Applied Soft Computing 131 (2022) 109752

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Deep Reinforcement Learning based dynamic optimization of bus timetable

Guanqun Ai [a,b], Xingquan Zuo [a,b,*], Gang Chen [c], Binglin Wu [a,b]

[a] *School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China*
[b] *Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, China*
[c] *School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand*

ARTICLE INFO

Article history:
Received 10 December 2021
Received in revised form 3 September 2022
Accepted 18 October 2022
Available online 25 October 2022

Keywords:
Bus timetable
Deep Reinforcement Learning
DQN
Online optimization

ABSTRACT

Bus timetable optimization is a key issue to reduce operational cost of bus company and improve the transit service quality. Existing methods optimize the timetable offline. However, in practice, the short-term passenger flow may change dramatically from time to time. Timetables generated offline cannot be adjusted in real time to handle the changed passenger flow. In this paper, we propose a Deep Reinforcement Learning based bus Timetable dynamic Optimization method (DRL-TO). In DRL-TO, the problem of bus timetable optimization is formulated as a Markov Decision Process (MDP). A Deep Q-Network (DQN) is applied as the agent to decide whether a bus departs at each minute during the service period. Therefore, departure intervals of bus services are determined in real time in accordance with passenger demand. We identify several new and useful state features for the DQN agent, including the load factor, the carrying capacity utilization rate, passengers' waiting time and the number of stranded passengers. Considering the interests of both the bus company and passengers, a reward function is designed, which includes metrics of full load rate, empty load rate, passengers' waiting time, and the number of stranded passengers. Experiments demonstrate that, in comparison to the timetable generated by offline optimization approaches and the manual method, DRL-TO can dynamically determine the departure intervals based on the real-time passenger flow, and generate a timetable with less departure time points (i.e., operational cost) and shorter passengers' waiting time (i.e., higher quality of service).

1. Introduction

Currently, the increasing number of private cars poses immense challenge on urban transportation and environment [1]. The bus system is an important part of urban transportation system, and has the characteristics of low price and large passenger capacity, which can ease traffic congestion, save energy and reduce pollution. However, in many cities, the transit service quality of the bus transportation system is poor. Long waiting time, crowded, and slow running speed increase passengers' preference over private cars [2]. Therefore, it is necessary to improve the transit service quality and operating efficiency of the bus system to attract more passengers to choose public transportation.

Optimizing bus departure timetable is an eximious way to achieve high transit service quality and operating efficiency. Considering the interests of both passengers and bus company, the bus timetable optimization method should optimize departure intervals according to passenger flow. This is vital to meet passengers' demand and save the operational cost of the bus company [3].

There were several prior bus timetable optimization methods reported, such as, Genetic Algorithm (GA) [4–6] and its variants [7,8], graphical heuristics algorithm [9], mathematical method [10], and exhaustive method [11]. All these approaches generate bus timetables in an offline manner according to historical passenger flow. Once the departure timetable is generated, it will be strictly followed by bus company. However, during the actual operation of bus system, passenger flow is often changed by unexpected events (e.g., when there will be a heavy rain in summer, students may go home earlier). Some activities can also change passenger flow (e.g., If an event is held at an exhibition center, the passenger flow of the line passing the center will increase) [12]. In these cases, the timetable generated based on historical data cannot satisfy the passengers' traveling demand.

To address the problem mentioned above, a bus Timetable dynamic Optimization method based on Deep Reinforcement Learning (DRL-TO) is proposed in this paper, which can optimize departure intervals (frequency) adaptively according to the real-time passenger flow.

[*] Corresponding author at: School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China.
E-mail address: zuoxq@bupt.edu.cn (X. Zuo).

https://doi.org/10.1016/j.asoc.2022.109752
1568-4946/© 2022 Elsevier B.V. All rights reserved.

DRL-TO is different from the prior bus timetable optimization methods mentioned above. In DRL-TO, the full time period of bus service is discretized into equal time steps, and the step size is one minute in this paper. Each step is considered as a decision point. A Deep Reinforcement Learning (DRL) based method is adopted as the controller, which decides whether a bus will depart from the starting station at each decision point. In this manner, a bus timetable is generated dynamically and adaptively with the flow of time. Thus, the bus timetable optimization problem is transformed into a real-time decision-making problem, which can be modeled as a discrete-time Markov Decision Process (MDP). To address this problem, Deep Q-Network (DQN) is employed as the agent. The carrying capacity utilization rate, the full load rate, the number of stranded passengers and passengers' waiting time are considered as the state features for the DQN agent. To give appropriate feedback to the agent, we design a reward function that considers empty load rate, full load rate, the number of stranded passengers and passengers' waiting time. The contributions of this paper are summarized as follows:

(1) The bus timetable optimization problem is modeled as a MDP and solved with a DRL based method. The existing bus timetable optimization methods, which optimize bus timetable in an offline manner, fail to handle the sudden changes of passenger flow.

(2) Several representative state features and a useful reward function are designed to guide the decisions of our new DRL system. A new design of state features and reward functions must be devised specially for this problem.

(3) DRL-TO is compared experimentally with the state-of-art methods and the timetables designed manually by domain experts. Experiments show that DRL-TO can be effectively utilized for the online bus timetable optimization with good performance.

The rest of this paper is arranged as follows. In Section 2, we briefly review the published bus timetable optimization methods and the departure timetable related applications of reinforcement learning. In Section 3, the bus timetable optimization problem is introduced in detail. Section 4 presents the bus timetable optimization method proposed in this paper. The experimental results are presented in Section 5. Finally, Section 6 concludes this study.

2. Related works

In this section we briefly review existing bus timetable optimization methods and the application of Reinforcement Learning (RL) to transportation timetable optimization.

2.1. Bus timetable optimization methods

A bus timetable includes a series of departure time points during a pre-defined service period of bus system (e.g., from 6:00 am to 11:00 pm everyday). Each departure time point is the planned time for a bus to depart from its starting station [8]. In [4–6], the bus timetable is optimized with uniform departure interval (e.g., departure time 6:00, 6:05, 6:10, 6:15). Meanwhile, other research works can generate a bus timetable with uniform departure interval (e.g., departure time 6:00, 6:04, 6:10, 6:17) [7,8,13].

The mathematical programming algorithms and heuristic algorithms are mainly used to generate the bus timetable with uniform intervals. In particular, the heuristic algorithms are more widely used to optimize the bus timetable. Many research works have been carried out to optimize the bus departure timetable by using GA or its upgraded version [4,6]. To improve transit service quality and cut down the operational cost of bus company, GA was employed by Han et al. [4] to optimize the objective function, which considered the passengers' cost and operator payment. A tabu search was embedded into GA by Zhou et al. [1] and an improved NSGA-II was proposed by Yang et al. [3] to reduce the operational cost of bus company and passengers' waiting time. The Non-dominated Sorting Genetic Algorithm-II (NSGA-II) was used to search for Pareto-optimal solutions of a bus timetable optimization model [14]. To improve the transit service quality, Wihartik et al. [5], Li et al. [15] and Tang et al. [16] presented an improved GA to optimize bus timetables, respectively. Considering the passengers' demand and the uncertain travel time, an improved GA was designed by Gkiotsalitis et al. [6] to optimize the bus timetables in an offline manner.

Besides, some other heuristic algorithms are also utilized for bus timetable optimization. Oudheusden et al. designed [17] a programming model and several heuristic methods to optimize the bus timetables in Bangkok in an offline manner. A graphical heuristic method was also used by Ceder et al. [9] to generate near-optimal solutions.

Mathematical programming methods are common in bus timetable optimization problems to generate the bus timetable with uniform departure intervals. Considering both the passengers' traveling cost and operational cost of bus company, Sun et al. [18] designed a model to control the bus departure interval. To improve bus operation efficiency and passengers' satisfaction, several numerical search methods were designed by Shang et al. [11]. Besides, Ceder et al. [19] proposed four passenger flow based methods to optimize bus departure timetables. The point test (the maximum load) data and the travel check (load curve) data were used as the benchmark for generating bus timetables.

The bus timetable with non-uniform interval departure has variable departure intervals in each time period. Generally, the bus timetable during a specific time period was optimized periodically. Different from that, the methods proposed by Sun et al. [13] and Li et al. [20] generated the bus timetable of a whole day with multiple runs. Considering the operational period as one period, the parallel genetic algorithm (PGA) [7] and a memetic algorithm (BTOA-MA) [8] generated the bus timetable for the whole day directly.

In summary, the bus timetable optimization methods mentioned above can only generate bus timetable in an offline manner. These methods cannot adjust the departure interval in real time if the actual passenger flow changes.

2.2. Departure timetable related applications of reinforcement learning

This subsection explores the related applications of RL in bus service as well as RL methods for optimizing subway and train timetables.

2.2.1. Reinforcement learning in bus service

Research on reinforcement learning in bus service management can be divided into two categories: (1) Optimizing parameters of other algorithms [21,22]; and (2) Optimizing parking times at each bus station for a fixed bus timetable [23]. To optimize the bus timetable, a meta-heuristic method was proposed in [21]. In [21], reinforcement learning was used as a selector that decides which genetic operator was used by the meta-heuristic method. Similarly, Matos et al. [22] employed boolean satisfiability problem model to optimize bus timetables. RL was used to optimize parameters of this model. Furthermore, the parking times at each bus station was optimized by RL in [23]. However, in [23] the bus timetable was fixed, which was not optimized by RL. Besides, DRL was employed to jointly optimize the bus network and timetable [24]. However, the method in [24] mainly used DRL to optimize the transit route network, not the timetable.

In [24], only the total number of departure time points in the bus timetable are determined for calculating the reward function of DRL.

In this paper, DRL is used for dynamic bus timetable optimization. Particularly, DRL-TO can be used to make online decisions regarding departure intervals of buses. In particular, DRL-TO is able to generate a timetable according to the real-time passenger flow. Generating the timetable online is very important in the case of uncertain events, since it can adjust the departure intervals of buses according to the real-time passenger flow. Existing approaches for bus timetable optimization are all offline approaches and generate a timetable in advance according to the historical passenger flow, which cannot handle uncertain events.

2.2.2. Optimization of train and subway timetables

As far as we know, there is only one existing work [25] that used DRL to determine subway departure intervals. In this work, the timetabling agent used State Action Reward State Action (SARSA) to make its decisions. The state space was composed of the operation information of the subway and the number of people waiting at each subway station. The operation information of the subway mainly included position and driving direction. The "departure" and "no departure" were used as actions. For the problem of subway timetable optimization, bus timetable optimization is another problem that will lead to some further research. There are two reasons as follows: (1) The operation of the subway is almost unaffected by traffic conditions because it runs on a fixed track. However, the operation of buses will be affected by road traffic conditions, which makes the optimization problem of bus timetable more complicated than that of subway. (2) The passenger flow of subway is generally stable. The problem focuses more on parking time at each station, which will affect subsequent subways. However, the optimization problem of bus departure timetable pays more attention to the departure intervals. In summary, the method in [25] cannot be used in this paper directly.

Reinforcement learning is also common in the Train Timetable Rescheduling (TTR) problem [26,27], and train stop time optimization problem [28]. The train timetable optimization problem concerns about the arrival time and departure time of the train at every station it passes through [28]. The TTR problem requires DRL agent to generate a feasible timetable in a short period of time, when the planned timetable no longer holds [26,27,29,30]. Their solutions cannot be applied to our problem directly.

3. The bus timetable optimization problem

The bus timetable optimization problem refers to determining the departure interval of a series of trips during the bus system service period according to the passenger flow. The problem considers the interests of both passengers and bus company.

The conflicting interests of passengers and bus company can be measured by several metrics. The passengers' interests are generally measured by the crowd degree and the waiting time, while that of bus company are measured by departure time points during a service period [1]. On the one hand, too less departure time points in the bus service period will result in a high bus crowd degree and prolong passengers' waiting time. On the other hand, too many points in the bus timetable may cause extra operational cost. Therefore, a properly designed bus departure timetable should improve the service level and reduce the operational cost of the bus company.

Dynamic optimization of bus departure timetable refers to dynamically setting the departure time of buses for a bus line according to the real-time passenger flow. The time interval between two consecutive departures must be greater than the minimum departure interval T_{min}, and less than the maximum departure interval T_{max}. T_{min} and T_{max} are the pre-set problem parameters.

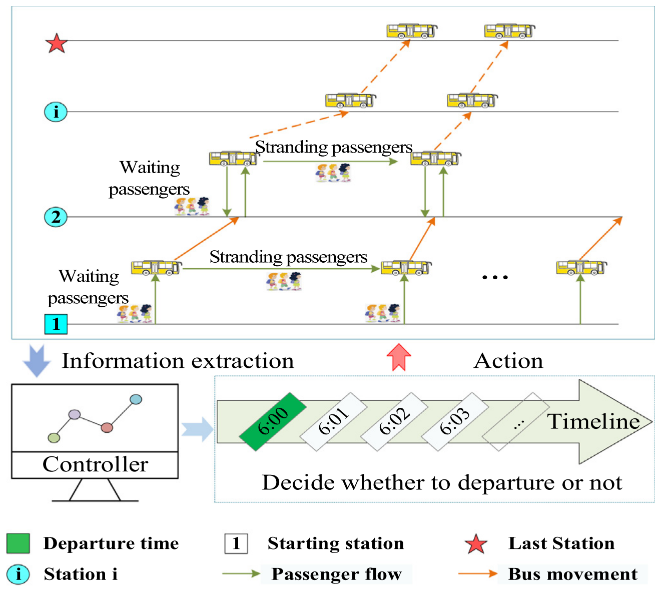


Fig. 1. The bus timetable optimization problem.

In practice, the bus line that does not share stations with other lines is common. Furthermore, for lines with a small proportion of transfer passengers, the optimization of timetable for a line has more practical research value than for multiple lines. Therefore, we study the optimization problem of the bus departure timetable of a single line in this paper.

4. Dynamic optimization of bus timetable based on deep reinforcement learning

There are usually two starting stations (control points) in a bus line. Each starting station has a bus departure timetable, which determines the departure time of buses in the corresponding direction (i.e., upward direction and downward direction). Fig. 1 presents an overview of the MDP model in this paper to optimize bus timetables in one direction.

According to the real-time passengers' demand and traffic condition, the controller adjusts the departure interval in real time by making a "depart" or "not depart" decision at each decision point. If the controller makes the decision "depart", a bus will depart from the starting station, and move forward according to the traffic conditions. When it arrives at one station, passengers onboard get off the bus first, and then the passengers waiting at the station get on the bus. Some passengers may be stranded whenever the arriving bus reaches its capacity limit, and they have to wait for the next bus.

The bus system changes in one-minute steps, similar to several existing works [3,18]. During this process, the environment executes the action given by the controller, and then the buses running on road moves forward according to the traffic conditions. More passengers may also arrive dynamically at multiple pick-up stations.

The passengers' actual demand can be measured by the number of passengers boarding and alighting buses at each station along the bus line, which is usually denoted by the origin–destination matrix (OD matrix). We use the travel time of buses between two adjacent stations to measure the traffic condition of the road segment. t_m^k denotes the travel time of the bus departs at

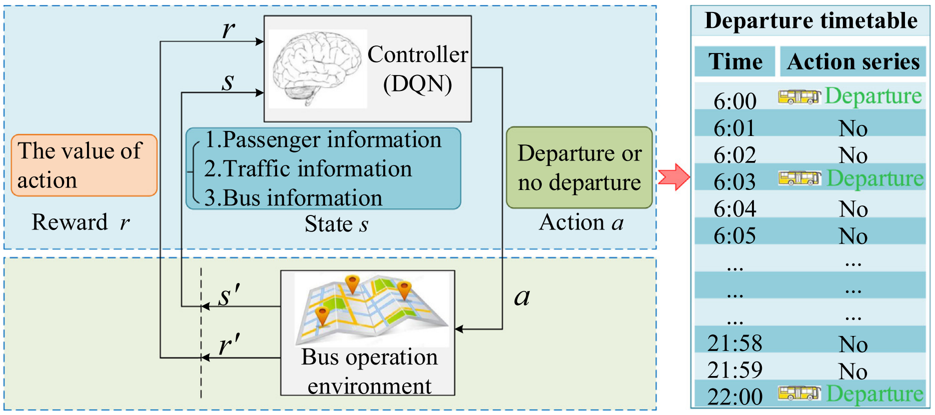


Fig. 2. Framework of the proposed method.

the mth minute from the kth station to the $(k+1)$th station. For convenience, we can track time in terms of minutes. For example, 360th minute corresponds to 6:00 am.

We formulate the sequential decision problem as a Markov Decision Process (MDP). The MDP model consists of five elements, i.e., $< S, A, P, R, \gamma >$. S is the state space. A is the action space, which includes two actions: "departure" and "no departure". P is the probability transition matrix, which is defined in Eq. (1). $P^a_{ss'}$ represents the expected probability of transferring to state s' after performing action a on state s. R is the reward function, which is defined as Eq. (2). R^a_s represents the expected reward that can be obtained at the next time $(t+1)$ after taking $A_t = a$ on state $S_t = s$ at the current time t. γ is the discount factor, we will discuss its influence in detail in Section 5.6.2.

$$P^a_{ss'} = \mathbb{P}\left[S_{t+1} = s' \mid S_t = s, A_t = a\right] \tag{1}$$

$$R^a_s = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right] \tag{2}$$

All elements of an MDP have a significant impact on the effectiveness of reinforcement learning for dynamic bus timetabling. Detailed design of each element will be presented in the following subsections.

4.1. Overview of the DRL-TO

Fig. 2 presents a high-level overview of the DRL-TO algorithm designed in this paper. The agent's decision-making (bus timetable optimization) process is as follows. At the first minute and the last minute in the bus timetable, a bus always departs from the starting station. Except for these two time points, other departure time points will be determined by the agent. At each decision point, the agent (controller) chooses an action ("departure" or "no departure") according to the current state. If the action "departure" is selected, the current minute will be taken as a time point and added to the bus timetable. Subsequently, the next state and a reward will be made known to the agent. The above process is repeated until the end of the service period.

As the number of stations increases, the size of the state space for the agent also increases, which may lead to the dimensionality disaster problem. To cope with this challenge, DQN [31] combines deep neural networks with Q-Learning to tackle the dynamic timetabling problem with discrete actions and a large state space.

4.2. Deep Q-Network

Compared with traditional reinforcement learning, DQN has two important improvements: the experience replay buffer and the target network. During learning, DQN obtains many experiences consist of current state s, action a, reward r and next state s'. These experiences are saved into the experience replay buffer as tuples in the form of (s, a, r, s'). In each training step, a mini-batch of the experiences are randomly selected and used to train DQN. This is shown to be effective at breaking the correlation between adjacent experiences, and accelerating the convergence of DQN [31]. Besides, another important improvement in DQN is the target network, which has the same structure as main network and is applied to estimate the state–action value [32].

DQN is a value-based reinforcement learning method, which is guided by the state–action value function to determine the actions to take ("departure" or "no departure") at every minute. The state–action value quantifies the expected long-term benefits of performing action a on state s, and can be calculated by the optimal Bellman equation:

$$q^*(s,a) = \sum_{s',r} p(s', r|s,a)[r + \gamma \times \max_{a'} q^*(s', a')] \tag{3}$$

where $q^*(s,a)$ indicates the maximum value of the expected return that can be obtained by performing action a on current state s. $q^*(s',a')$ is the maximum value of the expected return of next state–action pair. $p(s', r|s, a)$ represents the probability of getting state s' and reward r after performing action a on state s. γ is the discount rate.

In order to reduce the explore space for DQN agent, we introduce domain knowledge to restrict action selection. We stipulate that the departure interval must be between the maximum departure interval $T_{\max}$ and the minimum interval $T_{\min}$. $T_{\max}$ and $T_{\min}$ are hyperparameters set based on human experience. If the interval since the last departure time point t_{ml} is between $T_{\min}$ and $T_{\max}$, the $\varepsilon - greedy$ strategy is used to select actions. If t_{ml} is less than $T_{\min}$, we force the agent to choose action "no departure" to avoid too many departures; if t_{ml} is greater than $T_{\max}$, we force the agent to choose action "departure" to ensure that passengers do not have to wait for a long time. The pseudo-code is shown in Algorithm 1.

4.3. Reward function design

The reward function is related to the goal, which guides the agent to select the correct action. In this paper, our goal is to ensure the transit service quality while cutting down unnecessary operational cost of bus company by setting suitable departure intervals. As mentioned in Section 3, the passengers' interests are measured by the bus crowd degree and the waiting time, while that of bus company are measured by the number of departure time points during a pre-determined period. In particular, the

Algorithm 1 Action selection of heuristic DQN agent.

```
Input: $\varepsilon \in (0, 1)$, state $s$
Output: action $a$
1: If random() < $\varepsilon$ : // random() is a random real number in [0, 1].
2:   if $t_{ml} > T_{\max}$ :
3:     $a \leftarrow 1$
4:   elseif $t_{ml} < T_{\min}$ :
5:     $a \leftarrow 0$
6:   else :
7:     $a \leftarrow random(s)$ // Select an action randomly.
8: Else :
9:   if $t_{ml} > T_{\max}$ :
10:    $a \leftarrow 1$
11:  elseif $t_{ml} < T_{\min}$ :
12:    $a \leftarrow 0$
13:  else :
14:    $a \leftarrow \arg\max_a q(s, a)$
```

bus crowd degree of a bus is measured by both the number of passengers onboard [8] and the probabilities for passengers to be stranded.

Only two actions "departure" and "no departure" can be selected by the agent. We let 1 denote "departure" and 0 denote "no departure". Action 0 and action 1 are mutually restrictive. When the passenger flow is small, the agent should give priority to action 0. That is, no bus departs from starting station, which can reduce the number of time points in the timetable and cut down the operational cost of the bus company. On the contrary, if the passenger flow is large, the agent should give priority to action 1, which can improve the service quality. So we give a reward $1 - (o_m/e_m)$ to agent for action 0 and another reward o_m/e_m for action 1. o_m/e_m represents the service utilization rate. e_m is the carrying capacity provided by the bus departing at the mth minute. o_m is the carrying capacity consumed by passengers in e_m, $o_m \leqslant e_m$. The larger the o_m/e_m is, the greater reward of action 1 is.

Note that the same seat on the bus may serve different passengers on different sections of the bus line. The carrying capacity does not refer to the maximum cross-sectional passenger flow that a bus can carry (i.e., the maximum number of passengers a bus can carry), but refers to the total number of stations that can be taken by all passengers (see Section 4.5). Each departure time point in a bus timetable corresponds to a bus trip, so the carrying capacity provided by the bus departure timetable is also represented by the carrying capacity of buses.

Passengers' waiting time is another metric of transit service quality. Action 1 reduces passengers' waiting time, and action 0 increases the waiting time. Thus, a penalty $\omega \times W_m$ is given to the agent, if it selects action 0. W_m represents all the passengers' waiting time who took the bus that departed at the mth minute. ω is the penalty factor of the waiting time, and $0 < \omega < 1$. We set ω to make $\omega \times W_m$ be in [0, 1].

Furthermore, it is an extremely poor experience for passengers to be stranded at the station. This may significantly increase the passengers' waiting time, and cause strong anxiety among stranded passengers [33]. The stranded passengers are undesired by both actions. Hence, we give a penalty $\beta \times ds_m$ to the agent for both actions 0 and 1. ds_m represents the total number of passengers who are stranded by the bus that departed at the mth minute, which can be calculated by Eq. (5). β is the penalty factor for the number of passengers stranded.

In summary, the reward function is shown in Eq. (4):

$$r_m(s_m, a) = \begin{cases} 1 - (o_m/e_m) - (\omega \times W_m) - (\beta \times ds_m) & , \quad a = 0 \\ (o_m/e_m) - (\beta \times ds_m) & , \quad a = 1 \end{cases} \tag{4}$$

where $r_m(s_m, a)$ represents the immediate reward the DQN agent obtains after taking action a on state s_m.

$$ds_m = \sum_{k=1}^{K-1} ds_{mk} \tag{5}$$

where K represents the total number of stations on the bus line, and ds_{mk} represents the number of stranded passengers at the kth station due to the bus departing at the mth minute reaching its capacity limit. For the bus with the capacity $C_{max} = 47$, we set $\beta = 0.2$. Due to both o_m/e_m and $\omega \times W_m$ in the range of [0,1], when $ds_m > C_{\max} \times 10.6\%$, the reward $r_m(s_m, a)$ will be negative.

4.4. State space design

A DRL agent percepts the environment through state. Hence, selecting informative state features from bus operation system is essential for DRL. A well-designed state not only contains information about the environment, but also can provide information for the reward function. As mentioned above, in the reward function, we consider several state features: the carrying capacity provided by a bus e_m, and of which the amount actually consumed o_m, all the passengers' waiting time who took the bus that departed at the mth minute W_m, and the total number of stranded passengers with respect to the bus that departing at the mth minute ds_m. Because the same state feature may appear at different times, we add time information to the state space. Additionally, for all state features, we normalize their values into the same range [0, 1].

In summary, the definition of state is shown as below.

$$s_m = [x_m^1, x_m^2, x_m^3, x_m^4, x_m^5] \tag{6}$$

where x_m^1 and x_m^2 are temporal dimensions, which are calculated by Eqs. (7) and (8).

$$x_m^1 = t_h \;/\; 24 \tag{7}$$

$$x_m^2 = t_m \;/\; 60 \tag{8}$$

where t_h indicates the hour of the current time, and t_m indicates the number of minutes after the hour.

The crowded degree is another important state feature, which indicates the passengers' comfort level. We measure it by the maximum load factor in this paper, which is calculated by Eq. (9).

$$x_m^3 = C_{\max}^m / C_{\max} \tag{9}$$

where $C_{\max}$ is the capacity of the bus, and $C_{\max}^m$ represents the largest cross-section passenger flow during the running of the bus departing at the mth minute, i.e., the number of passengers onboard. If $x_m^3 = 1$, it means that the number of passengers onboard reaches the capacity of the bus, and some passengers waiting at stations may be stranded.

x_m^4 represents normalized waiting time, and its calculation is given in Eq. (10):

$$x_m^4 = W_m / \mu \tag{10}$$

where μ is the normalization factor, which can normalize passengers' waiting time into the range [0, 1]. W_m denotes the total waiting time of passengers riding the same bus that departed at the mth minute, which can be calculated as below.

$$W_m = \sum_{k=1}^{(K-1)} \sum_{i=1}^{l_m^k} (t_b^{m,i,k} - t_a^{m,i,k}) \tag{11}$$

where $t_a^{m,i,k}$ and $t_b^{m,i,k}$ represent respectively the arrival time and boarding time of passenger i at the kth station. Meanwhile m in $t_a^{m,i,k}$ and $t_b^{m,i,k}$ indicates that the bus that passenger i takes is departed at the mth minutes. l_m^k denotes the total number of passengers boarding the bus at the kth station.

The carrying capacity utilization rate is an important feature, showing how well service is matched to demand. Thus, we consider x_m^5 as a state feature, which is calculated by Eq. (12).

$$x_m^5 = o_m/e_m \tag{12}$$

These features are calculated at each decision point and used by DQN agent to choose its action.

From Eq. (9)–(12), we can see that the states are calculated based on the passenger flow in the period from the current time (when DRL-TO makes a decision) to the end time of the trip departing from the current time. Hence, DRL-TO needs to know the current passenger flow and the predicted passenger flow within a short time (the time for a bus to travel from one CP to another CP). The length of "short time" is related to the length of the bus line and the traffic conditions. We assume that the bus stops at the starting station. For the passengers waiting at the starting station, the time from bus departing to serving them is negligible. Thus, DRL-TO requires to know the current passenger flow at the starting station. For any other stations, DRL-TO must predict the passenger flow of the respective stations in the near future. This is because non-negligible time will be spent for the bus departing from the CP to arrive at these stations. The decision will lag behind the changes of passenger flow if the DRL controller only knows the passenger flow at the current time. For example, when the passenger flow increases significantly at the starting station of a bus line at 8:00 am, the bus departing at the same time will serve the waiting passengers immediately. However, if the passenger flow increases significantly at the 8th station of a bus line at 8:00 am, it is too late for the DRL controller to start to increase the bus departure frequency from 8:00 am, because it takes non-negligible time for buses departing after 8:00 am to reach the 8th station.

Many methods with high accuracy for passenger flow prediction have been proposed [34–36]. In this paper, we ignore the gap between the predicted passenger flow and the actual passenger flow. We assume that the passenger flow has been accurately predicted and is used to create corresponding state features, i.e., x_m^3, x_m^4, x_m^5.

4.5. Carrying capacity calculation

Both passengers' demand and the carrying ability of the bus can be measured by the carrying capacity. A method to calculate the carrying capacity was proposed in [8]. It defined the carrying capacity demanded by passengers as the total traveling distance of all onboarding passengers. Meanwhile, the carrying capacity that provided by a bus is defined as the product of the capacity and length of the bus line [8]. The calculation is based on the assumption that all passengers waiting at a station can board the first bus passing the station, without considering the number of stranded passengers. However, in practice, if passengers are frequently stranded, this will lead to an extremely poor transit service quality [37].

In this paper, we limit the capacity of a bus. The boarding and alighting process must be considered when calculating the number of stranded passengers at each station. Besides, the distance between two stations is proportional to the number of stations between them. Thus, to simplify the calculation of the carrying capacity, we use the number of stations to replace the distance.

In summary, we propose a new method to calculate the carrying capacity in this paper. Particularly, the carrying ability supplied by a bus trip (i.e., a departure time point in bus timetable), e_m, is calculated by

$$e_m = \alpha \times C \times (K-1) \tag{13}$$

where C is the number of seats on a bus. In addition to seats, standing areas can also be used to carry passengers. α is a relaxation coefficient, which indicates that the bus capacity is α times the number of seats on the bus. Generally, α is set to 1.5 [8]. Moreover, the carrying capability supplied by a time period $[t_a, t_b]$ in the departure timetable can be calculated as below.

$$\sum_{\{t_a \le m < t_b\}} e_m \tag{14}$$

If the mth minute is a departure time point in the bus timetable, o_m denotes the carrying capacity actually consumed by passengers taking the bus departing at the mth minute, which is defined as the total riding stations of all passengers. The definition of o_m is shown in Eq. (15)

$$o_m = \sum_{k=1}^{K-1} [(c_m^k + l_m^k - h_m^k) \times 1] = \sum_{k=1}^{K-1} (c_m^k + l_m^k - h_m^k) \tag{15}$$

where c_m^k is the number of passengers onboard when the bus departing at the mth minute arrivals at the kth station. c_m^k must be no greater than C_{max}. l_m^k and h_m^k denote respectively the number of passengers boarding and alighting the bus departing at the mth minute at the kth station. Similarly, the carrying capability consumed during the time period $[t_a, t_b]$ is computed by Eq. (16).

$$\sum_{\{t_a \le m < t_b\}} o_m \tag{16}$$

5. Experimental results

5.1. Real-world dataset

We select the card swiping records of a day in June, 2018 for three bus lines 2, 230 and 239 in Xiamen city, China (all the data are available at: https://blog.csdn.net/qq_24791311?spm=1000.2115.3001.5343). The information regarding those bus lines and the number of card swiping records can be found in Table 1. Each record contains information regarding the passengers' label, boarding time, boarding station and alighting station.

The number of bus seats is 31 for the three lines. We set the relaxation coefficient $\alpha = 1.5$. For the passengers boarding on the same bus at the same station, we assume that their arrival time is uniformly distributed. Thus, the arrival time of each passenger can be estimated via the boarding time and the time when the previous bus left the station, and is added in the record.

5.2. Comparison algorithms

DRL-TO is trained through direct interaction with a simulation environment. The DRL-TO makes decisions in the "real-time" simulation environment to generate a bus timetable in an offline manner. The generated timetable is compared against the timetables generated by a genetic algorithm (GA) and a memetic

Table 1
Information of bus lines.

Lines	Direction	Service time	Stations	Records
2	Upward	6:20–22:00	37	4968
	Downward	6:30–22:00	36	4515
230	Upward	6:00–22:00	33	7739
	Downward	6:45–22:00	33	6818
239	Upward	6:00–23.05	35	5896
	Downward	6:40–23:40	36	5802

Table 2
Parameters of DQN.

Parameters	Values
Learning rate	0.01
Initial weight value	Normal Initialization
Activation function	ReLU
Number of hidden layers	10
Number of hidden units	300
Batch size	32
Discount rate	0.4
Experience memory size	3000

algorithm (BTOA-MA) [8] as well as the manually designed bus timetables (Manual method). Note that all the competing methods generate timetables based on historical data.

BTOA-MA was proposed in [8], which is a state-of-the-art algorithm for bus timetable optimization, and GA [38,39] is the classical and most widely used method to optimize the bus timetable [5,6]. GA in this paper is constructed by removing the local search technique from BTOA-MA. The manually designed timetable is made by engineers according experience, and is currently used by the three lines in practice.

As GA, BOTA-MA and the Manual method can only generate timetables based on historical passenger flow, the comparison with these methods is only performed for the offline learning stage. Due to lack of existing studies that optimize the bus timetable in real-time manner, we compare the performance of DRL-TO before and after the changing the passenger flow.

5.3. Experimental settings

We build a bus operation simulation environment (all the codes are available at: https://blog.csdn.net/qq_24791311?spm=1000.2115.3001.5343) to simulate the real-world operation of a bus system. Passengers' card swiping records are used to reproduce the real-time passenger flow in the simulation environment. DRL-TO is trained by offline learning and applied in online manner. The training and application of DRL-TO are performed in this environment.

For BTOA-MA and GA [8]: the number of individuals is 50, the crossover and mutation probabilities are 0.5 and 0.002, respectively. The maximum number of generations is set to be 5000. The local search in BTOA-MA is conducted every 20 generations, and the number of local search iterations is 11. The code of GA and BTOA-MA was implemented using C++ language. The result are obtained from BTOA-MA and GA based on 30 independent runs.

The DQN network is constructed as a fully connected network [31]. Table 2 summarizes the parameters of DQN. All methods are experimented on a desktop computer, of which the CPU frequency is 3.20 GHz and the RAM capacity is 8G. DRL-TO is implemented in Python using the Pytorch library.

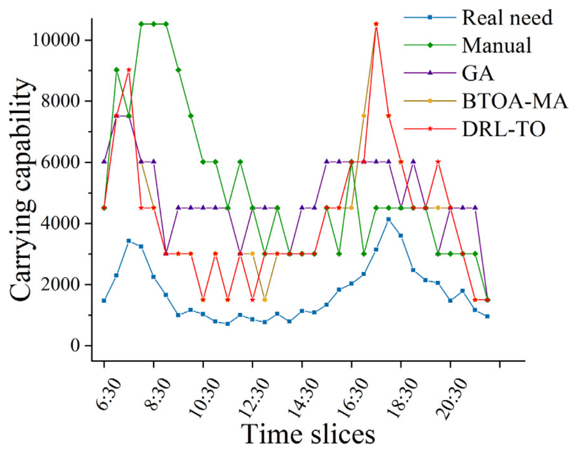


Fig. 3. Actual need of passengers on carrying capability vs. carrying capability provided by the optimized timetable of downward direction of bus line 230.

5.4. Experimental results of offline learning

Table 3 shows the performance of bus timetables obtained by GA, BTOA-MA, Manual method and DRL-TO respectively. The average waiting time (AWT), the number of departure time points in timetable (NDT) and the number of stranded passengers (NSP) are compared. The average waiting time is measured in minutes. AWT and NDT are two contradictory metrics. By changing the value of ω in Eq. (4), we create two versions of DRL-TO: DRL-TO(W) and DRL-TO(D). The former prioritizes the optimization of AWT, while the later prioritizes the optimization of NDT. Table 3 shows that, DRL-TO(W) outperforms Manual method in terms of AWT, NDT and NSP. Except for upward direction of line 2, DRL-TO(W) outperforms GA in terms of AWT, NDT and NSP. DRL-TO(D) outperforms BTOA-MA in all metrics. Overall, the timetable generated by the proposed method has less number of departure time points (less operational cost), the shortest passengers' waiting time and minimum number of stranded passengers (higher quality of service). Thus, DRL-TO can solve the problem of offline bus departure timetable optimization effectively.

Fig. 3 shows the carrying capability matching result of all algorithms. The horizontal axis is time, the vertical axis is the carrying capacity in the next half hour. The carrying capacity provided and required can be calculated by Eqs. (14) and (16). As shown in Fig. 3, the timetable generated by DRL-TO can satisfy the passengers' demand using less carrying capability. As shown in Fig. 4, DRL-TO has the fastest change in departure interval, suggesting that it is more sensitive to changes of passenger flow compared to other methods.

There are two reasons why DRL-TO can outperform GA and BOTA-MA.

(1) GA and BOTA-MA assume that all passengers waiting at any station can board on the first passing bus. However, due to the capacity restriction in practice, the number of passengers onboarding any bus is restricted by the remaining capacity of the bus. Therefore, the number of stranded passengers caused by GA and BOTA-MA are higher than DRL-TO. Meanwhile, the stranded passengers must wait extra time for another bus, resulting in extended waiting time for some passengers.

(2) The departure intervals generated by GA and BOTA-MA are largely uniform. Hence, the departure intervals do not match well with the actual passenger flow that varies significantly at different time of a day. Specifically, during off-peak hours when the passenger flow is low, GA and BOTA-MA may dispatch buses at higher frequency than necessary, resulting in wastage of carrying

Table 3
Experimental results of DRL-TO, BTOA-MA, GA, and the Manual timetable.

Lines		2		230		239	
Direction		Up	Down	Up	Down	Up	Down
Manual	NDT	72	72	104	110	119	109
	AWT(m)	6.2	7.0	5.2	7.6	4.0	10.6
	NSP	0	0	24	611	0	1635
GA	NDT	78	72	99	101	117	110
	AWT(m)	5.6	6.7	4.8	6.4	4.0	8.4
	NSP	0	0	14	370	0	1048
BTOA-MA	NDT	66	66	86	92	107	103
	AWT(m)	7.3	7.4	6.2	7.3	4.6	9.7
	NSP	0	20	50	574	0	1207
DRL-TO (D)	NDT	65	65	80	90	101	94
	AWT(m)	6.3	6.8	5.6	6.0	4.3	6.0
	NSP	0	0	0	21	0	77
	ω	1/4000	1/3000	1/4000	1/4000	1/1000	1/1500
DRL-TO (W)	NDT	71	70	88	99	111	107
	AWT(m)	5.8	6.1	4.7	5.1	3.8	5.2
	NSP	0	0	0	3	0	77
	ω	1/3000	1/1500	1/2000	1/2000	1/500	1/1000

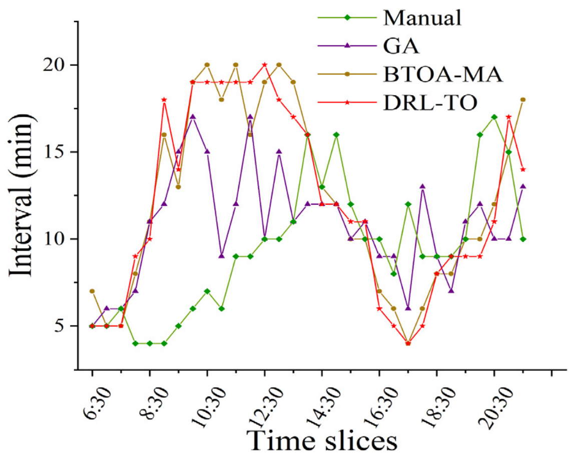


Fig. 4. The comparison of average interval in each half hour of the departure timetable generated by DRL-TO, GA, BTOA-MA and Manual method.

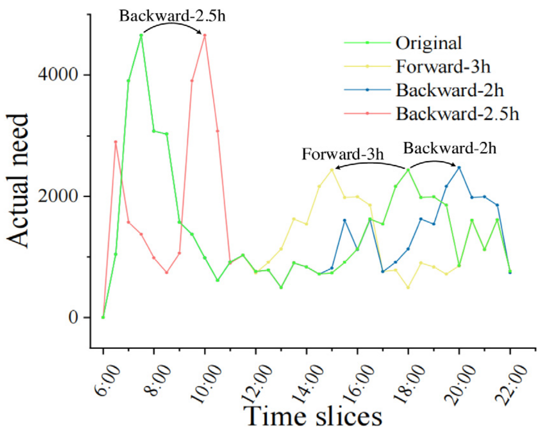


Fig. 5. Different time shifts on the peak of passenger flow for the upward direction of bus line 230 in time dimension.

capacity. On the other hand, during peak hours, GA and BOTA-MA may dispatch buses at lower frequency than requested, leading to stranded passengers and prolonged passenger waiting time. In our experiments, passengers at the 1st, 7th, 8th and 15th stations in the downward direction of line 239 accounted for 31.8% of the total number of passengers of the day. During the morning and afternoon peak hours, BOTA-MA and GA produced a high number of stranded passengers. The waiting time for passengers is also significantly long.

5.5. Experimental results of online application

In the online application stage, the trained DRL-TO is used to optimize the bus timetable in real-time parallel to the bus system. That is, DRL-TO makes decision in real-time to determine whether a bus should depart for each minute according to the real-time passenger flow. In this paper, we mimic the real-time real-world bus system in the simulation environment, where the real-time change to the passenger flow is mimicked by changing the historical passenger flow. We make two modifications to historical passenger flow to mimic sudden changes of passenger flows: (1) moving the peak of passenger flow forward or backward; and (2) reducing or increasing the peak passenger flow.

Note that whenever the passenger flow peak is shifted (or the peak passenger flow is increased and decreased), the environment for online application is different from the training environment.

5.5.1. Mimic sudden changes of passenger flow

For example, due to a new weather forecast on a upcoming heavy rain, students may go home earlier in afternoon. In this case, the passenger flow peak will move forward. To simulate the changes in passenger flow, we move the morning peak backward for 0.5, 1.5, and 2.5 hours, respectively, and move the evening peak forward (backward) for 1, 2 and 3 (1 and 2) hours, respectively. Fig. 5 shows the case that the passenger flow peak moves backward for 2.5 h in the morning and backward (forward) for 2 (3) hours in the afternoon.

If an event is held at an exhibition center, the passenger flow of the line passing the exhibition center will increase. To simulate the changes of the passenger flow like this, we decrease or increase the historical peak passenger flow by a passenger swiping records selection method (RSM).

Table 4
Experimental results of DRL-TO with the variable moving of peak passenger flow of upward direction of bus line 230.

	B(M) 0.5 h	B(M) 1.5 h	B(M) 2.5 h	B(E) 1 h	B(E) 2 h	F(E) 1 h	F(E) 2 h	F(E) 3 h
NDT	86	86	84	81	86	94	87	90
AWT	5.1	5.0	5.1	5.6	5.0	4.5	5.1	4.7
NSP	0	0	0	0	0	0	0	0

Note: “B”: backward. “F”: forward. “M”:morning. “E”:evening

Table 5
Experimental results of DRL-TO with variable peak passenger flow of bus line 2 upward direction.

Selection ratio	50%	90%	130%	170%	210%	250%	290%
NDT	58	63	65	69	73	75	77
AWT	8.1	7.0	6.6	6.2	5.8	5.7	5.5
NSP	0	0	0	0	0	0	0

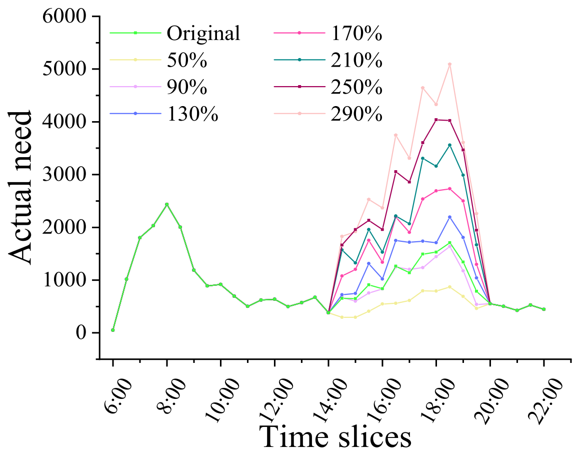


Fig. 6. The increase or decrease of carrying capacity of bus line 2 upward direction in the afternoon.

The operation process of RSM is as follows. For example, to change the passenger flow of the upward direction of line 2 during the period between 14:00 pm and 20:00 pm, we consider all the card swiping records in this period as a pool. Each time we randomly select a passenger card swiping record from the pool until the number of selected records reaches certain percent (selection ratio) of the total number of records during this period. Note that a record may be selected for more than one time. The selection ratio greater (less) than 1 is used to increase (decrease) the passenger flow.

The selection ratio is chosen as 50%, 90%, 130%, 170%, 210%, 250% and 290%, respectively, and the corresponding passenger flow is shown in Fig. 6 (all the data are available at: https://blog.csdn.net/qq24791311?spm365=1000.2115.3001.5343).

5.5.2. Performance of DRL-TO in online application

We study the ability of DRL-TO to cope with passenger flow changes. As shown in Fig. 7 (a) and (b), we swap the passenger flow in 12:00–15:00 pm and 15:00–20:00 pm in line 230 upward direction, to move the passenger flow peak forward for about three hours. From the comparison between Fig. 7 (c) and (d), we can see that when the passengers’ demand changes, DRL-TO can respond quickly and effectively to match the changed passenger flow. This means that the temporal changes of passenger flow will not impact the performance of the bus timetable generated by DRL-TO, which is also clearly evidenced by the experimental results reported in Table 4. When the peak of passenger flow moved, NDP, AWT and NSP of DRL-TO are not changed significantly. This shows that DRL-TO can make instantaneous adjustments to maintain transit service quality.

Fig. 8 and Table 5 show the ability of DRL-TO to handle both the increasing and decreasing passenger flow. As shown in Fig. 8 (a) and (b), we decreased the passenger flow of 14:00–20:00 pm to 50% of original. From the comparison of Fig. 8 (c) and (d), it can be seen that when the actually needed carrying capacity is reduced, DRL-TO can adjust the departure interval in real-time to match the changed passenger flow. Table 5 shows with the increase of passenger flow during 14:00–20:00 pm, the number of departures in the timetable generated by DRL-TO increases, resulting in the reduction of the average waiting time. That means that the bus timetable generated by DRL-TO can meet the passengers’ carrying capacity demand. This proves that DRL-TO can dynamically adjust the departure timetable according to the changes of passenger flow.

5.6. Performance analysis of DRL-TO

In this section, we will first analyze the impact of passenger flow prediction accuracy on the performance of DRL-TO. Next, the sensitivity of some hyper-parameters is discussed. Finally, ablation experiments are conducted.

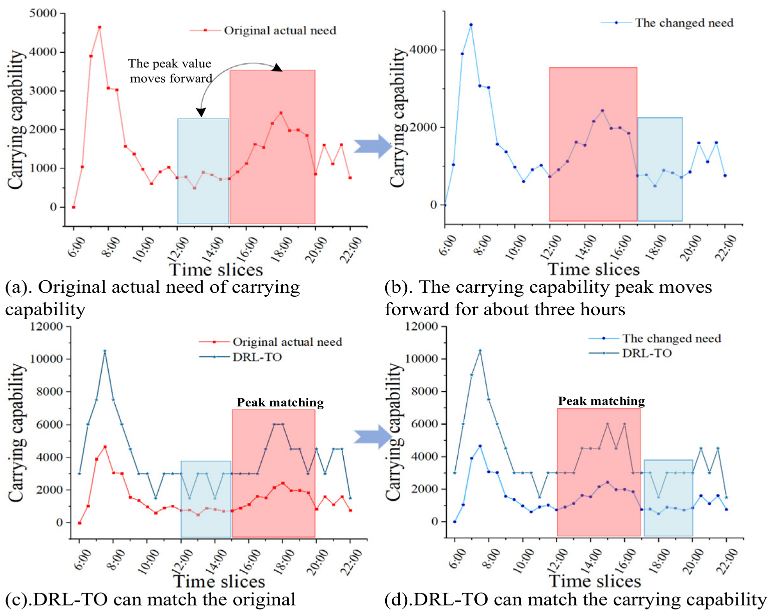
5.6.1. Effect of inaccurate prediction on the performance of DRL-TO

Some state features (i.e., x_m^3, x_m^4, and x_m^5) introduced in Section 4.4 depends on the predicted passenger flow. The prediction may be inaccurate, resulting in a decrease in the performance of DRL-TO. To verify the ability of DRL-TO to deal with inaccurate prediction, we add random prediction error to the input of DRL-TO to create a gap between the predicted passenger flow and the actual passenger flow (i.e., actual passenger flow = predicted passenger flow + random prediction error). The performances of DRL-TO on the predicted and actual passenger flow are observed.

$\mathcal{A}$. Mimic inaccurate prediction

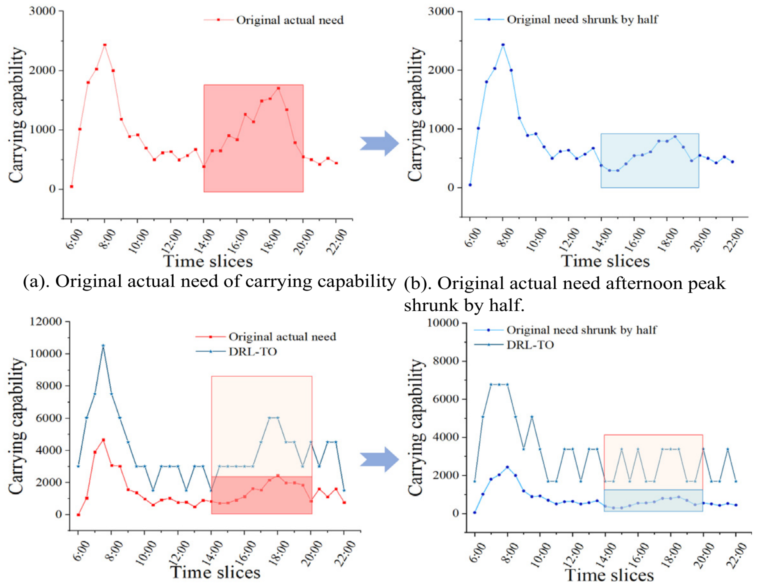
The passenger flow input to DRL-TO is the predicted passenger flow. To create a gap between the predicted passenger flow and the actual passenger flow, we simulate the actual passenger flow in two ways. Many existing studies showed that the accuracy of the existing short-term passenger flow predicting methods is generally 80%–95% [35,36,40,41]. In line with these findings, we create a gap between the predicted passenger flow and the actual passenger flow based on the accuracy.

a. If the predicted passenger flow is higher than actual passenger flow, the performance may be affected due to unnecessary operational cost. We randomly select a proportion of passengers by RSM. The selection ratios are set to 95%, 90%, 85%, and



(a). Original actual need of carrying capability
(b). The carrying capability peak moves forward for about three hours
(c).DRL-TO can match the original carrying capacity need.
(d).DRL-TO can match the carrying capability when peak hours move forward.

Fig. 7. The performance of DRL-TO before and after the forward movement of peak passenger flow.



(a). Original actual need of carrying capability
(b). Original actual need afternoon peak shrunk by half.
(c).DRL-TO can match the original carrying capacity need.
(d).The capacity provided by DRL-TO can change with the change of actual need.

Fig. 8. The performance of DRL-TO before and after the reduction of peak passenger flow.

80%, respectively. Thus, the gaps between actual and predicted passenger flows are −5%, −10%, −15%, and −20%, respectively. Here "−" means that the actual passenger flow is lower than the predicted one.

b. If the predicted passenger flow is lower than actual passenger flow, the performance may be affected due to unexpected passenger flow. We randomly select a proportion of passengers by RSM. The selection ratios are set to 105%, 110%, 115%, and 120%, respectively. Thus, the gaps between actual and predicted passenger flows are +5%, +10%, +15%, and +20%, respectively. Here "+" means that the actual passenger flow is higher than the predicted one.

The experiments are conducted in the downward direction of line 2. 10 passenger flows are generated independently for each selection ratio. Figs. 9 and 10 show the examples that the actual passenger flow is lower and higher than the predicted one, respectively.

B. Performance of DRL-TO under inaccurate prediction

To observe the performance of DRL-TO when the prediction is inaccurate, two versions of DRL-TO (i.e., DRL-TO(D) and DRL-TO(W)) are tested. DRL-TO(D) and DRL-TO(W) generate bus departure timetables based on the predicted passenger flows. The timetables generated by DRL-TO(D) and DRL-TO(W) have 65 and 70 departure times, respectively. The average, minimum and maximum values of AWT for 10 independent actual passenger flows of each selection ratio are reported in Tables 6 and 7.

As shown in Table 6, as the prediction error increases (i.e., the prediction accuracy decreases), the AWT of DRL-TO(D) does not change significantly. That is because the actual required carrying capacity is lower than that provided by the bus timetable. In

Table 6
The performance of DRL-TO when the predicted passenger flow is higher than the actual passenger flow.

Prediction error		0	−5%	−10%	−15%	−20%
DRL-TO (D)	AWT (average)	6.89	6.85	6.85	6.87	6.85
	AWT (minimum)	–	6.82	6.82	6.79	6.82
	AWT (maximum)	–	6.88	6.91	6.93	6.89
DRL-TO (W)	AWT (average)	6.13	6.09	6.09	6.10	6.08
	AWT (minimum)	–	6.07	6.02	6.05	6.05
	AWT (maximum)	–	6.12	6.16	6.13	6.17

Table 7
The performance of DRL-TO when the predicted passenger flow is lower than the actual passenger flow.

Prediction error		0	+5%	+10%	+15%	+20%
DRL-TO (D)	AWT (average)	6.89	6.87	6.85	6.87	6.85
	AWT (minimum)	–	6.83	6.80	6.82	6.79
	AWT (maximum)	–	6.90	6.89	6.92	6.94
DRL-TO (W)	AWT (average)	6.13	6.11	6.11	6.11	6.09
	AWT (minimum)	–	6.07	6.08	6.08	6.02
	AWT (maximum)	–	6.14	6.15	6.16	6.12

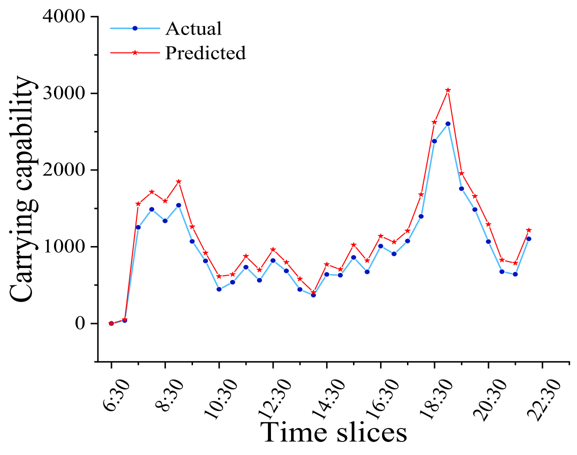


Fig. 9. Predicted passenger flow is higher than actual passenger flow by 15%.

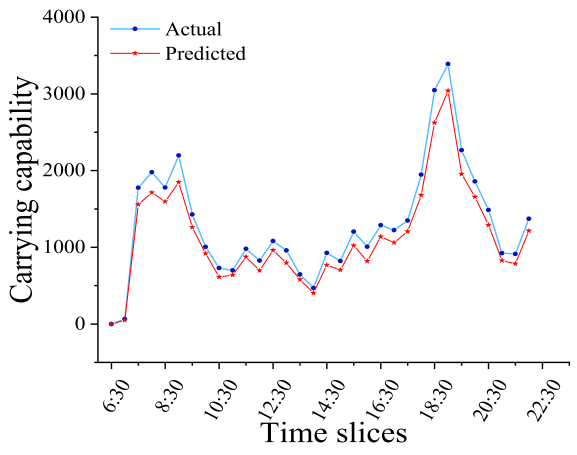


Fig. 10. Predicted passenger flow is lower than actual passenger flow by 15%.

this case, the service quality does not decrease. However, some carrying capacity provided by the bus timetable may be wasted. This is unavoidable to maintain a high level of service quality in practice.

As shown in Table 7, as the actual passenger flow increases, the performance of DRL-TO does not degrade significantly. The maximum value of AWT is only 0.7% higher than the case without random prediction error. This is because buses are not fully loaded and there is extra carrying capacity to serve new passengers. For that reason, small increase in passenger flow usually does not lead to significant increase in AWT. This can also be verified from the gap between the actual demand and the carrying capacity provided by DRL-TO in Fig. 3.

The results above show that DRL-TO can generate desirable departure timetables, even when the passenger flow prediction is inaccurate.

5.6.2. Hyper-parameter sensitivity analysis

In this section we will discuss some important hyperparameters. ω is the penalty factor for waiting time in the reward function (see Eq. (4)), which determines the preference of DRL-TO between the interests of passengers and the interests of the bus company. It is necessary to evaluate the influence of ω experimentally.

For this purpose, we conduct experiments on the downward direction of line 2. Other parameters are fixed as explained in Section 5.3 while ω is varied, and the changes in AWT and NDT are monitored. As shown in Table 8 and Fig. 11, increasing ω can increase NDT and decrease AWT. If bus company only cares about the cost, $\omega = 0$ is the best choice. In other cases, ω can be selected to meet the desirable trade-off between the cost and service quality.

In our experiments, we found that the discount factor γ has a significant influence on the convergence stability of DRL-TO. Thus, we study the relationship between stability and the discount factor γ. The standard deviation of NDT across 25 independent episodes is used to measure the stability of DRL-TO. As we can see in Fig. 12, $\gamma = 0.4$ corresponds to the smallest standard deviation, that is, when $\gamma = 0.4$, DRL-TO is the most stable. Thus, in other experiments, we set γ to be 0.4 consistently.

5.6.3. Ablation experiment

The state space in Section 4.4 and reward function in Section 4.3 are carefully designed in this paper. To prove their effectiveness, another two design schemes of state space and reward function are compared with DRL-TO.

Table 8
Experimental results of NDT and AWT with varied values of ω.

ω	NDT			AWT		
	Max	Min	Mode	Max	Min	Average
1/500	107	98	102	4.9	4.0	4.1
1/1000	92	84	86	5.6	4.6	4.9
1/1500	82	75	79	6.2	5.9	5.4
1/2000	78	70	74	7.2	5.5	5.9
1/3000	73	68	70	6.7	6.0	6.2
1/4000	72	65	67	6.8	6.1	6.4
1/5000	72	62	67	8.0	6.2	6.7
1/7000	68	61	65	7.7	6.5	6.9
1/9000	68	62	64	7.2	6.5	6.8
1/11000	69	61	64	7.7	6.7	7.0
0	62	59	60	8.0	7.1	7.6

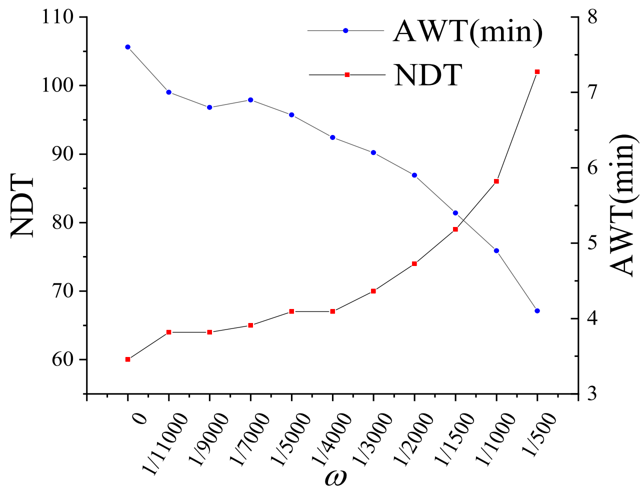


Fig. 11. The AWT and NDT with varied values of ω.

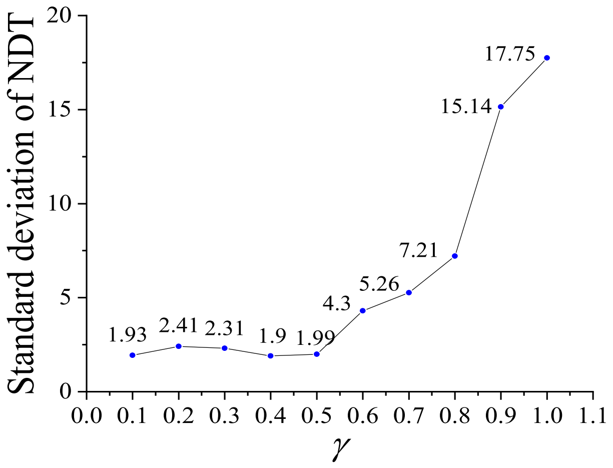


Fig. 12. The stability of DRL-TO with varied values of γ.

$\mathcal{A}$. Scheme one

State space: We use the real-time information of passengers, buses and stations to compose the state s_e, which is defined as below.

$$s_e = [s_e^{m-3}, s_e^{m-2}, s_e^{m-1}, s_e^{m}] \tag{17}$$

where each element (i.e., $s_e^{m-3}, s_e^{m-2}, s_e^{m-1}, s_e^{m}$) of s_e consists of four vectors, the length of which are all $(K-1)$. Vectors at the same position in the elements represent the information of same object (i.e., the passengers, buses or stations) at different time. For example, $s_e^m = [s_e^{m,1}, s_e^{m,2}, s_e^{m,3}, s_e^{m,4}]$. The kth dimension of $s_e^{m,1}$ denotes the number of passengers waiting at the kth station at the mth minute. The kth dimension of $s_e^{m,2}$ denotes the number of passengers who are expected to arrive at the kth station in the next quarter. The kth dimension of $s_e^{m,3}$ indicates whether there is a bus running between the kth station and the $(k+1)$th station. The kth dimension of $s_e^{m,3}$ is 1 if so; 0 otherwise. The kth dimension of $s_e^{m,4}$ denotes the traveling distance from the kth station, if the kth dimension of $s_e^{m,3}$ is equal to 1. If there are multiple buses between the kth station and the $(k+1)$th station, the closest one to the $(k+1)$th station is selected.

Reward function: For the problem of bus timetable optimization, to reflect the ideal situation that the buses running on the road are fully loaded and the number of passengers waiting at each station is 0, the following reward function is used.

$$r_m = -(C_m^l + \sum_{k=1}^{K} d_m^k) \tag{18}$$

where r_m is the immediate reward the DQN agent obtains at the mth minutes. C_m^l represents the total number of remaining seats on the bus running on the line at the mth minute, and d_m^k is the number of passengers waiting for the bus at the kth station at the mth minute.

$\mathcal{B}$. Scheme two

State space: In Scheme one, when the action is taken by the DQN agent, only the first dimension of vectors in $s_e^{m,1}, s_e^{m,2}, s_e^{m,3}$ and $s_e^{m,4}$ can be changed by the action directly. Other dimensions of the vectors are changed by the buses running on line and dynamic passenger flow. There is no strong correlation between actions and state changes.

In view of the above, a new state is designed. First, we add the time mark of bus system, i.e., t_h refers to the hourly period of the current time. Secondly, the cross-section passenger flow without limited capacity of buses is added. In addition, the interval between the current time and the previous departure time is also added. Therefore, the new state $s_{e'}^m$ is defined as below.

$$s_{e'}^m = [t_h, \max_i p_m(i), t_m^l] \tag{19}$$

where t_m^l represents the interval between the current minute (the mth minute) and the last departure point. Assume that there is a bus departs at current minute, $p_m(i)$ denotes the cross-section passenger flow without limited capacity of buses, which is calculated as below.

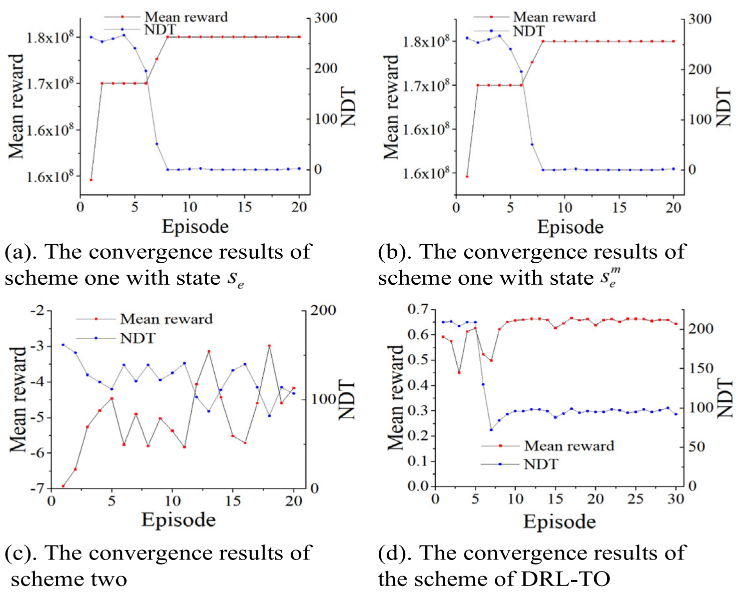
$$p_m(i) = \sum_{k=1}^{i} (l_m^k - h_m^k) \tag{20}$$

where i represents the ith station, l_m^k and h_m^k denote respectively the number of passengers boarding and alighting the bus departing at the mth minute at the kth station.

Reward function: We design a reward function as below.

$$r_m = -\left(\sum_{n\in N} C_{max} \times z_n - \sum_{n\in N}\sum_{k=1}^{z_n} (c_n^k + l_n^k - h_n^k)\right) - \sum_{k=1}^{K} w_m^k \tag{21}$$

where r_m is the immediate reward the DQN agent obtains at the mth minutes. N represents the set of buses running online at the mth minute. z_n represents No. of the station that the nth running bus just passes. For example, if the 2th bus is running between the 3th station and the 4th station, $z_2 = 3$. C_{max} is the capacity of a bus. l_n^k and h_n^k denote respectively the number of passengers boarding and alighting the nth bus in N at the kth station. c_n^k is the number of passengers onboard when the nth bus arrivals at the



(a). The convergence results of scheme one with state s_e

(b). The convergence results of scheme one with state s_e^m

(c). The convergence results of scheme two

(d). The convergence results of the scheme of DRL-TO

Fig. 13. Convergence comparison of different schemes of state and reward function.

kth station. $\sum_{k=1}^{K} w_m^k$ is the total number of passengers waiting at all stations at the mth minute.

C. Results discussion

As shown in Fig. 13, the average reward and NDT of per episode of DQN agent are shown in the Fig. 13 (a) and (b). The state and reward function follow scheme one. The results in Fig. 13 (a) is obtained based on the state s_e, while the results in Fig. 13 (b) is obtained by using the state s_e^m. For the same reward function, the number of departure time points reported in Fig. 13 (a) and (b) all converge to 0. A possible reason is because there is no strong correlation between actions and state changes. As we can see in Fig. 13 (c), DQN agent can converge based on the state and reward function designed in scheme two. However, the performance of DQN is not stable. The reason for this situation may be that we did not limit the capacity of buses, making feature $\max_i p_m(i)$ unbounded. Our experiment results confirm that only the state space and reward function proposed in Section 4.4 and Section 4.3 can enable DQN to achieve reliable and effective bus timetabling in both offline and online settings. Our proposed design of state space and reward function can provide effective feedback information for the DQN agent, as clearly evidenced by the convergence curves regarding the number of departure time points and average reward in Fig. 13 (d).

6. Conclusion

In order to meet the demand of suddenly changed passenger flows, this paper proposes a dynamic generation method of bus timetable based on deep reinforcement learning. We regard the bus timetable optimization problem as a sequential decision problem, and use reinforcement learning to determine whether to depart at the current time, thus to determine the departure interval in real time. We select the current time, load factor, carrying capacity utilization rate, and the number of stranded passengers as the states of the reinforcement learning, and "departure" or "no departure" as the actions. Taking into account the interests of both the bus company and passengers, a reward function is designed, which includes the metrics of full load rate, empty load rate, waiting time, and the number of stranded passengers.

A series of experiments on real datasets demonstrate that the method proposed in this paper is effective. DRL-TO can generate bus departure timetable dynamically based on real-time passenger flow, meanwhile the number of departure time points in bus timetable is reduced and passengers' waiting time is shorten.

CRediT authorship contribution statement

Guanqun Ai: Conceptualization, Software, Methodology, Validation, Writing – original draft. **Xingquan Zuo:** Conceptualization, Methodology, Writing – review & editing, Supervision, Funding acquisition. **Gang Chen:** Methodology, Investigation, Writing – review & editing. **Binglin Wu:** Investigation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have given the link to the data in this paper.

Acknowledgment

This work was supported by National Natural Science Foundation of China under Grant 61873040.

References

[1] Z. Qian, W. Fenglian, L. Ju, A bus headway optimization model based on genetic taboo algorithm, Transp. Sci. Eng. 31 (2) (2015) 81–86, http://dx.doi.org/10.16544/j.cnki.cn43-1494/u.2015.02.016.
[2] C. Wang, X. Zuo, Solving multi-depot electric vehicle scheduling problem by column generation and genetic algorithm, Appl. Soft Comput. 112 (2021) 107774(1–14), http://dx.doi.org/10.1016/j.asoc.2021.107774.
[3] J. Tang, Y. Yang, W. Hao, F. Liu, Y. Wang, A data–driven timetable optimization of urban bus line based on multi-objective genetic algorithm, IEEE Trans. Intell. Transp. Syst. 22 (4) (2021) 2417–2429, http://dx.doi.org/10.1109/TITS.2020.3025031.
[4] L. Shen, Y. Han, X. Jiang, Study on method of bus service frequency optimal modelbased on genetic algorithm, Procedia Environ. Sci. 10 (2011) 869–874, http://dx.doi.org/10.1016/j.proenv.2011.09.139.
[5] F.D. Wihartiko, A. Buono, B.P. Silalahi, Integer programming model for optimizing bus timetable using genetic algorithm, IOP Conf. Ser. Mater. Sci. Eng. 166 (1) (2017) 1–9, http://dx.doi.org/10.1088/1757-899x/166/1/012016.
[6] K. Gkiotsalitis, F. Alesiani, Robust timetable optimization for bus lines subject to resource and regulatory constraints, Transp. Res. Part E: Logist. Transp. Rev. 128 (2019) 30–51, http://dx.doi.org/10.1016/j.tre.2019.05.016.

[7] B. Yu, Z. Yang, X. Sun, B. Yao, Q. Zeng, E. Jeppesen, Parallel genetic algorithm in bus route headway optimization, Appl. Soft Comput. 11 (8) (2011) 5081–5091, http://dx.doi.org/10.1016/j.asoc.2011.05.051.

[8] P. Gao, X. Zuo, Q. Bian, X. Zhao, A memetic algorithm to optimize bus timetable with unequal time intervals, Proc. Genet. Evolut. Comput. Conf. Companion (2019) 1336–1344, http://dx.doi.org/10.1145/3319619.3326844.

[9] C. Avishai, H. Stephan, D. Blanche, Approaching even-load and even-headway transit timetables using different bus sizes, Public Transp. 5 (3) (2013) 193–217, http://dx.doi.org/10.1007/s12469-013-0062-z.

[10] H. Dong, J. Kong, Q. Liu, A bus departure time interval transition model considering traffic congestion, J. Transp. Syst. Eng. Inf. Technol. 16 (3) (2016) 101–106, http://dx.doi.org/10.16097/j.cnki.1009-6744.2016.03.015.

[11] H.Y. Shang, H.J. Huang, W.X. Wu, Bus timetabling considering passenger satisfaction: An empirical study in beijing, Comput. Ind. Eng. 135 (2019) 1155–1166, http://dx.doi.org/10.1016/j.cie.2019.01.057.

[12] L. Tiljari, T. Cari, B. Abramovi, T. Fratrovi, Traffic state estimation and classification on citywide scale using speed transition matrices, Sustainability 12 (18) (2020) 72–78, http://dx.doi.org/10.3390/su12187278.

[13] D.J. Sun, Y. Xu, Z.R. Peng, Technology, Timetable optimization for single bus line based on hybrid vehicle size model, J. Traffic Transp. Eng. (English Edition) 3 (2015) 179–186, http://dx.doi.org/10.1016/j.jtte.2015.03.006.

[14] J. Lu, Z. Yang, H. Timmermans, W. Wang, Optimization of airport bus timetable in cultivation period considering passenger dynamic airport choice under conditions of uncertainty, Transp. Res. C 67 (2016) 15–30, http://dx.doi.org/10.1016/j.trc.2016.01.012.

[15] X. Li, H. Du, H. Ma, C. Shang, Timetable optimization for single bus line involving fuzzy travel time, Soft Comput. 22 (21) (2018) 6981–6994, http://dx.doi.org/10.1007/s00500-018-3266-y.

[16] J. Tang, Y. Yang, Y. Qi, A hybrid algorithm for urban transit schedule optimization, Physica A 512 (2018) 745–755, http://dx.doi.org/10.1016/j.physa.2018.08.017.

[17] D.L. van Oudheusden, W. Zhu, Trip frequency scheduling for bus route management in bangkok, European J. Oper. Res. 83 (3) (1995) 439–451, http://dx.doi.org/10.1016/0377-2217(94)00362-G.

[18] W. Sun, X. Wang, G. Qiao, Study on departing time interval control of bus dispatching, in: 2007 International Conference on Service Systems and Service Management, 2007, pp. 1–5, http://dx.doi.org/10.1109/ICSSSM.2007.4280138.

[19] A. Ceder, Bus frequency determination using passenger count data, Transp. Res. Part A: General 18 (5) (1984) 439–453, http://dx.doi.org/10.1016/0191-2607(84)90019-0.

[20] J. Li, J. Hu, Y. Zhang, Optimal combinations and variable departure intervals for micro bus system, Tsinghua Sci. Technol. 22 (3) (2017) 282–292, http://dx.doi.org/10.23919/TST.2017.7914200.

[21] H. Elbaz, A. Elhilali Alaoui, G. Bencheikh, The synchronization bus timetabling problem, modeling and resolution by the multi-agent approach, in: 2018 4th International Conference on Logistics Operations Management, GOL, 2018, pp. 1–6, http://dx.doi.org/10.1109/GOL.2018.8378098.

[22] G.P. Matos, L.M. Albino, R.L. Saldanha, E.M. Morgado, Solving periodic timetabling problems with sat and machine learning, Public Transp. (2020) 1–24, http://dx.doi.org/10.1007/s12469-020-00244-y.

[23] J. Wang, L. Sun, Dynamic holding control to avoid bus bunching a multi agent deep reinforcement learning framework, Transp. Res. C 116 (2020) 102661(1–12), http://dx.doi.org/10.1016/j.trc.2020.102661.

[24] A. Darwish, M. Khalil, K. Badawi, Optimising public bus transit networks using deep reinforcement learning, in: 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 2020, pp. 1–7, http://dx.doi.org/10.1109/ITSC45102.2020.9294710.

[25] L. Zou, J. Xu, L. Zhu, Light rail intelligent dispatching system based on reinforcement learning, in: 2006 International Conference on Machine Learning and Cybernetics, 2006, pp. 2493–2496, http://dx.doi.org/10.1109/ICMLC.2006.258785.

[26] L. Ning, Y. Li, M. Zhou, H. Song, H. Dong, A deep reinforcement learning approach to high-speed train timetable rescheduling under disturbances, in: 2019 IEEE Intelligent Transportation Systems Conference, ITSC, 2019, pp. 3469–3474, http://dx.doi.org/10.1109/ITSC.2019.8917180.

[27] G. Yang, F. Zhang, C. Gong, S. Zhang, Application of a deep deterministic policy gradient algorithm for energy-aimed timetable rescheduling problem, Energies 12 (18) (2019) 3461(1–19), http://dx.doi.org/10.3390/en12183461.

[28] Y. Guo, A reinforcement learning approach to train timetabling for inter-city high speed railway lines, in: 2020 IEEE 5th International Conference on Intelligent Transportation Engineering, ICITE, 2020, pp. 493–497, http://dx.doi.org/10.1109/ICITE50838.2020.9231418.

[29] R. Wang, M. Zhou, Y. Li, Q. Zhang, H. Dong, A timetable rescheduling approach for railway based on monte carlo tree search, in: 23rd IEEE Intelligent Transportation Systems Conference, 2019, pp. 3738–3743, http://dx.doi.org/10.1109/ITSC.2019.8916963.

[30] Y. Zhu, H. Wang, R.M. Goverde, Reinforcement learning in railway timetable rescheduling, in: IEEE International Conference on Intelligent Transportation Systems, 2020, pp. 1–6, http://dx.doi.org/10.1109/ITSC45102.2020.9294188.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533, http://dx.doi.org/10.1038/nature14236.

[32] I. Osband, C. Blundell, A. Pritzel, B. Van Roy, Deep exploration via bootstrapped DQN, Adv. Neural Inf. Process. Syst. 29 (2016) 4026–4034, http://dx.doi.org/10.5555/3157382.3157548.

[33] R.A. Matthews, The science of murphy's law, Sci. Am. 276 (4) (1997) 88–91, http://dx.doi.org/10.1038/scientificamerican0497-88.

[34] J. Zhang, D. Shen, et al., A real-time passenger flow estimation and prediction method for urban bus transit systems, IEEE Trans. Intell. Transp. Syst. (11) (2017) 3168–3178, http://dx.doi.org/10.1109/TITS.2017.2686877.

[35] Y. Liu, C. Lyu, X. Liu, Z. Liu, Automatic feature engineering for bus passenger flow prediction based on modular convolutional neural network, IEEE Trans. Intell. Transp. Syst. 22 (4) (2020) 2349–2358, http://dx.doi.org/10.1109/TITS.2020.3004254.

[36] W. Wu, Y. Xia, W. Jin, et al., Predicting bus passenger flow and prioritizing influential factors using multi-source data: Scaled stacking gradient boosting decision trees, IEEE Trans. Intell. Transp. Syst. 22 (4) (2020) 2510–2523, http://dx.doi.org/10.1109/TITS.2020.3035647.

[37] V.B.E.C. Gkiotsalitis K, An analytic solution for real-time bus holding subject to vehicle capacity limits, Transp. Res. C 121 (2020) 102815., http://dx.doi.org/10.1016/j.trc.2020.102815.

[38] X. Zuo, C. Chen, W. Tan, M. Zhou, Vehicle scheduling of an urban bus line via an improved multiobjective genetic algorithm, IEEE Trans. Intell. Transp. Syst. 16 (2) (2015) 1030–1041, http://dx.doi.org/10.1109/TITS.2014.2352599.

[39] C. Wang, H. Shi, X. Zuo, A multi-objective genetic algorithm based approach for dynamical bus vehicles scheduling under traffic congestion, Swarm Evol. Comput. 54 (2020) http://dx.doi.org/10.1016/j.swevo.2020.100667, 100667(1–11).

[40] X. Li, Z. Chen, et al., Short-term bus passenger flow forecast based on deep learning, in: 2018 International Conference on Security, Pattern Analysis, and Cybernetics, 2018, pp. 372–376, http://dx.doi.org/10.1109/SPAC46244.2018.8965619.

[41] B. Du, H. Peng, S. Wang, et al., Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction, IEEE Trans. Intell. Transp. Syst. 21 (3) (2019) 972–985, http://dx.doi.org/10.1109/TITS.2019.2900481.