Operational Research (2022) 22:2943–2972
https://doi.org/10.1007/s12351-021-00639-0

ORIGINAL PAPER

Time-dependent rural postman problem: time-space network formulation and genetic algorithm

Jianbin Xin[1] · Benyang Yu[1] · Andrea D'Ariano[2] · Heshan Wang[1] · Meng Wang[3]

Received: 10 August 2020 / Revised: 25 March 2021 / Accepted: 10 April 2021 /
Published online: 3 May 2021

Abstract

In this paper, a new time-space network model is proposed for addressing the time-dependent rural postman problem (TDRPP) of a single vehicle. The proposed model follows the idea of arc-path alternation to form a feasible and complete route. Based on the proposed model, the time dependency of the TDRPP is better described to capture its dynamic process, compared to the existing methods using a piecewise constant function with limited intervals. Furthermore, the property of first-in-first-out (FIFO) can be satisfied with the time spent on each arc. We investigate the FIFO property for the considered time-dependent network and key optimality property for the TDRPP. Based on this property, a dedicated genetic algorithm (GA) is proposed to efficiently solve the considered TDRPP that suffers from computational intractability for large-scale cases. Comprehensive simulation experiments are conducted for various time-dependent networks to show the effectiveness of the proposed GA.

Keywords Rural postman problem · Time-space network model · Time-dependent network · Genetic algorithm

✉ Heshan Wang
whs7713578@zzu.edu.cn

Jianbin Xin
j.xin@zzu.edu.com

Benyang Yu
ybybloom@163.com

Andrea D'Ariano
a.dariano@ing.uniroma3.it

Meng Wang
m.wang@tudelft.nl

[1] School of Electrical Engineering, Zhengzhou University, Science Road 100, Zhengzhou 450001, China

[2] Dipartimento di Ingegneria, Università Degli Studi Roma Tre, via della Vasca Navale, 79 - 00146 Roma, Italy

[3] Department of Transport and Planning, Delft University of Technology, Mekelweg 2, 2628 CN Delft, The Netherlands

1 Introduction

Arc routing problems have been extensively investigated in the last decade. Arc routing problems are essentially routing problems in which the tasks to be performed are located on the arcs of the network. Many practical problems (such as trash collection, road gritting, and newspaper delivery) can be categorized into typical arc routing tasks (Corberán et al. 2021; Corberán and Laporte 2014; Corberán and Prins 2010).

Arc routing is a generalized problem that can be further divided into rural postman problem (RPP) and Chinese postman problem (CPP). In the resulting network, if some (not necessarily all) of the edges (for an undirected graph) or arcs (for a directed graph) are serviced, the corresponding formulation is an RPP. If all the edges or arcs must be visited, the planning problem is a CPP. Analogous to vehicle routing problems (VRPs), arc routing problems can be further described with additional constraints, such as time windows, constrained capacities, or time-dependent service costs (Tagmouti et al. 2010, 2011). For arc routing problems, the traversal can be completed by a single vehicle or a fleet of vehicles.

As an essential branch of arc routing problems, the RPP has received much attention from many scholars. The RPP was initially proposed as the routing problem of a single vehicle for a single depot (Orloff 1974), and the arc cost is fixed. Later, scholars extended the RPP for multiple vehicles (Quirion-Blais et al. 2017), multiple depots (Fernández et al. 2018) and changeable arc costs. An example of the RPP with varying arc costs is the literature that considers the RPP with various costs passing through the same arc in different directions, which is called the windy rural postman problem (Nossack et al. 2017). Furthermore, some scholars (Colombi et al. 2017) consider that the traversal priority of different arcs in the network is not the same, which is thus called hierarchical rural postman problem.

The time-dependent RPP (TDRPP), regarded as a special RPP, has several challenges for providing efficient models and algorithms. The TDRPP aims to find a shortest-time tour that services all the required arcs, and the time spent on each arc depends on its arrival time on the arc. The first challenge is that arcs in a time-dependent network can be traversed multiple times, so the number of arcs traversed in an optimal solution cannot be known in advance when modeling. Another challenge is that the time spent on each arc is a nonlinear function. Thus, the optimization problem cannot be easily solved by a commonly used mixed integer programming (MIP) solver.

Due to the challenges above, only a very few works have been proposed to solve the TDRPP (Tan and Sun 2011; Tan et al. 2013; Zanotti et al. 2019; Calogiuri et al. 2019). For the TDRPP, the first-in-first-out (FIFO) property (for each link, an earlier arrival time leads to an earlier departure time) should be guaranteed (Gendreau et al. 2015). For the existing literature (Tan et al. 2013; Calogiuri et al. 2019), the dynamic process of the service/travel time function in the network is not sufficiently represented under the FIFO condition. To achieve a suitable mathematical formulation (that better represents the time dependency under the FIFO condition) and to efficiently solve the resulting optimization problem, we make the following contributions:

- A new time-space network model is proposed for the formulation of time-dependent rural postman problems. The proposed model uses a discrete-time network that can represent a time-varying physical network. Therefore, the time dependency of the TDRPP is better described to capture its dynamic process, compared to the existing methods, by using a piecewise constant function with a limited number of intervals. Moreover, the FIFO property can be satisfied for the time spent on each arc using the proposed model.
- An optimality property is provided for the TDRPP that satisfies the FIFO rule for the time-dependent network. The optimality property is used to develop a customized GA to efficiently solve the considered TDRPP. To the best of our knowledge, no metaheuristic algorithm has been developed to solve a TDRPP problem. The GA proposed in this paper can provide near-optimal solutions in a reasonable computation time in comparison to a commercial solver and two commonly-used methods.

In the proposed time-space network model, based on the principle of the arc-path alteration, the route consists of service arcs and transition paths between every two successive service arcs. Several time slots discretize the planning horizon, and a dynamic physical network is mapped into a static time-space network to establish an integer programming formulation for the considered TDRPP. The developed GA is used to address the computational intractability of large-scale cases for the considered TDRPP. Comprehensive simulation experiments have been carried out to show the effectiveness of the proposed algorithm.

The remaining parts of this paper are organized as follows: Sect. 2 reviews the related literature. In Sect. 3, a time-space network model is proposed to formulate the considered TDRPP. In Sect. 4, the key optimality property of the solution is analyzed, and a customized GA is developed. Section 5 presents a comparison of the simulation results obtained via various solving methods. The conclusion and future research directions based on extensions of our methodology are given in Sect. 6.

2 Related work

The investigated routing problem can be regarded as a particular arc routing problem (Corberán and Prins 2010) with time-varying travel time for each link in the network. The following literature review relates to time-dependent arc routing problems and routing problems with time-dependent travel time functions.

2.1 Time-dependent arc routing problems

As mentioned in the introduction, arc routing problems aim to find a set of vehicle routes to traverse all predefined arcs in a given network (Gendreau et al.

2015). For arc routing problems, three problem types are identified: capacitated arc routing problem (CARP), Chinese postman problem and rural postman problem (Mour ao and Pinto 2017). Here, we focus on the literature on the issues with time dependencies.

The CARP is a particular arc routing problem in which the service process is subject to vehicle capacity constraints. In the direction of time-dependent CARPs, Tagmouti and his team have made noticeable contributions. Motivated by winter gritting applications, they studied the arc routing problem with a capacity limitation and a time-dependent service cost (Tagmouti et al. 2007). The studied arc routing problem is transformed into a node routing problem that is solved by using a column generation algorithm. In Tagmouti et al. (2010), a variable neighborhood descent heuristic algorithm is developed to solve the previously mentioned problem. Furthermore, a dynamic variant of arc routing problems with a capacity limitation is proposed in Tagmouti et al. (2011). Black et al. (2013) studied a kind of prize-collecting arc routing problem in which the transport manager must choose how to carry out a large number of truckloads and deliveries in the road network since the travel time of the road network varies over each day. Vincent and Lin (2015) also studied the problem of prize-collecting arc routing and proposed an iterative greedy heuristic algorithm.

The CPP is an arc routing problem in which a single vehicle needs to traverse all streets, and this problem was first proposed by Guan (1962). The first mathematical model for the time-varying CPP was proposed in Wang and Wen (2002) and their plan includes fuzzy time windows. However, the latter model can only solve the particular case in which all loops in the network pass through the constraints related to an original vertex 0. For the CPP with time-dependent travel times, Sun et al. (2011a, 2011b) proved that the time-dependent CPP is NP-hard and a dynamic programming method as well as a branch-and-bound algorithm were proposed. Sun et al. (2015) proposed a linear integer programming formulation, namely, the cyclic path formulation, which could be seen as an extended version of the method in Wang and Wen (2002).

The RPP is a more generalized CPP, where a subset of required arcs needs to be serviced. For TDRPPs, only a few works (Tan and Sun 2011; Tan et al. 2013; Calogiuri et al. 2019) have been published. An integer programming model using the arc-path alternation is established, and service arcs and travel arcs are defined to distinguish different patterns depending on whether each arc is serviced (Tan and Sun 2011; Tan et al. 2013). It is observed from (Tan et al. 2013) that the service/travel time spent on each arc is represented by a piecewise constant function with a limited number of intervals. This representation cannot sufficiently describe the dynamic process of the service/travel time function that changes continuously in practice. Moreover, the FIFO property of the time-dependent function cannot be guaranteed. In Calogiuri et al. (2019), a branch-and-bound algorithm is proposed for the TDRPP, where the FIFO conditions are satisfied. However, the mathematical representation is not provided for modeling the TDRPP. Furthermore, for the same service arc, the service time and the travel time have different patterns and this difference is not provided in Calogiuri et al. (2019).

Currently, the TDRPP problem lacks a mathematical model for better representing the time dependency of the service/travel time function in the network under the FIFO condition. Moreover, no efficient metaheuristic algorithm is available to solve the associated problem.

2.2 Time-dependent travel time function

In time-dependent routing problems, a key element is to define the travel time function of each arc in the network and to incorporate the time dependencies into the routing problem. Therefore, special attention has been paid to the study of travel time functions.

The concept of time-varying travel time was first proposed by (Beasley 1981) in which the travel time between customers is modeled by means of two sets of values in two nonoverlapping periods. Later, a robust and straightforward monotonicity property of the travel time function was identified in (Ahn and Shin 1991) to simplify the computational complexity since time-varying traffic conditions are involved. This property means that a vehicle arrives in a particular arc at an earlier time and it should also leave at an earlier time. The latter property is also referred to as first-in-first-out (FIFO).

The FIFO property represents a generalized travel time, and many scholars worked on deriving the travel time function before solving time-dependent routing problems. Until now, the majority of the travel time functions are obtained from the computation of speed variations (Sung et al. 2000; Horn 2000; Ichoua et al. 2003). Among them, Ichoua et al. (2003) proposed a model to generate the travel time function by satisfying the FIFO property based on a predicted speed function. This model, called the IGP model, divides the time into multiple periods during the planning horizon, and vehicles travel on a fixed-length street. The travel speed is considered as a piecewise linear function. Another method to obtain the FIFO travel function is to define a smoothed travel time function based on a step function of travel times (Fleischmann et al. 2004).

The easily-implemented IGP model has been widely used for computing the travel time on each arc of time-dependent routing problems. Typical examples are the time-dependent traveling salesman problem (Cordeau et al. 2014), the time-dependent vehicle routing problem (Huang et al. 2017) and the time-dependent rural postman problem (Calogiuri et al. 2019). More details regarding the travel time function can be found in (Gendreau et al. 2015). The IGP model is useful for generating the FIFO discrete service/travel time on each arc in our considered TDRPP.

2.3 Overall assessment of the literature

As a summarizing remark, TDRPPs have not been sufficiently investigated in the literature, and a mathematical model that better captures the time dependency of the TDRPP is needed. In addition, the FIFO property should be satisfied when representing the TDRPP. An efficient metaheuristic algorithm is needed for addressing the TDRPP that satisfies the FIFO property.

3 Problem setting and formulation

In this section, a time-space network model is proposed to formally describe the considered TDRPP problem. The first part defines the research problem, and the second part gives the detailed mathematical formulation of the studied routing problem.

3.1 Problem statement

In this paper, we investigate a time-dependent RPP problem. In the defined research problem, a shortest-time route is searched for traversing a subset of all the directed arcs in a time-dependent network. The route departs from the depot node at time 0 and returns to the depot node within the entire planning horizon. The arcs in the subset are referred to as service arcs. Because the service may need to be visited multiple times, each service arc has a service time (with service) and a travel time (without service). The remaining arcs only have a travel time as the service is not provided. Discrete-time-dependent functions represent both the travel and service times.

For the considered TDRPP problem, important assumptions are given as follows:

- Both the service time function and the travel time function of each arc satisfy the FIFO property, and these functions are assumed to be known in advance;
- The network is strongly connected. In other words, for any two nodes in the network, there exists a connected path between these two nodes.
- Each service arc must be serviced only once.
- Each arc can be visited more than once.

3.2 Graph representation

The network is regarded as a directed network $G(V, A)$, where V is a finite set of nodes, and A is a finite set of arcs $A = \{(i,j) | i \in V, j \in V, i \neq j\}$ between different adjacent nodes.

In the considered TDRPP, two types of arcs are identified: service arc and travel arc. The service arc is a particular arc that needs to be serviced, and the travel arc is a general arc without any service. In general, a particular arc (i, j) can be both a service arc and a travel arc, since it may be necessary to visit the same arc more than once to traverse all the service arcs. However, the time spent on the same arc can be different. If we define functions $\eta_{i,j}$ and $\tau_{i,j}$ as the service time and the travel time for the same arc (i, j), the value of $\eta_{i,j}$ is typically larger than the one of $\tau_{i,j}$ when entering arc (i, j) at the same time. Since we consider a time-dependent routing problem, these two functions can be extended as $\tau_{i,j}(t)$ and $\eta_{i,j}(t)$, where t is the arrival time of the corresponding vehicle at arc (i, j).

Assume that there are K service arcs. To facilitate the modeling, we define the origin node as node 1 and introduce a dummy node 0. Two dummy service arcs, (0,1) and (1,0), are included in the set of service arcs. The set of service arcs is defined as A_R ($A_R \subseteq A \cup \{(0,1),(1,0)\}$); A_R thus contains $\{(0,1),(1,0)\}$.

For the set A_R, the sequence of service arcs to be visited is defined as $\pi_0 = \{a_0, a_1, ..., a_{K-1}\}$ ($a_k \in A_R, k = 0, 1, ..., K-1$), where a_0 and a_{K-1} are dummy arcs (0,1) and (1,0). We observed that in A_R, there are $K-2$ real service arcs and 2 dummy service arcs.

Following the idea of arc-path alternation proposed in Tan and Sun (2011), a transition path defined as p_k is inserted between two successive service arcs a_{k-1} and a_k ($k = 1, 2, ..., K-1$). The transition path p_k, which connects the end node of service arc a_{k-1} and the start node of service arc a_k, could include several travel arcs in the network. We noted that p_k is an empty path if the end node of a_{k-1} and the start node of a_k are the same. Then, we define (p_k, a_k) ($k = 1, 2, ..., K-1$) as the k-th step. In particular, the 0-th step is (a_0). Based on the idea of arc-path alternation, each arc can be visited multiple times for flexible operations in practice. This means that a particular arc can appear in different steps of a complete route.

Based on the defined service arc sequence π_0 and the transition path p_k, a feasible complete route is defined as $\pi = \{a_0, p_1, a_1, p_2, a_2, ..., p_k, a_k, ..., p_{K-1}, a_{K-1}\}$. Because we consider a time-dependent routing problem, the complete route π is further represented as $\pi(t)$, indicating that route π starts at time t.

3.3 Time-space network model

In this subsection, based on the idea of arc-path alternation, a time-space network model is proposed to formulate the considered TDRPP.

Time-space network models have been widely used in the transportation domain to formulate the routing problems, including time constraints and location constraints (Yang and Zhou 2014; Meng and Zhou 2014). The time-space network framework is a discrete-time network. In the time-space network, the time-varying physical network is mapped to multiple time-invariant physical networks at different time instants. Therefore, the correlation between the time and the location can be captured when variations are considered.

For the time-space network modeling framework, a continuous-time planning horizon is discretized into several time slots, denoted as $\{0, \Delta t, 2\Delta t, ..., T \times \Delta t\}$. The notation Δt denotes a time slot (e.g., 1 minute) during which no perceptible changes of travel times are assumed to occur in a transportation network. The parameter T is a sufficiently large positive integer, and t ($t \in \{1, 2, ..., T\}$) represents the time when a vehicle enters arc (i, j). As a result, the time-space network decomposes the vehicle's overall routing process into several time slots. An example is illustrated in Sect. 3.4.

Before introducing the mathematical formulation, input parameters, general subscripts, and decision variables for the considered TDRPP are given in Tables 1 and 2. By using a time-space network framework, the TDRPP is formulated as follows:

Table 1 Input parameters and general subscripts

Symbol	Description
t_s	Start time within the planning horizon
$\pi(t_s)$	The complete TDRPP route starting at time t_s
V	Set of nodes in the directed network
A	Set of arcs in the directed network
A_R	Set of K service arcs. $A_R \subseteq A \cup \{(0,1),(1,0)\}$
Φ_t	Set of time indices $\{1, 2, ...T\}$, T is the largest time index
Φ_s	Set of step indices $\{0, 1, 2, ...K-1\}$, K is the number of service arcs
i, j	Node index, $i, j \in V$
(i, j)	Arc index, $(i, j) \in A$
t	Time slot index, $t \in \Phi_t$
k	Index used for the service arc and the transition path, $k \in \Phi_s$
$\tau_{i,j}(t)$	Travel time function, travel time on arc (i, j) at time t
$\eta_{i,j}(t)$	Service time function, service time on arc (i, j) at time t

Table 2 Decision variables

Symbol	Description
$X^{k,t}_{(i,j)}$	Binary variables. $X^{k,t}_{(i,j)} = 1$, if service arc (i, j) ($(i, j) \in A_R$) is served at step k and time t, otherwise is 0
$Y^{k,t}_{(i,j)}$	Binary variables. $Y^{k,t}_{(i,j)} = 1$, if travel arc $(i, j) \in A$ is traversed at step k and time t, otherwise is 0

$$\min \sum_{t\in\Phi_t} t \times X^{K-1,t}_{(1,0)} \tag{1}$$

subject to

$$\sum_{t\in\Phi_t} X^{0,t}_{(0,1)} = 1 \tag{2}$$

$$\sum_{t\in\Phi_t} X^{K-1,t}_{(1,0)} = 1 \tag{3}$$

$$\sum_{k\in\Phi_s} \sum_{t\in\Phi_t} X^{k,t}_{(i,j)} = 1, \quad \forall (i,j) \in A_R \tag{4}$$

$$\sum_{(i,j)\in A_R} \sum_{t\in\Phi_t} X^{k,t}_{(i,j)} = 1, \quad \forall k \in \Phi_s \tag{5}$$

$$\sum_{i\in V}\sum_{t\in\Phi_t} Y^{k,t}_{(i,j)} + \sum_{i\in V\cup 0}\sum_{t\in\Phi_t} X^{k-1,t}_{(i,j)} = \sum_{i\in V}\sum_{t\in\Phi_t} Y^{k,t}_{(j,i)} + \sum_{i\in V\cup 0}\sum_{t\in\Phi_t} X^{k,t}_{(j,i)}, \quad \forall j\in V, \forall k\in \Phi_s\backslash 0 \tag{6}$$

$$\sum_{i\in V}\sum_{t\in\Phi_t} t\times Y^{k,t}_{(j,i)} + \sum_{i\in V\cup 0}\sum_{t\in\Phi_t} t\times X^{k,t}_{(j,i)} = \sum_{i\in V}\sum_{t\in\Phi_t}(t+\tau_{i,j}(t))\times Y^{k,t}_{(i,j)} + \sum_{i\in V\cup 0}\sum_{t\in\Phi_t}(t+\eta_{i,j}(t))\times X^{k-1,t}_{(i,j)}, \quad \forall j\in V, \forall k\in\Phi_s\backslash 0 \tag{7}$$

$$X^{0,t_s}_{(0,1)} = 1 \tag{8}$$

$$X^{k,t}_{(i,j)} \in \{0,1\}, (i,j)\in A_R, k\in\Phi_s, t\in\Phi_t \tag{9}$$

$$Y^{k,t}_{(i,j)} \in \{0,1\}, (i,j)\in A, k\in\Phi_s, t\in\Phi_t \tag{10}$$

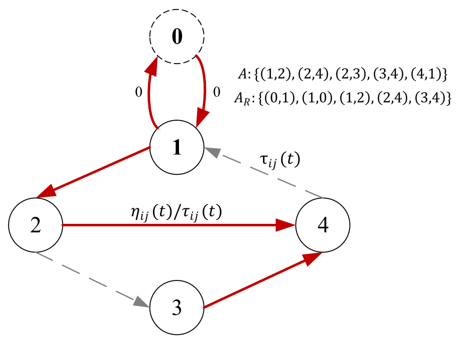
where the objective function presented in (1) is to minimize the total time spent to complete all the service arcs. This total time, which is defined as $f_\pi(t_s)$, is equivalent to the time of reaching the dummy service arc (1, 0) at the last step $K-1$. Constraints (2) and (3) guarantee that the two dummy arcs (0, 1) and (1, 0) can only be visited in the first step 0 and the last step $K-1$. Equality (4) ensures that each required arc needs to be serviced only once. Equality (5) constrains that only one service arc is included in each step. Constraint (6) guarantees that the number of arcs reaching a node in the route started from $t = t_s$ is equal to the number of arcs leaving that node. Constraint (7) guarantees that the time of each arc corresponds to a service/travel discrete-time-dependent function under the FIFO condition. Note that if the FIFO does not hold on this network, the equality sign should be replaced by the sign of 'greater or equal'. This modification allows the model to have a more general representation of time dependency. Equation (8) describes when the routing starts based on the value of t_s. Equation (9) and (10) indicate that the decision variables are binary values.

The optimization problem above is an integer programming (IP) that can be solved efficiently by commercial solvers (e.g., Gurob i (Gurobi 2018)) only for small-scale problems. In Sect. 4, an efficient algorithm will be developed to solve large-scale problems efficiently.

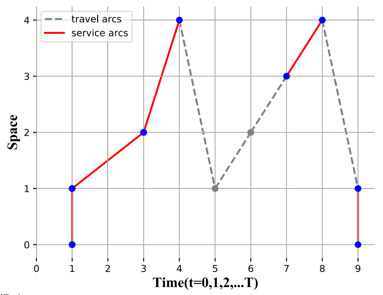
3.4 Illustrative example

In this subsection, an example is proposed to illustrate the routing process by using the proposed time-space network model. In this example, a simple network, including four nodes and one dummy node, is considered; and the routing of this network is presented in Fig. 1.

Figure 1a gives the topology of the example network, which includes four nodes 1-4 and one dummy node 0, which is connected to origin node 1. The feasible arcs and the


0
0
0
1
2
3
4
A: {(1,2), (2,4), (2,3), (3,4), (4,1)}
A_R: {(0,1), (1,0), (1,2), (2,4), (3,4)}
$\tau_{ij}(t)$
$\eta_{ij}(t)/\tau_{ij}(t)$

(a) A directed time-dependent network $G(V, A)$.



(b) A feasible route is mapped to the time-space network.

Fig. 1 Conceptual illustration of a feasible route for the TDRPP

required service arcs are given in set A and set A_R, respectively. The set A_R contains two dummy arcs (0,1) and (1,0), and their travel times are always 0. The solid lines and the dash lines represent the required service arcs and travel arcs, respectively.

Figure 1b maps a feasible route by using the proposed time-space network model. This model can clearly represent the specific location of the corresponding vehicle at a time. Figure 1b shows that the vehicle serves arc (1, 2) at time $1 \rightarrow 3$, and the service time is $\eta_{1,2}(1)$. The vehicle travels throughout arc (1, 2) at time $5 \rightarrow 6$, and then the travel time is $\tau_{1,2}(5)$.

Figure2 illustrates the feasible route of Fig.1b in more detail. In Fig.2, the route is divided into 5 steps ($K = 5$), and in each step, a service arc is included. The variables $Y^{k,t}_{(i,j)}$ and $X^{k,t}_{(i,j)}$ correspond to the choice of service arcs and travel arcs at step k and time t, respectively.

4 Solution algorithm

The previous section introduced our time-space network model for the considered TDRPP. Due to the computational intractability of the TDRPP, as suggested in Tan et al. (2013), solving such a model by using a commercial solver could take

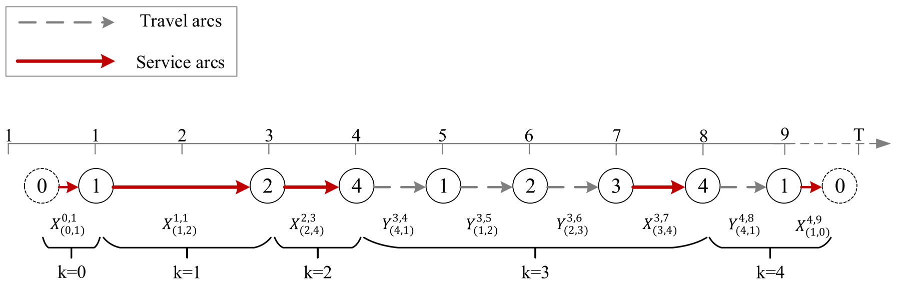


Fig. 2 Composition of a feasible route by using the proposed time-space network model

an unacceptable computation time when dealing with larger-scale problems. At the same time, the FIFO property should be satisfied in the considered TDRPP. For these reasons, we will first analyze how to incorporate the FIFO property in the TDRPP, and we will then design an efficient metaheuristic to solve the TDRPP with FIFO.

4.1 Property of the FIFO TDRPP

In this part, we study a crucial optimality property of the proposed TDRPP in which FIFO is satisfied both for the travel time function $\tau_{i,j}(t)$ and the service time function $\eta_{i,j}(t)$. It is worth noting that this optimality property is valid in a discrete-time network. The analysis of this property will help construct the solution of the considered TDRPP.

For a time-dependent network that satisfies FIFO, we define function $\rho_{i,j}(t)$ as the time spent on arc (i, j) with the arrival time t. If the service operation is performed for arc (i,j), $\rho_{i,j}(t)$ is equivalent to $\eta_{i,j}(t)$; otherwise, $\rho_{i,j}(t)$ is equal to $\tau_{i,j}(t)$. As a FIFO network is considered, both functions $\eta_{i,j}(t)$ and $\rho_{i,j}(t)$ should also satisfy FIFO.

In a discrete-time-dependent network, every arc satisfying the FIFO property can be described in the following form:

$$\text{if } t_1 \le t_2,\ t_1 + \rho_{i,j}(t_1) \le t_2 + \rho_{i,j}(t_2), \forall t_1, t_2 \in \{0, 1, ..., T\}, \tag{11}$$

where condition (11) means that, on each arc an earlier arrival leads to an earlier departure and a later arrival results in a later departure. In what follows, we will extend it from the arc between two nodes into the path between two nodes in the network.

Lemma 1 *In a strongly connected network that satisfies the FIFO property, for any two nodes i and j, the path between these two nodes also satisfies the FIFO property.*

Proof For this lemma, two conditions needs to be considered.

1. If nodes i and j are directly connected, i.e., $(i,j) \in A$, the path satisfies FIFO following Equation (11).
2. If nodes i and j are not directly connected, we let $v_1 = i$, $v_n = j$ and there are $n - 1$ arcs between these two nodes. The path can be described as $\{v_1, v_2, ...v_n\}$ and $(v_k, v_{k+1}) \in A, k = 1, .., n - 1$. The total time spent on this path is denoted as $f_{\{v_1,v_2,...,v_n\}}(t_s)$. We also know that $f_{(v_1,v_2)}(t_s) = t_s + \rho_{v_1,v_2}(t_s)$. Consider t_1 and t_2 $(t_1 \le t_2)$; we have the following conditions:

$$\begin{cases} f_{\{v_1,v_2\}}(t_1) = t_1 + \rho_{v_1,v_2}(t_1) \\ f_{\{v_1,v_2\}}(t_2) = t_2 + \rho_{v_1,v_2}(t_2). \end{cases} \tag{12}$$

Based on Equation (11) on arc (v_1, v_2), one can obtain the following inequality:

$$f_{\{v_1,v_2\}}(t_1) \leq f_{\{v_1,v_2\}}(t_2). \tag{13}$$

Furthermore, we extend the arc (v_2, v_3)

$$\begin{cases} f_{\{v_1,v_2,v_3\}}(t_1) = f_{\{v_1,v_2\}}(t_1) + \rho_{v_2,v_3}(f_{\{v_1,v_2\}}(t_1)) \\ f_{\{v_1,v_2,v_3\}}(t_2) = f_{\{v_1,v_2\}}(t_2) + \rho_{v_2,v_3}(f_{\{v_1,v_2\}}(t_2)). \end{cases} \tag{14}$$

Based on formula (13) and (14), we deduce that $f_{\{v_1,v_2,v_3\}}(t_1) \leq f_{\{v_1,v_2,v_3\}}(t_2)$, and recursively, we have that $f_{(v_1,\ldots v_n)}(t_1) \leq f_{\{v_1,\ldots,v_n\}}(t_2)$.

□

Remark 1 The route $\{v_1, v_2, ..., v_n\}$, $(v_k, v_{k+1}) \in A, k = 1, .., n-1$ has the FIFO property and this property can be extended for the route $\{v_1, v_2, ..., v_n\}$ in which node v_k and node v_{k+1}, $k = 1, .., n-1$ are directly or indirectly connected, following the results of Lemma 1.

Remark 2 We re-define $\{v_1, v_2, ..., v_n\}$ as the route in which every two successive nodes are directly or indirectly connected. Since this route has the FIFO property, the function $f_{\{v_1,\ldots,v_n\}}(t_s)$ is a nondecreasing function with respect to t_s.

For the considered TDRPP problem, there are K service arcs and $K-1$ transition paths. The complete route $\{a_0, p_1, a_1, p_2, a_2, ..., p_k, a_k, ..., p_{K-1}, a_{K-1}\}$ is equivalent to the route $(v_1, v_2, ..., v_{2K}, v_{2K+1})$, if we let $a_{k-1} = (v_{2k-1}, v_{2k})$ $(k = 1, 2, ..., K)$ and p_k be the path from v_{2k} to v_{2k+1} $(k = 1, 2, ..., K-1)$. As a result, one can observe that $f_\pi(t)$ is a nondecreasing function when the route of arc-path alternate is considered. With this observation, we can prove that the complete route π has a key optimality property, given as in Theorem 1.

Theorem 1 *In the time-dependent network that satisfies the FIFO property, the transition path between every two successive service arcs in the TDRPP shortest-time* $\pi(t_s)$ *is the corresponding shortest-time path.*

Proof The shortest-time TDRPP route is denoted as $\pi^*(t_s)$. Assuming that the k-th transition path p_k in the shortest-time route π^* is not the shortest-time path between arcs a_{k-1} and a_k, there is a shortest-time path s_k. Replace p_k with s_k to obtain a new route $\pi(t_s)$. The time spent on the two routes is expressed by Equation(15):

$$\begin{cases} f_{\pi^*}(t_s) = f_{\{a_0,p_1,a_1,\ldots,a_{k-1},p_k\}}(t_s) + f_{\{a_k,p_{k+1},\ldots,p_{K-1},a_{K-1}\}}(\dot{t}) \\ f_\pi(t_s) = f_{\{a_0,p_1,a_1,\ldots,a_{k-1},s_k\}}(t_s) + f_{\{a_k,p_{k+1},\ldots,p_{K-1},a_{K-1}\}}(\ddot{t}). \end{cases} \tag{15}$$

where $\dot{t}$ represents $f_{\{a_0,p_1,a_1,\ldots,a_{k-1},p_k\}}(t_s)$, while $\ddot{t}$ represents $f_{\{a_0,p_1,a_1,\ldots,a_{k-1},s_k\}}(t_s)$. Because s_k is the shortest-time path, $\ddot{t} < \dot{t}$ holds. $f_{\{a_k,p_{k+1},\ldots,p_{K-1},a_{K-1}\}}(t)$ is a nondecreasing function with respect to t, and Equation (16) holds:

$$f_{\{a_k,p_{k+1},\ldots,p_{K-1},a_{K-1}\}}(\ddot{t}) \leq f_{\{a_k,p_{k+1},\ldots,p_{K-1},a_{K-1}\}}(\dot{t}). \tag{16}$$

Following formula (15), we obtain $f_\pi(t_s) < f_{\pi^*}(t_s)$, indicating that π^* is not a shortest-time route. This contradicts the assumption that π^* is the shortest-time route. □

Theorem 1 indicates that for the FIFO TDRPP, the shortest-time transition path is a necessary condition to obtain the shortest-time route π^* and the minimum travel time on each transition path is needed. In the following subsection, we will use this property to design an efficient algorithm to solve the FIFO TDRPP.

4.2 Algorithm description

This section describes the genetic algorithm (GA) developed in this paper to solve the FIFO TDRPP. The first subsection gives a general description of the encoding procedure of our GA based on the optimality property proposed in Sect. 4.1, while the second subsection details how the GA is customized to solve the problem studied in this paper.

Genetic algorithms are random search techniques based on natural selection. GAs have been successfully applied to solve practical combinatorial optimization problems (Rabbouch et al. 2019; Shakibayifar et al. 2020). The GA in general uses a set of feasible solutions, which is defined as a population, and iteratively finds new high-quality solutions through the following customized operation: selection, crossover, and mutation. The selection operation is to pick up high-quality solutions from the current population and use them during the next iteration. The crossover and mutation operations generate new solutions to enlarge the diversity of the population with the aim of accelerating the convergence of the algorithm, i.e., improving local optimal solutions.

Encoding the solution of the considered TDRPP is complicated because we cannot know a priori how many arcs the optimal solution contains. Therefore, we cannot explicitly determine the solution's structure. We propose a simplified encoding based on Theorem 1 for constructing the solution. Based on the built solution structure, initial solutions are generated, and then these solutions are iteratively improved by the formulated selection, crossover, and mutation strategies in the GA framework.

4.2.1 Encoding

As introduced in Sect. 3, A_R represents the set of K required service arcs ($A_R = \{a_0, a_1, ..., a_{K-1}\}$). a_0 and a_{K-1} are dummy arcs. A transition path p_k is needed to connect two successive service arcs, and the shortest-time path p_k^* is needed to obtain the shortest-time route π^*. The p_k^* is obtained by shortest-path algorithms, such as time-dependent Dijkstra's algorithm (Kaufman and Smith 1993). This method is effective in time-dependent networks that satisfy FIFO.

The shortest-time route π^* of the TDRPP is obtained by integrating the optimal sequence of service arcs and the shortest-time transition paths between every two successive service arcs. Here, we focus on determining the optimal sequence of service arcs by using the proposed genetic algorithm. The detailed encoding for this sequence is as follows:

$$a_0, a_1, ..., a_{K-1}, \tag{17}$$

where the order $a_1, ...a_{K-2}$ represents the sequence of real service arcs. The locations of a_0 and a_{K-1} are fixed in the beginning and the end of the sequence.

An encoding example (0, 1, 2, 3, 4) is presented in Fig. 3, where dummy arcs 0 and 4 are fixed. By changing the travel order of arcs 1 and 2, we obtain another available sequence (0, 2, 1, 3, 4). In the next section, this sequence is used for the fitness evaluation of GA. Each available sequence corresponds to a complete TDRPP route π. The fitness evaluation is to calculate the total routing time of the TDRPP.

4.2.2 Genetic algorithm

Based on the designed encoding, we now describe the framework details of our GA to solve the considered TDRPP.

Algorithm 1 GA Procedures for the considered TDRPP

```
Input: $G(V, A)$: Time-dependent network. $P$: population.
Output: Route $\pi$: The route corresponding to the best solution. $f_\pi$: Its fitness value (i.e., the total routing time).
1: Initialize $P(iter)$,
2: while $iter \leq iter_{max}$ do
3:     Evaluate the fitness $f_\pi$ of each solution in population $P(iter)$,
4:     Divide $P(iter)$ evenly and randomly into $N$ groups,
5:     for $i = 1 : N$ do
6:         Select the best and the second-best chromosomes from $P_i(iter)$ as $c_i^1(iter)$ and $c_i^2(iter)$,
7:         Crossover over $c_i^1(iter)$ and $c_i^2(iter)$ as $c_i^3(iter)$ and $c_i^4(iter)$,
8:         Mutation over $c_i^1(iter)$ and $c_i^2(iter)$ as $c_i^5(iter)$ and $c_i^6(iter)$,
9:         $P_i(iter) = c_i^1(iter) \cup c_i^2(iter) \cup ...c_i^6(iter)$,
10:    end for
11:    $P(iter + 1) = P_1(iter) \cup P_2(iter) \cup ... \cup P_N(iter)$,
12: end while
13: return route $\pi$ and fitness $f_\pi$.
```

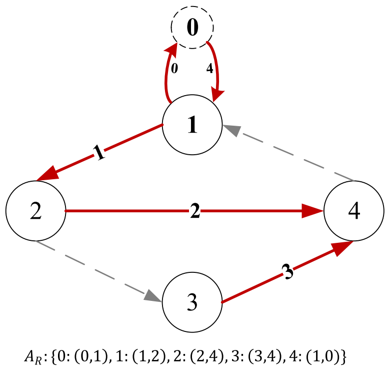


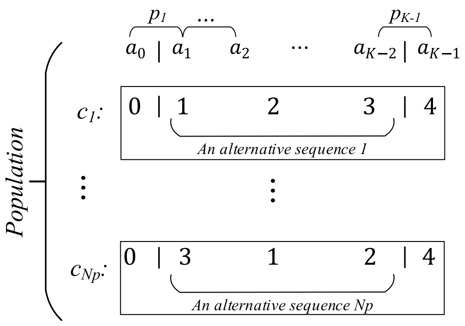
Fig. 3 Example of encoding with the arc number

Algorithm 1 gives the pseudocode of our GA. This algorithm contains the operations of selection, crossover and mutation based on the fitness evaluation of population $P(iter)$ at each iteration. The population contains N_p chromosomes, each of which represents a feasible sequence of service arcs. Initially these chromosomes are generated randomly. The fitness of each chromosome refers to the total time spent $f_\pi(t_s)$ on a feasible TDRPP route. According to Theorem 1, we insert the shortest-time transition path between two successive service arcs to obtain a complete route π^*, and this integration is given in Fig. 4.

In Algorithm 1, one key point is to evaluate the fitness of each chromosome in the population. In this study, the service time $\eta_{i,j}(t)$ of each service arc(i, j) is different from the travel time $\tau_{i,j}(t)$ (usually greater than the travel time). In this way, after a given start time t_s, the fitness value of each chromosome is composed of the service time of the service arc and the transition path time between the two successive service arcs. The calculation details of fitness values are given in Algorithm 3 in the appendix.

The selection operation is based on a tournament selection strategy (Kramer 2017). In this strategy, population P is divided into N groups evenly and randomly to maintain population diversity. In each group, the chromosomes with the lowest and the second lowest fitness value during the current iteration *iter* are selected as the candidate chromosome for the next iteration $iter + 1$. Crossover and mutation operations for the next iteration generate the remaining chromosomes in all the groups.

For the crossover operation, David's order-based strategy is considered in each group since this strategy is fast and efficient, as suggested by Kramer (2017). The idea behind this strategy is to transmit information about relative ordering to the offspring. Under this strategy, the crossover operation is performed based on two chromosomes with the lowest fitness values that are regarded as parents. For each group, the best chromosome and the second-best chromosome are used. Let c_1 and c_2 be the chromosomes with the lowest fitness value and the second lowest one, respectively. Two random crossover points are created, and the arc numbers between these two points are copied from c_1 to a new chromosome in the same positions. Then, remove these copied numbers from c_2 and insert the remaining arc numbers in c_2 into the new chromosome in order from the second crossover point. When the second half of the new chromosome is filled, fill it from the first half. The crossover process is illustrated in Fig. 5. In particular, 0 and 9 represent dummy arcs fixed on both sides



p_1
…
p_{K-1}
a_0 | a_1 a_2 ··· a_{K-2} | a_{K-1}
Population
c_1:
0 | 1 2 3 | 4
An alternative sequence 1
c_{Np}:
0 | 3 1 2 | 4
An alternative sequence Np

Fig. 4 Population composition of the proposed genetic algorithm

of the chromosome. The two random crossover points are 4 and 7 so that (3, 2, 7) in c_1 is copied to the same position on the new chromosome. Then, remove (3, 2, 7) in c_2, and put the number (1, 4, 5, 6, 8) into the remaining part of the new chromosome.

Regarding the mutation operation, the swap strategy is employed because it provides a new order but makes small changes to the existing order of the considered TDRPP solution. In this strategy, two positions in the chromosome are selected randomly, and their arc numbers are swapped.

4.2.3 Initial solution

Initial solutions are important for the performance of a metaheuristic algorithm (Sharma et al. 2011). To improve the performance of the proposed GA, we use the result of the greedy algorithm (Yu and Yang 2019), which is regarded as a simple and fast heuristic, as the initial solution.

The greedy algorithm is used to construct a complete sequence of service arcs for the considered TDRPP. Starting from the first dummy service arc, this sequence depends on the shortest transition path between two successive service arcs. The shortest-path is computed by using the Dijkstra's algorithm. The initial solution will be a chromosome in the initial population of the proposed GA.

5 Computational results

In this section, extensive numerical experiments are conducted to compare the proposed GA with two commonly-used metaheuristic algorithms. These three metaheuristic algorithms are also compared with the commercial solver by using different models. First of all, experimental settings are presented. Then, the computational results for small and large case studies are discussed.

5.1 Setup of the experiments

This part introduces the experimental settings for the assessment of the proposed methods to solve the FIFO TDRPP. Section 5.1.1 introduces the scenarios generation and the overall experimental setup. Section 5.1.2 discusses the parameter selection of three metaheuristic algorithms.

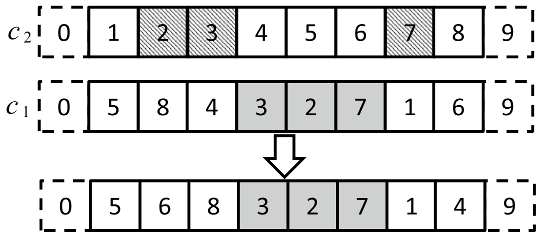


Fig. 5 Illustrative crossover operation (The locations of 0 and 9 are fixed, as they represent dummy service arcs)

5.1.1 Scenarios generation and overall setting

In this paper, the network $G(V, A)$ consists of multiple nodes and directed arcs that simulate various types of streets. The network scale is the following: 5 *km* by 5 *km* urban area, while all the nodes are randomly generated within this range. First, a closed-loop, including $|V|$ nodes, is created. The loop ensures that the generated network is strongly connected. The remaining $|A| - |V|$ arcs are randomly generated based on the created loop. The service arcs are randomly selected from A in different proportions $R = (30\%, 50\%, 70\%)$. This generation method is consistent with existing research (Tan et al. 2013; Calogiuri et al. 2019). In this time-dependent network, each time unit represents 1 minute. The travel speed of each arc is between [0.2,0.8] *km/min* with an average value of 0.5. The speed is updated every five time units. For small-scale and large-scale scenarios, we set up the planning horizon (T) of 200 and 1200 time units, and these numbers ensure finding feasible solutions. Here, we consider the service time for each service arc as twice the corresponding travel time, as suggested by Tan et al. (2013). The detailed configurations of the studied scenarios are given in Table 3. The scenario names are represented by the form G_{a-b}, where a represents a specific network, and b corresponds to a different proportion of service arcs in the network. The time-dependent network data generated can be found in Yu (2020).

In the considered TDRPP, the travel time function of each arc in the network G is generated using the IGP modeling approach proposed in Ichoua et al. (2003). The IGP approach considers a continuous-time function that satisfies the FIFO property. In this paper, we use a discrete-time model. We directly round the continuous-time function generated by the IGP model and modify it through Algorithm 2 to obtain a discrete-time function that satisfies the FIFO.

Algorithm 2 Discrete time function correction process

```
Input: $t_i$: Time to reach node $i$ of arc $(i,j)$. $\tau_{i,j}(t_i)$: Travel time of arc $(i,j)$ at $t_i$. $\eta_{i,j}(t_i)$: Service time of arc $(i,j)$ at $t_i$.
Output: $\tau_{i,j}(t_i+1), \eta_{i,j}(t_i+1)$,
1: if $t_i + 1 + \tau_{i,j}(t_i+1) < t_i + \tau_{i,j}(t_i)$ then
2:     $\tau_{i,j}(t_i+1) = t_i + \tau_{i,j}(t_i) - t_i - 1$,
3: end if
4: if $t_i + 1 + \eta_{i,j}(t_i+1) < t_i + \eta_{i,j}(t_i)$ then
5:     $\eta_{i,j}(t_i+1) = t_i + \eta_{i,j}(t_i) - t_i - 1$,
6: end if
7: return $\tau_{i,j}(t_i+1), \eta_{i,j}(t_i+1)$.
```

The results on the proposed GA are compared with two commonly-used metaheuristic algorithms, variable neighborhood search (VNS) and ant colony optimization (ACO), and a commercial solver Gurobi. These four methods are all based

Table 3 Setting of case studies by involving various $R\%$

Scenarios			$\lvert V\rvert$	$\lvert A\rvert$
$R = 30\%$	$R = 50\%$	$R = 70\%$		
G_{1-1}	G_{1-2}	G_{1-3}	5	6
G_{2-1}	G_{2-2}	G_{2-3}	5	9
G_{3-1}	G_{3-2}	G_{3-3}	5	12
G_{4-1}	G_{4-2}	G_{4-3}	5	15
G_{5-1}	G_{5-2}	G_{5-3}	5	18
G_{6-1}	G_{6-2}	G_{6-3}	10	20
G_{7-1}	G_{7-2}	G_{7-3}	10	30
G_{8-1}	G_{8-2}	G_{8-3}	10	40
G_{9-1}	G_{9-2}	G_{9-3}	20	40
G_{10-1}	G_{10-2}	G_{10-3}	20	60
G_{11-1}	G_{11-2}	G_{11-3}	20	80
G_{12-1}	G_{12-2}	G_{12-3}	30	60
G_{13-1}	G_{13-2}	G_{13-3}	30	90
G_{14-1}	G_{14-2}	G_{14-3}	30	120

on the proposed TSN model. For the three metaheuristic algorithms (GA, VNS and ACO), the result resulting from the greedy algorithm is all implemented as the initial solution. The method using the solver Gurobi based on the TSN model is referred to as TSN-G, for short. The algorithms VNS and ACO are briefly described as follows:

- VNS is regarded as a simple and effective metaheuristic, and the VNS consists of two phases (the improvement phase for the local search and the shaking phase for the global search). Following the property of the FIFO TDRPP and the encoding method presented by (17) in Sect. 4.2.1, the general VNS discussed in Hansen et al. (2017) is implemented here.
- ACO is another commonly-used metaheuristic, which is inspired by the behavior of the ant colony searching for food. The considered ACO here also uses the property of the FIFO TDRPP and the encoding method scheme and the algorithm framework can be found in Halim and Ismail (2019).

In addition to these above methods based on the proposed TSN model, the results of the piecewise constant (PWC) model (Tan et al. 2013) and the RPP model are also included for the comparison:

- In the PWC model, both the travel time function and the service time function of the TDRPP are approximated as piecewise constant functions (typically 4 segments for each function). The optimization problem is formulated as mixed integer linear programming and the route is obtained by the solver Gurobi. This method is notated as PWC-G for short.

- The RPP method transforms the TDRPP into a time-invariant RPP, following the idea adopted by Ichoua et al. (2003) for the time-dependent vehicle scheduling problem. The time spent on each arc is averaged over the whole planning. The time-invariant routing problem is relatively easier to be solved than the time-dependent routing problem. The resulting integer programming formulation is solved by the Gurobi solver. The integer programming formulation adopts the time-invariant model in Tan et al. (2013). This method is hereinafter called RPP-G.

All the numerical simulations are performed on a computer with 2.60 GHz CPU and 8GB of memory. The solver Gurobi 9.0 is used for solving the optimization problem resulting from the TSN, PWC and RPP models. For each scenario, three metaheuristic algorithms (GA, VNS, ACO) are computed 25 times. The maximum computation time is set to 2 hours. All the algorithms are implemented by Python 3.7, and the library Numba is used to speed up the computations.

5.1.2 parameter setting of metaheuristic algorithms

For the considered metaheuristic algorithms, parameters should be selected carefully. In this part, we set up important parameters of the GA, VNS and ACO algorithms suitable for the considered TDRPP.

In the GA, the population size and the maximum iteration number should be decided. We select the small/middle scenarios G_{2-3}, G_{3-3},G_{4-3}, G_{5-3}, G_{6-3} and G_{7-3} to evaluate the fitness convergence curves under different population sizes. Figure 6 gives the compared curves for these scenarios. From Fig.6, we can see that both the accuracy and convergence speed of GA increase when the population size becomes larger. However, as shown in Table 4, the calculation time also increases as the population size grows. In order to balance the calculation time and the solution quality, the population size and the number of iterations are set to be 360 and 400.

For a fair comparison, we use the maximum fitness evaluations (MaxFEs), which is a common criterion (Huang et al. (2019)) to terminate different metaheuristic algorithms. The same MaxFEs value is set when comparing the GA, VNS and ACO algorithms. The key parameters of the VNS and ACO algorithms are shown in Table 12 in the appendix.

5.2 Results on small-scale cases

In this subsection, the experimental results on the small-scale scenarios are presented using six different methods. The compared results are reported in Tables 5, 6 and 7, respectively.

Table 5 compares the averaged objective function value (i.e., the total time spent in the network to serve all service arcs), for each small-scale scenario presented in Table 3. In this table, the averaged objective values for all small-scale scenarios are also included at the bottom. In Tables 5 and 8, the bold value represents the best solution among all the methods, and the performance of the corresponding algorithm is better. The minimal and maximal values obtained from the three metaheuristic algorithms are recorded in Table 6.

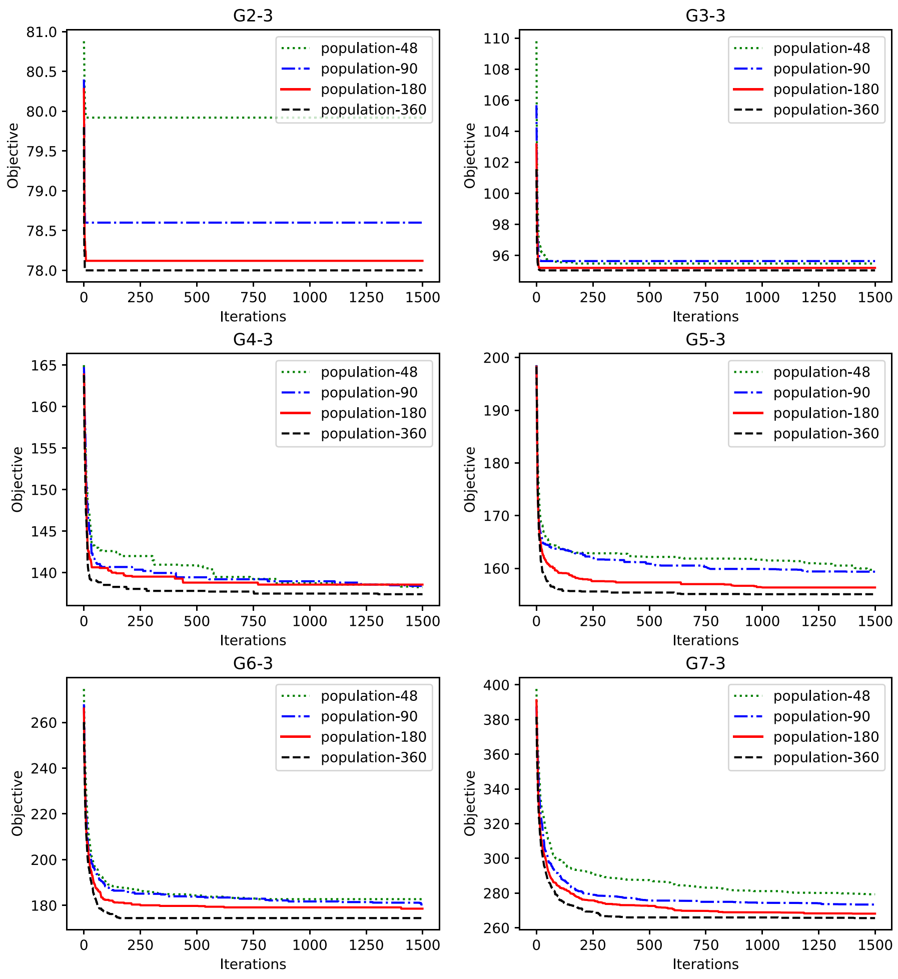


Fig. 6 Convergence curves of fitness values for different population sizes

Table 4 Average computation time of different population sizes in 400 iterations. (Unit: sec)

Scenarios	Population			
	48	90	180	360
G_{2-3}	1.96	3.48	6.97	13.34
G_{3-3}	2.33	4.23	8.86	17.78
G_{4-3}	2.20	4.43	8.68	17.15
G_{5-3}	2.45	4.60	8.58	18.29
G_{6-3}	8.18	15.25	29.22	56.62
G_{7-3}	12.25	22.38	43.22	83.41

Table 5 shows that the proposed GA is better than the other five methods. The algorithms based on the TSN model (GA, VNS, ACO, and TSN-G) provide better objective values than the methods using the PWC model and the RPP model. This means that considering time dependence in RPP can further reduce the value of the objective function. The TSN-G method can return optimal solutions by using the solver Gurobi. However, the TSN-G method fails to return feasible solutions within 2 hours as the scenario scale becomes larger (i.e., G_{3-2}, G_{3-3}, G_{4-2}, G_{4-3}, G_{5-2}, and G_{5-3}). For these cases where the TSN-G method can find the optimal solution, the GA and VNS algorithms reach the same objective value. The GA and VNS can give a high-quality solution for all small-scale cases. The ACO, in general, outperforms the RPP-G and PWC-G methods, but sometimes gives a poor solution.

Table 6 records the minimal/maximal value of three metaheuristic algorithms (GA, VNS, and ACO). This table indicates that the proposed GA provides better minimal or maximal values than the VNS and the ACO when the solver Gurobi cannot return the optimal solution.

Table 7 presents the computation times of the different methods for the reported small-scale cases. Table 7 shows that the RPP-G method has considerably shorter computation times compared to the other algorithms as the problem complexity is reduced. The TSN-G method takes the longest computation time to get the exact optimal solution. The computation time of the PWC-G is between the RPP-G

Table 5 Averaged objective value of different methods for the small-scale cases. (Units: Minutes)

Model	TSN				PWC	RPP
Scenarios	Algorithm					
	GA	VNS	ACO	TSN-G	PWC-G	RPP-G
G_{1-1}	45.00	45.00	45.00	45	45	45
G_{1-2}	52.00	52.00	52.00	52	52	52
G_{1-3}	62.00	62.00	62.00	62	62	62
G_{2-1}	67.00	67.00	67.00	67	67	67
G_{2-2}	**67.00**	**67.00**	**67.00**	**67**	**67**	75
G_{2-3}	**78.00**	**78.00**	90.20	**78**	81	81
G_{3-1}	53.00	53.00	53.00	53	53	53
G_{3-2}	**87.00**	**87.00**	**87.00**	–	100	102
G_{3-3}	**95.00**	**95.00**	95.52	–	109	112
G_{4-1}	**72.00**	**72.00**	95.00	**72**	95	95
G_{4-2}	**119.00**	**119.00**	125.00	–	144	151
G_{4-3}	**131.12**	131.60	132.00	–	157	156
G_{5-1}	**95.00**	**95.00**	**95.00**	**95**	99	100
G_{5-2}	**137.00**	**137.00**	153.40	–	152	166
G_{5-3}	**145.60**	146.80	147.00	–	160	179
Average	**87.05**	87.16	91.07	–	96.27	99.73

Table 6 The minimal and the maximal values of three metaheuristic algorithms for the small-scale cases. (Units: Minutes)

Scenarios	Algorithm		
	GA	VNS	ACO
	(min,max)	(min,max)	(min,max)
G_{1-1}	[45,45]	[45,45]	[45,45]
G_{1-2}	[52,52]	[52,52]	[52,52]
G_{1-3}	[62,62]	[62,62]	[62,62]
G_{2-1}	[67,67]	[67,67]	[67,67]
G_{2-2}	[67,67]	[67,67]	[67,67]
G_{2-3}	[78,78]	[78,78]	[81,91]
G_{3-1}	[53,53]	[53,53]	[53,53]
G_{3-2}	[87,87]	[87,87]	[87,87]
G_{3-3}	[95,95]	[95,95]	[95,96]
G_{4-1}	[72,72]	[72,72]	[95,95]
G_{4-2}	[119,119]	[119,119]	[124,126]
G_{4-3}	[129,132]	[129,136]	[132,132]
G_{5-1}	[95,95]	[95,95]	[95,95]
G_{5-2}	[137,137]	[137,137]	[151,155]
G_{5-3}	[142,147]	[144,148]	[147,147]
Average	[86.67,87.20]	[86.80,87.53]	[90.20,91.33]

Table 7 Averaged computation times of different methods for the small-scale cases. (Units: sec)

Model	TSN				PWC	RPP
Scenarios	Algorithm					
	GA	VNS	ACO	TSN-G	PWC-G	RPP-G
G_{1-1}	2.052	2.512	3.592	17.352	0.011	0.001
G_{1-2}	2.861	2.965	6.035	6.925	0.030	0.001
G_{1-3}	2.234	2.891	6.852	16.035	0.064	0.001
G_{2-1}	3.488	3.744	7.551	127.341	0.069	0.001
G_{2-2}	3.845	3.932	9.350	266.833	0.271	0.001
G_{2-3}	3.476	3.841	16.018	86.107	0.355	0.002
G_{3-1}	2.745	2.911	7.992	36.419	0.194	0.001
G_{3-2}	4.863	5.293	18.482	–	2.163	0.002
G_{3-3}	4.834	4.771	25.567	–	5.698	0.003
G_{4-1}	3.235	3.296	8.612	240.945	0.167	0.001
G_{4-2}	4.223	4.756	26.582	–	2.055	0.003
G_{4-3}	4.516	5.314	36.882	–	10.421	0.004
G_{5-1}	3.796	4.396	13.662	972.239	1.025	0.002
G_{5-2}	4.931	5.581	33.012	–	18.894	0.004
G_{5-3}	4.730	5.442	59.857	–	1005.926	0.007
Average	3.722	4.110	18.670	–	69.823	0.002

method and the TSN-G method. The proposed GA uses less than 5 seconds of computation time for all small-scale scenarios. The computation time of VNS is close to that of GA, while the ACO algorithm computes longer than the GA and the VNS. The computation time of ACO increases more quickly than the GA and the VNS as the case scale grows.

5.3 Results on large-scale cases

This part compares the experimental results concerning the large-scale scenarios ($G_6 - G_{14}$). The computational results of the compared methods are reported in Tables 8, 9 and 10. Because the TSN-G method cannot find any solution within the maximum computation time, the results of the TSN-G method are not listed in these tables.

Table 8 compares the averaged objective function values of five different methods (i.e., the developed GA, VNS, ACO, RPP-G, and PWC-G) for each large-scale scenario. As the scale of the scenario increases, the problem becomes increasingly complex. The proposed GA also has shorter objective values on average than the other methods. The PWC-G method obtains a solution only for a small part of the large-scale scenarios. Since the RPP-G method does not consider the time dependence, a feasible solution can always be given in large-scale scenarios. However, in large-scale cases, the objective function values obtained by the three metaheuristic algorithms (GA, VNS, and ACO) considering time dependence are significantly better than those obtained by the RPP-G method. For the three metaheuristic algorithms, the proposed GA has better objective values for most large-scale cases. For a few cases, the ACO algorithm has a slightly smaller objective value than the proposed GA, but the computation time of the ACO is significantly longer than the other two metaheuristic algorithms. Furthermore, the proposed GA has the best average performance.

Table 9 reports the minimal and maximal values obtained from the three metaheuristics. This table shows that the proposed GA and the VNS have better performance for most cases concerning this criterion. In general, the proposed GA has a lower minimal or maximal objective against the VNS and the ACO. It is noted that the ACO has lower minimal or maximal objective values for a few cases (e.g., G_{7-3}), than the proposed GA and the VNS. However, on average, the proposed GA outperforms the VNS and the ACO.

Table 10 presents the computation times of different algorithms for large-scale cases. Table 10 shows that the proposed GA provides high-quality solutions within a reasonable computation time. The computation time of all compared methods grows as the case becomes larger. In Table 10, the computation time of the RPP-G is very short, but the solution quality is the worst (as shown in Table 8). The computation times of the proposed GA and the VNS are close to each other and considerably shorter than the ACO algorithm.

Table 8 Averaged objective value of different methods for the large-scale cases. (Units: Minutes)

Model	TSN			PWC	RPP
Scenarios	Algorithm				
	GA	VNS	ACO	PWC-G	RPP-G
G_{6-1}	**126.00**	**126.00**	126.96	126	132
G_{6-2}	131.04	**131.00**	134.00	160	136
G_{6-3}	**170.96**	175.80	178.40	185	184
G_{7-1}	**119.00**	119.12	**119.00**	129	119
G_{7-2}	**175.96**	180.84	181.00	–	234
G_{7-3}	267.80	270.16	**264.92**	–	307
G_{8-1}	**142.24**	142.48	143.84	156	154
G_{8-2}	**236.12**	238.32	247.20	–	312
G_{8-3}	345.72	354.12	**338.16**	–	432
G_{9-1}	224.56	**213.60**	300.96	227	227
G_{9-2}	**290.40**	296.16	297.10	–	343
G_{9-3}	376.35	385.76	**369.00**	–	401
G_{10-1}	**219.92**	222.32	222.80	–	240
G_{10-2}	**363.52**	367.52	364.40	–	433
G_{10-3}	559.04	563.08	**548.60**	–	605
G_{11-1}	**301.88**	302.08	301.90	–	348
G_{11-2}	589.00	601.48	**571.40**	–	672
G_{11-3}	**819.12**	847.40	820.60	–	897
G_{12-1}	374.60	367.76	**361.00**	–	410
G_{12-2}	**454.08**	463.24	454.20	–	525
G_{12-3}	**675.20**	703.04	703.60	–	761
G_{13-1}	**443.52**	454.16	452.80	–	513
G_{13-2}	691.76	716.84	**680.00**	–	781
G_{13-3}	**1015.52**	1021.32	1041.60	–	1093
G_{14-1}	**469.84**	478.44	472.00	–	496
G_{14-2}	**779.04**	784.92	799.00	–	857
G_{14-3}	**1075.84**	1085.76	1104.00	–	1146
Average	**423.63**	430.10	429.56	–	472.52

5.4 Summary of experimental results

In this section, we summarize the experimental results as follows:

1. The experimental results show that the RPP-G method has the fastest solution speed, but this does not consider the time dependence. Therefore, this method cannot generate high quality solutions for the considered TDRPP.

Table 9 The minimal and the maximal values of different methods for the large-scale cases. (Units: Minutes)

Scenarios	Algorithm		
	GA	VNS	ACO
	(min,max)	(min,max)	(min,max)
G_{6-1}	[126,126]	[126,126]	[126,129]
G_{6-2}	[131,134]	[131,131]	[134,134]
G_{6-3}	[170,179]	[170,182]	[176,180]
G_{7-1}	[119,119]	[119,122]	[119,119]
G_{7-2}	[171,187]	[171,189]	[181,181]
G_{7-3}	[257,277]	[259,282]	[254,275]
G_{8-1}	[142,143]	[142,143]	[143,152]
G_{8-2}	[230,248]	[231,252]	[237,253]
G_{8-3}	[332,364]	[340,376]	[326,349]
G_{9-1}	[214,226]	[212,225]	[291,306]
G_{9-2}	[284,295]	[284,315]	[286,306]
G_{9-3}	[365,391]	[365,421]	[366,372]
G_{10-1}	[218,233]	[218,236]	[222,226]
G_{10-2}	[351,376]	[352,382]	[356,367]
G_{10-3}	[538,565]	[535,575]	[530,557]
G_{11-1}	[285,312]	[289,314]	[294,302]
G_{11-2}	[559,614]	[581,633]	[565,584]
G_{11-3}	[778,853]	[807,892]	[813,830]
G_{12-1}	[361,385]	[361,388]	[361,361]
G_{12-2}	[435,473]	[435,500]	[444,462]
G_{12-3}	[652,696]	[666,728]	[684,720]
G_{13-1}	[423,472]	[434,473]	[443,461]
G_{13-2}	[674,715]	[693,767]	[680,680]
G_{13-3}	[986,1032]	[988,1034]	[1024,1047]
G_{14-1}	[457,485]	[455,496]	[467,479]
G_{14-2}	[737,809]	[757,809]	[790,805]
G_{14-3}	[1043,1103]	[1051,1126]	[1100,1114]
Average	[408.81,437.48]	[413.78,448.78]	[422.67,464.48]

2. The PWC-G method approximates the time function of each arc of the TDRPP to a piecewise constant function. The results show that this method obtains better objective values than the RPP-G method. However, the objective value obtained by this method still cannot compete (in terms of solution quality) with the metaheuristics based on the TSN model.
3. The TSN-G method obtains the optimal objective value in the small-scale cases, but this method cannot find feasible solutions for the large-scale cases.

Table 10 Averaged computation times of the different methods for the large-scale cases. (Units: sec)

Model	TSN			PWC	RPP
Scenarios	Algorithm				
	GA	VNS	ACO	PWC-G	RPP-G
G_{6-1}	13.412	15.042	48.691	1.114	0.047
G_{6-2}	15.121	16.621	105.914	7.702	0.060
G_{6-3}	13.024	18.347	194.214	35.655	0.110
G_{7-1}	11.665	14.652	86.492	7.251	0.070
G_{7-2}	16.597	20.893	228.491	–	0.117
G_{7-3}	19.856	25.142	411.372	–	0.194
G_{8-1}	14.951	18.417	146.332	24.016	0.108
G_{8-2}	23.587	27.649	373.945	–	0.224
G_{8-3}	25.921	33.992	714.635	-	0.293
G_{9-1}	56.062	61.289	1236.041	63.95	0.139
G_{9-2}	81.253	88.951	1221.761	–	0.219
G_{9-3}	110.831	113.251	2268.591	–	0.395
G_{10-1}	91.285	91.674	1017.372	–	0.245
G_{10-2}	122.291	122.256	2664.914	–	0.497
G_{10-3}	153.950	170.412	5102.382	–	0.969
G_{11-1}	94.967	106.634	1917.673	–	0.511
G_{11-2}	128.181	143.962	4591.321	-	1.047
G_{11-3}	185.112	230.452	7200	–	1.775
G_{12-1}	171.665	173.427	1956.714	–	0.328
G_{12-2}	228.132	226.381	5137.281	–	0.641
G_{12-3}	360.138	340.091	7200	–	1.141
G_{13-1}	257.675	286.112	4496.481	–	1.201
G_{13-2}	346.856	408.982	7200	–	1.845
G_{13-3}	505.279	507.271	7200	–	5.257
G_{14-1}	275.675	300.912	7200	–	1.423
G_{14-2}	385.142	380.962	7200	–	5.984
G_{14-3}	498.129	494.531	7200	–	21.264
Average	155.806	164.382	3122.986	–	1.708

4. The proposed GA obtains the same optimal solution provided by the TSN method for the small-scale cases, and the computation time is less than 5 seconds. For the large-scale cases, the proposed GA outperforms the VNS and the ACO for most cases and has the strongest comprehensive performance in terms of the solution quality and the computation time. (i.e. the proposed GA outperforms the other algorithms when looking at the average results)

6 Conclusion and future research

In this paper, a new time-space network model is proposed for addressing time-dependent rural postman problems (TDRPPs). The proposed model is based on the idea of the arc-path alternation. This model can represent any form of travel time function and is thus considered as a more general model than the existing models. Based on a time-space network representation, an integer programming formulation is established to solve the studied problem. The tested scenarios become computationally intractable as the number of variables and constraints grows considerably (see more details in the Appendix).

We investigate the FIFO property of the considered time-dependent network and a crucial optimality property of the FIFO TDRPP. Based on the optimality property, a well-customized genetic algorithm (GA) is proposed to efficiently solve the considered TDRPP. Comprehensive simulation experiments have been conducted to show the effectiveness of the proposed GA on different types of scenarios. The simulation results show that our GA provides high-quality solutions in a reasonable computation time compared to two metaheuristic algorithms (i.e., the VNS and ACO algorithms) and the commercial solver Gurobi when considering the different investigated models.

Future research should extend the considered single-vehicle TDRPP problem into a TDRPP with multiple vehicles. Furthermore, the time-space network model developed for the TDRPP could be integrated with dynamic discretization discovery (a framework for solving routing problems based on discrete-time networks) proposed in Boland and Savelsbergh (2019) to possibly find the exact solution to the studied routing problem.

Appendix

- The total time spent $f_\pi(t_s)$ can be calculated by Algorithm 3. In Algorithm 3, the term $\{v_1^k, v_2^k, ..., v_{n_k}^k\}$ represents the node sequence corresponding to transition path p_k, while n_k represents the number of nodes included in p_k.
- The number of constraints and decision variables of the proposed TSN model for each scenario is provided in Table 11.
- A description of the VNS and ACO algorithms used in this paper can be found in Hansen et al. (2017); Halim and Ismail (2019). In order to compare with GA fairly, the maximum fitness evaluation times of the two algorithms are 144000(360*400). The neighborhood structure used by the VNS algorithm in the shaking and improvement procedure is “2-opt move”, “Insertion-1 move” and “Insertion-2 move”. The ant colony size and iteration number of the ACO algorithm are 360 and 400, which are consistent with those of the GA. The other parameters of ACO are optimized by the cross validation, and the specific parameters are shown in Table 12.

Algorithm 3 Calculate the fitness of the developed GA

```
Input: $c$: One chromosome containing $K$ service arcs. $t_s$: start time.
Output: $f_\pi(t_s)$: Its fitness value (i.e. the total time spent).
 1: $t = t_s$,
 2: for $k = 1 : K-2$ do
 3:     Define the transition path $p_k$ as $\{v_1^k, v_2^k, ..., v_{n_k}^k\}$,
 4:     for $i = 1 : n_k - 1$ do
 5:         $t = t + \eta_{v_i^k, v_{i+1}^k}(t)$,
 6:     end for
 7:     Let arc $(i, j) \triangleq a_k$,
 8:     $t = t + \tau_{i,j}(t)$,
 9: end for
10: Define the transition path $p_{K-1}$ as $\{v_1^{K-1}, v_2^{K-1}, ..., v_{n_K}^{K-1}\}$,
11: for $i = 1 : n_K - 1$ do
12:     $t = t + \eta_{v_i^{K-1}, v_{i+1}^{K-1}}(t)$,
13: end for
14: $f_\pi(t_s) = t$,
15: return fitness $f_\pi(t_s)$.
```

Table 11 The number of constraints and decision variables corresponding to the time-space network model

Scenarios	Constraints	Variables	Scenarios	Constraints	Variables
G_{1-1}	42	8000	G_{8-1}	292	907200
G_{1-2}	53	11000	G_{8-2}	468	1636800
G_{1-3}	65	14400	G_{8-3}	644	2520000
G_{2-1}	53	14000	G_{9-1}	552	907200
G_{2-2}	65	18000	G_{9-2}	888	1636800
G_{2-3}	90	27200	G_{9-3}	1224	2520000
G_{3-1}	66	21600	G_{10-1}	804	1920000
G_{3-2}	90	32000	G_{10-2}	1308	3532800
G_{3-3}	114	44000	G_{10-3}	1812	5491200
G_{4-1}	66	25200	G_{11-1}	1056	3307200
G_{4-2}	114	50000	G_{11-2}	1728	6148800
G_{4-3}	138	64800	G_{11-3}	2399	9604800
G_{5-1}	78	35000	G_{12-1}	1184	1920001
G_{5-2}	126	63800	G_{12-2}	1928	3532801
G_{5-3}	174	99000	G_{12-3}	2672	5491201
G_{6-1}	160	268800	G_{13-1}	1742	4141200
G_{6-2}	248	460800	G_{13-2}	2857	7726800
G_{6-3}	336	691200	G_{13-3}	3973	12090000
G_{7-1}	226	541200	G_{14-1}	2299	7204800
G_{7-2}	358	958800	G_{14-2}	3787	13540800
G_{7-3}	490	1462800	G_{14-3}	5275	21259200

Table 12 VNS and ACO algorithm parameters

Method	VNS		ACO	
	Parameters	Value	Parameters	Value
	Iteration	144000(360×400)	Iteration	400
		2-opt move	Num of ants	360
	Neighborhood structure	Insertion-1 move	evaporation rate	0.01
		Insertion-2 move	heuristic exponent	10
			pheromone exponent	1

Acknowledgements This research is supported by the National Natural Science Foundation of China under Grant 61703372 and the Outstanding Foreign Scientist Project in Henan Province under Grant GZS2019008.

References

Ahn BH, Shin JY (1991) Vehicle-routeing with time windows and time-varying congestion. J Opera Res Soc 42:393–400

Beasley J (1981) Adapting the savings algorithm for varying inter-customer travel times. Omega 9:658–659

Black D, Eglese R, Wøhlk S (2013) The time-dependent prize-collecting arc routing problem. Comput Oper Res 40:526–535

Boland NL, Savelsbergh MW (2019) Perspectives on integer programming for time-dependent models. Top 27:147–173

Calogiuri T, Ghiani G, Guerriero E, Mansini R (2019) A branch-and-bound algorithm for the time-dependent rural postman problem. Comput Oper Res 102:150–157

Colombi M, Corberán Á, Mansini R, Plana I, Sanchis JM (2017) The hierarchical mixed rural postman problem. Transp Sci 51:755–770

Corberán Á, Eglese R, Hasle G, Plana I, Sanchis JM (2021) Arc routing problems: a review of the past, present, and future. Networks 77:88–115

Corberán Á, Laporte G (2014) Arc routing: problems, methods, and applications. MOS-SIAM Series on (Optimization)

Corberán A, Prins C (2010) Recent results on arc routing problems: an annotated bibliography. Networks 56:50–69

Cordeau JF, Ghiani G, Guerriero E (2014) Analysis and branch-and-cut algorithm for the time-dependent travelling salesman problem. Transp Sci 48:46–58

Fernández E, Laporte G, Rodríguez-Pereira J (2018) A branch-and-cut algorithm for the multidepot rural postman problem. Transp Sci 52:353–369

Fleischmann B, Gietz M, Gnutzmann S (2004) Time-varying travel times in vehicle routing. Transp Sci 38:160–173

Gendreau M, Ghiani G, Guerriero E (2015) Time-dependent routing problems: a review. Comput Oper Res 64:189–197

Guan M (1962) Graphic programming using odd and even points. Chinese Math. 1:237–277

Gurobi (2018) Gurobi optimizer reference manual. http://www.gurobi.com

Halim AH, Ismail I (2019) Combinatorial optimization: comparison of heuristic algorithms in travelling salesman problem. Arch Comput Methods Eng 26:367–380

Hansen P, Mladenović N, Todosijević R, Hanafi S (2017) Variable neighborhood search: basics and variants. EURO J Comput Optim 5:423–454

Horn ME (2000) Efficient modeling of travel in networks with time-varying link speeds. Networks Int J 36:80–90

Huang T, Gong YJ, Kwong S, Wang H, Zhang J (2019) A niching memetic algorithm for multi-solution traveling salesman problem. IEEE Trans Evol Comput

Huang Y, Zhao L, Van Woensel T, Gross JP (2017) Time-dependent vehicle routing problem with path flexibility. Trans Res Part B Methodol 95:169–195

Ichoua S, Gendreau M, Potvin JY (2003) Vehicle dispatching with time-dependent travel times. Eur J Oper Res 144:379–396

Kaufman DE, Smith RL (1993) Fastest paths in time-dependent networks for intelligent vehicle-highway systems application. J Intell Transp Syst 1:1–11

Kramer O (2017) Genetic algorithm essentials. Springer, Berlin

Meng L, Zhou X (2014) Simultaneous train rerouting and rescheduling on an n-track network: a model reformulation with network-based cumulative flow variables. Trans Res Part B Methodol 67:208–234

Mourao MC, Pinto LS (2017) An updated annotated bibliography on arc routing problems. Networks 70:144–194

Nossack J, Golden B, Pesch E, Zhang R (2017) The windy rural postman problem with a time-dependent zigzag option. Eur J Oper Res 258:1131–1142

Orloff C (1974) A fundamental problem in vehicle routing. Networks 4:35–64

Quirion-Blais O, Langevin A, Lehuédé F, Péton O, Trépanier M (2017) Solving the large-scale min-max k-rural postman problem for snow plowing. Networks 70:195–215

Rabbouch B, Saâdaoui F, Mraihi R (2019) Efficient implementation of the genetic algorithm to solve rich vehicle routing problems. Oper Res, pp 1–29. https://doi.org/10.1007/s12351-019-00521-0

Shakibayifar M, Sheikholeslami A, Corman F, Hassannayebi E (2020) An integrated rescheduling model for minimizing train delays in the case of line blockage. Oper Res 20:59–87

Sharma D, Deb K, Kishore N (2011) Domain-specific initial population strategy for compliant mechanisms using customized genetic algorithm. Struct Multidiscip Optim 43:541–554

Sun J, Meng Y, Tan G (2015) An integer programming approach for the Chinese postman problem with time-dependent travel time. J Combin Optim 29:565–588

Sun J, Tan G, Hou G (2011a) A new integer programming formulation for the chinese postman problem with time dependent travel times. World Academy of Science, Engineering and Technology. Int J Comput Electrical Autom Control Inf Eng 5:410–414

Sun J, Tan G, Qu H (2011b) Dynamic programming algorithm for the time dependent Chinese postman problem. J Inf Comput Sci 8:833–841

Sung K, Bell MG, Seong M, Park S (2000) Shortest paths in a network with time-dependent flow speeds. Eur J Oper Res 121:32–39

Tagmouti M, Gendreau M, Potvin JY (2007) Arc routing problems with time-dependent service costs. Eur J Oper Res 181:30–39

Tagmouti M, Gendreau M, Potvin JY (2010) A variable neighborhood descent heuristic for arc routing problems with time-dependent service costs. Comput Ind Eng 59:954–963

Tagmouti M, Gendreau M, Potvin JY (2011) A dynamic capacitated arc routing problem with time-dependent service costs. Transp Res Part C Emerg Technol 19:20–28

Tan G, Sun J (2011) An integer programming approach for the rural postman problem with time dependent travel times. In: International Computing and Combinatorics Conference. Springer, pp 414–431

Tan G, Sun J, Hou G (2013) The time-dependent rural postman problem: polyhedral results. Optim Methods Softw 28:855–870

Vincent FY, Lin SW (2015) Iterated greedy heuristic for the time-dependent prize-collecting arc routing problem. Comput Ind Eng 90:54–66

Wang HF, Wen YP (2002) Time-constrained Chinese postman problems. Comput Math Appl 44:375–387

Yang L, Zhou X (2014) Constraint reformulation and a lagrangian relaxation-based solution algorithm for a least expected time path problem. Transp Res Part B Methodol 59:22–44

Yu B (2020) Data set of time-dependent rpp. https://github.com/momoyby/TDRPP

Yu G, Yang Y (2019) Dynamic routing with real-time traffic information. Oper Res 19:1033–1058

Zanotti R, Mansini R, Ghiani G, Guerriero E (2019) A kernel search approach for the time-dependent rural postman problem. In: WARP3 Proceedings, Pizzo, Italy