Data-Driven Flexible Vehicle Scheduling and Route Optimization

Yongxuan Lai, Fan Yang, Ge Meng, and Wei Lu

***Abstract*—The flexible transit service reflects a trend of demand on the flexibility and convenience in urban public transport systems, within which the vehicle scheduling and passenger insertion are two challenging issues. Especially, finding the optimal solution for a flexible transit system can be viewed as an extension of the traveling salesman problem which is NP-complete. Yet most of the existing research mainly focuses on one aspect, i.e. route planning, stop selection or vehicle scheduling, where a combined integration and optimization of the whole system is largely neglected. In this paper, we propose a data-driven flexible transit system that integrates the origin-destination insertion algorithm and the *milp*-based (mixed-integer linear programming) scheduling scheme. Specifically, stops are mined from the historical datasets and some stops act as *backbone* stops that should be visited by the vehicles; and a heuristic backbone-based origin-destination insertion algorithm is proposed to schedule the routing path of vehicles, where the time loss caused by the optimal insertion positions is calculated for the vehicles to decide whether to accept the requests or not when constructing a path for the flexible routes. Moreover, a vehicle scheduling model based on *milp* is proposed to minimise the gap between the passenger flow and available seats. The proposed flexible transit systems are simulated in real-world taxi datasets, and experimental results show that the proposed flexible transit system can effectively increase the delivery ratio and decrease the passengers' waiting time compared with existing methods.**

***Index Terms*—Flexible transit system, data-driven route scheduling, heuristic origin-destination insertion, mixed-integer linear programming.**

I. Introduction

With the development of mobile-oriented wireless networks and technologies, in recent years there has been increasing research on improving the flexibility and availability of urban public transportation systems [1], [2], [3], [4], [5]. According to the service mode, urban public transportation systems can be roughly divided into the $fixed$ and $flexible$ transit systems [6], [7]. The fixed transit services have fixed routes and fixed service schedules, which are more common and have already been operated for more than a century. The flexible transit service, however, is a more recent innovation of service driven by the development of transit technologies, mobile communications, and big data technologies. It reflects a trend of flexibility and convenience in urban public transport systems. Users could subscribe and order a transit request, and the transportation system would fulfil this request by dynamically scheduling the trips and vehicles through wireless communications and back-end data processing [5], [8], [9]. This brings about the concept of Flexible Bus and Flexible Vehicle scheduling.

Compared with the fixed transit system that contains fixed vehicles and routing lines, the flexible vehicle has several advantages: 1) the transit lines are flexible according to the distribution of requests and passengers. The lines and stops change accordingly with the demands, and vehicles can avoid stops or areas that are without boarding or alighting activities; 2) the number of vehicles serving the transit routes and the time interval between sequential vehicles are dynamically adjusted according to time and requests; 3) the stops are flexible and could be optimized to be located to places that are most convenient for the passengers. In this way, passengers could walk shorter distances to reach the stops to be picked up, and their waiting time could also be reduced. These two factors are critical for the comfort of passengers [10]. In this way, the overall efficiency of the vehicle company and the quality of service could both be improved. Flex bus or flex vehicles [6], [11], [12], [13], [14] solve the imbalance of spatial passenger flow in the suburban areas or other lower populated places. But we found that such flexible mode of transportations is also needed in urban scenarios, as the passenger flow is largely spatially and temporally imbalanced.

However, the vehicle scheduling and passenger insertion are two challenging issues for the flexible transit system. The problem of finding optimal solution for a flexible transit system, also known as the Dial-A-Ride Problem (DARP) [15], [16], can be viewed as an extension of the traveling salesman problem. It is NP-complete, which involves calculating an optimum journey for visiting a number of predetermined nodes on a network. Existing research on flexible public transport

Manuscript received 5 January 2021; revised 29 August 2021, 17 November 2021, 11 February 2022, and 9 May 2022; accepted 24 August 2022. This work was supported in part by the Natural Science Foundation of Guandong under Grant 2021A1515011578; in part by the Longyan Institute of Industry and Education Integration, Xiamen University, under Grant 20210302; in part by the Natural Science Foundation of China under Grant 61872154 and Grant 61972403; in part by the Natural Science Foundation of Fujian under Grant 2018J01097; and in part by the Shenzhen Basic Research Program under Grant JCYJ20190809161603551. The Associate Editor for this article was H. Khayyam. *(Corresponding authors: Yongxuan Lai; Fan Yang.)*

Yongxuan Lai and Ge Meng are with the School of Informatics and the Shenzhen Research Institute, Xiamen University, Xiamen 361005, China, and also with the School of Mathematics and Information Engineering, Lonyan University, Longyan 364000, China (e-mail: laiyx@xmu.edu.cn; 573810157@qq.com).

Fan Yang is with the Department of Automation, Shenzhen Research Institute, Xiamen 361005, China, and also with the Xiamen Key Laboratory of Big Data Intelligent Analysis and Decision-Marking, Xiamen University, Xiamen 361005, China (e-mail: yang@xmu.edu.cn).

Wei Lu is with the Key Laboratory of Data Engineering and Knowledge Engineering, Ministry of Education, and the Department of Computer Science, Renmin University of China, Beijing 100872, China (e-mail: lu-wei@ruc.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TITS.2022.3204644, provided by the authors.

Digital Object Identifier 10.1109/TITS.2022.3204644

provides methods on the route planning [17], [18], stop selection [19] and vehicle scheduling [20], [21]. Yet they only focus on one aspect of the problem, which lacks a combined optimization of the whole problem. Also, in most of the existing approaches, the datasets of vehicles and passengers are not well used and integrated into the flexible transit system, which deters the overall optimization of the flexible transit system.

In this paper, we propose a data-driven flexible transit system that integrates the origin-destination insertion algorithm and the *milp*-based (mixed-integer linear programming) scheduling scheme. We define factors related to the service area, slack ratio of traveling, and passengers' pick up time window, and take them into account for the algorithm design. The proposed demand-responsive traveling scheme assumes a time-dependent road network model for the estimation of the operating cost, and models the vehicle scheduling as a Mixed-Integer Linear Programming problem. The major contributions of this paper are as follows:

- We model the flexible transit system and define several concepts for the system based on trajectory datasets. The historical trajectory data are clustered to identify and generate a series of popular candidate stops called *backbone* stops. The service area, flex route, and the slack ratio and etc. are also formalized and defined.
- We propose a heuristic backbone-based origin-destination insertion algorithm to schedule the routing path of vehicles. The algorithm integrates the historical patterns of the request datasets to efficiently construct a path for the flex route, and calculates the time loss caused by the optimal insertion position of origin and destination of new request to decide whether to receive the request.
- We propose a vehicle scheduling model based on *milp* to minimise the gap between the passenger flow and available seats. An objective function that represents the resource utilization in *milp* is adopted to build the vehicle scheduling model.
- We combine the origin-destination insertion algorithm and the *milp*-based scheduling into an integrated scheme, and conduct extensive simulations to verify the effectiveness of the scheme. Experimental results show that the proposed flexible transit system can effectively increase the delivery ratio and decrease the passengers' waiting time compared with existing systems.

The rest of the paper is structured as follows: section II describes the related work; section III introduces some preliminaries and defines the model; section IV presents the detailed modelling of the flexible vehicle transit system, which includes flexible route scheduling, optimal path of flex route calculation, and origin-destination insertion; section V describes the environmental setup and analyzes the simulation results, and finally section VI concludes the paper.

II. Related Work

In this section we review four categories of related works, and position our work in the research community.

A. Demand-Responsive Transit Service

Demand-responsive transit service is an alternative travel method to personal vehicles, carpool/vanpool and regular transit service. It is comprised of a number of customer requests that need to be served door-to-door or curb-to-curb by a set of vehicles [1], [2].

One important issue in demand-responsive transit service is to devise a real-time matching algorithm that determines the best vehicle (taxi, cab, vehicle) to satisfy incoming service requests. Dijoseph *et al.* [22] proposed a mathematical model to optimize the social and fiscal sustainable operation of a feeder bus system considering realistic network and heterogeneous demand. Ma *et al.* [23] proposed a taxi searching algorithm using a spatio-temporal index to quickly retrieve candidate taxis that are likely to satisfy a user request. The algorithm checks each candidate vehicle and inserts the query's trip into the schedule of the taxi that satisfies the query with minimum additional incurred travel distance. Based on [23], Ma *et al.* [8] reported a real-time taxi-sharing system based on the mobile-cloud architecture. Drivers and passengers exchange services and demands using an application installed on their smart phones, and the taxi that minimizes the increased travel distance of the ride request would be selected to pick up the new passenger. Gomes *et al.* [24] designed a heuristic approach that involves the construction of a feasible route through a greedy randomized procedure, followed by a local search phase, and a Decision Support System was also embedded in the simulation [25]. Zhu *et al.* [26] proposed a path planning strategy that focuses on a limited potential search area for each vehicle by filtering out requests that violate passenger service quality level, and studied the joint transportation and charging scheduling for public vehicle systems to balance the transportation and charging demands, ensuring the long-term operation [27].

B. Flexible Transit and Customized Vehicle

Flexible Transit service is firstly adopted in low-demand areas (e.g. the suburbs of a city and industrial parks). The demand for public transport is relatively low and distributed. To cut down the operation cost and to increase the degree of passenger satisfaction, flexible transit service adds flexibility to transit routes and schedules. The vehicle routes, vehicle schedule, vehicle stops, or vehicle types could be changed by the operator; and the overall system cost could be significantly reduced by effectively integrating conventional and flexible services in comparison with conventional or flexible services [13]. Recently, a fully flexible transit system such as the DIDI mini-vehicle system has been brought into market in Beijing and Chengdu, China. The DIDI mini-vehicle is a seven-seat car without a fixed route, and it would pick up the passengers who send requests to the system in realtime. One main drawback of the mini-vehicle is that it cannot pick up many passengers or provide transit over long ranges. Martínez *et al.* [6] presented the formulation of a new optimization problem designated as the express mini-vehicle problem. It clusters small groups of clients with compatible boarding/exiting points in time and space for a new type of urban mobility service.

Recently, the concept of customized vehicle is introduced and operated in many large and medium-sized cities. The operational activity of a customized vehicle is planned by aggregating space–time demand and similar passenger travel demands. The first customized vehicle was implemented in Beijing in October 2013, and 287 customized bus lines have been implemented in Beijing since December 2015 [7]. Ma *et al.* [7] proposed an improved immune genetic algorithm to solve the model with regard to the problems associated with the operation of customized vehicles, such as stop selection, line planning and timetables. However, they only used some demand data or passenger flow data for the simulation. The impact of demand data on the designing is absent from these discussions. Mahrsi *et al.* [28] proposed two approaches to cluster smart card data to extract mobility patterns in a public transportation system. Ma *et al.* [14] proposed a data mining method to identify travel patterns for individual transit riders using a large smart card dataset. Nourbakhsh *et al.* [12] proposed a structured flexible-route transit system where the bus tubes form a "grand" structure that includes a grid tube network that provides double coverage to passengers in the central part of the city, and a hub-and-spoke tube network that provides a single coverage in the peripheral part. Boyer *et al.* [29] proposed a method to deal with the Flexible Vehicle and Crew Scheduling Problem. It aimed for high quality and fast to compute solutions for resources (vehicles and drivers) assignment to cover timetables generated at the tactical level, and adopted a mixed-integer linear programming model and a variable neighborhood search for this problem. Repoux *et al.* [30] proposed a type of semi-autonomous transportation system that consists of convoys composed of one human-driven lead vehicle guiding several autonomous small capacity trailers. The trailers can detach from a convoy and travel autonomously in a protected environment before attaching later to another convoy.

C. Vehicle Routing Problem With Pickup and Delivery

The flexible vehicle network design can be formulated as a vehicle routing problem with pickup and delivery [31]. The objective could be either minimizing the operation cost, maximizing satisfied demand, or maximizing the quality of service [16]. These objectives are optimized separately or simultaneously. Given the objectives and constraints, the vehicle routing problem is usually formulated as the mixed-integer programming model with routing and scheduling variables [32]. For instance, Cordeau *et al.* [31] introduced a mixed-integer programming formulation of the problem and used a new valid inequalities for the dial-a-ride problem. The VRPPD is NP-hard since it is the generalization of Vehicle Routing Problem [33]. So for instances with a large number of requests, heuristics or metaheuristics approaches are adopted to deal with the large-scale real-life applications. Dondo *et al.* [17] proposed a two-phase heuristics algorithm to deal with the instance with a large number of passenger requests, where Phase I aims to identify a set of cost-effective feasible clusters while Phase II assigns clusters to vehicles and sequences them on each tour by using the cluster-based *milp* formulation. Zhu *et al.* [18] proposed a heuristic precedence constrained origin-destination insertion algorithm for the public vehicle system to minimize vehicles' total travel distance with service guarantee such as low detour ratio. Sun *et al.* [34] developed a mixed integer non-linear model for optimizing multi-terminal customized bus service in an urban setting. According to the estimated spatio-temporal passenger demand, the objective total cost, consisting of supplier's and users' costs, is minimized subject to capacity and time window constraints. Wang *et al.* [35] studied the last-mile problem that concerns the provision of travel services from the nearest public transportation node to a passenger's home or other destination. An exact mixed-integer programming (MIP) model and feasible heuristic approaches are developed and implemented to evaluate the system's performance.

D. Vehicle Scheduling

Some research has been done on setting the departure interval when scheduling the vehicles. Lee *et al.* [20] mainly analyzed the relationship between departure interval and passenger flow demand, and proposed an optimization method of vehicle scheduling based on the delay of departure interval, so as to reduce the probability of vehicle overload and other situations. The problem of the best vehicle scheduling mode is also solved by establishing models. Zhu *et al.* [36] optimized the sum of the operating cost of vehicle company in a whole day and the cost of passengers waiting for the car and the transfer. And under the premise of reasonable assumptions, they established an optimization model of the vehicle departure interval. Zhang *et al.* introduced the comfort of passengers in the vehicle scheduling optimization model in [37], which is based on the full consideration of the vehicle company operating costs and the passengers' waiting costs. Tan *et al.* [38] made a multi-object genetic algorithm optimizing model, including the passengers and vehicle companies, and used the genetic algorithm's global optimization search to deal with the operate vehicles on one vehicle line, obtaining the optimal solution of vehicle departure interval. Ma [39] put forward a hybrid departure scheduling model for different vehicle models of the same vehicle line, which could be solved by a genetic algorithm. Tephan *et al.* [40] mainly aimed at optimizing the operation cost of public transport companies, and established the timetable model of public transport based on the factors such as passenger waiting cost and vehicle empty seat punishment, and solves the model to obtain the optimal vehicle schedule. Hoo and Ong *et al.* [21] mainly considered the impact of urban traffic congestion on vehicle scheduling. An optimization model based on vehicles and other vehicles was proposed. The effectiveness of the model for reducing traffic congestion is verified by simulation experiments.

E. Positioning of Our Work

Demand-responsive transit service could be abstracted as a member of the general class of the Dial-a-Ride Problem [15], [16], which focuses on scenarios of planning schedules for vehicles, subject to the time constraints on pickup and delivery events.

Different from the above mentioned research [6], [11], [12], [13], [14], [29], the proposed scheme adopts a data-driven approach for the flexible scheduling of vehicles. The differences lie in three aspects: 1) in most of the existing research the terminal stations are predefined and given as external parameters, while we adopt a data-driven approach to deduce these parameters. For instance, the stations and stops are mined from real-world datasets and could be adjusted according to real-time requests. Also, the mobility or travel patterns could be beneficial to understand the variability of urban travel behavior and facilitating network design. But analyses of existing works are all based on dataset from fixed route transits. In this research we conduct the clustering algorithms on the real-world origin-destination datasets of taxis, which are more likely to be replaced by the flexible vehicles and mini-vehicle systems. 2) at existing research the stops are either fixed or decided by ad hoc requests, both of which lack a degree of flexibility. In our approach, all the stops are mined from the datasets and are defined as candidate pick-up locations. Vehicles would visit the backbone stops to collect the patterned or predicted requests, and would only visit part of the candidate stops to collect the predicted or ad hoc requests. In this way, both the static patterned requests and dynamic ad hoc requests could be handled efficiently and effectively; 3) in previous research, the route scheduling and path routing were investigated separately. Our framework integrates the flexible route scheduling and optimal path routing, where a data-driven approach is adopted based on the real-world OD datasets. And existing schemes mainly considered how to meet the requests from passengers, yet the utilization of seats was largely neglected. In this paper, the proposed scheduling scheme aims to minimise the gap between the demand and supply of seats, and the number of allocated vehicles could be customized and dynamically adapted according to the demand; 4) travel time variability would significantly affect the routing and scheduling of flexible public transport. Our approach is able to deal with the impact of variability of travel time as well as the distribution of passenger and vehicle arrivals. Vehicles travel with different speeds at different time spans when passing different road segments, and our data-driven approach could capture the pattern and evolve with the requests and travelling time by adjusting the locations of stops and the scheduling of vehicles.

Also, while most of the previous flexible transit systems are adopted in the suburban area or other lower populated places, our study shows that flexible mode of transportation is feasible at urban scenarios and is able to handle the spatially and temporally imbalanced passenger flow.

III. Model Description

In flex-route transit service, vehicles travel among stations while responding to demands. A vehicle, denoted by c, follows the following two basic rules to provide the public transit service: 1) while traveling along route $u \to v$, c receives on-demand requests within its service area. A request might be accepted or rejected by the vehicle; 2) if the vehicle accepts a request, it travels to the place and picks up the rider; else, it sends a reject message to the rider. In this section we introduce some concepts and definitions of this model.

A. Stations and Stops

Vehicles pick up and drop off demand responsive riders at $stops$, which are temporary locations between station u and v. We denote the set of all possible stops as E.

Both stations and stops could be predefined, or extracted from historical trajectories. In this research we adopt a data-driven strategy which identifies stations and stops through clustering the OD datasets.

B. Flex Route

Symbol $r(u, v, t)$ denotes a route from station u to station v starting at the scheduled departure time t. When a vehicle is assigned to route $r(u, v, t)$, it travels along path from u to v. We denote the path from u to v through the shortest path by $u \to v$, and its traveling time is denoted by $tt(r)$. Yet a vehicle along the route would respond to riders' requests, so it would go to pick up the riders. The actual travelling path is denoted by $u \hookrightarrow v$, and its actual running time is denoted by $at(r)$. Also, a route has a scheduled running time $rt(r)$, which means the vehicle along route r should arrive v before the scheduled time $rt(r)$. The following formula holds:

$$tt(r) \leq at(r) \leq rt(r) \tag{1}$$

Here we assume the times $tt(r), at(r), rt(r)$ take a predefined time slot as the unit. A time slot is denoted by U and it could be 5 or 10 minutes. The actual running time would increase as new requests are inserted into the route. So a request would be rejected when, if it is accepted, the actual running time is larger than the scheduled time.

C. Slack Time

Slack time is denoted by $st(r)$ and defined as follows:

$$st(r) = rt(r) - tt(r) \tag{2}$$

It is the extra time to serve on-demand requests within the service area of the route. Also, the slack ratio is denoted by α:

$$\alpha = \frac{st(r)}{tt(r)} \tag{3}$$

In this study we assume $\alpha > 0$ is a predefined parameter for all the routes in the flex transit system. The running time could be calculated by the following formula:

$$rt(r) = \lceil tt(r) * (1 + \alpha) \rceil \tag{4}$$

where $\lceil x \rceil$ denotes the ceiling of the x in unit time slot.

D. Request

A request is denoted by $req(t, o, d, w)$, where t is the time when the request is submitted, o is the pickup location, d is the dropoff location, and w is the constraint time window for the pickup. A request might either be accepted or rejected by the vehicle.

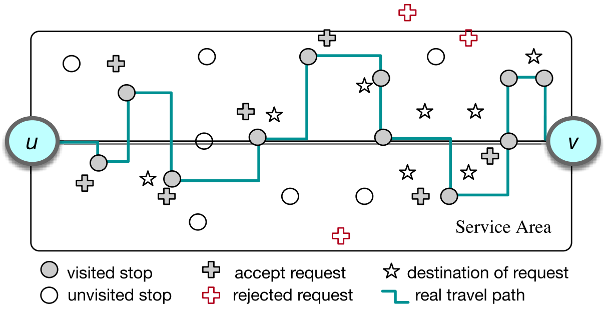


Fig. 1. Illustration of a flex route $u \rightarrow v$. The vehicle travels to pick up demand-responsive riders, and drop them to station or stops that are nearest to their destinations.

E. Service Area

Service area is usually represented by an extended rectangle area along path $u \rightarrow v$. The width of service area defines how far away from the standard route a vehicle may deviate to pick up or drop off passengers. Fig. 4 illustrates a flex route $u \rightarrow v$ and its service area.

F. Operation Policies

The vehicles are not required to follow a specific route and could have a different route from time to time. The only constraint in the service is that all flex routes are required to start and end at stations, and depart and arrive within their scheduled running time.

We assume riders get on and off vehicles at the departure and destination stations. They also issue demand-responsive requests so that they can get on and off vehicles at some predetermined locations. We call these locations the *dynamic stops*, which are extracted from the trajectories logs. Request $req(t, o, d, w)$ would be transformed to $req(t, o', d', w)$, where o', d' are the nearest dynamic stops to locations o, d. The flex route system would calculate whether a request is compatible with a route. A request $req(t, o, d, w)$ is *compatible* with a route r if it meets the following conditions:

$$t + w(l_c, o') \in r.w; \tag{5}$$

$$cost(path(r)_{o',d'}) \leq rt(r) \tag{6}$$

where l_c is the current location of vehicle, $w(l_c, o')$ is the time cost of traveling from l_c to o', $path(r)_{o',d'}$ is the path after inserting o' and d' on route r. Condition (5) means the vehicle should travel to o' to pick up the rider on its constraint time window w, condition (6) means the total traveling time of the path after inserting o', d' should be within the running time of route r, i.e. $rt(r)$.

The flex route system would accept a request if the request is compatible with the route. Then the system would send an "accept" message to the rider and guide the rider to walk to o' for the pickup. And the vehicle would drop off the rider at d', where the rider could walk to his/her destination. If the request is not compatible with the route, the system would send a "reject" message to the rider immediately.

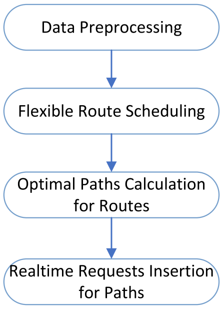

Data Preprocessing
Flexible Route Scheduling
Optimal Paths Calculation for Routes
Realtime Requests Insertion for Paths

Fig. 2. Main steps of the data-driven flexible vehicle system.

IV. Flexible Vehicle Modelling

In this section, we present the detailed description of the flexible vehicle system which consists of 4 steps: the data preprocessing, the flexible route scheduling, the optimal path calculation, and the request insertion (Fig. 2). The data preprocessing step is described in the next section, while the metrics that evaluate the designed transit service are first introduced.

A. Performance Metrics

We aim to study the feasibility of a public vehicular system that provides another public transportation method other than the vehicle or responsive taxies. So in this study we use the sharing amplifier (sa), rider delivery ratio (dr), and the average walking distance (wd) as three main metrics for the performance measures.

The sharing amplifier sa is defined as follows:

$$sa = \frac{\sum_{c \in F} tl(c)}{\sum_{r \in D} l(r.o, r.d)} \tag{7}$$

where D is the set of successfully delivered requests, $l(r.o, r.d)$ is the distance of the shortest path from the origin $r.o$ to the destination $r.d$, F is the set of vehicles, and $tl(c)$ is the total traveling distance of vehicle c. sa actually represents the average number of passengers sharing the vehicle in the whole trip. The rider delivery ratio (dr) is calculated as:

$$dr = \frac{|D|}{|R|} \tag{8}$$

where $|D|$ is the number of successful deliveries, $|R|$ is the number of all requests. The average walking distance wd is calculated as:

$$wd = \frac{\sum_{r \in D} l(r.o, r.\widehat{o}) + l(r.d, r.\widehat{d})}{|D|} \tag{9}$$

where $r.\widehat{o}$, $r.\widehat{d}$ are the real pickup and dropoff locations for request r.

Other factors such as the average waiting time and detour ratio are also important indicators of the QoS (quality of service) of passengers. We would discuss them at the experimental analysis.

TABLE I

MAIN NOTATIONS IN THE PAPER

$r(u, v, t)$	a route from station u to station v starting at the scheduled departure time t
$tt(r)$	(minute), travelling time of $r(u, v, t)$
$at(r)$	(minute), actual running time of $r(u, v, t)$
$rt(r)$	scheduled time of $r(u, v, t)$
$st(r)$	(minute), slack time of $r(u, v, t)$
α	slack ratio of a route $r(u, v, t)$
K	number of subareas
U	length of time slot unit
C	capacity of seating and standing riders
F^u_{ij}	number of demand flow that departs from station u at interval i and arrives at station v at interval j
F^v_{ij}	number of demand flow that departs from station v at interval i and arrives at station u at interval j
x^u_{ij}	number of vehicles that are scheduled to depart from u at interval i and arrives v at interval j
x^v_{ij}	number of vehicles that are scheduled to depart from v at interval i and arrives u at interval j
cap	capacity of the vehicle
κ	≥ 1, empirical factor as there are get-ons and get-offs during the trip
N	number of vehicles of the fleet
cap^u	capacity of station u
cap^v	capacity of station v
s^u_i	number of vehicles at station u at interval i
s^v_i	number of vehicles at station v at interval i
A_i	the i-th subarea
G_i	set of grid covered by A_i
θ	threshold of overlap area between a grid g and a subrea A_i
$fg(z, t)$	flow of a grid
$fw(A_i, t)$	weight of flow in subarea A_i that begins at time t
f_t	flow from u to v at time t
$N_0(s)$	number of currently received pickup and drop off requests at stop s
$N_1(s)$	flow weight of grid which stop s belong to
β	a balance factor for $N_0(s)$ and $N_1(s)$
BS	set of backbone stations
bs_i	the i-th backbone
w	constraint time window for the pickup

B. Flexible Route Scheduling

Given a flex route from station u to v, the operation time domain is split into n slots. For example, if there are 16 hours of operation time for the route, and each slot is two minutes, then there are $n = 480$ (16*60/2) time slots. x^u_{ij} denotes the number of vehicles that are scheduled to depart from u at interval i and arrive v at interval j. Similarly, x^v_{ij} denotes the number of vehicles that are scheduled to depart from v at interval i and arrive u at interval j. Fig. 3 illustrates the slots and trip scheduling problem. The scheduling is modelled as a Mixed-Integer Linear Programming (MILP) problem based on the historical demand of flows:

$$\text{minimize}: \sum_{i}^{n}\sum_{j}^{n} |F^u_{ij} - C * x^u_{ij}| + |F^v_{ij} - C * x^v_{ij}| \tag{10}$$

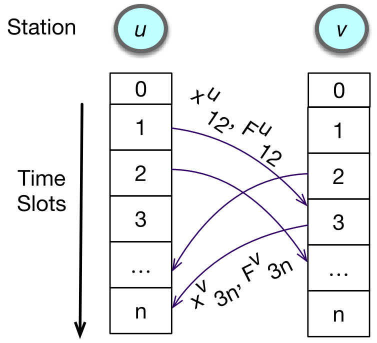


Fig. 3. Illustration of timeslots and trips corresponding to the flexible route scheduling problem.

$$\text{subject to:}\quad s^u_{i+1} = s^u_i - x^u_{ij} + \sum_{z=1}^{n} x^v_{zi}, \quad i = 1..n-1 \tag{11}$$

$$s^v_{i+1} = s^v_i - x^v_{ij} + \sum_{z=1}^{n} x^u_{zi}, \quad i = 1..n-1 \tag{12}$$

$$s^u_1 + s^v_1 \leq N \tag{13}$$

$$0 \leq s^u_i \leq cap^u, \quad i = 1..n \tag{14}$$

$$0 \leq s^v_i \leq cap^v, \quad i = 1..n \tag{15}$$

$$x^u_{ij}, x^v_{ij} \geq 0, \quad i = 1..n, \ j = 1..n \tag{16}$$

The objective (10) is to minimise the gap between the demand and supply of vehicle seats. F^u_{ij} is the number of demand flow that departs from station u at interval i and arrives at station v at interval j; F^v_{ij} is the number of demand flow that departs from station v at interval i and arrives at station u at interval j. F^u_{ij} and F^v_{ij} are given as constant variables for the model and we will discuss their calculation in the next section. C is the supply of seats in a single vehicle, which could be empirically calculated by $cap * \kappa$, where cap denotes the capacity of the vehicle and $\kappa \geq 1$ denotes the empirical factor as there are get-ons and get-offs during the trip.

s^u_i denotes the number of vehicles at station u at interval i. Constraints (11) and (12) imply the change of vehicles at u and v at interval i by subtracting the departed vehicles and adding the arrived vehicles. Constraint (13) ensures at the very beginning vehicles at station u and v are within the range of N, which is the number of vehicles of the fleet. Constraints (14) and (15) imply the number of vehicles at station u and v should be greater than zero and smaller than the capacity of the stations, i.e. cap^u and cap^v. Constraint (16) defines the value of the decision variables x^u_{ij} and x^v_{ij}, which should be zero or positive integers.

The flow of trips is a key factor when scheduling the routes, which are calculated based on the OD dataset. As mentioned previously, the OD dataset contains origin and destination GPS points, and the points are indexed by a set of grids on the map. We further split the OD pairs by time slots, so the set of OD pairs are stored and indexed according to spacial and temporal dimensions. We use a table, denoted by $\Gamma(z)$, for the storage,

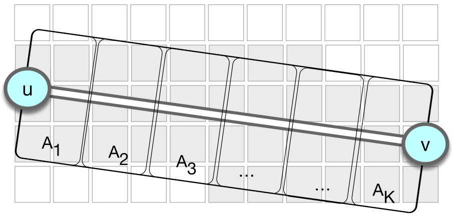


Fig. 4. Illustration of service area and subareas. The traffic flow is calculated based on grids and OD points in the service area.

and each tuple is in the form $< t, g_o, g_d, num >$, where z is the grid, t is the time slot, g_o, g_d denote the grid of the origin and destination respectively, and num is the number of the corresponding OD pairs.

Supposed the route under consideration is $r(u, v, t)$, which is from station u to v and departs at time t. Then the trip flow F_t is then calculated as follows:

1) Split the service area into subareas. K denotes the number of subareas and is calculated as:

$$K = \lceil \frac{tt(r) * (1 + \alpha)}{U} \rceil \tag{17}$$

where $tt(r)$ is the traveling time of r and α is the slack ratio, U is the length of time slot unit. So the service area is equally split into K rectangle subareas $\{A_1, A_2, .., A_K\}$, which is illustrated in Fig. 4.

2) For each subarea A_i, get the set of its covered grids G_i. A grid g is covered by A_i if the following condition holds:

$$\frac{ar(A_i \cap g)}{ar(g)} \geq \theta \tag{18}$$

where $ar(X)$ denotes the area of polygon X, $\theta \in [0, 1]$ is the a predefined threshold.

3) For each grid z within subarea A_i, calculate its trip flow that begins at time t. The flow of a grid is defined as $fg(z, t)$:

$$\begin{aligned} fg(z,t) &= \sum_{x \in \Gamma(z)} x.num, \quad x.t \in slot(t,i),\ x.g_d \in G \\ slot(t,i) &= [t + (i-1) * U,\ t + i * U), \\ & \quad i = 1, 2, \ldots, K \end{aligned} \tag{19}$$

where x is a tuple in table $\Gamma(z)$, $G = G_1 \cup G_2 \ldots \cup G_K$ is the set of all the grids in the service area of r', $slot(t, i)$ is the i^{th} time slot that begins at t and has an interval U. Then the weight of flow in subarea A_i that begins at time t is defined as:

$$fw(A_i, t) = \sum_{z \in G_i} fg(z, t) \tag{20}$$

4) Calculate the demand flow from u to v at time t, which is defined as follows:

$$f_t = \sum_{i=1}^{K} fw(A_i, t) \tag{21}$$

And f_t is mapped to F^u_{ij} as follows:

$$F^u_{ij} = \begin{cases} f_t, & i = to\_slot(t) \ and \ j = to\_slot(t + rt(r)) \\ 0, & i \neq to\_slot(t) \ or \ j \neq to\_slot(t + rt(r)) \end{cases} \tag{22}$$

where $to\_slot(t)$ is a function that maps time t to the index of time slot, $rt(r)$ is the running time of vehicle that travels along route $r(u, v, t)$. Similarly, flow F^v_{ij} is calculated based on route $r(v, u, t)$ that departs v for u at time t.

C. Optimal Path of Flex Route

When a route $r(u, v, t)$ is scheduled, information about its departure time and destination would be notified to potential riders. A rider might either go to the backbone stops to get on the vehicle, or just send a request $req(t, o, d, w)$ trying to be picked up, where the request might be accepted or rejected by the flex vehicle system.

We adopt a data-driven approach in this study. The pattern of OD (origin-destination) pairs is mined to identify dynamic stops. With a large number of origins and destinations of the travel demands, we could cluster these points collectively to represent potentially meaningful places. These places are the potential locations for the stops.

The shared nearest neighbors (SNN) is adopted as the basis of distance measure between two GPS points. Given two points A and B, the distance is defined as:

$$dist(x, y) = 1 - \frac{w(N_k(x) \cap N_k(y))}{w(N_k(x) \cup N_k(y))} \tag{23}$$

where $N_k(x)$ is the set of k nearest neighbours of x, $w(Q)$ is the total weight of points in set Q. This distance meets several requirements for the spatial clustering of origin/destination points: 1) a cluster would meet a minimum size constraint k, and each cluster is spatially contiguous; 2) it preserves the data resolution by constructing as many clusters as possible; 3) it identifies clusters of different point densities and different shapes. So the summary statistics (e.g. net flow ratio) for each cluster are meaningful and usually stable.

$$\text{maximize}: \ \sum_{i \in V} x^+_i \tag{24}$$

$$x^+_i \in [0, num^+(i)] \tag{25}$$

$$x^-_i \in [0, num^-(i)] \tag{26}$$

$$f^+_i + x^+_i = f^+_j \tag{27}$$

$$f^-_i + x^-_i = f^-_j \tag{28}$$

$$\sum_{(u,j) \in E} a_{i,j} = 1, \quad j \in V - v \tag{29}$$

$$\sum_{(i,v) \in E} a_{i,j} = 1, \quad i \in V - u \tag{30}$$

$$\sum_{(i,j) \in E} a_{i,j} \leq 1, \quad i \in V - v \tag{31}$$

$$\sum_{(i,j) \in E} a_{i,j} \leq 1, \quad j \in V - u \tag{32}$$

$$b_{i,j} \geq a_{i,j}, \quad i, j \in V \tag{33}$$

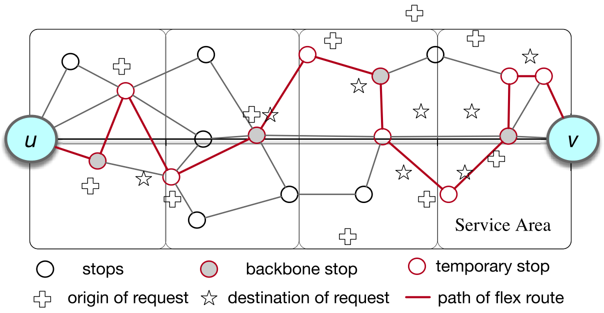


Fig. 5. Illustration of a route graph and the actual path of flex route. The path consists of backbone stops, which are selected based on historical OD data, and the ad-hoc stops, which are determined in real time.

$$b_{i,j} + b_{j,i} \leq 1, \quad i, j \in V \tag{34}$$

$$b_{i,j} + b_{j,j'} + b_{j',j} \leq 2, \quad i, j \in V \tag{35}$$

$$e_0 + f_i^+ - f_i^+ + x_i^+ - x_i^- \leq c_p, \quad i \in V \tag{36}$$

$$\sum_{(i,j)\in E} a_{i,j} * t(i,j) < rt(r) \tag{37}$$

$$a_{i,j}, b_{i,j} \in \{0, 1\} \tag{38}$$

$$f_i^+, f_i^- \geq 0 \tag{39}$$

Given a route $r(u, v, t)$, and a set of stops Q within the service area W, we define finding the optimal path of this route as the OPFR problem (Optimal Path of Flex Route). First we model the stops in the service area as vertices in a directed graph $G_r(V, E)$, which is also called the *route graph*. V is defined as $\{u, v\} \cup Q$, an edge (i, j) is added to E if the distance $l(i, j)$ is less than a threshold, where $i, j \in V$. Then finding a path in the service area can be modelled as a Mixed-Integer Linear Programming (MILP) problem.

Table II denotes the meanings of the symbols. The goal (24) is to maximise the number of served requests. Constraint (25) and (26) ensure that the vehicle selectively picks up or drops off riders that belong to that stop. Constraint (27) and (28) imply the total number of picked or dropped riders when vehicle traverses the edge (i, j). Constraint (29) and (30) mean the path should start at u and end at v. Constraint (31) means any location from $V - u$ has one successor, and (32) means any location from $V - v$ has one precursor. Constraint (33) implies the relationship between $a_{i,j}$ and $b_{i,j}$, which could be inferred from their definition. Constraint (34) implies that (i, j) and (j, i) could not both on the path, constraint (35) implies no circles on the path. Constraint (36) ensures the number of riders is smaller than the vehicle capacity at any stop. Constraint (37) ensures that the cost of traveling time is less than the running time. Finally, constraints (38) and (39) define the nature of the variables.

D. Origin-Destination Insertion Based on Backbone Stops

The problem of finding optimal solutions for a flexible transit system is a *dial-a-ride* problem [15], [16] and is NP-complete. When real-time requests are received by the flex route system, the set of R changes accordingly, and the running time varies with the time. So in real situations the problem has larger complexity with dynamic finite capacity and with more constraints (e.g., time). In this section we present a heuristic algorithm that integrates the historical OD patterns to construct a path for the flex route.

TABLE II

Notations in the OPFR Problem

x_i^+	number of requests picked up at i	e_0	number of on-board riders at u
x_i^-	number of requests dropped off at i	c_p	capacity of vehicle
f_i^+	number of picked up requests at i (i is not included)	$num^+(i)$	number of desired pickups at i
f_i^-	number of requests dropped off at i (i is not included)	$num^-(i)$	number of desired drop offs at i
$a_{i,j}$	is 1 if (i, j) is on the path; else 0	$t(i, j)$	traveling time from i to j
$b_{i,j}$	is 1 if both i and j are on the path and i precedes (not necessarily immediately) j; else 0	V	set of stops within service area plus u and v

1) Backbone Stops: The stations are divided into three types with different degrees of popularity: the start/end stations, backbone stops, and ordinary stops. Both backbone stops and ordinary stops are mined from the request dataset, where backbone stops are locations with more requests than other locations. In more detail, clustering method is used to cluster the requests of the whole area at a certain time, and the cluster centers are mapped as stops in the road network. Flexible vehicle would stop at the backbone stops, which provides some certainty to the riders on the location dimension. Backbone stops can also be manually designated if the operator considers flexible vehicles must stop at some locations to pick up riders. On the contrary, ordinary stops are candidate stops that might not have so many requests; but if there are requests, the flexible vehicles would still stop and pick up the requests. Whether vehicles would stop at the ordinary stops or not depends on the real-time requests, so the ordinary stops provide some degree of flexibility to the route planning.

The service area of a flex route consists of K rectangle subareas. For each subarea A_i of route $r(u, v, t)$, we define a *backbone stop* bs_i:

$$\begin{aligned} bs_i &= \underset{s}{\text{argmax}} \{\beta * N_0(s) + (1 - \beta) * N_1(s) : s \in S_i\} \\ N_0(s) &= num^+(s) + num^-(s), \quad N_1(s) = fw(s.grid, t) \end{aligned} \tag{40}$$

where $\beta \in [0, 1]$ is a balance factor, S_i is the set of stops within the subarea A_i. $N_0(s)$ is the number of currently received pickup and dropoff requests at stop s, $N_1(s)$ is the flow weight of $s.grid$, which is denoted by $fw(s.grid, t)$ and defined at (19). Here $s.grid$ is the grid that s belongs to.

The set of backbone stops together with the stations u, v are denoted by $BS = \{bs_0, bs_1, bs_2, \ldots bs_K, bs_{K+1}\}$, where $bs_0 = u, bs_{K+1} = v$. Stops in BS are arranged in topological order of the directed acyclic graph, and the shortest path $\{bs_i \rightarrow bs_{i+1}\}$ between bs_i and $bs_{i+1}, i = 0, \ldots, K$ could be calculated. So the initial path for the flex route is generated by iteratively connecting the backbone stops and their temporary stops between them. As illustrated in Fig. 6(a), the shortest path between a and e is $(a - d - e)$, so d is also added to the path of the route. We denote the path of route r by $path(r)$ and denote the set of all stops in $path(r)$ by $S(path, r)$. As there is only one backbone stop at each subarea within the service area, we assume that initial $path(r)$ would always satisfy the running time constraint.

2) Path Insertion: The initial path is then extended, i.e. new pickup and dropoff stops are inserted into the path, as new requests are coming in. Give a request $req(o, d, w)$, suppose o' and d' are the nearest stops to o and d, and d' is *behind* o' in topological order, there are three cases when inserting stops to the path:

- When both o' and d' are in set $S(path, r)$, the request is immediately accepted. The insertion of request does not add extra cost to the path. If o' or d' is not a backbone stop, it becomes a backbone stop, and the stop is moved to BS.
- When only one stop, either o' or d', is in BS, the flex route system would check whether the stop is feasible to be added to the path. Without loss of generality, suppose the dropoff stop d' is already in the path, yet the pickup stop o' is to be checked. Suppose the subarea that contains o' is A_i, and the set of backbone stops in $path(r)$ contained in A_i is $BS(A_i) = \{bs_{j+1}, bs_{j+2}, .., bs_{j+m}\}$, then the possible insertion positions are: $I_0,\ I_1,\ I_2, \ldots, I_m$, where $I_k = (bs_{j+k}, bs_{j+k+1})$, $k = 0, 1, \ldots, m$, and bs_j is the last backbone stop at subarea A_{i-1}. Fig. 6 illustrates an example of an insertion into the path, where $(a, e), (e, f), (f, g)$ are the possible insertion positions for stop c.

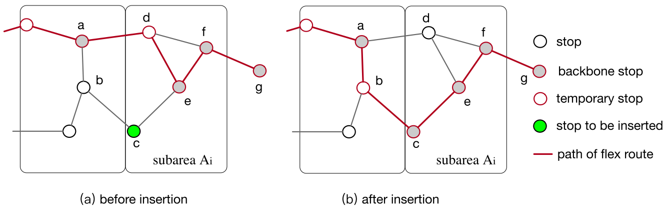
  For every possible insertion position, a new path is built to contain the new requested stop o'. The new path after insertion at I_x is denoted by $path(r)_x$, and the insertion position is selected by the following formula:

  $$k_1 = \mathop{\mathrm{argmin}}_{x} \{cost(path(r)_x) : cost(path(r)_x) < rt(r), \quad x = 0, \ldots, m\} \tag{41}$$

  where $cost(path(r)_x)$ is the cost of traveling along path $path(r)_x$, $rt(r)$ is the running time of route r. The path with insertion I_{k_1} has the least traveling time. The path is selected and it is feasible, i.e. meets the running time constraint.

  If there is a feasible path after insertion, the request would be accepted; otherwise, the request would be rejected. When o' is inserted at $I_{k_1} = (bs_{j+k_1}, bs_{j+k_1+1})$, stop o' is added to BS and the path is updated by following operations:

  $$\begin{aligned} path(r, o') &= path(r) - \{bs_{j+k_1} \rightarrow bs_{j+k_1+1}\} \\ path(r, o') &= path(r) + \{bs_{j+k_1} \rightarrow o'\} + \{o' \rightarrow bs_{j+k_1+1}\} \end{aligned} \tag{42}$$



(a) before insertion

(b) after insertion

Fig. 6. An example of an insertion into the path. $(a, e), (e, f), (f, g)$ are the possible insertion positions for stop c. Yet (a, e) is the insertion position, new paths are built by adding shortest paths $a \rightarrow c$ and $c \rightarrow e$, and removing $a \rightarrow e$.

  where $\{a \rightarrow b\}$ is the shortest path from a to b. In Fig. 6, (a, e) is the insertion position, new paths are built by adding shortest paths $a \rightarrow c$ and $c \rightarrow e$, and removing $a \rightarrow e$.
- When both stop o' or stop d' are not in BS, two insertion positions are identified and the feasibility of a new path after insertions is checked. The insert procedure is similar to case (2). Suppose the subarea that contains o' is A_i, and the set of backbone stops in $path(r)$ contained in A_i is $BS(A_i) = \{bs_{j+1}, bs_{j+2}, .., bs_{j+m}\}$. The possible insert position of o' is defined as $I_0, I_1, I_2, \ldots, I_m$, where $I_k = (bs_{j+k}, bs_{j+k+1})$, $k = 0, 1, \ldots, m$, and bs_j is the last backbone stop at subarea A_{i-1}. Similarly, we define the subarea that contains d' as $A_{i'}$, and the possible insert positions of d' is defined as $I'_0,\ I'_1,\ I'_2, \ldots, I'_n$, where $I'_k = (bs_{j'+k}, bs_{j'+k+1})$, $k = 0, 1, \ldots, n$. $bs_{j'}$ is the last backbone stop at subarea $A_{i'-1}$. Then the insert positions for o' and d' are calculated by the following formula:

  $$\begin{aligned} (k_1, k_2) = \mathop{\mathrm{argmin}}_{(x,y)} \{&cost(path(r)_{x,y}) : \\ &cost(path(r)_{x,y}) < rt(r)\}, \\ &\times x = 0, \ldots, m, \ \ y = 0, \ldots, n \end{aligned} \tag{43}$$

  where $path(r)_{x,y}$ denotes the path of r if inserting o' at I_x and d' at I_y. Only when both insertions are allowed, the request is accepted; otherwise, the request is rejected. The stops would be added to set BS if the request is accepted.

3) Algorithm Description: Algorithm 1 in the Appendix is the pseudocode of the origin-destination insertion algorithm based on the backbone stops.

V. Performance Evaluation

We conducted experiments on real-world road networks and trajectory datasets to verify the performance of the proposed scheme. The schemes are implemented in Java 1.8 and experiments are run on a notebook computer with Intel Core i7 CPU, 2.6 GHz, 16 G RAM under Windows 10.

A. Environmental Setup

1) Road Networks: The road network of the Xiamen City, Fujian Province, China is used for the simulation, which contains 24750 road vertices and 32364 road segments. By default, we set the average vehicle speed to 35 km/h in the urban area and set the traveling time of each road segment as its

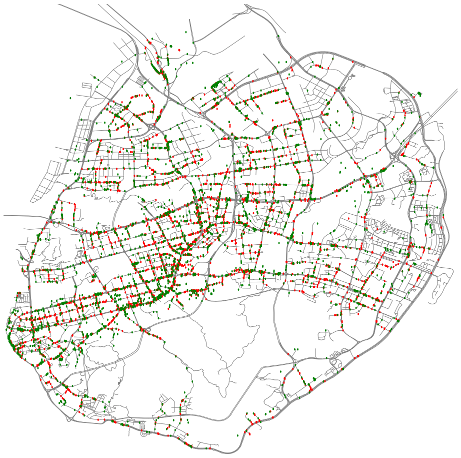


Fig. 7. Distributions of origins (in red) and destinations (in green) of trips during 6:00 a.m. to 10:00 a.m. on July 2, 2014 in the Xiamen Island, Fujian Province, China.

weight. We get the map of the Xiamen city ([118.0660E, 118.1980E]*[24.4240N, 24.5600N]) from OpenStreetMap and load the graph into the memory using the JGraphT framework[1] 1.3.0 so as to make efficient shortest path queries on the road network. The road network is divided into 100*100 grids, which is to facilitate the calculation of the request density and stops of different regions.

2) Datasets: It is hard to forecast the transit demand of requests for flexible vehicles or customized buses due to the lack of real-world OD datasets that reflects the pattern of requests. But as there is a large overlap on the customers of taxies and flexible vehicles, in this research we adopt the Xiamen Taxi Dataset[2] for the simulation, which consists of one-month trajectory data of about 5,000 taxicabs in Xiamen city, China during July 2014. There are about 220 million GPS position records and 8 million live trips. The trajectory reporting frequency is 1–2 times per minute. For this simulation we extracted trajectory data from 6:00 a.m. to 10:00 a.m. on July 2, 2014, including 59311 trajectories for performance evaluation. Fig. 7 shows the distribution of GPS points in the dataset.

B. Data Preprocessing

The stops are mined from the trajectory dataset, and the preprocessing includes three steps: 1) matching GPS points; 2) calculating the flow of passengers in each grid, 3) defining the service area, and 4) identifying the backbone stops.

1) Matching GPS Points: For each GPS record, it is inefficient to match every possible road segment. Rather, we only need to identify a few road segments that cover all possible segments for the GPS record while filtering others. According to [41], [42], GPS location errors can be as large as 100 meters in a city with dense tall buildings and viaducts. 100 meters can be roughly regarded as 0.001 latitude or longitude. So imaging there is a circle of radius 0.001 latitude or longitude centered at the GPS record, the GPS record can only reside on the road segments that intersect or tangent to the circle. From our investigation, 99.27% road segments in our road network are less than 0.005 latitude or longitude long. Such a circumstance is described in Fig. 8. The maximum distance appears when the road segment is tangent to the circle.

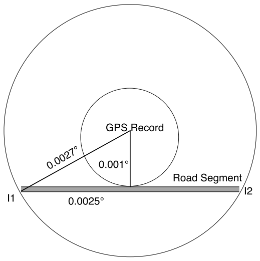


Fig. 8. The maximum distance appears when the road segment is tangent to the circle and the tangent point turns to be the midpoint.

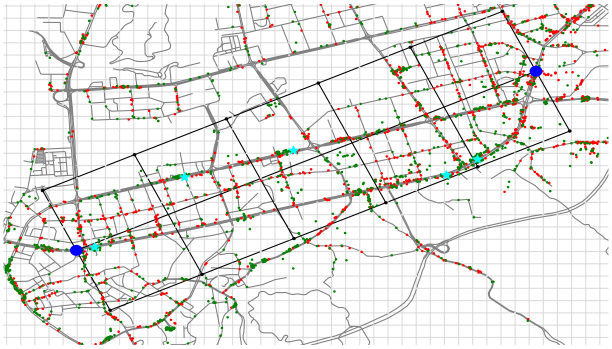


Fig. 9. Terminal stations u, v(in blue) and service area (rectangle in black, with 8.91 km in length and 0.8 km in width) in the Xiamen City, China. The map is divided into 10000 grids (in light grey), and the backbone stations are illustrated by stars.

We test each road segment on the following criteria: whether there exists a road segment that meets the condition that the distance between the endpoint and the GPS record is less than 0.0027 latitude or longitude long. For most cases, the GPS record would select the nearest road segment. But in some cases, the timestamps of the GPS record are also considered to discard anomalies or to select the road segments. When the forward and backward conjunctive GPS records reside at the same road segment, then the middle GPS point would locate at the same road segment. Once the road segment is determined, the tangent point to the circle is set to be the calibrated GPS point. But if the GPS is more than 0.0027 latitude or longitude long away from the road segment, it would be discarded.

2) Calculate the Flow of Passengers: The road network is divided into 10000 (100 * 100) grids. The GPS locations of origin-destination pairs are calibrated and mapped to the nearest vertexes in the road network. The flow of passengers is calculated by accumulating the number of origins or destinations within the grid. The time of each request is set w earlier according to the constraint of pickup window.

3) Define the Service Area: The terminal station u and v are defined as in Fig. 9, where the width is set as 1.6 km, and the time unit U is set to 5 minutes to divide the service domain into subareas according to Eq. 17. The slack ratio (α) of the flex route is set 0.6.

[1] https://jgrapht.org/

[2] http://mocom.xmu.edu.cn/project/show/xmdataset

4) Identify Backbone Stops: The backbone stops are calculated according to Eq. 18-22 in each subarea. The threshold θ and β are 0.5, and the values of them are standardized in range [0, 1] to make $N_0(s)$ and $N_1(s)$ in the same order of magnitude. Fig. 9 illustrates the service area and backbone stops.

C. Compared Schemes and Metrics

Besides the proposed Flex vehicle scheme, other origin-destination insertion schemes and scheduling schemes are also conducted for performance comparison.

1) Insertion Schemes:

- Fixed pickup/pickoff (*fixed*): pickup and pickoff passengers along fixed stations. Stops between u and v locate 300 meters away from their adjacent stops.
- Greedy algorithm with origin-destination insertion (*greedy*): according to the topological order of the stops in the road network, the adjacent stops with the most requests and their corresponding destination stops are selected to be inserted to the route.
- Probability-based spreading (*B-planner* [19]): cluster "hot" areas and split hot areas into clusters to identify candidate vehicle stops. It uses a bidirectional probability-based spreading algorithm to generate candidate vehicle routes, and adds constraints of time window to the requests.
- Heuristic backbone-based origin-destination insertion algorithm (*backbone*): it is the insertion algorithm proposed in this paper.

2) Scheduling Schemes: Besides the *milp* scheduling described in section IV, two other scheduling schemes are also implemented for comparison:

- Fixed time slot scheduling (*fixed-slot*): set a fixed departure interval for adjacent vehicles. It serves as the baseline of the schemes.
- Genetic algorithm-based scheduling (*genetic*): setting the operating cost as the objective value, it uses the genetic algorithm to construct the scheduling model.

3) Traveling Schemes: Different insertion and scheduling schemes are combined to form traveling schemes. Besides the proposed Flex vehicle system, which is actually a combination of the *backbone* insertion scheme and *milp* scheduling scheme (*backbone-milp*), we also conduct other three vehicle traveling schemes for comparison.

- *fixed-fixed-slot*: vehicle traveling along fixed stations and scheduling according to fixed time slots scheme.
- *fixed-milp*: vehicle traveling along fixed stations and scheduling according to the *milp* scheme.
- *backbone-fixed-slot*: vehicle traveling according to the *backbone* insertion scheme and scheduling according to the *fixed-slot* scheme.
- *backbone-genetic*: vehicle traveling according to the *backbone* insertion scheme and scheduling according to the *genetic* scheme.

TABLE III
PERFORMANCE OF THE TRAVELING SCHEMES

Schemes /Metrics	**Sharing Amplifier**	**Delivery Ratio(%)**	**Average Walking Distance(m)**	**Average Waiting Time(min)**
fixed-fixed-slot	1.88	18.64	433.60	11.46
fixed-milp	2.80	26.15	442.52	9.73
backbone-fixed-slot	4.28	26.96	154.33	7.18
backbone-genetic	5.88	46.54	**152.62**	7.05
backbone-milp	**7.37**	**50.51**	153.87	**6.22**

4) Metrics: The sharing amplifier (sa), rider delivery ratio (dr), and the average walking distance (wd) are three main metrics introduced in Section IV.A. As we aim to study the gap between the demand and supply of vehicle seats, the value of (10), denoted as seat-request-gap (srg), and the operating costs (oc) are adopted as metrics for the scheduling. srg is defined as:

$$srg = \sum_{i}^{n} \sum_{j}^{n} |F_{ij}^{u} - C * x_{ij}^{u}| + |F_{ij}^{v} - C * x_{ij}^{v}| \qquad (44)$$

And the calculation of operating cost is as follows:

$$oc = \delta * wc + (1 - \delta) * tc \qquad (45)$$

where $\delta \in [0, 1]$ is a balance factor, wc and tc are the waiting cost of passengers and the traveling cost of vehicles respectively. wc is calculated according to the average monthly income of Xiamen:

$$wc = \frac{wage}{21.42 * 8 * 60} \qquad (46)$$

where $wage$ is the average monthly wage, which is to be divided by the average working days per month (21.42), the working hours per day (8), and the minutes per hour (60). The traveling cost tc is obtained from historical vehicle operation data. In this research δ is set 0.5, $wage$ is 8000 RMB per month, and tc is 6.14 RMB/km.

D. Experimental Analysis

1) Overall Performance: There are 10 vehicles deployed by default, where each terminal station u and v has 5 vehicles scheduled for operation at the initial time, i.e. cap^u and cap^v are both set 5. The capacity of the vehicle cap is 35, the empirical factor κ is 1.5, the slack ratio α is 0.6, and the pick up time window is set to 10 minutes, U is set to 10 minutes by default.

Table III shows the overall performance of the five different traveling systems. The simulation was conducted based on the Xiamen Taxi Dataset from 6:00 am to 10:00 am on July 2, 2014. The *backbone-milp* and *backbone-genetic* have the highest sharing amplifier, which are 7.37 and 5.88 respectively. Meanwhile, *backbone-milp* has the highest delivery ratio at 50.51%, which is about 170% higher than the *fixed-fixed-slot* scheme. The *backbone-milp* scheme also has the shortest average waiting time at 6.22 minutes, and the average walking

TABLE IV

PERFORMANCE OF THE OD INSERTION SCHEMES

Schemes /Metrics	Sharing Amplifier	Delivery Ratio(%)	Average Walking Distance(m)	Average Waiting Time(min)
fixed	3.10	2.74	252.62	12.77
B-planner	5.17	3.13	157.00	10.63
greedy	5.42	2.10	167.35	7.82
backbone	**7.10**	**6.49**	**154.12**	**5.55**

TABLE V

PERFORMANCE OF THE SCHEDULING SCHEMES

Schemes /Metrics	Seats-Requests-Gap	Delivery Ratio(%)	Operating Cost(RMB)
fixed-slot	1533	21.00	2147.9
genetic	912	22.69	1887.7
milp	**727**	**26.95**	**1700.8**

distance is 153.87 meters. The *fixed-fixed-slot* scheme is set as the baseline. It can be found that when the scheduling scheme is changed from *fixed-slot* to *milp*, the sharing amplifier and delivery ratio increase by about 48% and 40% respectively. This is mainly because *milp* would schedule more vehicles when there is more demand in service area to maximize the objective function. When changing the traveling scheme to the proposed *backbone* scheme, the average walking distance of passengers is reduced by about 38%. The stops in the *backbone* scheme are carefully calculated and generated from mining the historical request data, therefore it reduces the paths passengers travel. Meanwhile, the sharing amplifier increases by 127% as the *backbone* scheme responds to the requests in a more immediate way and can fulfil more demands. The average waiting time is reduced by about 39%, due to the fact that *backbone* minimizes the time loss caused by each newly added origin-destination pairs and a delay time window is set. From the performance comparison, we can see that the proposed data-driven insertion scheme and optimized scheduling scheme together bring higher benefits to the vehicle operation.

To study the performance of the insertion schemes, the number of trips is set to 1. Table IV displays the overall performance of different insertion schemes when the vehicle only goes once from u to v. The *backbone* scheme has the highest sharing amplifier at 7.10, which is about 129% and 37% higher than *fixed* and *B-planner* respectively. The *backbone* scheme also has the highest delivery ratio, which is about 136% and 107% higher than the *fixed* and *B-planner* scheme. This is because vehicles in the *backbone* scheme do not stop at the fixed stop, the flexibility provides more opportunities for nearby requests to be fulfilled. The *greedy* scheme has the lowest delivery ratio, as it inserts stops that have the highest number of requests and this may miss requests in other locations. The average walking distance of passengers received by vehicles in *backbone* and *B-planner* is 154.12 and 157.00 meters respectively, which is about 39% less than that of *fixed*. The main reason is that the set of stops in *backbone* and *B-planner* are both mined from the historical request data, where the surrounding areas of these stops have high request density. Passengers can reach the nearest vehicle stops with a shorter distance. Also, the *backbone* scheme has the shortest average waiting time for passengers at 5.55 minutes, about 56% shorter than that of the $fixed$ scheme.

Table V displays the performance of three different scheduling schemes when the insertion scheme is fixed, i.e. adopting the *fixed* scheme from 6:00 a.m. to 10:00 a.m. on July 2, 2014. The *milp* scheme has the minimal gap between the number of seats and the requests, which is about 53% and 21% lower than that of the *fixed-slot* and *genetic* scheduling scheme. Meanwhile, it has the highest delivery ratio at 26.95%. The *genetic* scheme has the lowest operating cost, which is 1700.8 RMB. Compared with *fixed-slot*, the gap between the demand and supply of vehicle seats is reduced by more than 40% when adopting the *genetic* and *milp* schemes. This is because both the *genetic* and *milp* schemes have the flexibility to adopt to the requests of passengers. The goal of the *genetic* scheme is to minimize the operating cost of vehicles, so when there are fewer demands generated in the service area, less vehicles will be provided. Similarly, the *milp* scheme sets this gap as its objective function. The *genetic* and *milp* schemes also have a delivery ratio about 8% higher than that of the *fixed-slot* scheme, since when there are more requests in the service area, more vehicles would be scheduled for passengers to be picked up. The overall operating cost of vehicles is also shown in Table V. The *genetic* scheme has the lowest operating cost as we set the operating costs of vehicles as objective function. Also, the proposed *milp* scheme has lower operating cost than that of the *fixed-slot* scheme owing to the flexibility of vehicle scheduling.

2) Impact Factors: We also vary several critical factors, i.e., the slack ratio α, the width of service area, and the length of pick up time window to study their impact on the proposed scheme.

Fig. 10 depicts the impact of slack ratio α. When α increases, the sharing amplifier and the delivery ratio both increase for the *backbone* and *greedy* schemes. Larger α means longer travelling time, so requests have more opportunity to be accepted. Also, when α is set at some value, e.g. 0.8, 1.1 and 1.7, some requests with shorter distance are also fulfilled in this scenario, so the sharing amplifier decreases slightly according to its definition defined at Eq. 7. The *backbone* scheme also outperforms *greedy* on the sharing amplifier and delivery ratio, especially when α becomes higher. Larger α means more chances for the proposed algorithm to optimise the insertion, and more requests could be accepted. Passengers' average walking distance on *backbone* and *greedy* reaches its minimal when α is 0.4, which are 144.09 and 160.43 respectively. Also, the passengers' average waiting time fluctuates as α increases. When more travelling time is provided, stops far away from the initial path can now be inserted, yet it also means the vehicle needs to take longer to reach these stops. The average waiting time of $backbone$ reaches the minimum value when α is 1.4, which is 3.95 minutes. On the contrary, the average waiting time

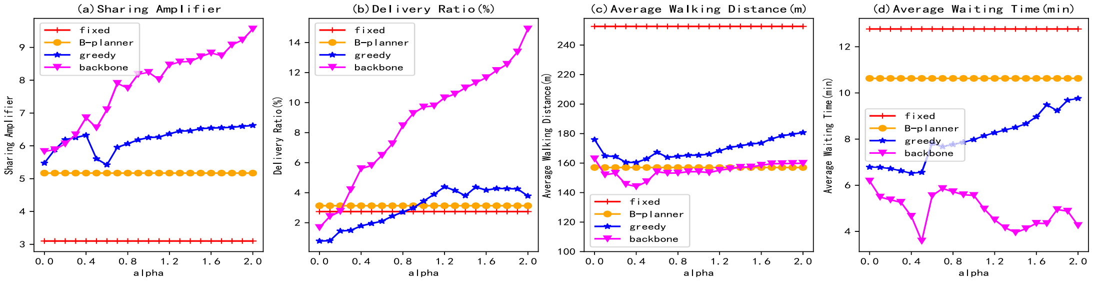


Fig. 10. Impact of slack ratio α on (a) sharing amplifier, (b) delivery ratio, (c) the average walking distance, and (d) the average waiting time.

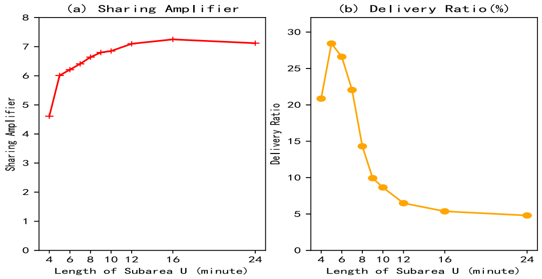


Fig. 11. Impact of the length of subarea U on the (a) sharing amplifier and (b) delivery ratio.

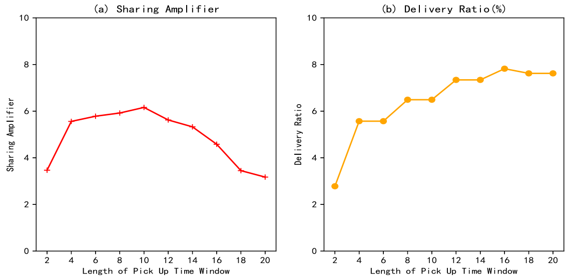


Fig. 12. Impact of length of pick up time window on (a) sharing amplifier and (b) delivery ratio.

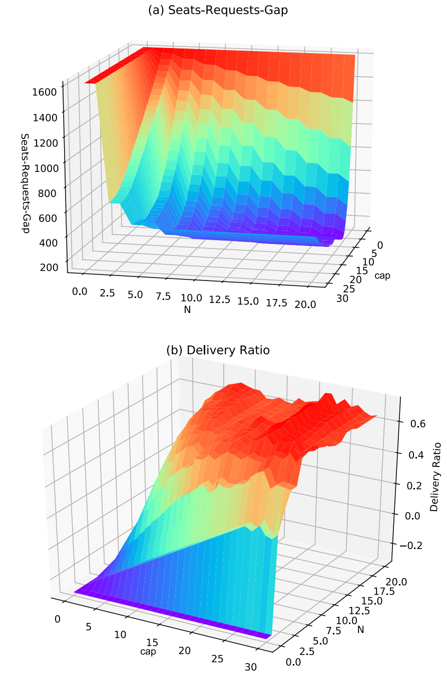


Fig. 13. Impact of the number of seats in a single vehicle *cap* and the number of planned vehicles N on the (a) seats-requests-gap, and (b) delivery ratio.

of *greedy* increases with α, which is because the time loss brought from inserting stops with the most number of requests is amplified when α increases.

Fig. 11 depicts the impact of the length of the subarea (U) on the sharing amplifier and delivery ratio. We can find that when U grows from 4 minutes to 12 minutes, the sharing amplifier increases from 4.61 to 7.02. A larger U means larger subarea, and hence fewer backbone stops that the vehicle should visit. So the route would have more flexibility and the sharing amplifier would increase. The sharing amplifier then goes to be stable, this is mainly due to the time constraint of the flexible route which makes it hard to accept more requests. Accordingly, the delivery ratio reaches a maximum value when U is about 8 minutes, and then goes down as the subarea grows larger.

Fig. 12 illustrates the impact of the pickup time window on the sharing amplifier and delivery ratio. We can find that the sharing amplifier increases significantly when the length of time window increases from 2 minutes to 6 minutes. After that, the increase on the length of time window has little impact. But as the length of time window increases, the delivery ratio also increases as longer waiting time gives requests more chance to be accepted.

We also vary the number of seats in a single vehicle *cap* and the number of planned vehicles N to study their impact on the resource utilization and the delivery ratio of the proposed scheme. The *fixed* travelling scheme is adopted by default, and *cap* is set from 0 to 30 and N is set from 0 to 20. From Fig. 13 (a), we can see that the gap between the seats and the requests decreases with *cap* and N. When more vehicles or more seats are provided, requests have a higher chance to be accepted. Fig. 13 (b) illustrates the impact of capacity of a vehicle on the delivery ratio. The delivery ratio increases with the number of seats in a vehicle and the number of available vehicles. More seats and more vehicles mean more requests can be accepted. However, small fluctuations occur on the delivery, which is because the objective function of the *milp* would make the provided resources of seats be as close as possible to the actual flow of passengers.

VI. Conclusion

We propose a data-driven flexible transit system that integrates the origin-destination insertion algorithm and the *milp*-

based scheduling scheme. The historical patterns of the request datasets are mined to efficiently construct a path for the flex route, and the time loss caused by the optimal insertion position of origin and destination of new request is calculated to decide whether to receive the request. And a vehicle scheduling model that minimises the gap between the passenger flow and available seats is adopted to reduce the operation cost. Experimental results show that the proposed flexible transit system can effectively increase the delivery ratio and decrease the passengers' waiting time compared with existing fixed or flexible transit systems.

The combination of flexibility and high sharing rate of this research is consistent with the development concept of "smart and green city", where resources are fully utilized and the cost of transportation is reduced. Our approach could evolve with the pattern of requests by adjusting the stop locations and the scheduling of vehicles. One drawback of our approach is that it needs extra tasks of data collection and pre-processing, which adds some degree of complexity to the whole public vehicle transportation system. The flexible vehicles system proposed in this research could be viewed as a special kind of the demand-responsive or Taxi system. If we can obtain more sources of data: existing transit modes, taxis, and other similar services such as uber/lyft, we would be able to estimate the demand and test our scheme in real-world scenarios. For the future work, we are going to investigate other factors, e.g. request priority and passenger utility, and propose efficient algorithms to recommend optimized pickup stops for riders to improve the overall quality of service of the transit system. Also, we are to study the problem of multi-line flexible transit system optimization in a data-driven approach.

References

[1] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks," *Proc. Nat. Acad. Sci. USA*, vol. 111, no. 37, pp. 13290–13294, 2014.

[2] Y. Huang, R. Jin, F. Bastani, and X. S. Wang, "Large scale real-time ridesharing with service guarantee on road networks," *Proc. VLDB Endowment*, vol. 7, pp. 2017–2028, Oct. 2013.

[3] Y. Lai, F. Yang, L. Zhang, and Z. Lin, "Distributed public vehicle system based on fog nodes and vehicular sensing," *IEEE Access*, vol. 6, pp. 22011–22024, 2018.

[4] N. K. Chowdhury and C. K.-S. Leung, "Improved travel time prediction algorithms for intelligent transportation systems," in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst.* Kaiserslautern, Germany: Springer, 2011, pp. 355–365.

[5] G. Ambrosino, J. D. Nelson, M. Boero, and I. Pettinelli, "Enabling intermodal urban transport through complementary services: From flexible mobility services to the shared use mobility agency: Workshop 4. Developing inter-modal transport systems," *Res. Transp. Econ.*, vol. 59, pp. 179–184, Nov. 2016.

[6] L. M. Martínez, J. M. Viegas, and T. Eiró, "Formulating a new express minibus service design problem as a clustering problem," *Transp. Sci.*, vol. 49, no. 1, pp. 85–98, Feb. 2015.

[7] J. Ma *et al.*, "A model for the stop planning and timetables of customized buses," *PLoS ONE*, vol. 12, no. 1, Jan. 2017, Art. no. e0168762.

[8] S. Ma, Y. Zheng, and O. Wolfson, "Real-time city-scale taxi ridesharing," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 7, pp. 1782–1795, Jul. 2015.

[9] C. Mulley and J. D. Nelson, "Flexible transport services: A new market opportunity for public transport," *Res. Transp. Econ.*, vol. 25, no. 1, pp. 39–45, Jan. 2009.

[10] B. Barabino, N. A. Cabras, C. Conversano, and A. Olivo, "An integrated approach to select key quality indicators in transit services," *Social Indicators Res.*, vol. 149, no. 3, pp. 1–36, 2020.

[11] J. Brake and J. D. Nelson, "A case study of flexible solutions to transport demand in a deregulated environment," *J. Transp. Geography*, vol. 15, no. 4, pp. 262–273, Jul. 2007.

[12] S. M. Nourbakhsh and Y. Ouyang, "A structured flexible transit system for low demand areas," *Transp. Res. B, Methodol.*, vol. 46, no. 1, pp. 204–216, Jan. 2012.

[13] M. Pei, P. Lin, and J. Ou, "Real-time optimal scheduling model for transit system with flexible bus line length," *Transp. Res. Record, J. Transp. Res. Board*, vol. 2673, no. 4, pp. 800–810, Apr. 2019.

[14] X. Ma *et al.*, "Mining smart card data for transit riders' travel patterns," *Transp. Res. C, Emerg. Technol.*, vol. 36, pp. 1–12, Nov. 2013.

[15] A. Attanasio, J.-F. Cordeau, G. Ghiani, and G. Laporte, "Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem," *Parallel Comput.*, vol. 30, no. 3, pp. 377–387, Mar. 2004.

[16] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: Models and algorithms," *Ann. Oper. Res.*, vol. 153, no. 1, pp. 29–46, 2007.

[17] R. Dondo and J. Cerdá, "A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows," *Eur. J. Oper. Res.*, vol. 176, no. 3, pp. 1478–1507, 2007.

[18] M. Zhu *et al.*, "Public vehicles for future urban transportation," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 12, pp. 3344–3353, Dec. 2016.

[19] C. Chen, D. Zhang, N. Li, and Z.-H. Zhou, "B-Planner: Planning bidirectional night bus routes using large-scale taxi GPS traces," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 4, pp. 1451–1465, Aug. 2014.

[20] L. Lee, V. Bagheri, Y. Sadullah, A. Farhan, and B. Mohd, "Analysis of headways on passenger loads for public bus services: Case study of Penang Island, Malaysia," *Eur. J. Sci. Res.*, vol. 45, no. 3, pp. 476–483, 2010.

[21] H. L. Khoo and G. P. Ong, "Bi-objective optimization approach for exclusive bus lane scheduling design," *J. Comput. Civil Eng.*, vol. 29, no. 5, Sep. 2015, Art. no. 04014056.

[22] P. DiJoseph and S. I.-J. Chien, "Optimizing sustainable feeder bus operation considering realistic networks and heterogeneous demand," *J. Adv. Transp.*, vol. 47, no. 5, pp. 483–497, Aug. 2013.

[23] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *Proc. IEEE 29th Int. Conf. Data Eng. (ICDE)*, Apr. 2013, pp. 410–421.

[24] R. Gomes, J. P. de Sousa, and T. G. Dias, "A GRASP-based approach for demand responsive transportation," *Int. J. Transp.*, vol. 2, no. 1, pp. 21–32, Apr. 2014.

[25] R. Gomes, J. P. D. Sousa, and T. Galvão, "An integrated approach for the design of demand responsive transportation services," in *Computer-based Modelling and Optimization in Transportation*. Berlin, Germany: Springer, 2014, pp. 223–235.

[26] M. Zhu, X.-Y. Liu, and X. Wang, "An online ride-sharing path planning strategy for public vehicle systems," 2017, *arXiv:1712.09356*.

[27] M. Zhu, X.-Y. Liu, and X. Wang, "Joint transportation and charging scheduling in public vehicle systems—A game theoretic approach," 2017, *arXiv:1712.07947*.

[28] M. K. E. Mahrsi, E. Come, L. Oukhellou, and M. Verleysen, "Clustering smart card data for urban mobility analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 3, pp. 712–728, Sep. 2017.

[29] V. Boyer, O. J. Ibarra-Rojas, and Y. Á. Ríos-Solís, "Vehicle and crew scheduling for flexible bus transportation systems," *Transp. Res. B, Methodol.*, vol. 112, pp. 216–229, Jun. 2018.

[30] M. Repoux, N. Geroliminis, and M. Kaspi, "Operational analysis of an innovative semi-autonomous on-demand transportation system," *Transp. Res. C, Emerg. Technol.*, vol. 132, Nov. 2021, Art. no. 103373.

[31] J.-F. Cordeau, "A branch-and-cut algorithm for the dial-a-ride problem," *Oper. Res.*, vol. 54, no. 3, pp. 573–586, 2006.

[32] D. Huang, Y. Gu, S. Wang, Z. Liu, and W. Zhang, "A two-phase optimization model for the demand-responsive customized bus network design," *Transp. Res. C, Emerg. Technol.*, vol. 111, pp. 1–21, Feb. 2020.

[33] D. Pisinger and S. Ropke, "A general heuristic for vehicle routing problems," *Comput. Oper. Res.*, vol. 34, no. 8, pp. 2403–2435, Aug. 2007.

[34] Q. Sun, S. Chien, D. Hu, G. Chen, and R.-S. Jiang, "Optimizing multi-terminal customized bus service with mixed fleet," *IEEE Access*, vol. 8, pp. 156456–156469, 2020.

[35] H. Wang, "Routing and scheduling for a last-mile transportation system," *Transport. Sci.*, vol. 53, no. 1, pp. 131–147, 2017.

[36] Q. Zhu and Y. Li, "Improved harmony search algorithm for bus scheduling optimization," in *Proc. Chin. Control Decis. Conf. (CCDC)*, pp. 815–818, 2019.

[37] Z. Quan, W. Xin, and N. Di, "Research on intelligent bus scheduling based on QPSO algorithm," in *Proc. Int. Conf. Intell. Comput. Technol. Automat. (ICICTA)*, 2015, pp. 730–733.

[38] D. Tan, J. Wang, H. Liu, and X. Wang, “The optimization of bus scheduling based on genetic algorithm,” in *Proc. Int. Conf. Transp., Mech., Elect. Eng. (TMEE)*, 2011, pp. 1530–1533.

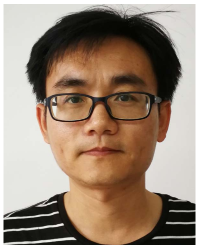
[39] M. Ye, “Novel field-bus real-time network scheduling method,” in *Proc. 2nd Int. Conf. Comput. Model. Simul.*, vol. 3, 2010, pp. 253–256.

[40] S. Hassold and A. Ceder, “Multiobjective approach to creating bus timetables with multiple vehicle types,” *Transp. Res. Rec.*, vol. 2276, pp. 56–62, Dec. 2012.

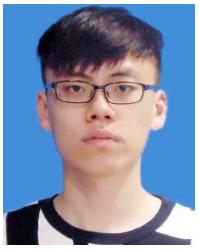
[41] H. Zhu, Y. Zhu, M. Li, and L. M. Ni, “SEER: Metropolitan-scale traffic perception based on lossy sensory data,” in *Proc. IEEE Int. Conf. Comput. Commun., Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, Apr. 2009, pp. 217–225.

[42] M. A. Quddus, “Validation of map matching algorithms using high precision positioning with GPS,” *J. Navigat.*, vol. 58, pp. 257–272, May 2005.

Yongxuan Lai received the bachelor’s degree in management of information system and the Ph.D. degree in computer science from the Renmin University of China in 2004 and 2009, respectively. He is currently a Professor with the Software Engineering Department, School of Informatics, Xiamen University, China. From September 2017 to September 2018, he was a Visiting Scholar with The University of Queensland, Australia. His research interests include network data management, intelligent transportation systems, and big data management and analysis.



Fan Yang received the Ph.D. degree in control theory and control engineering from Xiamen University in 2009. He is currently an Associate Professor with the Department of Automation, Xiamen University. His research interests include feature selection, ensemble learning, and intelligent transportation systems.



Ge Meng received the B.S. degree from the Department of Software Engineering, Lanzhou University of Technology, in 2017. He is currently pursuing the Ph.D. degree with the School of Informatics, Xiamen University. His research interests are machine learning and intelligent transportation systems.

Wei Lu received the bachelor’s and master’s degrees from the China University of Geoscience, Beijing, and the Ph.D. degree from Renmin University. He is currently a Professor with the Computer Science Department, Renmin University. He worked as a Research Follow of the Computer Science Department, NUS, from July 2011 to September 2014. He was a Research Assistant with the DKE Group, The University of Queensland, Australia, from December 2007 to December 2008, and Database Group, National University of Singapore, from June 2010 to December 2010. His research interests include data mining and big data analysis, cloud database systems, and spatial and textual data management.