A deep reinforcement learning approach for solving the pickup and delivery problem with drones and time windows

Lu Fu-qiang[1], Chen Nan[1], Bi Hua-ling[1]
(1. School of Economics and Management, Yanshan University, Qinhuangdao 066004, China.)

Abstract: As modern logistics enterprises seek new ways to meet complex customer demands, Unmanned Aerial Vehicles (UAVs) have become a popular topic both in academic discussions and practical applications. Pickup and delivery problems (PDPs), which generalizes the vehicle routing problem and applicable to various practice logistic problems, have gained increasing attention. Due to the NP-hard nature of PDPs and the main challenge of using drones with limited capacity and limited flying time, how to find optimal route is a crucial issue. This paper addresses a pickup and delivery problem with drones and time windows (PDP-DTW) for minimized total distance, where energy consumption and capacity are considered. The problem is to design a set of routes for drone fleets servicing pickup and delivery nodes with defined demands and time windows according to orders. A mixed integer nonlinear programming model is established for PDP-DTW and an attention-based learning approach called Graph Attention Model is proposed to cope with its characteristic. Comparative experiments indicate that the proposed method performs better than classical heuristics and graph convolutional networks for our problems. Extensive experiments have also been conducted to confirm the applicability of the proposed model in various scenarios, including the unequal number of facilities and customers, changes in the capacity and energy consumption of drones, and the impact of time windows.

Keywords: pickup and delivery problem; drones; reinforcement learning; neural networks; attention mechanism; graph attention model

1 Introduction

As modern logistics enterprises seek new ways to meet complex customer demands, Unmanned Aerial Vehicles (UAVs) have become a popular topic both in academic discussions and practical applications. Their advantages, together with their less cost or labor than trucks, have made them flexible in state-of-the-art commercial last-mile deliveries (Aurambout J P et al., 2019). Many well-known enterprises such as Google, Walmart and Federal Aviation Administration (FAA), have begun testing their delivery by drones(Zheng K et al., 2020). While challenges in drone delivery limit its further development, trajectory optimization is worth studying through scientific means. Many scholars have shown deep interest in various extensions of VRP, such as the VRP with Time Window (VRPTW), where drones are required to visit customers within a specific period of time(Okulewicz M and Mańdziuk J, 2019). Pickup and delivery problems (PDPs), which generalize the vehicle routing problem and are applicable to various practical logistic scenarios (Ropke S, Cordeau J F, 2009), have gained increasing attention. The primary concern of PDPs is how to assign a fleet of heterogeneous vehicles to pickup and delivery requests over time, considering capacities, speed, and locations. In the medical field, PDPs meet the requirements of medical supplies delivery in public health emergencies, assuming that each facility can serve all the products ordered by the customers(Shi Y et al., 2022;Gómez-Lagos J et al.,2022). In ride-sharing, platforms like Lyft and Uber can also be considered instances of PDPs, where the loads of each node are equal to the maximum capacity of one vehicle(Toth et al.,2014). In online delivery services(e.g. UberEats and GrubHub), person-to-person trading also is a example of PDPs, requiring pickups from multiple locations and deliveries to corresponding customers(Jun S and Lee S, 2022). The utilization of drones is ideal for this type of logistics operations(Bi, H. et al.,2023), we specially for efficiently transporting small-sized and lightweight parcels. However, the main challenge of using drones in PDPs arises from limited capacity and flying time, as well as the interdependency between pickup and delivery in urban environments.

Due to the NP-hard nature of PDPs, issues such as the complexity of modeling and computation persist when attempting to solve them. For the real-world problems, algorithms that need to run repeatedly and consume excessive time when the information is updated are inappropriate(Berbeglia G et al., 2010). In recent years, some researchers have addressed various extensions of VRPs (e.g.

VRP with capacity constraints or time windows) through reinforcement learning(Nazari M et al., 2018) and neural network(James J Q et al., 2019). Deep neural network is a feasible way to directly made decision for a tour or multiple sequential tours by defining network state and reward(Qureshi A H et al., 2018). As a combinatorial optimization problem, PDPs also related to decision-making orders, which can be regarded as a sequential decision-making problem or Markov Decision Process(MDP). Our goal is to devise an end-to-end deep learning model for PDPs that can address the challenges mentioned above.

In this paper, the main contributions are summarized as follows: (1) The introduction of a pickup and delivery problem with drones and time windows (PDP-DTW) aimed at minimizing the total distance while considering time windows; (2) The establishment of a Mixed-Integer Nonlinear Programming (MINLP) model for PDP-DTW; (3) The proposal of a Graph Attention Model (GAM) designed to address the characteristics of PDP-DTW, utilizing attention mechanisms and a pointer network decoder; (4) Comparative experiments offering valuable insights into the application of PDP-DTW and demonstrating the advantages of GAM.

The remainder of the paper will proceed as follows: In Section 2, we provide a literature review related to PDP-DTW and learning methods for solving routing problems. Section 3 formally defines PDP-DTW. Section 4 presents the Graph Attention Model and details the training of our model through reinforcement learning. Section 5 showcases the main computational results for PDP-DTW and additional computational experiments. Finally, in Section 6, we conclude with final remarks and discussions.

2 Literature review

2.1 Pickup and delivery problem with drones

The Pickup and Delivery Problems (PDPs), in which each transportation request has a single origin and a single destination pair in general, are also related to the vehicle routing problem. Mitrovi´c-Mini´c et al. (2004) focuses on the dynamic Pickup and Delivery Problem with Time Windows (PDPTW), which is faced by the transport of small parcels during the whole day. Nagy G et al. (2015) study vehicle routing problem with divisible deliveries and pickups (VRPDDP), aiming to find the benefit from two separate visits of one customer. Wolfinger D et al. (2021) solved the VRPDDP by an arc-based mixed-integer formulation they proposed. Some researchers also consider the multiple distribution stations near urban residential communities and multiple trips in the case of insufficient capacity(Zhen L et al., 2022). Wang Y et al.(2021) proposed a mix of multi-depot multi-trip vehicle routing and pickup and delivery problem with time window, considering the delivery of e-commerce and O2O parcels in a last-mile system. Considering the loading cost, Xue L et al.(2016) propose a branch-and-cut algorithm and a branch-and-price algorithm to solve the pickup and delivery problem with a loading cost (PDPLC). Shi J et al.(2019) study the PDPs while operating an electric vehicle fleet to provide ride-hailing services to local residents. All the above studies derive from the classic PDP, but they not be fully applicable in the case of drones.

The core elements of PDPs follow the traditional VRP concepts, but they remain challenging in practice due to multiple operational characteristics(Tongren Yan et al.,2022;Fuqiang Lu et al.,2020). Cheng C et al.(2018) introduced a Multi-Trip Drone Routing Problem (MTDRP) that considers the influence of limited payload. Liu Y (2018) presented a model for delivering on-demand meals, considering the limited capacity of drones and splitting a single order into multiple drones according to meals. Troudi A et al.(2018) introduced a Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) that considers the influence of energy constraints and battery capacity. Dorling K et al.(2016) proposed the drone routing problems and calculated the energy consumption through a linear approximation function based on the weight of drones. Jung S and Kim H(2016) studied on the drone routing problems with time windows and trip duration. To solved a similar problem related to energy, the author used a similar linear approximation function. Cheng C et al.(2020) also studied a the drone routing problem considering payload and travel time due to battery constraints. Following the linear approximation function by Dorling K et al.(2016), the author used an exact algorithm to solve the models. As the related drone routing literature indicates, two differentiating factors exist when considering only drones. They suggest that drone efficiency benefits from low delivery payloads and energy model, unlike ground vehicles in traditional VRP(Kyriakakis N A et al., 2023;Feng Wenjing et al.,2022). Therefore, many variants of the PDP are yet to be studied, such as the one proposed in this paper, the PDP-DTW, which considers pickups and deliveries according to orders while accounting for capacity and energy constraints of drones.

2.2 Deep reinforcement learning approach

The pickup and delivery problem with drones and time windows (PDP-DTW) is a combinatorial optimization problem which belongs to NP-hard. The research streams related to VRP or PDP are summarized in Table.1. There exist two streams of researchers in solving such NP-hard problem (Duan L et al.,2020). One is based on the techniques of operations research (OR), often specializing in exact algorithm and heuristic algorithm. In the case of exact approaches, mathematical models and dynamic programming have been studied to find the optimal PDP solutions(Fuqiang Lu et al. 2023). Xue L et al.(2016) and Mahmoudi M et al.(2016) addressed the general PDP and proposed mathematical model for them. In their research, branch-and-cut algorithms and dynamic programming have been studied to find the optimal solution. However, as the number of customer increases, finding the optimal solution may take several days. In this situation, it is better to find a near-optimal solution via heuristic algorithm, although overcoming the difficulty of designing a heuristic algorithm with good performance is necessary. Murray C C et al.(2020) presented a Mixed-Integer Linear Programming (MILP) model for flying sidekick TSP. Considering a last-mile delivery system in which trucks operate in coordination with drones, they develop a three-phase heuristic solution approach. Mehlawat M K et al.(2019) designed a hybrid intelligent heuristic solution approach to computing the expected values of such fuzzy variables. In their article, a vehicle routing problem (VRP) with multiple depots to pickup and multiple customers to delivery is studied. Lu Fuqiang et al.(2022) proposed An ant colony system-improved grey wolf optimization in their study of fourth-party logistics routing problem.

Due to the computational complexity of NP-hard problems and the state-of-the-art achievements in other fields, such as natural language processing, there is a growing interest in new approaches based on machine learning (ML). Another stream of research is related to reinforcement learning (RL)(Duan L ,2020), which efficiently solves the same and similar problems using trained policies. Reinforcement learning approaches have gained popularity in solving various NP-hard problems in ride-hailing services such as Shi J et al.(2018), and in Traveling Salesman Problem (TSP) such as Bello I et al.(2016). Furthermore, an end-to-end model based on the encoder-decoder structure, trained through reinforcement learning, has become a choice for recent research. In terms of routing problems such as VRP and TSP, Nazari M et al.(2018) proposed a novel end-to-end framework for solving them using reinforcement learning. The Attention Model (AM) proposed by Vaswani A et al.(2017), with several layers of multi-head attention to find the most critical parts between inputs, outperforms than other neural network architectures (e.g. LSTM) according to Kool W et al.(2018). While focusing on routing a fleet of drones, it is a single-agent model similar to Bogyrbayeva A et al.(2023). However, at each step of prediction, the decoder of the original AM conditions only on the current location of the drones and the current state of customer nodes, ignoring the difficulty brought by orders. The ordering of past actions is relevant to future actions in PDP-DTW because drones need to fly to pick up at a facility and deliver it to customers. Therefore, the approaches of all the above-mentioned papers are not well-suited for the PDP-DTW we proposed. This paper focuses on developing an end-to-end model that achieves both solution quality and computational efficiency while solving PDP-DTW.

Table.1 Summary of the study on PDP

Authors	Objective function	Pickup and Delivery	Drone energy	Drone capacity	Time windows	Approach
Xue L et al.(2016)	routing and loading cost	Yes	No	Yes	No	Exact
Mahmoudi M et al.(2016)	routing cost	Yes	No	Yes	Yes	Exact
Cheng C et al.(2016)	travel distance	No	Yes	Yes	Yes	Exact
Neira D A et al.(2020)	travel distance	No	No	Yes	Yes	Exact
Bouman P et al.(2018)	completion time	No	No	No	No	Exact
Li M P et al.(2017)	completion time	Yes	No	No	No	Heuristics
Aleksandrov M D et al.(2017)	number of vehicles	Yes	No	No	No	Heuristics
Mehlawat M K et al.(2019)	traveling time and total type score	Yes	No	No	No	Heuristics
Curtois T et al.(2018)	total distance	Yes	No	No	Yes	Heuristics
Xu X et al.(2018)	routing cost	No	No	No	No	Heuristics
Murray C C et al.(2020)	completion time	No	No	Yes	Yes	Heuristics
Salama M et al.(2020)	delivery costs	No	No	Yes	No	Heuristics
Nazari M et al.(2018)	total distance	No	No	No	Yes	Machine learning
Shi J et al.(2018)	completion time and costs	Yes	Yes	No	No	Machine learning
Bello I et al.(2016)	total distance	No	No	No	Yes	Machine learning
Kool W et al.(2018)	total distance	No	No	No	No	Machine learning
Duan L et al.(2020)	total distance	No	No	No	No	Machine learning
Bogyrbayeva A et al.(2023)	completion time	No	Yes	Yes	No	Machine learning

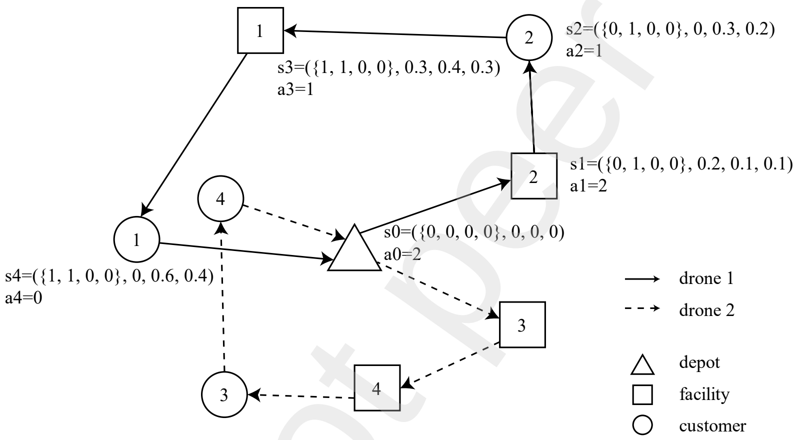

Fig.1 An example of PDP-DTW with state $s_t = (o_{\pi_t}; w_{\pi_t}; e_{\pi_t}; t_{\pi_t})$ and action $a_t = \pi_t$ on 4 orders.

3 Problem formulation

In this section, we will provide the mathematical formulation for PDP-DTW. Fig.1 illustrates an example and explains the Markov Decision Process of PDP-DTW. As an extension of PDPs, PDP-DTW is under the following scenarios or assumptions: (1) In a given region, there exists a center for drones to take off and land, referred to as the depot. There is also a homogeneous and adequate fleet of drones capable of meeting the needs of all customers and their orders. (2) Each order specifies the location for picking up a parcel, the weight of the parcel, the delivery location, and service time windows ensuring that drones complete the delivery tasks within specified timeframes. (3) Each drone has limited capacity and limited energy comparing with trucks in traditional VRP. Therefore we assume that a drone needs to pick up and deliver parcels serval times in one trip to utilize its capacity and battery. (4) All drones take off simultaneously, pick up a parcel at the facility, travel to another facility for another parcel or the customer for delivery, and return to the depot after completing their orders. (5) The time required to load parcels at facility nodes and unload parcels at customer nodes is ignored. Regarding to drone flight condition and our assumptions, the mathematical formulation are as follows:

3.1 Network definitions

F is the set of n facilities, C is the set of n customers. $\{0\}$ and $\{n+1\}$ denotes the depot to take-off and the same depot to landing. $N = \{0\} \cup F \cup C \cup \{2n+1\}$ is the set of nodes. $A = \{(i,j),\ i \neq j,\ (i,j) \in N \times N,\}$ is the set of arcs. The graph can be write into $G = (N, A)$.

For each node $i \in N$:

1) r_i^x is the weight of parcel belongs to x order at node i.
2) a_i^t indicates the earliest start time at node i.
3) b_i^t indicates the latest start time at node i.
4) t_i indicates the start of service time at node i.
5) b_i is the drone energy consumption when it arrive at customer node i.

For each arc $(i,j) \in A$:

1) $v_{i,j}$ is a binary variable that indicates drone move from i to j.
2) $d_{i,j}$ is the distance from i to j.
3) $o_{i,j}^x$ is the parcel weight of x order carried by drone fighting in (i,j), $x \in \{1, \dots, n\}$. If a drone pickup the parcel of first order at node i and travels to j, can be express as $o_{i,j}^1 = 2kg$. Consequently, $\delta_{i,j} = \sum_1^n o_{i,j}^x$ denotes the parcel weight carried by drone during it travels from i to j.

3.2 Network Graph Constraints

In order to ensure that every route is valid through, network graph constraints are expressed as Eq. 1 - Eq. 6.

$$\sum_{j \in N, (i,j) \in A} v_{i,j} = 1 \;\; \forall i \in C \tag{1}$$

$$\sum_{j \in N, (i,j) \in A} v_{j,i} = 1 \;\; \forall i \in C \tag{2}$$

$$\sum_{j \in N, j \neq 0} v_{0,j} - \sum_{j \in N, j \neq n+1} v_{j,n+1} = 0 \tag{3}$$

$$\sum_{j \in N, j \neq 0} v_{j,0} = 0 \tag{4}$$

$$\sum_{j \in N, j \neq n+1} v_{n+1,j} = 0 \tag{5}$$

$$\sum_{j \in N, j \neq 0} v_{0,j} \geq 1 \tag{6}$$

3.3 Pickup and Delivery Constrains

According to different demand information provide with orders, drone will plan the optimal vehicle routing. In each tour, one drone will pickup parcel at facility and delivery it to customer without exceeding its limited capacity ζ. The formula of constraints are shown as Eq. 7 - Eq. 9:

For each order $x \in \{1, \dots, n\}$:

$$\sum_{k \in N, (j,k) \in A} o_{j,k}^x - \sum_{i \in N, (i,j) \in A} o_{i,j}^x = r_i^x \quad \forall j \in F \cup C \tag{7}$$

$$\delta_{i,j} \leq v_{i,j} \zeta \;\; \forall (i,j) \in A \tag{8}$$

$$\sum_{j \in N, (0,j) \in A} o_{0,j}^x = 0 \tag{9}$$

3.4 Energy Consumption Constraints

Energy consumption is a key factor related to drone payload which lead to limited flight time and delivery distance. We use Eq. 10 to represent the drone energy consumption from i to j. λ is a fixed constant calculated by gravity, the fluid density of wind and the area of spinning blade rotor, the number of helicopter rotor. F is the drone framework weight. V is the velocity of drones. B is the battery weight. M is a sufficiently high value. Q is the energy quantity of drone carried each time.

$$e_{i,j} = \lambda \cdot \frac{d_{i,j}}{V} \cdot (F + B + \delta_{i,j})^{\frac{3}{2}} \;\; \forall (i,j) \in A \tag{10}$$

$$b_i + e_{i,j} \le b_j + M(1 - v_{i,j}) \ \forall (i,j) \in A \tag{11}$$

$$b_{n+1} \le Q \tag{12}$$

3.5 Service Time Windows Constraints

In PDP-DTW, the orders should be completed by drones within designated service time windows. Given the hard time window $[a_i^t, b_i^t]$ of order i, the constraints can be rewritten as follow.

$$t_i + \frac{d_{i,j}}{V} \le t_j + M(1 - v_{i,j}) \ \forall (i,j) \in A \tag{13}$$

$$a_i^t \le t_i \le b_i^t \ \forall i \in C \tag{14}$$

3.6 Objective Function

To solve the PDP-DTW in real world, the objective function aims to minimize the total route distance of drone while completing all the pickup and delivery tasks. Our objective function is:

$$min \sum v_{i,j}\, d_{i,j} \ \forall (i,j) \in A$$
$$s.t. (1) - (14)$$

The goal of PDP-DTW is to pursue the minimum drone flight distance with given the orders of each customer. As a extension case of PDPs, PDP-DTW is a combinatorial optimization belongs to be NP-hard. The constrains come from orders, energy consumption and time windows make it more difficult to solve than traditional VRP.

4 A deep reinforcement learning approach

In this section, we present the Graph Attention Model (GAM) to address our problem, following the general encoder-decoder perspective. As a network architecture inspired by attention model, it consists of two components. The first components is a graph embedding encoder (Fig.2), which produces representations of the nodes and graph in PDP-DTW. Then we using a sequential prediction decoder to take the embeddings as input and produces a output sequence as a solution for PDP-DTW.

4.1 Encoder

Given the graph $G = (N, A)$ of a PDP-DTW instance, the input representation for each node is associated with orders (i.e., the coordination of facility and customer, the weight of parcel and the service time windows) and the depot 0 is only associated with the coordination. Encoder of our network will generates the embedded map of (depot, facility and customer) nodes. We will use W_i, b_i to denotes the trainable parameters and use [;] to denotes the concatenation operation. For different nodes in PDP-DTW, we initialed they as d_x-dimensional vectors as follows:

$$h_i^0 = \begin{cases} W_1 x_0^c + b_1, & i = 0 \\ W_2[x_i^c; x_i^w] + b_2, & i \in F \\ W_3[x_i^c; x_i^t] + b_3, & i \in C \end{cases} \tag{15}$$

where x_i^c represents the coordination of nodes, x_i^w represents the weight of parcel at facility nodes and x_i^t represents the service time window at customer nodes. In a encoder with L layers, embeddings will be processed by a multi-head attention (MHA) and node-wise feed-forward (FF) sublayer for $l(l = 1,2,\dots,L)$ times. Formally, we define d_k-dimensional vectors q_i^{l-1} and k_i^{l-1} and a d_v-dimensional vector v_i^{l-1} as follows:

$$q_i^{l-1} = W_4 h_i^{l-1} \tag{16}$$

$$k_i^{l-1} = W_5 h_i^{l-1} \tag{17}$$

$$v_i^{l-1} = W_6 h_i^{l-1} \tag{18}$$

Then we can compute the compatibilities u_{ij}^{l-1} by the scaled dot-product attention and the attention weight a_{ij}^{l-1} via softmax function:

$$u_{ij}^{l-1} = \frac{\left(q_i^{l-1}\right)^T k_j^{l-1}}{\sqrt{d_k}} \tag{19}$$

$$a_{ij}^{l-1} = \frac{e^{u_{ij}^{l-1}}}{\sum_{n=0}^{n=j} e^{u_{in}^{l-1}}} \tag{20}$$

That is said, the embedding of node i at l layer is:

$$h'^{l}_{i} = \sum_{n=0}^{n=j} a_{in}^{l-1} v_{n}^{l-1} \tag{21}$$

Above Eq.16 to Eq.21 shows the the attention mechanism by Vaswani A et al.(2017) and we made some modifications on Eq.19 so that message passing between all nodes. To allow nodes to receive more messages from other nodes, it is beneficial to have multiple attention heads(We using $H = 8$).

$$\hat{h}_i^l = BN^l\left(h_i^{l-1} + MHA_i^l\left(h_0^{l-1}, \ldots, h_{2n}^{l-1}\right)\right) \tag{22}$$

$$MHA_i^l\left(h_0^{l-1}, \ldots, h_{2n}^{l-1}\right) = \sum_{h=1}^{H} W_h^H h'^{l}_{i} \tag{23}$$

where W_h^H is trainable matrices and h'^{l}_{i} is calculated by attention mechanism for H times with different parameters. The output of each l layer with two sublayers is:

$$h_i^l = BN^l\left(\hat{h}_i^l + FF^l\left(\hat{h}_i^l\right)\right) \tag{24}$$

With adding a skip-connection(He K et al.,2016) and batch normalization(Ba J L et al.,2016) operations for L layers, we denote with h_i^L the final node embedding produced by layer L (We using $L = 3$). The graph embedding h_G is:

$$h_G = \frac{1}{n} \sum_{i=0}^{2n} h_i^L \tag{25}$$

Above node embedding and graph embedding lays the foundation for the decoding process via attention mechanism, so that the decoder can search the solution of PDP-DTW.

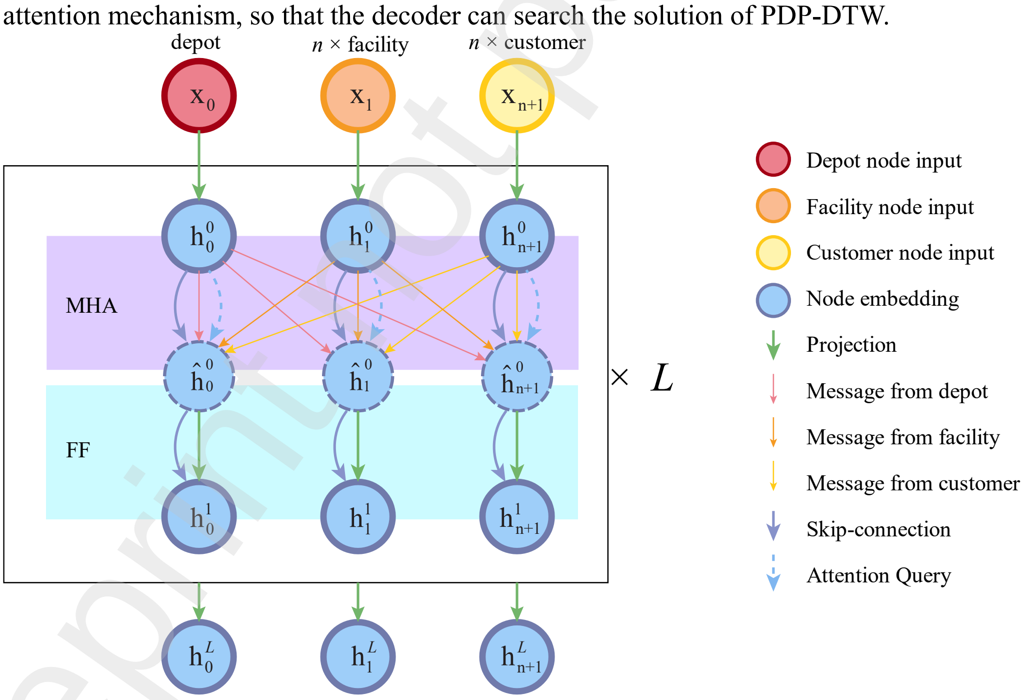

Fig.2 Attention based encoder.

4.2 Sequential prediction decoder

We using a recurrent neural networks(RNN) with a self-attention mechanism to map the nodes embeddings to the solution of PDP-DTW. We using $\pi = \{\pi_0, \pi_1, \pi_2, \dots, \pi_T\}$ denotes the route in PDP-DTW, while $\pi_t \in N$ is the output of decoder at step t. Given the output of encoder, decoder need to generate a sequence which length larger than $2n+1$ because drones will travel back to depot and may need several tours to complete all task. It is pointer network make it possible to generate such a sequence with uncertain length. Firstly, decoder with pointer network will calculate the attention scores before applying softmax function to get the probability distribution. Then with a probability distribution about PDP-DTW, decoder will select a node to be the output π_t at decoding step t. Similarly, the policy or the route we needed will be given after repeating choosing node for enough times. The policy we want to find through GAM network is related with the joint probability is expressed by θ as follow:

$$p_\theta(\pi|s) = \prod_{t=0}^{T} p_\theta(\pi_{t+1}|S, \pi_t) \tag{26}$$

The decoder context at step t is not only based on the output of encoder but also the output of decoder at step $(t-1)$. The network should recognize that the environment is changing while drone travels from one node to another node, i.e. a parcel has been pick up or delivery. The formula of context is as follows:

$$h_t^c = \begin{cases} [h_G; h_{\pi_t}^L; o_{\pi_t}; w_{\pi_t}; e_{\pi_t}; t_{\pi_t}], t \geq 1 \\ [h_G; h_0^L; o_0; w_0; e_0; t_0], t = 0 \end{cases} \tag{27}$$

where $h_{\pi_t}^L$ is the embedding of node π_t and $o_{\pi_t}, w_{\pi_t}, e_{\pi_t}, t_{\pi_t}$ denotes the state transition variables of PDP-DTW. o_{π_t} is a vector records the parcel carry by drone at node π_t. It can be formulated as follows:

$$o_{\pi_t} = \left(f_1^{\pi_t}, f_2^{\pi_t}, \dots, f_n^{\pi_t}\right) \tag{28}$$

$$f_i^{\pi_t} = \begin{cases} \max\left(1, f_i^{\pi_{t-1}}\right), \pi_t = i \\ 0, else \end{cases} \tag{29}$$

For example, if a drone has load the parcel of order 1 and load a new parcel at facility 3, then $o_3 = (1, 0,1,0,0, \dots,0)$ will record what happens. We also draw the concept of other variables into establishing the state transition equation of PDP-DTW. When the order is certain, the total weight of parcel, the energy consumption, the elapsed time and are expressed as w_{π_t}, e_{π_t} and t_{π_t}. The state transition equations are updated as follows:

$$w_{\pi_t} = \begin{cases} w_{\pi_{t-1}} + r_{\pi_t}^{\pi_t}, \pi_t \in F \cup C \\ 0, else \end{cases} \tag{30}$$

$$g_{\pi_{t-1},\pi_t} = \lambda \cdot \frac{d_{\pi_{t-1},\pi_t}}{V} \cdot \left(F + B + w_{\pi_{t-1}}\right)^{\frac{3}{2}} \tag{31}$$

$$e_{\pi_t} = \begin{cases} e_{\pi_{t-1}} + g_{\pi_{t-1},\pi_t}, \pi_t \in F \cup C \\ 0, else \end{cases} \tag{32}$$

$$t_{\pi_t} = \begin{cases} t_{\pi_{t-1}} + \dfrac{d_{\pi_{t-1},\pi_t}}{V}, \pi_t \in F \cup C \\ 0, else \end{cases} \tag{33}$$

In each tour, $\pi_t = 0$ indicates the initial state of drone, which carry nothing and fully charging before take off. With the drone travel from π_{t-1} to π_t, context are updated through these equations. To record the state, we keep track of the variables and pack they into the context. A single-head attention mechanism is used to calculate the $p(\pi_{t+1}|S, \pi_t)$ in Eq.26. Different from hat we do in Eq.16 to Eq.21, the computational procedure at decoder step t is as Eq.32 to Eq.36:

$$q_t^c = W_7 h_t^c \tag{34}$$

$$k_t^c = W_8 h_i^L \tag{35}$$

In order to facilitate the constraints of PDP-DTW, masking will ensure network output the regular solution. At decoding step t, the set of masked nodes at step t is:

$$S_t = \begin{cases} \{0\} \cup C, \pi_{t-1} = 0 \\ S_{t-1} \cup \{i | w_i \geq \zeta \; or \; e_i \geq Q \; or \; t_i \geq l_i\}, else \end{cases} \tag{36}$$

$$u_i^c = \begin{cases} -\infty, \;\; \forall i \in S_t \\ C \cdot tanh\left(\frac{(q_t^c)^T k_t^c}{\sqrt{d_k}}\right), else \end{cases} \tag{37}$$

where using activation function $tanh$ with value range from $-C$ to C. A softmax function is applied to get the the probability of the final output vector, as follows:

$$p(\pi_{t+1} | S, \pi_t) = \frac{e^{u_{\pi_t}^c}}{\sum_{j=0}^{j=2n} e^{u_j^c}} \tag{38}$$

See Fig.3(1-5) for an illustration of the decoding process and how a tour $\pi = \{0,2,4,1,3,0\}$ is constructed in PDP-DTW. In this example, the order 1 request pickup at facility 2 and deliver to customer 4, while order 2 request pickup at facility 1 and deliver to customer 3.

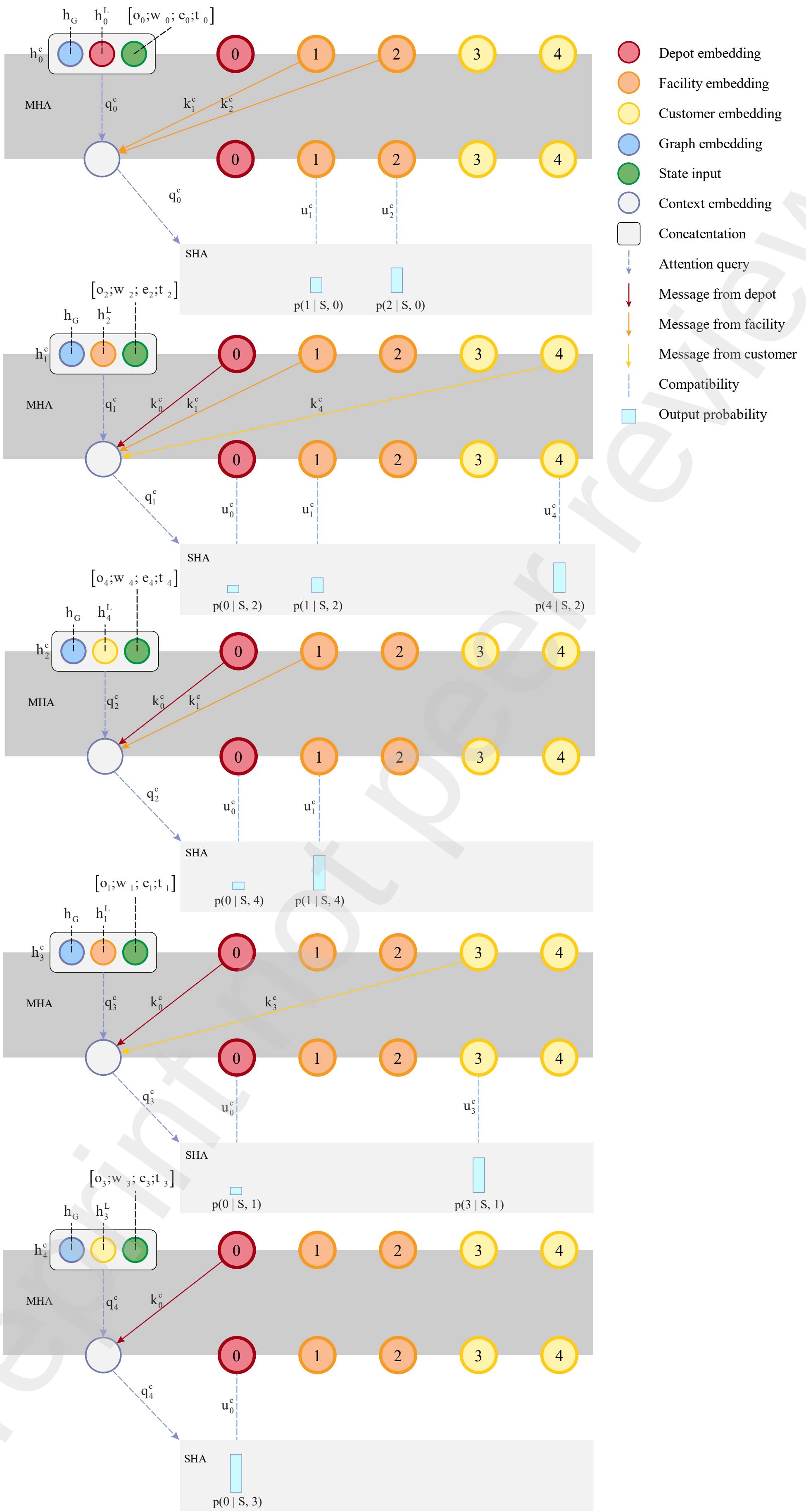

Fig.3 Attention based decoder.

4.3 Reinforce with rollout baseline

Section 4.1 and 4.2 defined our model and it will provide a probability distribution $P(\pi|s)$ to sample for obtaining a tour in PDP-DTW after training. In order to train GAM, we define the reinforced loss of the decoder as the expected cost:

$$\begin{aligned}\mathcal{L}_r(\theta|S) &= \sum_{S} \mathbb{E}_{\pi\sim p_\theta(\pi|s)}\big(L(\pi) - b(s)\big) \\ &= \sum_{S}\big(L(\pi) - b(s)\big)\sum_{i=1}^{T}\log p(\pi_{t+1}|S,\pi_t) \end{aligned} \tag{39}$$

where $L(\pi)$ is the total cost of the solution π and $b(s)$ is the cost of a solution from a deterministic greedy rollout policy. The baseline is fixed for each epoch by freezing greedy rollout policy. At the end of every epoch, the baseline policy will be updated if the current trained policy improves significantly (according to a paired t -test). In the following, we denote the trained policy by $\mathbb{P}_\theta$, and the baseline policy by $\mathbb{P}_{\theta^{BL}}$.

Algorithm 1 The learning procedures of GAM

Inputs: number of epochs E, steps per epoch T, batch size B, significance α (for paired *t*-test)
Initialization: $\theta, \theta^{BL} \leftarrow \theta$
for epoch = $1,\dots,E$ **do**
    for step = $1,\dots,T$ **do**
        $s_i \leftarrow RandomInstance()\ \forall i \in \{1,\dots,B\}$
        $h_i \leftarrow Encoder(s_i)\ \forall i \in \{1,\dots,B\}$
        $\pi_i \leftarrow SampleRollout(h_i, \mathbb{P}_\theta)\ \forall i \in \{1,\dots,B\}$
        $\pi_i^{BL} \leftarrow GreedyRollout(h_i, \mathbb{P}_{\theta^{BL}})\ \forall i \in \{1,\dots,B\}$
        $\mathcal{L}_r(\theta) \leftarrow ReinforcedLoss(\pi_i, \pi_i^{BL})$
        $\theta \leftarrow Adam\big(\theta, \nabla\mathcal{L}_r(\theta)\big)$
    if OneSidedPairedTTest$(\theta, \theta^{BL}) < \alpha$ **then**
        $\theta^{BL} \leftarrow \theta$

5 Computational experiment

In this section, we test our method on a set of randomly instances with different number of customers and facilities. In section 5.2, numerical experiment will compare the metrics between GAM and other solutions. Section 5.3 carries out further analysis on the performance of GAM and another deep learning model. Finally, we conduct sensitivity analysis in section 5.4 by varying the parameters in PDP-DTW.

5.1 Experimental Settings

For the reason that there have not been any proposed PDP-DTW benchmark test suites, we generate the set of randomly instances. The test benchmarks contain instance sets with different number of nodes: 20 nodes (10 facilities and 10 customers), 40 nodes(20 facilities and 20 customers), 60 nodes(30 facilities and 30 customers) and 80 nodes(40 facilities and 40 customers). Specifically, each problem encompasses various customer locations, facility locations, parcel weights, and time windows to simulate diverse real-world conditions.

The x, y coordinates of each facility and customer location were generated based on a uniform distribution over the $[0,4]\times[0,4]$ square, the weight r_i^x of order x at node i was generated from $[20\%, 40\%]$ of capacity ζ. For time window, the earliest start time a_i^t was set to 0 and the latest start time b_i^t was generated from $[0.4, 0.6]\ hours$. Following the research of Dorling K et al.(2016), we set $\lambda = 20,\ Q = 0.97kWh,\ F + B = 3kg,\ \zeta = 10kg,\ V = 60km/h$.

As an alternative method to solve PDP-DTW, we consider the following methods as our baseline: graph convolutional networks (GCN), mixed integer linear program (MILP), Greedy Randomized Adaptive Search Procedures(GRASP), Ant Colony Optimization (ACO) were implemented based on the previous studies. For MILP, IBM CPLEX optimizer was used based on the default settings, and 3,600 second was set as the time limit. All algorithms were coded in Python and run on an Intel i7-6700HQ 2.6 GHz processor with 16 GB RAM and the GTX 3090 graphic card.

5.2 Computational Results of PDP-DTW

5.2.1 Small size problem

This section explains the performance of solving small size PDP-DTW problem. The results are shown in Table.2(1-2), where columns show the values of the objective function and runtime. Furthermore, columns of MILP show the values of GAP found by CPLEX for each instance. For other approach, let $O_A(i)$ be the solution's value founded by algorithms i and O_C the value of the solution founded by CPLEX on the same instance. Their gaps are calculated as follows:

$$GAP(i) = \frac{O_A(i) - O_C}{O_C} \times 100\% \tag{40}$$

The results show that there exist different difficult between different instances. On instance 10f10c2, CPLEX can obtain exact solution while GAM take a few seconds to obtain the same result. On instance 10f10c5, GAM obtains better result than CPLEX. But for more instances, CPLEX can obtain feasible solutions in one hour which better than GAM. The commercial solver continues to demonstrate its effectiveness in consistently solving small-size problems. Regarding heuristic algorithms, GRASP was also able to obtain feasible solutions in an acceptable time frame compared to ACO.

Table.2 Results for small size problems

instance	GAM		GCN		MILP		GRASP		ACO	
	obj.	gap*	obj.	gap*	obj.	gap	obj.	gap*	obj.	gap*
10f10c1	4.994	0.37%	5.334	7.21%	4.975	10.39%	5.450	9.55%	8.104	62.88%
10f10c2	4.993	0.00%	6.308	26.33%	4.993	0.00%	5.274	5.62%	9.431	88.87%
10f10c3	5.344	13.62%	5.406	14.95%	4.703	8.37%	5.204	10.65%	6.934	47.43%
10f10c4	4.876	0.85%	5.629	16.43%	4.835	13.72%	4.900	1.35%	6.878	42.25%
10f10c5	5.578	-2.19%	6.503	14.03%	5.703	16.05%	6.053	6.14%	7.542	32.26%

Table.3 Runtime for small size problems

instance	GAM	GCN	MILP	GRASP	ACO
	rt.	rt.	rt.	rt.	rt.
10f10c1	3.69	8.04	3600.49	650.27	100.79
10f10c2	3.38	3.15	724.60	647.79	98.26
10f10c3	3.50	3.40	3601.67	656.79	93.16
10f10c4	3.63	3.19	3601.81	624.47	94.07
10f10c5	3.99	3.26	3600.52	602.78	92.39

5.2.2 Middle and Large problem

Based on the computational results in Table.2 and Table.3, it is evident that CPLEX is unable to obtain the exact solutions for all small-sized problems. When we applied CPLEX on larger-size problem, specifically instances containing 40 nodes, the computational difficulty of NP-hard became apparent and feasible solution could not be obtained through MILP in acceptable time. Therefore, the results using CPLEX are omitted in this section, and an additional metric is introduced for comparison. Let $O_A(i)$ be the solution's value founded by algorithms i and O_B the value of the best solution on the same instance. The GAPs are calculated as follows:

$$GAP(i) = \frac{O_A(i) - O_B}{O_B} \times 100\% \tag{41}$$

As can be seen from the computational results in Table.4 and Table.5, for the PDP-DTW proposed in this paper, the solution performance of GAM and GCN are better than other method, both in terms of solution quality and runtime. Regarding heuristic, GRASP still outperformed than ACO and can obtain better solution than GCN on instances with 40 nodes. However, as the number of nodes increased to 60 and 80, the gap of GCN decrease and can obtain better solution than GAM we proposed. This phenomenon may be attributed to the novel architecture of GCN, which has been proved to be success and efficient in solving VRP with 50 and more nodes.

Table.4 Results for middle and large size problems

instance	GAM		GCN		GRASP		ACO	
	obj.	gap	obj.	gap	obj.	gap	obj.	gap
20f20c1	10.2377	0.00%	13.6974	33.79%	13.0222	27.20%	19.6543	91.98%
20f20c2	11.0905	0.00%	15.5899	40.57%	13.6154	22.77%	20.9077	88.52%
20f20c3	11.5809	0.00%	12.7765	10.32%	13.1807	13.81%	19.9094	71.92%
20f20c4	11.7823	0.00%	15.3435	30.23%	13.5518	15.02%	18.5023	57.03%
20f20c5	10.0720	0.00%	13.1498	30.56%	13.8074	37.09%	19.7997	96.58%
Avg.	10.9527	0.00%	14.1114	29.09%	13.4355	23.18%	19.7547	81.21%
30f30c1	16.2204	0.00%	18.4509	13.75%	20.2813	25.04%	32.3497	99.44%
30f30c2	17.2856	0.00%	18.4390	6.67%	20.8220	20.46%	32.4270	87.60%
30f30c3	17.5188	0.49%	17.4331	0.00%	19.9541	14.46%	32.2697	85.11%
30f30c4	18.1012	0.00%	21.8142	20.51%	21.0542	16.31%	31.1362	72.01%
30f30c5	18.0774	1.83%	17.7519	0.00%	19.3671	9.10%	31.7144	78.65%
Avg.	17.4407	0.47%	18.7778	8.19%	20.2958	17.07%	31.9794	84.56%
40f40c1	24.3648	5.23%	23.1546	0.00%	30.8248	33.13%	48.1317	107.87%
40f40c2	24.0648	0.00%	26.2995	9.29%	29.9596	24.50%	47.0571	95.54%
40f40c3	25.4444	0.73%	25.2597	0.00%	31.2822	23.84%	48.6533	92.61%
40f40c4	25.8793	0.00%	26.7733	3.45%	28.6121	10.56%	46.5507	79.88%
40f40c5	23.0397	0.00%	23.3750	1.46%	29.8692	29.64%	45.7932	98.76%
Avg.	24.5586	1.19%	24.9724	2.84%	30.1096	24.33%	47.2372	94.93%

Table.5 Runtime for middle and large size problems

instance	GAM	GCN	GRASP	ACO
	rt.	rt.	rt.	rt.
20f20c1	4.97	4.63	1396.39	191.09
20f20c2	5.02	4.74	1308.19	197.58
20f20c3	5.36	4.55	1337.88	190.50
20f20c4	5.40	5.05	1382.38	185.96
20f20c5	5.01	4.89	1352.57	194.67
Avg.	5.15	4.77	1355.48	191.96
30f30c1	5.88	7.05	2123.77	291.09
30f30c2	6.54	6.19	2118.49	281.87
30f30c3	6.32	7.34	2138.80	291.57
30f30c4	5.88	6.81	2059.00	289.90
30f30c5	6.16	6.33	2081.65	282.57
Avg.	6.16	6.74	2104.34	287.40
40f40c1	9.67	8.23	2971.65	386.17
40f40c2	9.56	7.52	2959.78	403.50
40f40c3	9.49	7.93	2899.89	405.39
40f40c4	9.09	8.11	2984.32	398.38
40f40c5	7.75	8.94	2944.26	390.99
Avg.	9.11	8.15	2951.98	396.89

5.3 Comparison of GAM and GCN on more instances

To provide readers with a better understanding of the differences between the Graph Attention Network and Graph Convolution Network, this section aims to conduct a more in-depth analysis of the performance of the two deep learning models we utilized to solve PDP-DTW.

In practical scenarios, it is more common for one facility to receive multiple orders from different customers or for one customer to receive several parcels from different facilities. Such scenarios, with an unequal number of facility nodes and customer nodes, deviate from the original PDP-DTW, yet they can still be effectively addressed using the same method we proposed. To verify the generalizability and evaluate the impact of solution quality on the performance of the deep learning model, this section assesses the solution quality across each set in Table 6-8, which contains 100 instances. Other settings remain consistent with Section 5.2.

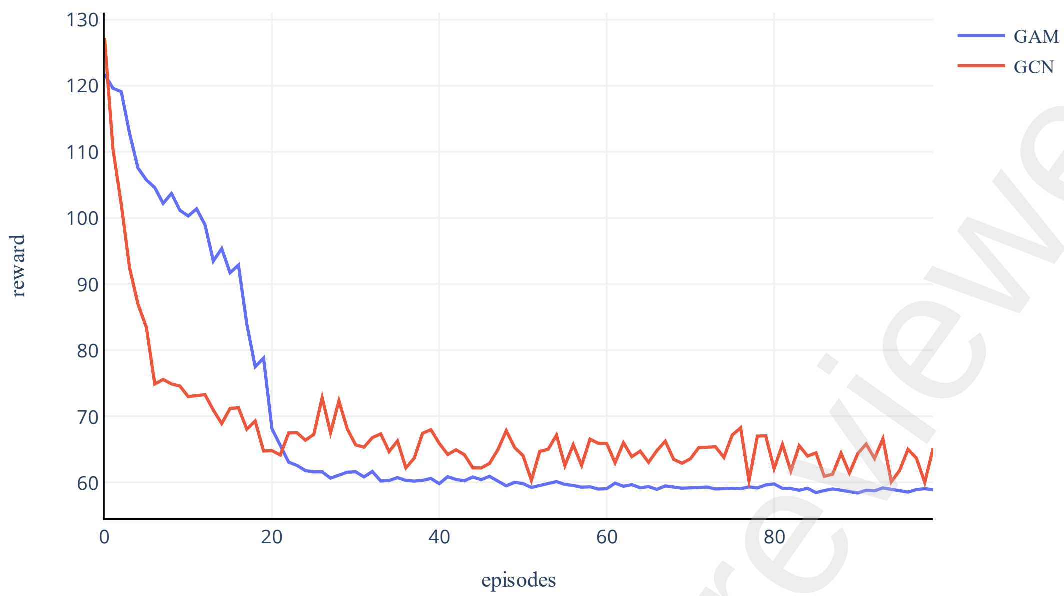

Fig.4 Training rewards of GAM and GCN

Table.6 Results of GAM-BS for PDP-DTW and its variants

set of instance	GAM-BS					
	min	Q1	mean	middle	Q3	max
f3_c10	3.0182	5.0446	5.7655	5.6721	6.3385	10.6612
f5_c10	3.8652	5.0596	5.6699	5.6351	6.2584	7.9410
f7_c10	3.9422	5.1252	5.7417	5.7471	6.2324	7.4167
f10_c10	4.2056	5.5451	5.8821	5.7963	6.3313	8.3257
f10_c7	3.8690	5.1726	5.7627	5.7188	6.2476	8.2648
f10_c5	3.6276	5.1561	5.6485	5.6918	6.2343	7.5872
f10_c3	3.7267	5.1544	5.7737	5.6699	6.1880	8.5755
f5_c20	8.3199	10.3149	11.2077	11.0486	11.9764	14.9119
f10_c20	8.3848	10.4911	11.2135	11.1227	11.9780	14.7418
f15_c20	8.2263	10.4114	11.3737	11.3668	12.2738	14.0343
f20_c20	8.6003	10.5039	11.4785	11.4009	12.5104	14.3197
f20_c15	7.9963	10.7153	11.3570	11.3524	11.9354	13.5307
f20_c10	8.8512	10.6364	11.3691	11.3320	12.1120	14.6102
f20_c5	8.7414	10.3088	11.3402	11.3150	12.3730	15.7908
f7_c30	13.9281	16.2693	17.3405	17.4229	18.3622	21.4457
f15_c30	13.8263	16.1240	17.2372	17.1734	18.5525	20.2069
f23_c30	14.0778	16.3048	17.2427	17.4077	18.0671	20.3263
f30_c30	14.1615	16.2252	17.3060	17.3811	18.2622	20.5679
f30_c23	14.2294	16.3316	17.2363	17.2718	17.9309	21.4004
f30_c15	13.6549	16.0161	17.0480	17.1254	17.9332	20.3698
f30_c7	13.9824	16.2207	17.1950	17.1859	18.0936	23.3986
f10_c40	16.9638	21.2230	22.6859	22.9041	24.0320	28.6977
f20_c40	17.6009	21.2634	22.2486	22.3366	23.3256	26.6722
f30_c40	17.9815	21.2698	22.3611	22.5469	23.5898	27.0927
f40_c40	18.8412	21.8542	22.9172	23.0161	24.0296	28.1909
f40_c30	18.5156	21.3688	22.7317	22.5807	23.9838	28.3407
f40_c20	17.7762	21.3637	22.6509	22.5432	23.8802	27.4728
f40_c10	18.3786	21.3448	22.9477	22.6159	24.7368	32.1385

Table.7 Results of GAM-GS for PDP-DTW and its variants

set of instance	GAM-GS					
	min	Q1	mean	middle	Q3	max
f3_c10	3.0182	5.0813	5.8172	5.7249	6.4679	9.2963
f5_c10	3.8652	5.2152	5.9049	5.9708	6.6462	8.4511
f7_c10	4.3866	5.4560	6.0209	5.9932	6.4433	8.8899
f10_c10	4.4103	5.7659	6.1477	6.0273	6.5811	8.4077
f10_c7	3.9316	5.2978	6.0248	6.0248	6.6740	8.4962
f10_c5	3.6277	5.3036	5.7988	5.8231	6.3441	7.7579
f10_c3	3.7270	5.2400	5.8435	5.6955	6.3844	8.7559
f5_c20	7.8563	10.4072	11.5920	11.5626	12.4528	15.7354
f10_c20	8.0926	10.5772	11.6017	11.6189	12.6617	14.4222
f15_c20	8.2512	10.8588	11.8160	11.7045	12.9162	15.1844
f20_c20	9.5120	10.9668	11.9216	11.9496	12.9306	15.3440
f20_c15	7.9964	10.8942	11.9265	11.8784	12.8571	15.9819
f20_c10	9.1890	10.9943	11.6908	11.6007	12.5402	15.0570
f20_c5	8.6682	10.3235	11.5288	11.5790	12.5436	14.8226
f7_c30	12.8694	16.0845	17.7054	17.4228	19.5757	23.6055
f15_c30	13.8696	16.2256	17.4890	17.6288	18.5815	21.3926
f23_c30	14.7959	16.8036	17.6692	17.6460	18.7445	21.6588
f30_c30	14.7243	17.0033	17.8415	17.7035	18.7217	21.7128
f30_c23	14.4094	16.8639	17.8327	17.7942	18.8724	22.3536
f30_c15	13.6836	16.4352	17.4661	17.3729	18.6480	21.6610
f30_c7	13.9729	15.9764	17.4420	17.4584	18.8871	22.0388
f10_c40	18.5402	21.2159	22.9630	22.7189	25.0345	29.2235
f20_c40	18.2134	21.2262	22.5469	22.2820	23.6135	28.4550
f30_c40	18.9257	21.2244	22.4605	22.2837	23.5783	28.4218
f40_c40	18.8640	22.1124	22.9651	22.9135	24.0253	27.1764
f40_c30	18.7074	22.2113	23.1876	23.1236	24.4072	27.1764
f40_c20	18.4646	21.8687	23.0880	23.3290	24.1955	27.4005
f40_c10	17.8756	21.7779	23.1153	22.9460	24.7336	28.2099

Table.8 Results of GCN for PDP-DTW and its variants

set of instance	GCN					
	min	Q1	mean	middle	Q3	max
f3_c10	3.4722	5.2684	6.1463	6.0781	6.6149	10.2423
f5_c10	3.8025	5.3802	6.0775	6.0166	6.6340	9.4830
f7_c10	4.4980	5.6554	6.1650	6.0714	6.7040	9.0497
f10_c10	4.7100	5.8138	6.3961	6.3850	6.9391	9.5807
f10_c7	4.1550	5.4731	6.1268	6.1037	6.6227	8.7850
f10_c5	3.9381	5.4294	6.1340	5.9802	6.6832	9.5205
f10_c3	4.0988	5.2939	6.1143	5.9637	6.6728	10.9684
f5_c20	8.8194	11.4601	12.6220	12.5418	13.5740	17.5766
f10_c20	8.9182	11.4822	12.6083	12.6000	13.7478	17.3107
f15_c20	10.5910	12.3532	13.0504	12.8209	13.7568	16.9391
f20_c20	10.0664	12.1070	13.0611	13.1495	13.8053	17.8585
f20_c15	9.7350	12.0289	12.8613	12.9037	13.6952	16.2545
f20_c10	9.5223	11.5295	12.6897	12.7619	13.8026	16.2683
f20_c5	8.3533	11.3761	12.5950	12.3839	13.8457	16.8889
f7_c30	12.8889	16.2656	17.6721	17.6568	19.0727	24.5642
f15_c30	13.9802	16.6998	17.9015	17.5715	19.0462	22.0629
f23_c30	12.5096	16.5862	17.6965	17.3685	18.7222	22.2701
f30_c30	13.8417	16.7052	17.8071	17.6209	18.8229	21.3764
f30_c23	13.9287	16.8751	17.9210	18.0865	18.9328	23.6368
f30_c15	13.6319	16.2112	17.4249	17.4015	18.5348	22.2464
f30_c7	13.7527	16.0743	17.6681	17.5355	19.2349	23.4081
f10_c40	18.0524	21.9573	23.4762	23.1974	25.0600	29.8043
f20_c40	18.0275	21.3652	22.8037	22.7997	24.2284	28.8510
f30_c40	18.1879	21.5716	22.7667	22.7253	24.0981	27.2144
f40_c40	17.8054	22.3801	23.4008	23.1718	24.3467	29.2638
f40_c30	18.9985	22.0486	23.2547	23.1583	24.4589	28.9674
f40_c20	19.2650	21.6941	23.3013	23.3561	24.8581	27.5101
f40_c10	18.5596	22.2070	23.3148	23.1570	24.7919	29.6357

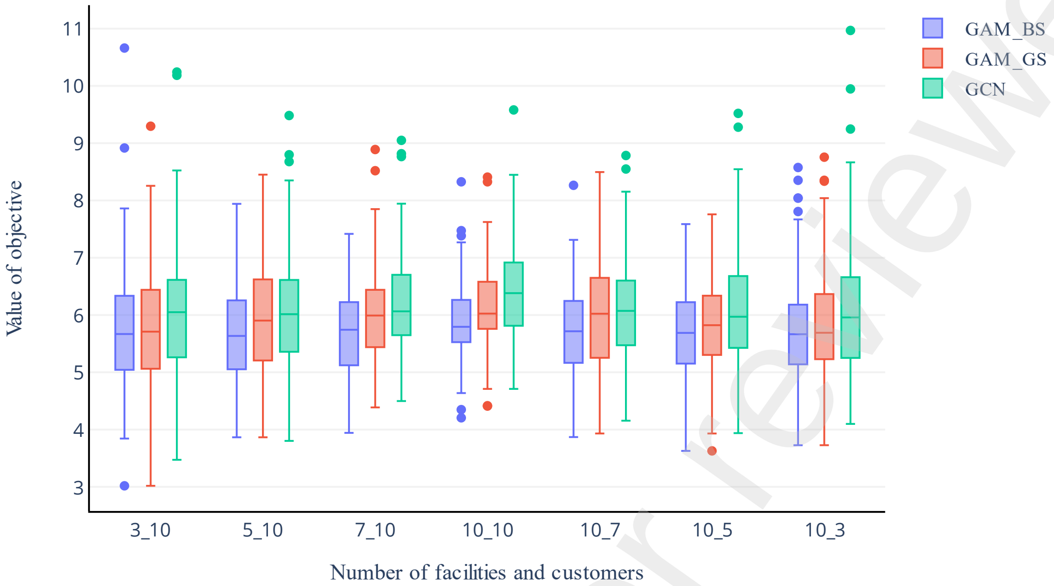

Fig.5 Box plots for PDP-DTW and its variants on 10 orders

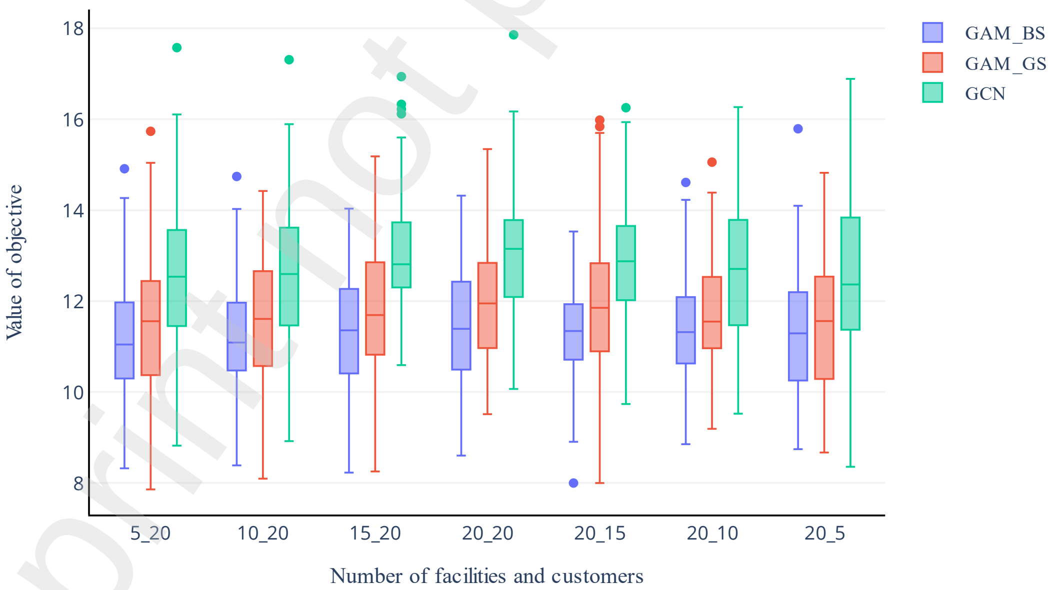

Fig.6 Box plots for PDP-DTW and its variants on 20 orders

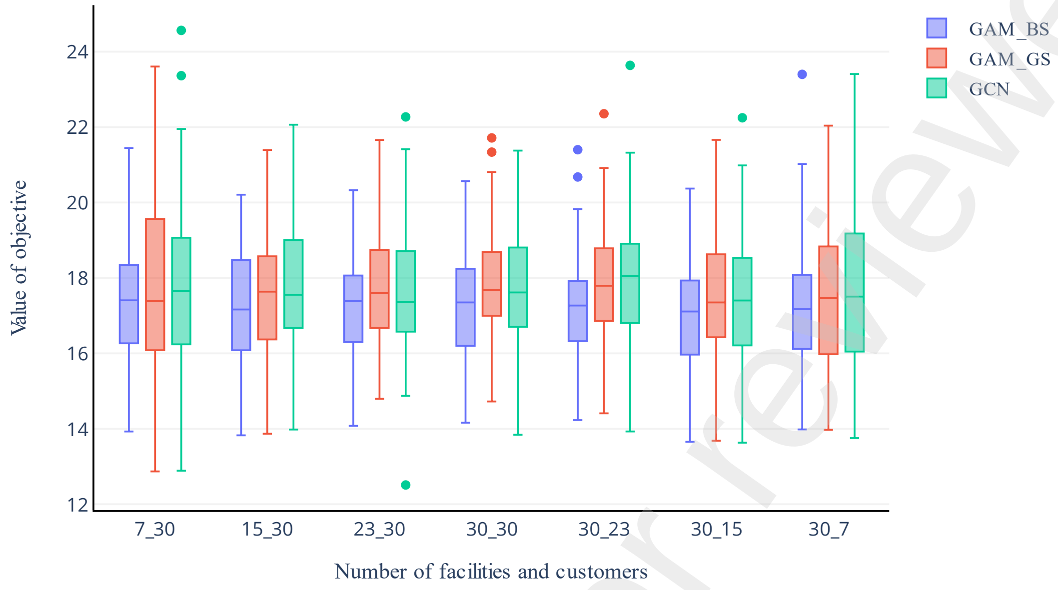

Fig.7 Box plots for PDP-DTW and its variants on 30 orders

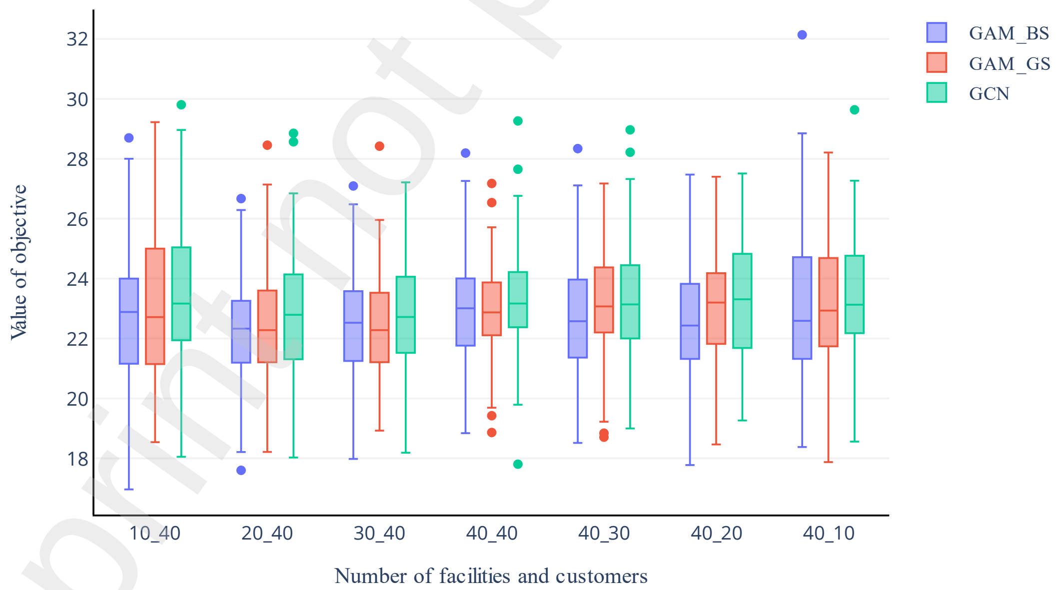

Fig.8 Box plots for PDP-DTW and its variants on 40 orders

Figure 4 illustrates the training rewards for the PDP-DTW. In general, using GAM results in better and more stable rewards compared to GCN, which incorporates more edge information into its structure. To compare the effect of decoding rules on the attention mechanism, GAM with beam search (GAM-BS) and GAM with greedy search (GAM-GS) are tested. Now we report the experiment results for the PDP-DTW under the scenario with an unequal number of facility nodes and customer nodes. In particular, the results (Fig.5, Fig.6, Fig.7, Fig.8) compare the performance among the solutions generated by GAM and GCN under different scenario of PDP-DTW and its variants, and show that pretrained model are capable of providing effective solutions. It can be seen that GAM generally outperforms than GCN on solving PDP-DTW and its variants. Furthermore, with the sum of facility nodes and customer nodes decreased, the objective value also decreases due to the fact that one drone can pick up or deliver multiple parcels at the same nodes, reducing the total distance. It appears that GAM is more adept at capturing this feature, resulting in better solutions than GCN when dealing with scenarios involving an unequal number of facilities and customers. Although GAM with beam search takes more time than greedy search, it still outperforms many other methods. For example, when the number of orders is lower than 30, beam search consistently leads to higher quality results than greedy search.

5.4 Sensitivity analysis on PDP-DTW

This section aims to further evaluate the results and conduct sensitivity analysis on three aspects of PDP-DTW. We investigate the impact of different parameters, including drone capacity, time windows, and energy consumption, on problems of various sizes to observe their performance. We generated four sets, each containing 100 instances, and calculated the mean solution for analysis.

5.4.1 Impacts of the capacity

The limited capacity is an important part of PDP-DTW, while the weight of parcel always varied from different customers. We using the ratio of the capacity to denotes that a drone can carry out how many parcel of different orders, where 0.5 denotes that one drone can carry 2 parcels mostly and 0.2 denotes 5 parcels. Fig.9 reveals the impact of changes in capacity (or parcel weights), which shows that the value of objective have all increased as the capacity decreased. It can be seen that when the ratio of capacity increases from 0.2 to 1.0, the travel distance have increased by 54.47% on small-sized problem and increased more with more orders. This is because as the capacity become enough, drones have more choices to pickup and delivery parcels. Moreover, when the ratio of capacity increased from 0.9 to 1.0, it result in the similar case that a drone can hardly pickup another parcel so that the travel distance will be maximum. Therefore, it is important for platform to choice the capacity or limited the weight of parcels to a certain threshold.

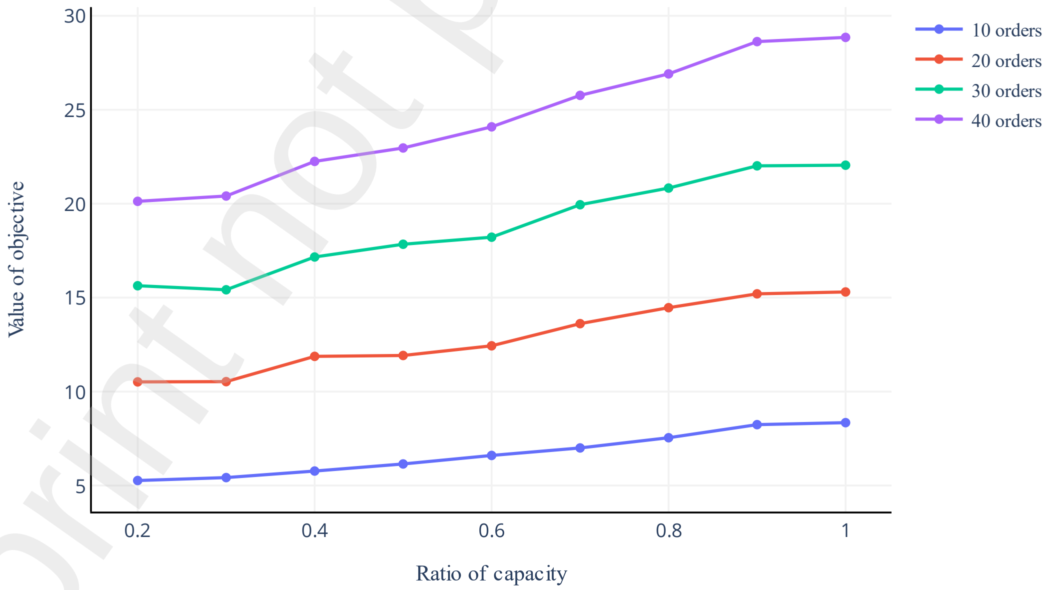

Fig.9 Sensitivity analysis of capacity of drones

5.4.2 Impacts of the time windows

The width of time windows is also crucial in PDP-DTW, and we vary the time windows by multiplying a ratio with the parameters used before. From Fig. 10, we observe that the travel distance decreases sharply at first and then becomes relatively flat. This is because, as the time windows widen, drones can deliver more parcels simultaneously. However, drone flying time is limited to the maximum travel distance even when time windows are wide enough, as it cannot complete all orders in one tour. As the number of orders increases, the travel distance also increases. This further emphasizes the importance of time windows in large-sized problems, highlighting the need for the platform to strike a balance between the cost implications of narrow time windows and the demand in larger scenarios.

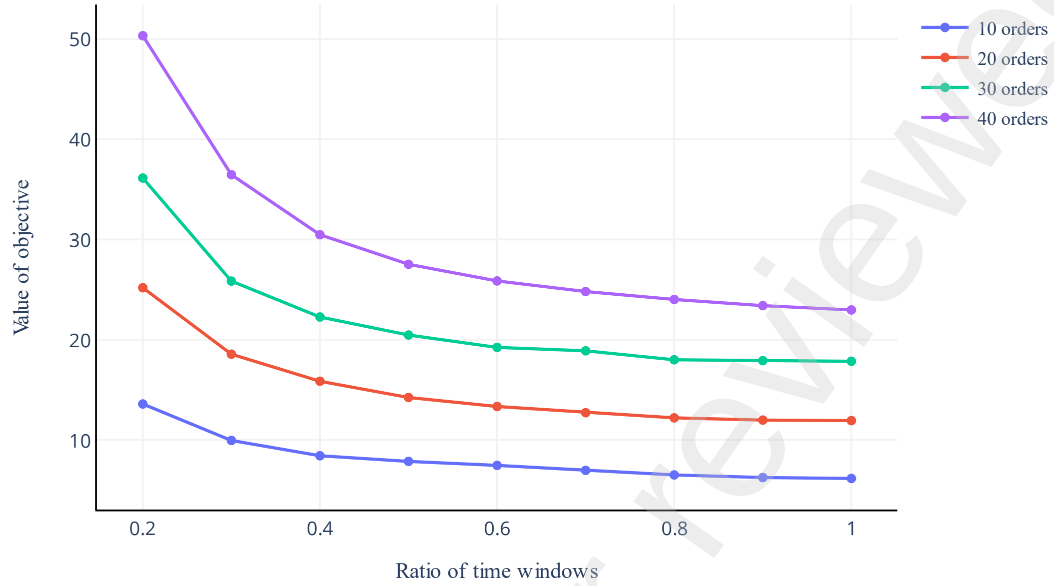

Fig.10 Sensitivity analysis of time windows

5.4.3 Impacts of the energy consumption

Drones, being electric vehicles, are sensitive to factors such as payload, flying range, battery life, and others, all of which are closely tied to energy consumption. To model real-life applications more accurately, this section takes into consideration the nonlinear energy constraints of PDDRPO-PDP-DTW. The results of multiplying different ratios on energy consumption are shown in Fig. 11. It can be observed that there are two distinct stages: the first stage is when the ratio varies from 1 to 4, and the second is from 5 to 10. During the first stage, the objective value remains stable, unaffected as the drone consumes more energy. A drone, equipped with a designated battery capacity, can handle tasks with a workload below the threshold achievable with a fully charged battery. Thus, during this stage, the drone's workload can be increased, including improvements to capacity, delivery distance, and other factors, thereby enhancing the overall efficiency of the logistics network. However, as the energy consumption changes according to a ratio from 5 to 10, it marks the point where the drone requires returning to the depot for recharging, constrained by its limited battery capacity. In this scenario, optimizing efficiency hinges on augmenting energy capacity or implementing additional charging stations between the depot and customers.

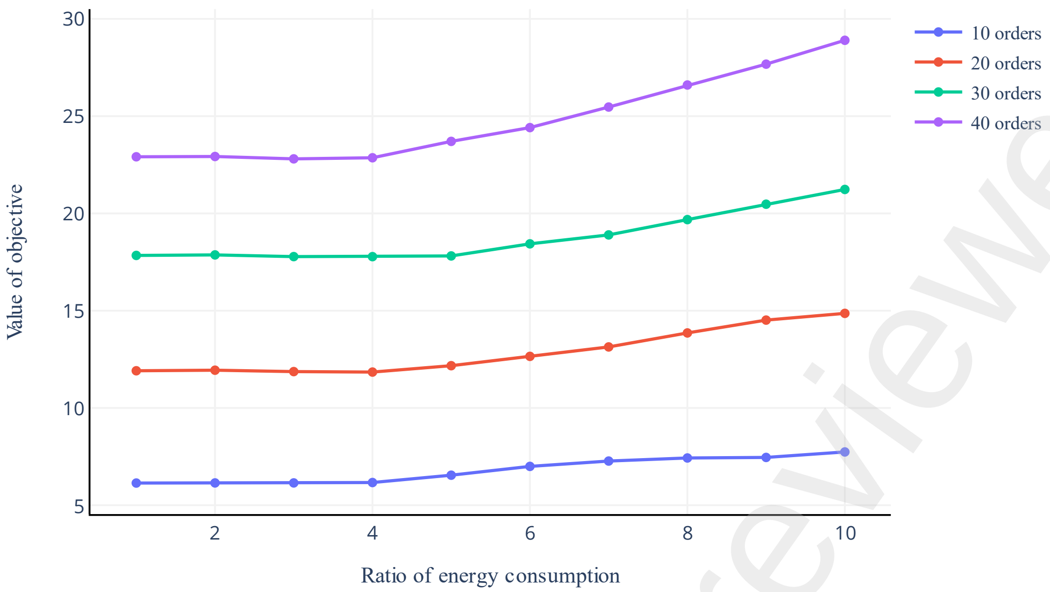

Fig.11 Sensitivity analysis of energy consumption

6 Conclusion

We herein addressed the PDP-DTW for minimization of total travel distance. In this problem, some of the orders request delivery in specific time windows; the energy consumption is considered as a nonlinear function, which causes the challenge of solving the proposed problem. We proposed an approach called GAM, a deep reinforcement learning approach to learn routing policies for PDP-DTW. The results of the numerical experiments are as follows. (1)By utilizing the proposed mathematical model, the proposed solution methods proved to be valid and efficient on obtaining optimal or best feasible solutions. (2)Sensitivity analysis help us model more accurately real-life applications. Regarding to future work, other extensions of the PDP-DTW, considering more complex situations, such as dynamic situation and traffic congestion, will also be conducted.

Acknowledgments

This work is supported by the National Key R&D Program of China under Grant No. 2020YFB1712802, the Research Program of the Regional Economic Development Research Center of Yanshan University under Grant No JJ2208.

Reference

[1] Aleksandrov M D, Barahona P, Kilby P, et al. Heuristics and policies for online pickup and delivery problems[C]//Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence. 2013.

[2] Aurambout J P, Gkoumas K, Ciuffo B. Last mile delivery by drones: An estimation of viable market potential and access to citizens across European cities[J]. European Transport Research Review, 2019, 11(1): 1-21. https://doi.org/10.1186/s12544-019-0368-2

[3] Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016. https://doi.org/10.48550/arXiv.1607.06450

[4] Bello I, Pham H, Le Q V, et al. Neural combinatorial optimization with reinforcement learning[J]. arXiv preprint arXiv:1611.09940, 2016. https://doi.org/10.48550/arXiv.1611.09940

[5] Berbeglia G, Cordeau J F, Laporte G. Dynamic pickup and delivery problems[J]. European journal of operational research, 2010, 202(1): 8-15. https://doi.org/10.1016/j.ejor.2009.04.024

[6] Bi, H.; Zhu, X.; Lu, F.; Huang, M. The Meal Delivery Routing Problem in E-commerce Platforms under the Shared Logistics Mode. J. Theor. Appl. Electron. Commer. Res. 2023, 18, 1799–1819. https://doi.org/10.3390/jtaer18040091

[7] Bogyrbayeva A, Yoon T, Ko H, et al. A deep reinforcement learning approach for solving the traveling salesman problem with drone[J]. Transportation Research Part C: Emerging Technologies, 2023, 148: 103981.https://doi.org/10.1016/j.trc.2022.103981

[8] Bouman P, Agatz N, Schmidt M. Dynamic programming approaches for the traveling salesman problem with drone[J]. Networks, 2018, 72(4): 528-542.https://doi.org/10.1002/net.21864

[9] Cheng C, Adulyasak Y, Rousseau L M. Drone routing with energy function: Formulation and exact algorithm[J]. Transportation Research Part B: Methodological, 2020, 139: 364-387.https://doi.org/10.1016/j.trb.2020.06.011

[10] Cheng C, Adulyasak Y, Rousseau L M. Formulations and exact algorithms for drone routing problem[M]. CIRRELT, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport= Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation, 2018.

[11] Curtois T, Landa-Silva D, Qu Y, et al. Large neighbourhood search with adaptive guided ejection search for the pickup and delivery problem with time windows[J]. EURO Journal on Transportation and Logistics, 2018, 7(2): 151-192.https://doi.org/10.1007/s13676-017-0115-6

[12] Dorling K, Heinrichs J, Messier G G, et al. Vehicle routing problems for drone delivery[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2016, 47(1): 70-85.https://doi.org/10.1109/TSMC.2016.2582745

[13] Duan L, Zhan Y, Hu H, et al. Efficiently solving the practical vehicle routing problem: A novel joint learning approach[C]//Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020: 3054-3063.https://doi.org/10.1145/3394486.3403356

[14] Feng Wenjing, Lu Fuqiang, Wang Suxin, Bi Hualing, Wang Leizhen, Research on UAV Distribution Path Planning Considering Charging Facilities, Control Engineering, 2022

[15] Fuqiang Lu, Weidong Chen, Wenjing Feng, Hualing Bi, 4PL routing problem using hybrid beetle swarm optimization, Soft Computing, 2023, https://doi.org/10.1007/s00500-023-08378-4.

[16] Fuqiang Lu, Wenjing Feng, Mengying Gao, Hualing Bi, Suxin Wang, The Fourth-Party Logistics Routing Problem Using Ant Colony System-Improved Grey Wolf Optimization, Journal of Advanced Transportation, 2020, 8831746: 1-15.https://doi.org/10.1155/2020/8831746

[17] Gómez-Lagos J, Rojas-Espinoza B, Candia-Véjar A. On a pickup to delivery drone routing problem: Models and algorithms[J]. Computers & Industrial Engineering, 2022, 172: 108632.https://doi.org/10.1016/j.cie.2022.108632

[18] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[19] James J Q, Yu W, Gu J. Online vehicle routing with neural combinatorial optimization and deep reinforcement learning[J]. IEEE Transactions on Intelligent Transportation Systems,

2019, 20(10): 3806-3817.https://doi.org/10.1109/TITS.2019.2909109
[20] Jun S, Lee S. Evolutionary neural network for learning of scalable heuristics for pickup and delivery problems with time windows[J]. Computers & Industrial Engineering, 2022, 169: 108282.https://doi.org/10.1016/j.cie.2022.108282
[21] Jung S, Kim H. Analysis of amazon prime air uav delivery service[J]. Journal of Knowledge Information Technology and Systems, 2017, 12(2): 253-266.
[22] Kool W, Van Hoof H, Welling M. Attention, learn to solve routing problems![J]. arXiv preprint arXiv:1803.08475, 2018.https://doi.org/10.48550/arXiv.1803.08475
[23] Kyriakakis N A, Aronis S, Marinaki M, et al. A GRASP/VND algorithm for the energy minimizing drone routing problem with pickups and deliveries[J]. Computers & Industrial Engineering, 2023: 109340.https://doi.org/10.1016/j.cie.2023.109340
[24] Li M P, Kuhl M E. Design and simulation analysis of PDER: A multiple-load automated guided vehicle dispatching algorithm[C]//2017 winter simulation conference (wsc). IEEE, 2017: 3311-3322.https://doi.org/10.1109/WSC.2017.8248048
[25] Liu Y. An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones[J]. Computers & Operations Research, 2019, 111: 1-20.https://doi.org/10.1016/j.cor.2019.05.024
[26] Lu Fuqiang, Yan Tongren, Bi Hualing, Feng Ming , Wang Suxin, Huang Min, A bilevel whale optimization algorithm for risk management scheduling of information technology projects considering outsourcing, Knowledge-Based Systems, 2022, 235(107600): 1-15
[27] Mahmoudi M, Zhou X. Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state–space–time network representations[J]. Transportation Research Part B: Methodological, 2016, 89: 19-42.https://doi.org/10.1016/j.trb.2016.03.009
[28] Mehlawat M K, Gupta P, Khaitan A, et al. A hybrid intelligent approach to integrated fuzzy multiple depot capacitated green vehicle routing problem with split delivery and vehicle selection[J]. IEEE Transactions on Fuzzy Systems, 2019, 28(6): 1155-1166.https://doi.org/10.1109/TFUZZ.2019.2946110
[29] Mitrović-Minić S, Krishnamurti R, Laporte G. Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows[J]. Transportation Research Part B: Methodological, 2004, 38(8): 669-685.https://doi.org/10.1016/j.trb.2003.09.001
[30] Murray C C, Raj R. The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones[J]. Transportation Research Part C: Emerging Technologies, 2020, 110: 368-398.https://doi.org/10.1016/j.trc.2019.11.003
[31] Nagy G, Wassan N A, Speranza M G, et al. The vehicle routing problem with divisible deliveries and pickups[J]. Transportation Science, 2015, 49(2): 271-294.https://doi.org/10.1287/trsc.2013.0501
[32] Nazari M, Oroojlooy A, Snyder L, et al. Reinforcement learning for solving the vehicle routing problem[J]. Advances in neural information processing systems, 2018, 31.
[33] Neira D A, Aguayo M M, De la Fuente R, et al. New compact integer programming formulations for the multi-trip vehicle routing problem with time windows[J]. Computers & Industrial Engineering, 2020, 144: 106399.https://doi.org/10.1016/j.cie.2020.106399
[34] Okulewicz M, Mańdziuk J. A metaheuristic approach to solve dynamic vehicle routing problem in continuous search space[J]. Swarm and Evolutionary Computation, 2019, 48: 44-61.https://doi.org/10.1016/j.swevo.2019.03.008
[35] Qureshi A H, Nakamura Y, Yoshikawa Y, et al. Intrinsically motivated reinforcement learning for human–robot interaction in the real-world[J]. Neural Networks, 2018, 107: 23-33.https://doi.org/10.1016/j.neunet.2018.03.014
[36] Ropke S, Cordeau J F. Branch and cut and price for the pickup and delivery problem with time windows[J]. Transportation Science, 2009, 43(3): 267-286.https://doi.org/10.1287/trsc.1090.0272
[37] Salama M, Srinivas S. Joint optimization of customer location clustering and drone-based routing for last-mile deliveries[J]. Transportation Research Part C: Emerging Technologies, 2020, 114: 620-642.https://doi.org/10.1016/j.trc.2020.01.019
[38] Shi J, Gao Y, Wang W, et al. Operating electric vehicle fleet for ride-hailing services with reinforcement learning[J]. IEEE Transactions on Intelligent Transportation Systems, 2019, 21(11): 4822-4834.https://doi.org/10.1109/TITS.2019.2947408

[39] Shi J, Gao Y, Wang W, et al. Operating electric vehicle fleet for ride-hailing services with reinforcement learning[J]. IEEE Transactions on Intelligent Transportation Systems, 2019, 21(11): 4822-4834.https://doi.org/10.1109/TITS.2019.2947408
[40] Shi Y, Lin Y, Li B, et al. A bi-objective optimization model for the medical supplies' simultaneous pickup and delivery with drones[J]. Computers & Industrial Engineering, 2022, 171: 108389.https://doi.org/10.1016/j.cie.2022.108389
[41] Tongren Yan, Fuqiang Lu, Suxin Wang, Leizhen Wang, Hualing Bi, A hybrid metaheuristic algorithm for the multi-objective location-routing problem in the early post-disaster stage, Journal of Industrial and Management Optimization, 2022, doi: 10.3934/jimo.2022145
[42] Toth, Paolo, and Daniele Vigo, eds. Vehicle routing: problems, methods, and applications. Society for industrial and applied mathematics, 2014.
[43] Troudi A, Addouche S A, Dellagi S, et al. Sizing of the drone delivery fleet considering energy autonomy[J]. Sustainability, 2018, 10(9): 3344.https://doi.org/10.3390/su10093344
[44] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.
[45] Wang Y, Lei L, Zhang D, et al. Towards delivery-as-a-service: Effective neighborhood search strategies for integrated delivery optimization of E-commerce and static O2O parcels[J]. Transportation Research Part B: Methodological, 2020, 139: 38-63.https://doi.org/10.1016/j.trb.2020.06.003
[46] Wolfinger D, Salazar-González J J. The pickup and delivery problem with split loads and transshipments: A branch-and-cut solution approach[J]. European Journal of Operational Research, 2021, 289(2): 470-484.https://doi.org/10.1016/j.ejor.2020.07.032
[47] Xu X, Li J, Zhou M C. Delaunay-triangulation-based variable neighborhood search to solve large-scale general colored traveling salesman problems[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(3): 1583-1593.https://doi.org/10.1109/TITS.2020.2972389
[48] Xue L, Luo Z, Lim A. Exact approaches for the pickup and delivery problem with loading cost[J]. Omega, 2016, 59: 131-145.https://doi.org/10.1016/j.omega.2015.05.012
[49] Zhen L, Baldacci R, Tan Z, et al. Scheduling heterogeneous delivery tasks on a mixed logistics platform[J]. European Journal of Operational Research, 2022, 298(2): 680-698.https://doi.org/10.1016/j.ejor.2021.06.057
[50] Zheng K, Zhang Z, Song B. E-commerce logistics distribution mode in big-data context: A case analysis of JD. COM[J]. Industrial Marketing Management, 2020, 86(1): 154-162.