, S^{RS})*, in which* S^{Rem} *is the set of remaining transportation requests that have not been assigned to generate a route segment, and* S^{RS} *is the set of route segments.*

6.1. The main idea and framework

We initialize the set of solution seeds as the first seed (S^{Rem}, S^{RS}), where S^{Rem} is the set of the first transportation requests of tourist groups, and S^{RS} is empty. For each seed in the set of solution seeds, a solution is obtained when all the remaining transportation requests of the seed are used to generate route segments heuristically (see Section 6.3). Simultaneously, several solution seeds were derived in the process of generating the solution. A mathematical model is used to calculate the objective value of the solution (see Section 6.4). To speed up the search, only if a solution outperforms the currently optimum solution can its derived solution seeds be used to generate solutions hereafter. If each seed in the set of solution seeds was generated as a solution, and the

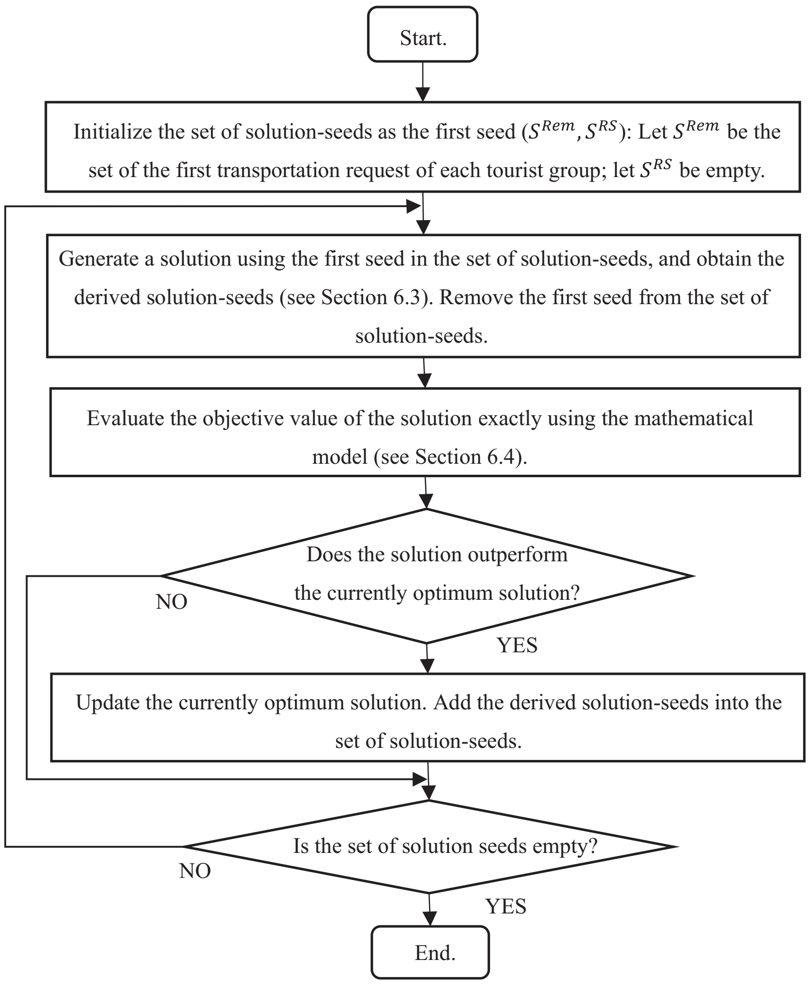
Start.
Initialize the set of solution-seeds as the first seed (S^{Rem}, S^{RS}): Let S^{Rem} be the set of the first transportation request of each tourist group; let S^{RS} be empty.
Generate a solution using the first seed in the set of solution-seeds, and obtain the derived solution-seeds (see Section 6.3). Remove the first seed from the set of solution-seeds.
Evaluate the objective value of the solution exactly using the mathematical model (see Section 6.4).
Does the solution outperform the currently optimum solution?
NO
YES
Update the currently optimum solution. Add the derived solution-seeds into the set of solution-seeds.
Is the set of solution seeds empty?
NO
YES
End.

Fig. 3. Framework of the math-heuristic route-segment generation (MRSG) algorithm.

solution was evaluated, the final solution with the optimum objective value was obtained, and the algorithm returned. Figure 3 presents the framework of the MRSG algorithm.

6.2. *Encoding and decoding*

A solution of the MRSG algorithm can be encoded as a set of route segments, where each transportation request $(i, i+n)$ of the problem must be handled in a route segment exactly once. A vehicle route is defined as the route in which the vehicle travels from the gate, and it also returns to the gate after serving a series of route segments in order. Therefore, the objective value can be obtained when a solution is decoded as a set of vehicle routes, where each route segment of the solution must be handled in a vehicle route exactly once.

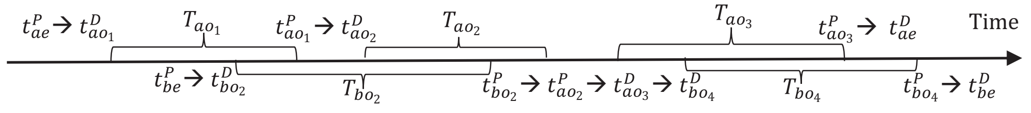
$t^P_{ae} \rightarrow t^D_{ao_1}$
T_{ao_1}
$t^P_{ao_1} \rightarrow t^D_{ao_2}$
T_{ao_2}
T_{ao_3}
$t^P_{ao_3} \rightarrow t^D_{ae}$
Time
$t^P_{be} \rightarrow t^D_{bo_2}$
T_{bo_2}
$t^P_{bo_2} \rightarrow t^P_{ao_2} \rightarrow t^D_{ao_3} \rightarrow t^D_{bo_4}$
T_{bo_4}
$t^P_{bo_4} \rightarrow t^D_{be}$

Fig. 4. A valid solution of Example $\mathcal{B}$.

An example (Example $\mathcal{B}$) with two tourist groups is presented here. The visit routes of Groups a and b are (e, o_1, o_2, o_3, e) and (e, o_2, o_4, e), respectively, where e is the gate and the others are the scenic spots. The notation $t^P_{ao_1}$ ($t^D_{ao_1}$) denotes the starting time of the shuttle task when a vehicle picks up (delivers) Group a at Spot o_1. T_{ao_1} is the visit duration of Group a at Spot o_1. The starting times of all shuttle tasks on the timeline in Fig. 4 are obtained according to the given routes and the duration of visits by tourists.

As shown in Fig. 4, a valid solution S^{RS} of the example can be encoded as

$$S^{RS} = \{t^P_{ae} \rightarrow t^D_{ao_1}, t^P_{be} \rightarrow t^D_{bo_2}, t^P_{ao_1} \rightarrow t^D_{ao_2}, t^P_{bo_2} \rightarrow t^P_{ao_2} \rightarrow t^D_{ao_3} \rightarrow t^D_{bo_4}, t^P_{ao_3} \rightarrow t^D_{ae}, t^P_{bo_4} \rightarrow t^D_{be}\},$$

where $t^P_{bo_2} \rightarrow t^P_{ao_2} \rightarrow t^D_{ao_3} \rightarrow t^D_{bo_4}$ is a route segment with two transportation requests. This indicates that an empty vehicle picks up Group b at Spot o_2 at Time $t^P_{bo_2}$, and stays there to pick up Group a until Time $t^P_{ao_2}$. The vehicle first delivers Group a to Spot o_3 at Time $t^D_{ao_3}$ and becomes empty after delivering Group b to Spot o_4 at Time $t^D_{bo_4}$. The other route segments are similar and thus omitted. The solution S^{RS} can be decoded as two vehicle routes as follows:

$$\left(e, t^P_{ae} \rightarrow t^D_{ao_1}, t^P_{ao_1} \rightarrow t^D_{ao_2}, t^P_{ao_3} \rightarrow t^D_{ae}, e\right),$$

and

$$\left(e, t^P_{be} \rightarrow t^D_{bo_2}, t^P_{bo_2} \rightarrow t^P_{ao_2} \rightarrow t^D_{ao_3} \rightarrow t^D_{bo_4}, t^P_{bo_4} \rightarrow t^D_{be}, e\right).$$

6.3. *The generation of a solution*

A solution can be generated using the solution seed (S^{Rem}, S^{RS}). The main idea of the sub-algorithm to generate a solution is as follows. The route segment in set S^{RS} is generated heuristically using the transportation requests in set S^{Rem}. The transportation requests in set S^{Rem} and their starting times are updated once a route segment has been generated, which guarantees the precedence constraints of all transportation requests. Furthermore, the derived solution seeds are selected during the generation of each route segment. As a result, S^{RS} is generated as a solution when S^{Rem} becomes empty. At the same time, a set of derived solution seeds, S^{D_Seed}, is obtained, and the sub-algorithm returns.

Definition 3. *An* extended insertion *of a route segment is defined as an extended route segment that is obtained by inserting a transportation request into the route segment.*

Let N^R be the number of transportation requests for a route segment. The number of extended insertions, $f(N^R)$, when extending the route segment using a transportation request is

$$f\left(N^R\right) = \begin{cases} 1, & N^R = 0, \\ 2, & N^R = 1, \\ 4N^R + 1, & N^R \geq 2. \end{cases} \tag{46}$$

For example, if the route segment is $(u, u+n)$ and the inserted transportation request is $(i, i+n)$, the number of extended insertions of the segment is two. They are $(u,\ i,\ i+n,\ u+n)$ and $(u, i, u+n, i+n)$.

In generating a route segment, each transportation request is arranged as a whole to guarantee that the P&D tasks in the transportation request are served in pairs with precedence constraints. The route segment is initialized as the transportation request with the earliest starting time in S^{Rem}. We enumerate all the extended insertions of the route segment using each transportation request in S^{Rem}. The maximum riding duration of any transportation request and the loading capacity of a vehicle for an extended insertion are checked. The feasible extended insertion with the maximum number of transportation requests, minimum traveling cost, and earliest ending time is selected as the generated route segment. The generated route segment is placed into S^{RS}, and the other feasible extended insertions are used to generate the derived solution seeds. Finally, the transportation requests in S^{Rem} and their starting times are updated according to the generated route segment.

The following steps are the details of this sub-algorithm:

STEP 1: Initialize a route segment r as the transportation request (j^*, j^*+n), where

$$j^* = \underset{(j,\ j+n)\in S^{Rem}}{\operatorname{argmin}}\ y_j. \tag{47}$$

Let y_j represent the time when task j is served by a vehicle. Let the traveling cost $L^R = t_{j^*,j^*+n}$, the ending time $t^R = y_{j^*} + t_{j^*,j^*+n}$, and $N^R = 1$.

STEP 2: Remove the transportation requests in route segment r from S^{Rem}. If set S^{Rem} is empty, go to Step 7. Otherwise, let m be the index of the transportation request in S^{Rem} and is initialized as 0.

STEP 3: If the mth transportation request is the last one in S^{Rem}, let $N^R = N^R + 1$, then go to Step 4. Otherwise, let $m = m + 1$, and enumerate the extended insertions of r using the mth transportation request in S^{Rem}. Let S_m^I be the set of feasible extended insertions and proceed to Step 5.

STEP 4: If N^R equals to the maximum number of transportation requests that are set for a route segment, go to Step 7; otherwise, go to Step 2.

STEP 5: If S_m^I is empty, go to Step 3; otherwise, let r_m^I with the minimum traveling time L_m^I and the earliest ending time t_m^I be the best-extended insertion in S_m^I. Remove r_m^I from S_m^I.

STEP 6: Update the route segment r and generate the derived solution seeds. If $L_m^I < L^R$, or $L_m^I = L^R$ and $t_m^I < t^R$, generate a derived solution seed of each extended insertion in $S_m^I \cup \{r\}$, and insert the derived seed into S^{D_Seed}. Let $r = r_m^I$, $L^R = L_m^I$, and $t^R = t_m^I$. Otherwise, generate a derived solution seed of r_m^I, and put it into S^{D_Seed}. Return to Step 3.

STEP 7: Let $S^{RS} = S^{RS} \cup \{r\}$ and record L^R and t^R. Update the transportation requests in S^{Rem}: For each transportation request $(j, j+n)$ in r, if $\beta(j+n) \neq 0$, let $y_{\theta(j+n)} = y_{j+n} + T_{\alpha(j)\beta(j+n)}$, and $S^{Rem} = S^{Rem} \cup \{\theta(j+n)\}$. If S^{RS} is empty, the sub-algorithm returns; otherwise, return to Step 1.

Given an extended insertion s, the procedure for generating a derived solution seed (S_s^{Rem}, S_s^{RS}) of s is described as follows:

STEP 1: Initialize S_s^{Rem} as S^{Rem} and S_s^{RS} as S^{RS}.
STEP 2: Place the extended insertion s into S_s^{RS}. For each transportation request $(j, j+n)$ in s, remove it from the set S_s^{Rem}. Update the transportation request in S_s^{Rem}: If $\beta(j+n) \neq 0$, let $y_{\theta(j+n)} = y_{j+n} + T_{\alpha(j)\beta(j+n)}$, and $S_s^{Rem} = S_s^{Rem} \cup \{\theta(j+n)\}$.
STEP 3: Place the derived solution seed into S^{D_Seed} and the procedure returns.

6.4. The evaluation of solutions

The evaluation of a solution is to determine a set of vehicle routes with the minimum objective value based on the route segments. A mathematical model can evaluate a solution exactly and quickly because the starting times of tasks, the loading capacity of vehicles, and the maximum driving duration of a transportation request are satisfied simultaneously when generating a solution as in Section 6.3.

In the evaluation, each route segment in the solution S^{RS} is regarded as a node on a directed graph. Therefore, the problem is described on a graph $G' = (V', A')$ where $V' = \{0, 1, \cdots, |S^{RS}|\}$ and $A' = \{(i, j) : i \in V', j \in V', i \neq j\}$. We obtain a simple multiple traveling salesman problem that can be easily formulated according to G'.

A node $i \in V' \setminus \{0\}$ in Graph G' has the following parameters: the original scenic spot, r_i^O; the terminal scenic spot, r_i^D; the starting time, t_i^O; the ending time, t_i^D and the traveling time, T_i^{RS}. For Node 0, the locations of the origin r_0^O and terminal r_0^D are both the gate. The traveling time between nodes $i \in V'$ and $j \in V'$ is $t_{ij}^{RS} = \tau(r_i^D, r_j^O)$.

The decision variable was introduced as

$$x'_{ij} = \begin{cases} 1, & \text{if a vehicle serves an arc } (i, j) \in A', \\ 0, & \text{otherwise.} \end{cases}$$

A solution in the MRSG algorithm can be evaluated using the following integer linear programming model (Model ILP-RS).

$$\min \ u_1 \sum_{j \in V' \setminus \{0\}} x'_{0j} + u_2 \sum_{(i,j) \in A'} \left(t_{ij}^{RS} + T_j^{RS} \right) x'_{ij}, \tag{48}$$

subject to

$$\sum_{j \in V' \setminus \{0\}} x'_{0j} = \sum_{j \in V' \setminus \{0\}} x'_{j0}, \tag{49}$$

$$\sum_{j \in V' \setminus \{0\}} x'_{ij} = 1, \quad \forall i \in V' \setminus \{0\}, \tag{50}$$

$$\sum_{j \in V' \setminus \{0\}} x'_{ij} - \sum_{j \in V' \setminus \{0\}} x'_{ji} = 0, \quad \forall i \in V' \setminus \{0\}, \tag{51}$$

$$t_i^D + t_{ij}^{RS} - t_j^O \leq M_{ij4}\left(1 - x'_{ij}\right), \quad \forall\, i \in V' \setminus \{0\}, j \in V' \setminus \{0\}. \tag{52}$$

In Model ILP-RS, Objective Function (48) refers to Objective Function (9). Constraint (49) limits both the origin and destination of each route being Node 0. Constraints (50) and (51) indicate that any node (route segment) should be handled just once. Specifically, Constraint (50) limits that the number of outgoing flows of each task node is one. Constraint (51) guarantees the conservation of incoming and outgoing flows of each node. Constraint (52) denotes the relationship between the starting times for any two continuous tasks in a route. Sub-tours can also be removed using Constraint (52), where

$$M_{ij4} = T^C + t_{ij}^{RS}, \quad \forall\, i \in V' \setminus \{0\}, j \in V' \setminus \{0\}.$$

If $x'_{ij} = 0$, Constraint (52) is automatically relaxed. If $x'_{ij} = 1$, Constraint (52) becomes

$$t_i^D + t_{ij}^{RS} \leq t_j^O, \quad \forall\, i \in V' \setminus \{0\}, j \in V' \setminus \{0\}.$$

7. Validation and evaluation

The setting of the hardware, software, and instances in the experiments is presented in Section 7.1. Section 7.2 validates Model ILP$_3$ and its strengthened version. Section 7.3 evaluates the MRSG algorithm. Section 7.4 presents the comparison with a benchmark model of the vehicle scheduling for tourists. Sensitivity analyses of the FVS-P problem are presented in Section 7.5.

7.1. Setting of experiments

A computer with an Intel Pentium processor (CPU G4400, two CPUs at 3.30 GHz) and 4.0 GB of memory and Windows 10 was used to implement all the experiments. All codes were written using the C++ language in Visual Studio 2013. IBM ILOG CPLEX 12.6.1 was used to solve the mathematical models and was configured as follows. The memory available for working storage was 1024 MB. The longest calculation time of each run was 3600 seconds. The strategy for selecting nodes was *depth-first* because it could find a feasible solution quickly with a lower memory usage.

Benchmark instances of the FVS-P problem do not exist. We cannot use the instances of other problems directly, even after minor modifications. Therefore, nine small-sized instances, called sm1–sm9, and two groups of large-sized instances, called LA1–LA7 and LB1–LB7 (the differences between these two types of large-sized instances are presented later), were randomly generated based on the typical scenic area at Qiandao Lake in China.

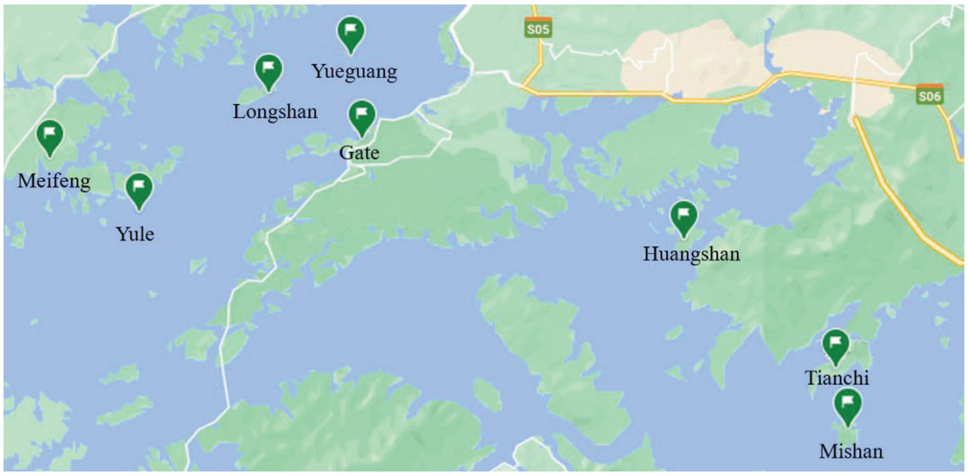

Fig. 5. Locations of scenic spots in the Baidu map of Qiandao Lake.

Table 1
The traveling times between scenic spots of Qiandao Lake

Traveling time (min)	Gate	Meifeng	Yule	Yueguang	Longshan	Huangshan	Tianchi
Meifeng	9						
Yule	7	3					
Yueguang	2	9	8				
Longshan	4	6	5	3			
Huangshan	9	18	16	11	13		
Tianchi	14	23	20	16	18	5	
Mishan	16	24	21	18	19	7	2

With regard to the parameters in the scenic area, let $[T^A,\ T^B]$ be 8:00–19:00, and T^C be 13:00 (in 24-hour format), daily in any instance. Seven scenic spots were considered. The locations of these scenic spots and the gate in the Baidu map are shown in Fig. 5. The distance between any two locations was set according to the data measured using the Baidu map. The distances were transferred into approximate integer values of the traveling time in Table 1, where the speed of vehicles is assumed to be 1 km/min.

Table 2 presents the number of transportation requests (n), which determines the scale of instances, and the number of groups ($|C|$) for every instance. The number of vehicles in K was 20, and the loading capacity of the vehicles was 40. The arrival time t_c of Group c was randomly generated from 8:00 to 13:00. The number of tourists q_c in Group c was randomly generated from 3 to 40. The longest riding time, L, was 25 minutes. For Instances sm1–sm9 and LB1–LB7, the number of scenic spots in O_c of Group c was set randomly from 1 to 5. The visiting duration T_{co} of Group c at Spot o was set randomly from 50 to 80 minutes. For Instances LA1–LA7, the number of scenic spots in O_c and the visiting duration T_{co} of Group c at Spot o were set randomly from 1 to 3 and from 90 to 120 minutes, respectively. The given data of all instances in Table 2 are available in the Figshare repository (https://doi.org/10.6084/m9.figshare.14954865).

Table 2
The numbers of transportation requests and tourist groups of each instance

Instance	$(n, \|C\|)$	Instance	$(n, \|C\|)$	Instance	$(n, \|C\|)$
sm1	(9, 3)	LA1	(59, 20)	LB1	(56, 15)
sm2	(10, 3)	LA2	(55, 20)	LB2	(55, 15)
sm3	(9, 3)	LA3	(62, 20)	LB3	(55, 15)
sm4	(9, 3)	LA4	(63, 20)	LB4	(70, 20)
sm5	(10, 3)	LA5	(57, 20)	LB5	(70, 20)
sm6	(10, 3)	LA6	(62, 20)	LB6	(71, 20)
sm7	(19, 5)	LA7	(56, 20)	LB7	(72, 20)
sm8	(14, 5)				
sm9	(16, 5)				

The unit of driving time is the minute. Let $u_1 = 40$ and $u_2 = 1$ per minute in all experiments considering the operation in practice. See Section 7.5.1 for a sensitivity analysis of u_1. We set the maximum number of transportation requests N^R of a route segment as three when generating a route segment in the MRSG algorithm. A route segment cannot comprise too many transportation requests considering the maximum riding duration of a transportation request and the loading capacity of a vehicle.

7.2. *Validation of Models ILP$_3$ and ILP$_3$_IR*

This section validates Models ILP$_3$ and ILP$_3$_IR using the CPLEX software. Instances sm1–sm9 and LA1–LA7 were used to validate the two models. Model ILP$_3$ cannot provide a feasible solution for a large-sized instance using the CPLEX software with the aforementioned set within 3600 seconds. Consequently, we provide an initial solution to the CPLEX software for each large-sized instance. The initial solution is provided by Model ILP-RS, in which each transportation request is regarded as a route segment directly.

For Instances sm1–sm9, both Models ILP$_3$ and ILP$_3$_IR can attain the optimal solutions within the time limit, which validates Models ILP$_3$ and ILP$_3$_IR. Moreover, Table 3 shows that the average computation time of Model ILP$_3$ is 26.65 seconds, while that of Model ILP$_3$_IR is only 0.40 seconds. In other words, the index-reduction strategy is effective in strengthening Model ILP$_3$ based on small-sized instances. For Instances LA1–LA7, neither models could attain the optimal solutions within the time limit. The feasible objective value and the gap between the objective value and the lower bound of each instance can be provided by the CPLEX software as shown in Table 4. In particular, the average gaps of Models ILP$_3$ and ILP$_3$_IR are 49.28% and 15.60%, respectively. Compared to Model ILP$_3$, the average improvement in the objective values of Model ILP$_3$_IR was 7.74%.

The size of models for small-sized Instances sm8–sm9 and large-sized Instances LA6–LA7 are presented to further compare Models ILP$_3$ and ILP$_3$_IR. The number of binary-integer variables and the number of constraints in Models ILP$_3$ and ILP$_3$_IR for each instance are shown in Figs. 6 and 7, respectively. As shown in Figs. 6 and 7, the index-reduction strategy is effective for strengthening Model ILP$_3$, especially considering large-sized instances. However, Model ILP$_3$_IR

Table 3
Performance for small-sized Instances sm1–sm9

	ILP_3		ILP_3_IR	
Instance	OBJ.	CPU (seconds)	OBJ.	CPU (seconds)
sm1	203	1.00	203	0.40
sm2	195	0.66	195	0.40
sm3	137	0.50	137	0.34
sm4	193	0.70	193	0.29
sm5	228	2.82	228	0.42
sm6	180	1.62	180	0.34
sm7	372	44.59	372	0.52
sm8	250	12.71	250	0.46
sm9	419	175.25	419	0.47
Average		26.65		0.40

Table 4
Performance for large-sized Instances LA1–LA7

	ILP_3		ILP_3_IR		Imp1[a]
Instance	OBJ.	Gap (%)	OBJ.	Gap (%)	(%)
LA1	1066	50.38	977	14.72	8.35
LA2	938	32.94	907	6.43	3.30
LA3	1070	50.47	972	15.64	9.16
LA4	1169	56.89	1041	14.70	10.95
LA5	786	44.91	742	19.41	5.60
LA6	976	60.45	895	19.66	8.30
LA7	1010	48.91	924	18.66	8.51
Average		49.28		15.60	7.74

[a] $\text{Imp1} = \frac{\text{OBJ. }(ILP_3) - \text{OBJ. }(ILP_3\_IR)}{\text{OBJ. }(ILP_3)} * 100\%$

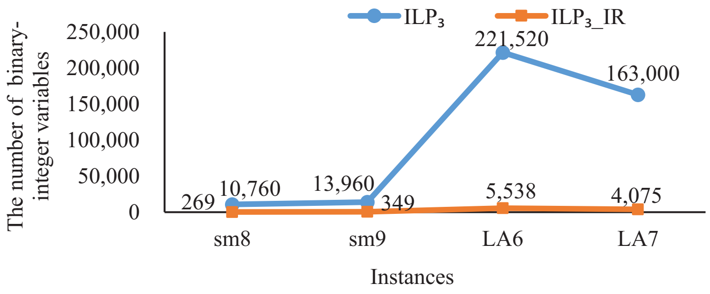

Fig. 6. The number of binary-integer variables in the instances solved by the three-indexed integer linear programming model (Model ILP_3) and two-indexed integer linear programming model (Model ILP_3_IR).

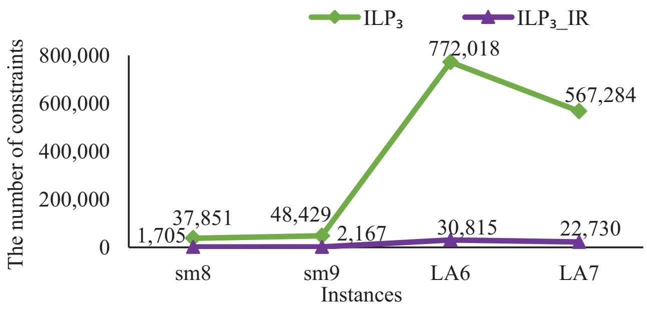

Fig. 7. The number of constraints in the instances solved by Models ILP_3 and ILP_3_IR.

for the large-sized instances is still too hard to be solved considering the complicated nature of the problem.

Figure 8 presents the scheduling result of Instance sm8 obtained from Model ILP_3_IR to further illustrate the FVS-P problem. In Fig. 8, different colors represent the trips of different vehicles. In Instance sm8, the given visiting routes of Groups 1–5 are (Gate, Tianchi, Meifeng, Gate), (Gate, Meifeng, Yule, Gate), (Gate, Yueguang, Yule, Gate), (Gate, Meifeng, Yule, Gate), and (Gate, Yule, Gate), respectively. The total number of transportation requests is 14.

It can be observed that three vehicles serve as shuttle vehicles for the five groups. The routes of Vehicle 1 can be formulated as: Vehicle 1 picks up Group 4 at Time 9:59 and delivers it to Meifeng at Time 10:08. The vehicle stays there, picks up Group 4 at Time 11:22, delivers it to Yule at Time 11:25, and travels to the gate without tourists. This vehicle picks up Group 2 at Time 11:54 and waits at the gate to pick up Group 5 until Time 12:04. Vehicle 1 first delivers Group 5 to Yule at Time 12:11, and then delivers Group 2 to Meifeng at Time 12:14. The vehicle stays there, picks up Group 2 at Time 13:05, and delivers it to Yule at Time 13:08. The vehicle stays at Yule, picks up Group 5 at Time 13:23, and finally delivers it to the gate at Time 13:30. The routes of the other vehicles are similar and omitted here.

The visiting route of Group 2 can also be formulated from another angle. First, Group 2 is picked up at the gate at Time 11:54 and delivered to Meifeng at Time 12:14 by Vehicle 1. When it finishes visiting Meifeng at Time 13:05, Vehicle 1 picks it up and delivers it to Yule at Time 13:08. When the visit is complete at Time 14:28, Vehicle 3 picks up the group and delivers it to the gate at Time 14:35. Note that vehicle serving Group 2 changes. The visiting routes of the other groups are similar and omitted here.

7.3. Validation of the MRSG algorithm

To validate the MRSG algorithm, all instances (Instances sm1–sm9, LA1–LA7, and LB1–LB7) were solved using Model ILP_3_IR, the LNS algorithm proposed in Zhang et al. (2020), and the MRSG algorithm, separately. We modified the main components of the LNS for solving the FVS-P problem as follows. First, a route segment is considered an element in the order-encoding scheme. Second, the initial solution is generated using Model ILP-RS, where a transportation request is set as a route segment directly. Third, several transportation requests are removed randomly, and

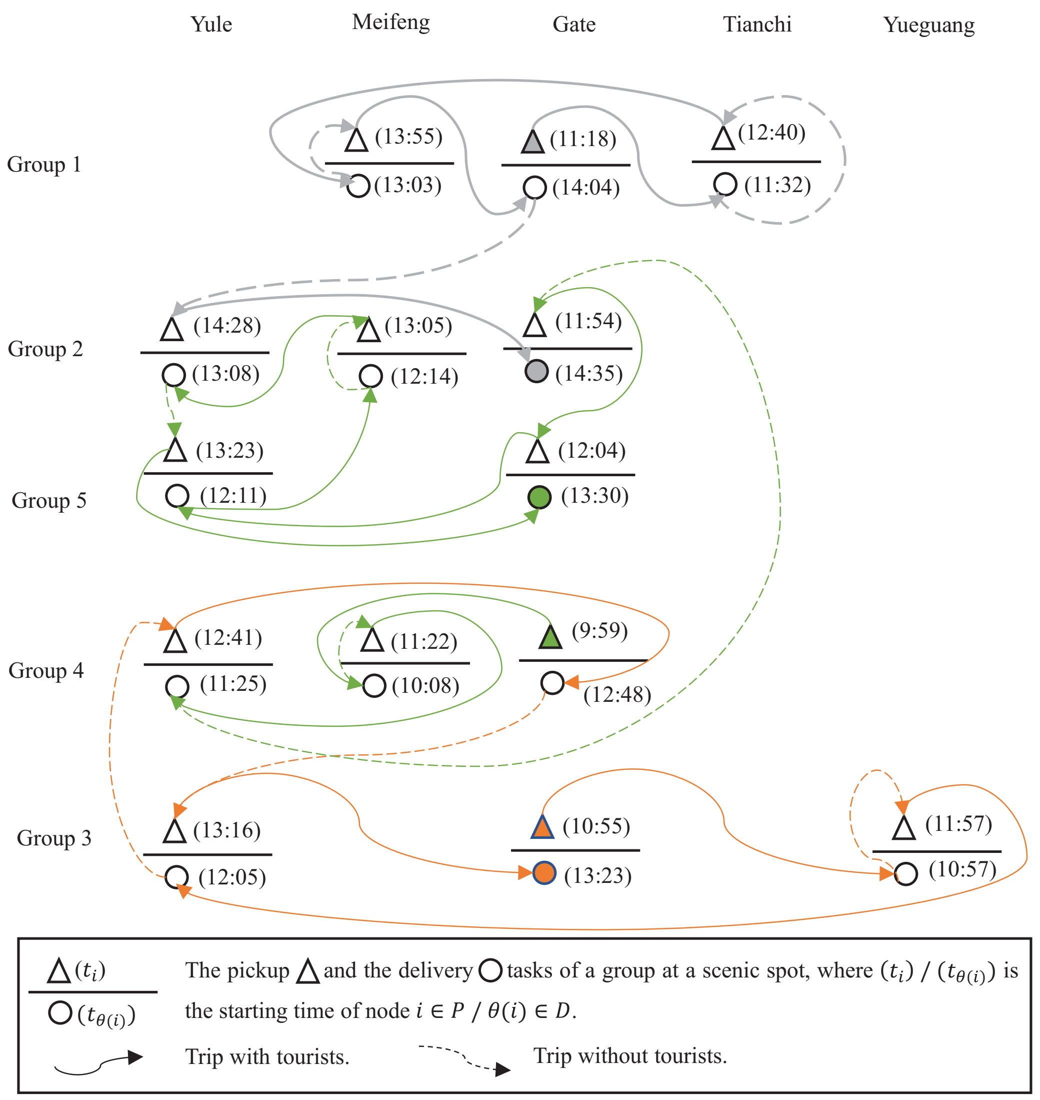

Fig. 8. A scheduling result of Instance sm8.

a removed request can be inserted into not only a route segment under the loading capacity of vehicles but also a position between two segments in a route. At last, the feasibility of a solution is checked by the precedence of the starting times of tasks. The objective value can be obtained according to Function (9).

The solutions provided by Model ILP$_3$_IR using the CPLEX software within 3600 seconds (if the optimal solution of an instance is found before 3600 seconds, the model stops directly) and by the LNS algorithm within 100 iterations were compared with the solutions of the MRSG algorithm, individually. As presented in Table 5, both Model ILP$_3$_IR and the MRSG algorithm provide optimal solutions for Instances sm1–sm9. The LNS algorithm provides optimal solutions

Table 5
Comparison results of Instances sm1–sm9

	ILP_3_IR		LNS		MRSG	
Instance	OBJ.	CPU (seconds)	OBJ.	CPU (seconds)	OBJ.	CPU (seconds)
sm1	203	0.40	203	4.32	203	0.22
sm2	195	0.40	195	4.08	195	0.32
sm3	137	0.34	137	3.96	137	0.28
sm4	193	0.29	193	3.86	193	0.38
sm5	228	0.42	228	4.19	228	0.57
sm6	180	0.34	180	4.44	180	0.58
sm7	372	0.52	372	5.08	372	0.29
sm8	250	0.46	254	4.37	250	3.30
sm9	419	0.47	419	4.94	419	0.56
Average		0.39		4.36		0.72

Abbreviations: ILP_3_IR, reformulated as a two-indexed integer linear programming model; LNS, large neighborhood search; MRSG, math-heuristic route-segment generation.

Table 6
Comparison results of Instances LA1–LA7

	ILP_3_IR		LNS		MRSG		OBJ.	OBJ.
Instance	OBJ.	Gap (%)	OBJ.	CPU (seconds)	OBJ.	CPU (seconds)	Imp1[a] (%)	Imp2[b] (%)
LA1	977	14.72	1066	35.51	922	21.93	5.63	13.51
LA2	907	6.43	938	30.78	881	30.50	2.87	6.08
LA3	972	15.64	1048	46.10	906	41.64	6.79	13.55
LA4	1041	14.70	1169	54.64	987	21.03	5.19	15.57
LA5	742	19.41	786	58.12	723	23.26	2.56	8.02
LA6	895	19.66	976	74.05	839	57.71	6.26	14.04
LA7	924	18.66	998	69.10	876	35.48	5.19	12.22
Average				52.61		33.08	4.93	11.85

Abbreviations: ILP_3_IR, reformulated as a two-indexed integer linear programming model; LNS, large neighborhood search; MRSG, math-heuristic route-segment generation.

[a]OBJ. Imp1 $= \frac{\text{OBJ. (ILP}_3\text{\_IR) − OBJ. (MRSG)}}{\text{OBJ. (MRSG)}} * 100\%$;

[b]OBJ. Imp2 $= \frac{\text{OBJ. (LNS) − OBJ. (MRSG)}}{\text{OBJ. (LNS)}} * 100\%$.

for eight (sm1–sm7 and sm9) out of nine instances and a worse solution for Instance sm8 with the gap of 1.57%. The average calculation time of the MRSG algorithm is slightly longer than that of Model ILP_3_IR. The main reason is that the iterative operation and random feature might make the MRSG algorithm take more time to find the optimal solutions for some small-sized instances. However, the average calculation time of the MRSG algorithm is shorter than that of the LNS algorithm. This validates the MRSG algorithm primarily.

The MRSG algorithm outperformed both Model ILP_3_IR and the LNS algorithm considering large-sized instances. As shown in Tables 6 and 7, the MRSG algorithm provides better solutions than the other two methods for both Instances LA1–LA7 and LB1–LB7. When compared to Model ILP_3_IR, the average improvements in the objective values for Instances LA1–LA7 and LB1–LB7

Table 7
Comparison results of Instances LB1–LB7

	ILP_3_IR		LNS		MRSG		OBJ.	OBJ.
Instance	OBJ.	Gap (%)	OBJ.	CPU (seconds)	OBJ.	CPU (seconds)	Imp1[a] (%)	Imp2[b] (%)
LB1	986	7.44	1073	33.69	954	41.35	3.25	11.09
LB2	827	17.45	909	30.50	795	24.22	3.87	12.54
LB3	928	21.79	1010	37.24	870	48.57	6.25	13.86
LB4	1127	25.93	1203	58.67	1026	35.35	8.96	14.71
LB5	1053	30.29	1169	68.67	983	119.59	6.65	15.91
LB6	1238	29.11	1363	87.67	1141	68.89	7.84	16.29
LB7	1205	33.81	1304	108.80	1064	55.33	11.70	18.40
Average				60.75		56.19	6.93	14.69

Abbreviations: ILP_3_IR, reformulated as a two-indexed integer linear programming model; LNS, large neighborhood search; MRSG, math-heuristic route-segment generation.
[a]OBJ. Imp1 $= \frac{\text{OBJ. (ILP}_3\text{\_IR) − OBJ. (MRSG)}}{\text{OBJ. (MRSG)}} * 100\%$;
[b]OBJ. Imp2 $= \frac{\text{OBJ. (LNS) − OBJ. (MRSG)}}{\text{OBJ. (LNS)}} * 100\%$.

are 4.93% and 6.93%, respectively. Furthermore, when compared to the LNS algorithm, the average improvements in the objective values for Instances LA1–LA7 and LB1–LB7 are 11.85% and 14.69%, respectively. As a comparison, the average calculation time of the MRSG algorithm for Instances LA1–LA7 and LB1–LB7 are 33.08 and 56.19 seconds, respectively. However, the calculation time of the CPLEX software for Model ILP_3_IR was 3600 seconds, and the average calculation time of the LNS algorithm for Instances LA1–LA7 and LB1–LB7 are 52.61 and 60.75 seconds, respectively. In summary, the MRSG algorithm achieves better solutions in a much shorter time when compared to both Model ILP_3_IR and the LNS algorithm for each type of large-sized instances.

7.4. Comparison to a benchmark model of the scheduling for tourists

We compare the FVS-P problem to a benchmark model of the scheduling for tourists, in which the combination of different tourist groups' transportation requests on a segment of a route is not considered and the waiting time of tourists in a transportation request is ignored. The benchmark model is a simplified version of Model ILP_3_IR, where the decision variable $x_{i,i+n}$ is set as one for each pickup task node $i \in P$, and both the decision variables v_i and z_i and the constraints, including Formulations (31)–(33), (38)–(40), (42)–(43), and (45), are removed. Furthermore, Constraint (34) is modified as $y_{i+n} - y_i = t_{i,i+n}$, $\forall i \in P$.

All instances were solved using CPLEX based on the mathematical benchmark model. Table 8 presents the comparative results regarding the average values of the optimal objective, fixed cost, and travel cost among all instances, separately. All the three average values of the FVS-P problem are better than those of the benchmark model, although the MRSG algorithm cannot guarantee to solve the optimal solutions for all instances in the FVS-P problem. This validates that the operation mode in the FVS-P problem performs better, compared to the benchmark mode. One main reason

Table 8
Comparative results to the benchmark model (average values of all instances)

The benchmark model			This research		
OBJ.	Fixed cost	Travel cost	OBJ.	Fixed cost	Travel cost
749.00	227.83	521.17	658.43	198.26	460.35

Table 9
Results under different values of u_1 for Instance LA7

u_1	OBJ.	Number of involved vehicles	Total traveling times of vehicles (minutes)
0	551	12	551
20	741	8	581
40	876	5	676
60	976	5	676

might be that the combination of different tourist groups on a route segment would decrease both the number and the total travel time of vehicles.

7.5. *Sensitivity analyses*

This section analyzes the sensitivities of the cost coefficient, u_1, for using a vehicle, and the maximum riding duration, L, of tourists. Instance LA7 was selected as the testing instance in the following experiments, and the MRSG algorithm was employed as the solution method.

7.5.1. Effect of u$_1$

We increased the value of u_1 from 0 to 60 at a step of 20, keeping $u_2 = 1$, and resolved Instance LA7. The number of involved vehicles decreases, and the total serving time of all vehicles increases in general, with the increasing value of u_1, as presented in Table 9. When u_1 increases to 60, the two components of the objective function remain because the minimum number of vehicles involved in Instance LA7 is 5. Fewer vehicles cannot provide feasible solutions. The cost coefficients of the two components of the objective function can be modified in actual applications.

7.5.2. The effect of parameter L

We increased the value of the maximum riding duration L in Instance LA7 from 0 to 30 at steps of 5. Notably, if $L < t_{i,i+n}$, the riding duration of the transportation request $(i, i+n)$ is $t_{i,i+n}$. Figure 9 shows that the objective value decreases when the value of L increases. That is, vehicles can serve more tourist groups on a trip with a longer riding duration of tourists, and thus the operational cost decreases. In particular, if $L \leq 5$, each tourist group must be delivered to its destination without waiting once it has been picked up at the request's origin. Consequently, more vehicles must be used

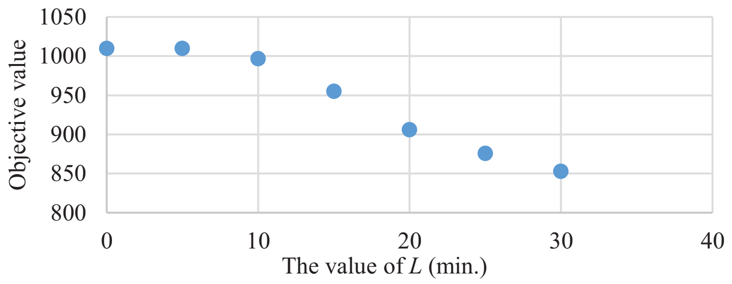

Fig. 9. Results with different values of L for Instance LA7.

with higher operational costs. However, if L is too long, it may influence the degree of satisfaction of tourists.

8. Conclusions and future research

We present a FVS-P problem that usually arises in scenic areas, where tourists must travel between scenic spots by shuttle vehicles, and the visit routes of groups are presented. A set of vehicles is flexibly scheduled to serve the precedence-constrained transportation requests with the longest riding time for requests, and the loading capacity of vehicles considered simultaneously. The problem minimizes the total serving costs of all the vehicles involved. The problem is formulated as a three-indexed mathematical model based on a graph-based description with a strengthened version. A MRSG algorithm is proposed to solve this problem. Both the mathematical models and the algorithm were verified extensively using numerous near-practical instances.

The experiments validated the strengthened mathematical model as well as math-heuristic algorithm. Importantly, the algorithm can provide better solutions in a shorter time than both the strengthened model and a LNS in the literature. The operation mode in this problem is better than a benchmark mode of vehicle scheduling for tourists in terms of the objective values. Sensitivity analyses indicate that managers of scenic areas can modify the cost coefficients of the two components in the objective function. Moreover, the waiting time of tourists and the operation cost of scenic areas can be balanced by adjusting the longest riding time of tourists.

Of course, this study has some limitations. For example, we assume that all the information is given in advance. In fact, uncertain scenarios are more general. However, both dynamic and precedence-constrained features are difficult to handle in the vehicle scheduling problems. The complicated features of the solutions presented herein might present challenges in solving a dynamic scheduling problem in scenic areas.

The scheduling problem for tourists can also be solved considering some other factors. First, heterogeneous vehicles can be introduced into the problem because it may help save operational costs in practice. However, a heterogeneous vehicle scheduling problem for tourists would be more difficult to be solved because the vehicle type of a route should also be determined. Furthermore, the match of tourist groups in a trip would be more complicated when considering heterogeneous vehicles with different loading capacities. Second, the maximum number of tourists at a scenic spot during a given period can be considered, which is an important constraint that influences the vehicle scheduling for tourists directly. Third, different solution strategies to handle similar problems may be designed in the future.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Nos. 71971050, 71831006).

References

Afifi, S., Dang, D.-C., Moukrim, A., 2015. Heuristic solutions for the vehicle routing problem with time windows and synchronized visits. *Optimization Letters* 10, 3, 511–525.

Ait Haddadene, S.R., Labadie, N., Prodhon, C., 2016. A GRASP $\times$ ILS for the vehicle routing problem with time windows, synchronization and precedence constraints. *Expert Systems with Applications* 66, 274–294.

Archetti, C., Guastaroba, G., Huerta-Muñoz, D.L., Speranza, M.G., 2021. A kernel search heuristic for the multivehicle inventory routing problem. *International Transactions in Operational Research* 28, 6, 2984–3013.

Bertazzi, L., Coelho, L.C., De Maio, A., Laganà, D., 2019. A matheuristic algorithm for the multi-depot inventory routing problem. *Transportation Research Part E: Logistics and Transportation Review* 122, 524–544.

Cappanera, P., Requejo, C., Scutellà, M.G., 2020. Temporal constraints and device management for the Skill VRP: mathematical model and lower bounding techniques. *Computers & Operations Research* 124, 105054.

Dohn, A., Rasmussen, M.S., Larsen, J., 2011. The vehicle routing problem with time windows and temporal dependencies. *Networks* 58, 4, 273–289.

Drexl, M., 2012. Synchronization in vehicle routing-A survey of VRPs with multiple synchronization constraints. *Transportation Science* 46, 3, 297–316.

Fazi, S., Fransoo, J.C., Van Woensel, T., Dong, J.-X., 2020. A variant of the split vehicle routing problem with simultaneous deliveries and pickups for inland container shipping in dry-port based systems. *Transportation Research Part E: Logistics and Transportation Review* 142, 102057.

Furtado, M.G.S., Munari, P., Morabito, R., 2017. Pickup and delivery problem with time windows: a new compact two-index formulation. *Operations Research Letters* 45, 4, 334–341.

Gunawan, A., Lau, H.C., Vansteenwegen, P., 2016. Orienteering problem: a survey of recent variants, solution approaches and applications. *European Journal of Operational Research* 255, 2, 315–332.

Hà, M.H., Nguyen, T.D., Nguyen Duy, T., Pham, H.G., Do, T., Rousseau, L.-M., 2020. A new constraint programming model and a linear programming-based adaptive large neighborhood search for the vehicle routing problem with synchronization constraints. *Computers & Operations Research* 124, 105085.

Hanafi, S., Mansini, R., Zanotti, R., 2020. The multi-visit team orienteering problem with precedence constraints. *European Journal of Operational Research* 282, 2, 515–529.

Ho, S.C., Szeto, W.Y., Kuo, Y.-H., Leung, J.M.Y., Petering, M., Tou, T.W.H., 2018. A survey of dial-a-ride problems: literature review and recent developments. *Transportation Research Part B: Methodological* 111, 395–421.

Kotiloglu, S., Lappas, T., Pelechrinis, K., Repoussis, P.P., 2017. Personalized multi-period tour recommendations. *Tourism Management* 62, 76–88.

Li, J., Qin, H., Baldacci, R., Zhu, W., 2020. Branch-and-price-and-cut for the synchronized vehicle routing problem with split delivery, proportional service time and multiple time windows. *Transportation Research Part E: Logistics and Transportation Review* 140, 101955.

Liu, M., Luo, Z., Lim, A., 2015. A branch-and-cut algorithm for a realistic dial-a-ride problem. *Transportation Research Part B: Methodological* 81, 267–288.

Lucci, M., Severín, D., Zabala, P., 2021. A metaheuristic for crew scheduling in a pickup-and-delivery problem with time windows. *International Transactions in Operational Research* 30, 2, 970–1001.

Luo, Z., Liu, M., Lim, A., 2019. A two-phase branch-and-price-and-cut for a dial-a-ride problem in patient transportation. *Transportation Science* 53, 1, 113–130.

Melis, L., Sorensen, K., 2022. The static on-demand bus routing problem: large neighborhood search for a dial-a-ride problem with bus station assignment. *International Transactions in Operational Research* 29, 3, 1417–1453.

Braekers, K., Ramaekers, K., Nieuwenhuyse, I.V., 2016. The vehicle routing problem: state of the art classification and review. *Computers & Industrial Engineering* 99, 300–313.

Parragh, S.N., Pinho De Sousa, J., Almada-Lobo, B., 2015. The dial-a-ride problem with split requests and profits. *Transportation Science* 49, 2, 311–334.

Persia, F., Pilato, G., Ge, M.Z., Bolzoni, P., D'auria, D., Helmer, S., 2020. Improving orienteering-based tourist trip planning with social sensing. *Future Generation Computer Systems-the International Journal of Escience* 110, 931–945.

Qiu, Y.Z., Zhou, D., Du, Y.N., Liu, J., Pardalos, P.M., Qiao, J., 2021. The two-echelon production routing problem with cross-docking satellites. *Transportation Research Part E-Logistics and Transportation Review* 147, 102210.

Rajendran, S., Srinivas, S., 2020. Air taxi service for urban mobility: a critical review of recent developments, future challenges, and opportunities. *Transportation Research Part E: Logistics and Transportation Review* 143, 102090.

Rasmussen, M.S., Justesen, T., Dohn, A., Larsen, J., 2012. The Home Care Crew Scheduling Problem: preference-based visit clustering and temporal dependencies. *European Journal of Operational Research* 219, 3, 598–610.

Sarasola, B., Doerner, K.F., 2019. Adaptive large neighborhood search for the vehicle routing problem with synchronization constraints at the delivery location. *Networks* 75, 1, 64–85.

Trachanatzi, D., Rigakis, M., Marinaki, M., Marinakis, Y., 2020. An interactive preference-guided firefly algorithm for personalized tourist itineraries. *Expert Systems with Applications* 159, 113563.

Wu, W., Lin, Y., Liu, R., Jin, W., 2022a. The multi-depot electric vehicle scheduling problem with power grid characteristics. *Transportation Research Part B: Methodological* 155, 322–347.

Wu, W., Ma, J., Liu, R., Jin, W., 2022b. Multi-class hazmat distribution network design with inventory and superimposed risks. *Transportation Research Part E: Logistics and Transportation Review* 161, 102693.

Wu, W., Zhou, W., Lin, Y., Xie, Y., Jin, W., 2021. A hybrid metaheuristic algorithm for location inventory routing problem with time windows and fuel consumption. *Expert Systems with Applications* 166, 114034.

Zhang, R., Huang, C., Wang, J., 2020. A novel mathematical model and a large neighborhood search algorithm for container drayage operations with multi-resource constraints. *Computers & Industrial Engineering* 139, 106143.

Zhang, R., Liu, Z., Feng, X., 2021a. A novel flexible shuttle vehicle scheduling problem in scenic areas: task-divided graph-based formulation and ALGORITHM. *Computers & Industrial Engineering* 156, 107295.

Zhang, R., Wang, D., Wang, J., 2021b. Multi-trailer drop-and-pull container drayage problem. *IEEE Transactions On Intelligent Transportation Systems* 22, 9, 5708–5720.

Zheng, W.M., Ji, H.P., Lin, C.R., Wang, W.H., Yu, B.L., 2020a. Using a heuristic approach to design personalized urban tourism itineraries with hotel selection. *Tourism Management* 76, 103956.

Zheng, W.M., Liao, Z.X., Lin, Z.B., 2020b. Navigating through the complex transport system: a heuristic approach for city tourism recommendation. *Tourism Management* 81, 104162.