Routing Planning for Last-Mile Deliveries Using Mobile Parcel Lockers: A Hybrid Q-Learning Network Approach

First and Corresponding Author

Yubin Liu
Department of Civil and Environmental Engineering
Imperial College London, London, UK, SW7 2AZ
Email: y.liu20@imperial.ac.uk

Co-authors List

Qiming Ye
Department of Civil and Environmental Engineering
Imperial College London, London, UK, SW7 2AZ
Email: qiming.ye18@imperial.ac.uk

Jose Escribano-Macias
Department of Civil and Environmental Engineering
Imperial College London, London, UK, SW7 2AZ
Email: jose.escribano-macias11@imperial.ac.uk

Yuxiang Feng
Department of Civil and Environmental Engineering
Imperial College London, London, UK, SW7 2AZ
Email: y.feng19@imperial.ac.uk

Panagiotis Angeloudis
Department of Civil and Environmental Engineering
Imperial College London, London, UK, SW7 2AZ
Email: p.angeloudis@imperial.ac.uk
ABSTRACT
Mobile parcel lockers (MPLs) have been recently proposed by logistics operators as a technology that could help reduce traffic congestion and operational costs in urban freight distribution. Given their ability to relocate throughout their area of deployment, they hold the potential to improve customer accessibility and convenience. In this study, we formulate the Mobile Parcel Locker Problem (MPLP), a special case of the Location-Routing Problem (LRP) which determines the optimal stopover location for MPLs throughout the day and plans corresponding delivery routes. A Hybrid Q-Learning-Network-based Method (HQM) is developed to resolve the computational complexity of the resulting large problem instances while escaping local optima. In addition, the HQM is integrated with global and local search mechanisms to resolve the dilemma of exploration and exploitation faced by classic reinforcement learning (RL) methods. We examine the performance of HQM under different problem sizes (up to 200 nodes) and benchmarked it against the exact approach and Genetic Algorithm (GA). Our results indicate that HQM achieves better optimisation performance with shorter computation time than the exact approach solved by the Gurobi solver in large problem instances. Additionally, the average reward obtained by HQM is 1.96 times greater than GA’s, which demonstrates that HQM has a better optimisation ability. Further, we identify critical factors that contribute to fleet size requirements, travel distances, and service delays. Our findings outline that the efficiency of MPLs is mainly contingent on the length of time windows and the deployment of MPL stopovers. Finally, we highlight managerial implications based on parametric analysis to provide guidance for logistics operators in the context of efficient last-mile distribution operations.

Keywords: Mobile Parcel Lockers, Urban Logistics, Last-Mile Delivery, Location Routing Problem, Reinforcement Q-Learning
1. Introduction

The recent growth in e-commerce and online shopping transactions has prompted a considerable growth in demand for fast and timely parcel deliveries and underpins a significant increase in last-mile freight activities in urban areas. Global parcel volumes are predicted to reach 211.5 billion by 2022, with B2C parcels accounting for 57% of the total with a 27% annual growth rate (Royal Mail, 2022). Last-mile freight distribution operations occupy around 40% of freight vehicles’ mileage (Macário, 2012). These patterns have been shown to have a knock-on effect on traffic congestion, as well as on the demand for parking spaces (Savelsbergh and Van Woensel, 2016).

One of the solutions that have been recently adopted by logistics operators and online retailers has been the deployment of stationary parcel lockers (SPLs), which enable unattended deliveries, simplifying delivery tours, and therefore reducing fleet size requirements (Deutsch and Golany, 2018). The service radius of such facilities, however, is limited by the walking distance of customers (Iwan et al., 2016). As a result, a very dense SPL network would have to be set up for such a scheme to be effective, requiring considerable capital investment and significant maintenance costs.

The emergence of increasingly capable electric and autonomous vehicle technologies has prompted the proposition of a new concept, namely mobile parcel lockers (MPLs), that has the potential to address some of the above limitations (Figure 1). MPL fleets could therefore respond to changing customer demand patterns, reduce facility number requirements, and increase accessibility for customers (J. Li et al., 2021). MPLs offer three potential advantages in urban parcel deliveries compared to traditional delivery vehicles. Firstly, MPLs are more flexible since they are not restricted to operating in highly congested areas (J. Li et al., 2021). Attempts will be made to facilitate MPL travel on pavements or bicycle lanes at slower speeds without occupying the main road that reducing traffic congestion. Secondly, MPLs can park in small curb side spaces, pavements, or public spaces outside buildings because of their small size, to increase customer convenience and accessibility. Finally, the mobility of MPLs enables them to provide multiple pick-up locations for customers, which has the potential to reduce repeated deliveries due to customer inconvenience. E-commerce operators have launched pilot studies to stimulate the broader application of MPLs in different regions, such as JD.com, Cainiao in China, and Google Mobile Delivery Platform in the US (Myllymaki, 2016). As another example, French car producer Renault introduced an autonomous driving platform which can load a parcel locker (EZ-Pro) that autonomously transfers packages towards customers (Joerss et al., 2016).

![Fig. 1 Illustration of MPLs](image)

Figure 2 provides the schematic operation of MPLs. Logistics operators determine the transient location and stopovers of MPLs based on characteristics of their customers (such as spatial distribution and demand) as well as attributes of parking spaces (such as duration and location). The logistics operators will notify
customers prior to the arrival of MPLs, so they may access the indicated locations to pick up parcels during the MPLs’ dwell time in the parking lots. Following the completion of a current task, the MPL will return to the depot to obtain parcels for the next delivery round.

Several studies have considered the prospect of applying MPLs in last-mile deliveries, but without fully exploiting the ability of this concept to dynamically adapt to varying levels of customer demand. For instance, Orenstein et al. (2019) propose a model of delivering parcels from a depot to multiple service point using MPLs, in which recipients can choose more than one possible pick-up location. However, the author only considers a single period of the delivery round with a deterministic demand pattern. Wang et al. (2020) introduce MPLs to supplement fixed lockers, while aggregating demands from multiple delivery providers to achieve joint distribution. Nevertheless, the model did not consider customers’ spatial distributions and the dynamic arrival of delivery requests. J. Li et al., (2021) proposes a two-echelon delivery network in which MPLs serve as an intermediate layer between couriers and depots, with the assumption, however, that customer locations and demands are static and deterministic. A more flexible version of this problem is studied by Schwerdfeger and Boysen (2020) with their Mobile Parcel Locker Problem (MPLP), where MPLs dynamically change their location in accordance with customer drop-off time/location preferences, and the objective of maximising end-user convenience. The limitation of the model is that the resupply of lockers with parcels is not considered, and the static characteristic of the model formulation make it difficult to capture such dynamic case.

Given the operational shortcomings of the above studies, this paper proposes a new parcel delivery framework that considers changing customers’ demand patterns and parcel resupply, which have not been addressed in the above studies, and identifies the critical factors contributing to MPLs’ efficient operations. As a primary feature of our research, we use MPLs to dynamically transfer parcels between depots and available parking spaces, with special consideration for parking space availability and the drop-off location/timing requests that have been submitted by the customers. We assume customers provide multiple time/location combinations within the planning horizon such that operators can synergistically determine the most appropriate delivery schemes. MPLs are then expected to deliver parcels to customers before returning to the depot so that MPLs can reload the next batch of parcels. Beyond the novel parcel delivery framework proposed above, we consider the dynamic planning technique that can resolve the time window conflict and delivery route adjustment, and we identify it as the critical difference between our model and the aforementioned MPL-related studies.
The solution approaches that have been previously used to solve MPLP include exact methods (Schwerdfeger and Boysen, 2020), traditional heuristics (Li et al., 2021), and meta-heuristics (Wang et al., 2020). However, the capability of the exact method is limited by the NP-hardness of the large problem instances, while heuristic algorithms are vulnerable to the presence of local optima. Reinforcement learning (RL) methods (e.g., Pointer networks, Attention mechanisms) have been previously used in Vehicle Routing Problems (VRPs), which provides the potential to be adopted to solve MPLP. However, these learning techniques are highly dependent on the quality and availability of training samples and their pre-processing, causing low sampling efficiency (Sutton and Barto, 2018). To tackle the weakness of the above techniques, we propose a Hybrid Q-Learning-Network-based Method (HQM) to improve the quality of solutions given its ability to exploit a wide spectrum of individual behaviour (Sutton and Barto, 2018). Further, the global and local search mechanisms are embedded within the HQM to improve local optima dilemma encountered by classic heuristics.

This study aims to develop an HQM that integrates global and local search mechanisms to solve MPLP considering customer demand patterns and time windows constraints. Concretely, the objectives of this paper are:

- **Model establishment**: establish a mixed integer programming model for MPLP that considers changing customers' demand patterns and parcel resupply.
- **Dynamic case modelling**: define dynamic route adjustment strategies to resolve time windows conflicts and on-demand MPLs allocation with varied conflicted scenarios.
- **Solution design (HQM)**: establish a novel encoder & attention-based decoder framework of HQM for improving the training sample dependency and the exploration-exploitation dilemma faced by classic RL methods.
- **Performance measurement and parametric analysis**: evaluate the performance of HQM comparing to an exact approach and a meta-heuristic algorithm (GA) using simulated data (see Appendix E) and define the effect of critical factors against fleet size, MPLs’ travel distance, and service delay. Providing managerial and actionable insights for logistics professions based on parametric analysis.

In achieving these objectives, this paper contributes to the existing literature in the following respects:

- **Theoretical contribution**: to the best of the authors’ knowledge, this research is the first attempt to apply an RL-based approach to solve MPLP. Our analysis reveals that HQM has better optimisation ability than the exact approach in large problem instances. Additionally, the average reward obtained by HQM is 1.96 times greater than that obtained by GA with improved sampling efficiency than meta-heuristic algorithms for solving large-scale MPLP problems regardless of the network type. We therefore aspire that it will serve as the foundation for future studies in the field of location-routing optimisation or VRPs in last-mile deliveries.
- **Algorithmic contribution**: our encoder & attention-based decoder framework of HQM is superior to other existing RL-based frameworks that use coordinates for training input as it can handle more decision parameters (as these parameters can be infinitely extended to the encoder structure) to resolve more complex multi-decision MPLPs and VRPs (e.g., the combined decision of task allocation and execution sequence). The elaborate description of the encoder & attention-based decoder framework will be presented in section 4.3.1 & 4.3.3.
- **Modelling contribution**: we consider the dynamic planning scenario in our model where two route adjustment strategies are developed within the HQM framework to dynamically resolve the time windows conflicts caused by service delay and traffic conditions, which has the potential to be adopted to other RL-based methods in solving time-windows conflicts of VRPs.

5
• **Managerial contribution:** Based on parametric analysis of key factors that influence MPL-based parcel deliveries, we provide managerial insight (Section 5.4.4) from the perspective of logistics operators to stimulate broader adoption of autonomous last-mile delivery modes.

The remainder of the paper is structured as follows. Section 2 provides a brief review of major modelling approaches and algorithms for MPLP deployments. Section 3 describes the problem definition and formulation of a mathematical model for the MPLP. Section 4 presents our proposed solution approach for the MPLP instances. Section 5 presents the results of our experiments and evaluates the performance based on different aspects (e.g., final reward, convergence, computing time), compared with the Gurobi solver and GA-based alternative, used as the baseline. We further identify key influence factors (time windows, network sizes, capacity, etc) from both MPLs and customers’ perspectives towards different cost units of the objective function. Additionally, we provide a managerial implication for future MPL operations based on the parametric analysis of different influencing factors. Finally, the conclusion section provides a summary of findings and highlights potential applications of our MPLP solution.

2. Literature review

This paper belongs to an emerging stream of research that focuses on the utilisation of autonomous vehicle technologies to Location-Routing Problems (LRPs) within the context of urban freight distribution. To the best of the authors' knowledge, the current literature on location-routing planning for MPLs is limited: only four pieces of literature focus on the last-mile deliveries using MPLs (J. Li et al., 2021; Schwerdfeger and Boysen, 2020; Wang et al., 2020; Orenstein et al., 2019). Because our proposed MPL instance involves the aspects of facility location selection and vehicle route planning, we include the following fields related to our research for the purpose of this review: (i) LRPs, (ii) MPLP, and (iii) VRPs with RL approaches.

2.1 Location-routing problems (LRPs)

LRPs comprises two sub-problems: (1) determining the location of several types of facilities (e.g., central depot, transshipment centre), and (2) assigning tasks and determining sequences of execution while fulfilling customer demands, which is modelled as a VRP. Since there is a wide range of literature concerning the LRP in terms of diverse models (e.g., single or multi-echelon network) and solving algorithms (e.g., large neighbourhood search, GA etc), we therefore mainly focus on studies that discuss the solving sequence between the location selection problem and VRP, as this would provide the basis for defining the solution structure for our model. The sequence for solving LRPs in most literature can be classified as sequential, iterative, and hierarchical methods (Nagy and Salhi, 2007).

Sequential methods provide a straightforward, order-based procedure for resolving LRPs: first by determining optimal locations for facilities, followed by determining routes according to the results obtained from the location selection phase, or vice-versa (Perboli et al., 2011). However, the sequential approach ignores the impact of the interaction between the location selection phase and the routing planning phase (Zhao et al., 2018). For instance, although the location determined by the facility location stage is optimal, such facility location may generate a longer route in the following routing planning stage, thus the global optimum cannot be achieved. Given this concern, the solution optimality of sequential methods requires a detailed enumeration of all potential demand patterns of the model, and each demand should be fulfilled within the service radius to compromise the less interaction between the location selection phase and routing planning phase (Salhi and Rand, 1989; Enthoven et al., 2020).

Iterative approaches that plan facility location and vehicle routing simultaneously address the above information insufficient exploitation between the two sub-problems (Jiang et al., 2019). Specifically, the iterative methods can solve the corresponding routing problem while considering the location decisions and assigning one depot to fulfil each customer cluster, or by solving the facility location problem and planning at least one
route for each opened depot (Escobar et al., 2013). Nevertheless, the capability of iterative approaches is limited by the increasing computing complexity of large problem instances and dependency on route estimation accuracy (Kaewploy and Sindhuchao, 2017; Prins et al., 2007).

In terms of hierarchical methods, the facility location problem is solved as the main problem, and the subsequent routing problem is solved as the subordinate problem (Escobar et al., 2013). A heuristic algorithm (e.g., GA, simulated annealing algorithms) is used as the primary algorithm to solve the location problem and then each step refers to the subroutine that solves the routing problem (Albareda-Sambola et al., 2005; Melechovský et al., 2005). However, the limitation is that such hierarchical methods rely on accurate route length estimation, the algorithm may fall into local optimum if the routing length is not predicted well in the location phase (Nagy and Salhi, 2007).

In our case, customer demand changes over time, and the location of parking spaces impacts their location for parcel collection in each stage, whereas we can fulfil a customer as long as they are within the service range of an MPL in one period. To the best of the authors’ knowledge, this characteristic has not been defined as a dynamic facility location problem. Therefore, considering that we have enumerated all demand patterns for each customer within the service radius and each customer will be served at a predefined parking space during the planning horizon, we propose using a sequential approach to reduce computing complexity and route estimation inaccuracy that iterative and hierarchical methods faced in large problem instances.

From the algorithmic perspective, the NP-hard property of the LRPs renders exact solutions inefficient and does not scale well when tackling large problem instances. Under this circumstance, heuristic (Duhamel et al., 2010), and meta-heuristic approaches (Yu et al., 2010) are regarded as suitable methods for solving intermediate and large LRPs instances since those heuristics can search for near-optimal solutions with less computing time. Recent studies of applying heuristics for solving LRP or VRP include: Lagrangian relaxation for logistics vehicle recharging station VRP (Yang et al., 2022), adaptive large neighbourhood search (ALNS) integrated with iterative greedy algorithm and Clarke & Wright saving algorithm (Guo et al., 2022), GA-Simulated Annealing algorithm (Yu et al., 2022a), ALNS for two-echelon VRP with time windows, covering options, and occasional drivers (Yu et al., 2021), and NSGA-II (Heidari et al., 2022). The above studies have revealed that the effectiveness of (meta-)heuristics on improving solution quality for LRP and VRP. Our study, therefore, will adopt a RL-based technique within the sequential solution framework to compromise the computing complexity and solution quality in large problem instances. As a complement for heuristics, we design a global and local search mechanism to improve solution quality.

We summarise the classic studies based on the category of the above solving sequence (Table 1). The main limitations of the reviewed studies in LRPs lie in three aspects: i) lack of exploiting information interaction between the two subproblems, resulting in a considerable deviation from the theoretically attainable optimal solution; ii) ignores the uncertainty from customers (e.g., demands, time slot, changing locations) that may cause missed deliveries; iii) the potential for splitting customers’ demand to reduce total route length. In our case, the limitation (i) and (iii) will be addressed by enumerating all possible pick-up locations/stopovers of customers before route planning phase, thus ensuring all potential pick-up locations of each customer can be considered for delivery scheme. And the demand within the same parking space can be fulfilled by different MPL to minimise the fleet size. To resolve limitation (ii), we develop different route adjustment strategies to reduce service delays caused by time window conflicts.
### Table 1. Summary table of classic LRP studies

| Category          | Reference                  | Main problem/Method                        | Network type/size                                      | Objective                                                                 | Key constraints and features                  |
|-------------------|----------------------------|--------------------------------------------|--------------------------------------------------------|---------------------------------------------------------------------------|-----------------------------------------------|
| Sequential        | Enthoven et al. (2020)     | 2E-VRP-COMIP/MIP, ALNS.                    | CDPs/1 depot, up to 200 customers, 50 lockers, 10 satellites. | Minimise the traveling costs and the connection costs.                  | Fleet size, flow conservation constraints, covering options, vehicle capacity. |
|                   | Perboli et al. (2011)      | CVRP/MIP formulation and valid inequalities. | 2E-DP/Up to 50 customers, 5 satellites.               | Minimises the traveling and handling operations costs.                   | Locker capacity, arc flow of each echelon, demand fulfilment. |
|                   | Zhao et al. (2018)         | CLRP/Cooperative approximation heuristic    | 2E-attended-and-unattended-CDPs/Up to 200 customers, 20 depots | Minimise vehicle routing costs.                                           | Heterogeneous fleet, carbon emissions, demand variations. |
| Iterative         | Eseobari et al. (2013)     | CLRP/Granular tabu search-based heuristics. | DPs/Up to 20 depots, 200 customers.                  | Minimise depot costs and the traveling costs of vehicles.               | Each cluster is assigned to exactly one depot, depot capacity. |
|                   | Prins et al. (2007)        | CLRP/Lagrangean granular tabu search analysis, DPs/Up to 10 depots, 200 customers. | Minimise depot fixed cost and traveling costs.        | Depot and vehicle capacities, subtour elimination, flow balance, demand fulfilment. |
|                   | Duhamel et al. (2010)      | VRPRB/Path relinking algorithm.            | DPs/Up to 5 depots, 200 customers.                   | Minimise depot fixed cost and traveling costs.                          | Depot and vehicle capacities, flow balance, demand fulfilment. |
|                   | Melechovský et al. (2005)  | LRIP/Non-linear MIP, MOGWO, NSGA-II        | Multi-echelon DPs/1 depot, 3 DC, 3 BC, 300 customers. | Minimise fixed, inventory, transportation, and emission costs.          | Demand fulfilment, flow balance, demand fulfilment. |
| Hierarchical      | Nadizadeh and Hosseini Nasab (2014) | CLRP-FD/Stochastic simulation, local search | DPs/Up to 5 depots, 100 customers.                   | Minimise fixed costs of depots and vehicles, traveling and opportunity lost costs | Fuzzy demands, depot and vehicle capacities, deliveries cannot be split, travel distance limitation. |
|                   | Alhareda-Sambola et al. (2005) | LRP/LP, Tabu search                       | DPs/Up to 5 depots, 30 customers.                    | Minimise depot fixed cost and traveling costs.                          | Deliveries cannot be split, each route starts and ends at the depot, vehicle capacity. |
|                   | Yu et al. (2010)           | CLRP/SA                                   | DPs, vehicle without return to depot/Up 4 depots, 318 customers. | Minimise depot fixed and operational costs, vehicle traveling costs.    | Demand fulfilment, vehicle and flow capacities, time-windows, inventory level, order value. |
|                   | Zhang et al. (2019)        | MOVPRPFlexTW/ACO, optimality              | DPs/Solomon benchmarks                                | Minimise routing costs. Maximise customer satisfaction.                 | Vehicle capacity, time windows, demand fulfilment. |

**Notes:** 1) ALNS=Adaptive large neighbourhood search; 2) ACO=Ant colony optimisation; 3) BC=Backup centre; 4) CDP=Collection and delivery problems; 5) CLRP-FD=Dynamic capacitated location-routing problem with fuzzy demands; 6) DC=Distribution centre; 7) DP=Delivery problems; 8) 2E-DP=2-Echelon delivery problems; 9) LRIP=Location, routing and inventory problem; 10) MOGWO=Multi-objective grey wolf optimisation algorithms; 11) MOVPRPFlexTW=Multi-objective VRP with flexible time windows; 12) NSGA-II=Non-Dominated Sorting Genetic algorithm; 13) VRPRB=VRP with route balancing;
2.2 Mobile parcel locker problem

The MPLP is a variant of LRPs, which is a two-stage optimisation problem. Compared to the traditional LRPs that regard the location selection problems as the main problem, the existing MPLP studies focus on the routing optimisation for MPLs (refer to as the main/core problem). Therefore, to solve MPLP, the decision-makers are required to define the optimal stopovers for lockers first, and then plan delivery routes for MPLs accordingly (following the sequential solving structure mentioned in Section 2.1). Schwerdfeger and Boysen (2020) introduced the first MPLP as a mixed-integer programming model that considers customers’ coverage and minimises the MPL fleet size by determining MPL stopovers and their corresponding routes.

While their exact solution approach shows substantial improvements compared to stationary locker systems, several key limitations arise: i) there is a lack of consideration of parcel resupply and stochastic factors (e.g., service delay) in the study. However, these uncertainties exist in real-world operations and will affect the dynamic adjustment of the scheduling strategy. ii) the operation details of a single MPL cannot be tracked, such as the service starting/ending time and the duration at parking spaces. Hence, the unavailability of operation details cannot support an in-deep route optimisation for a single MPL, leading to larger gap towards optimal solution. iii) from the perspective of solving method, the computational efficiency of the proposed exact solution approach is significantly lower (computing time \( \approx 60\text{min} \)) than most evolutionary algorithms in large networks (nodes \( \geq 100 \)).

We identify the difference between our research and Schwerdfeger and Boysen (2020) as follows. We address the aforementioned limitation (i) by introducing service delay within the objective function, followed by different route adjustment strategies to tackle latency. To overcome limitation (ii), we design a state-based structure for solution extraction such that the delivery scheme of each MPL can be interpreted and revised by the proposed HQM. To tackle the last limitation, we integrate HQM with global & local search mechanisms has shown the potential to achieve a better compromise between solution quality and problem size compared with traditional exact and heuristics approaches.

Orenstein et al. (2019) introduced a flexible parcel delivery problem that enables customers to provide couriers with multiple delivery locations; each parcel will only be delivered to a subset of the locker network. The solution approach is based on a savings heuristic combined with tabu search with a joint objective of large neighbourhoods to optimise the delivery task with lower cost and shorter delivery times. The results show that the proposed method could be adapted to dynamic and stochastic environments. However, the model was not adopted for the occasions where customers may change their preferred destinations if parcels fail to be delivered. This limitation is worth discussing since customers may request parcel redelivery if the parcel failed to be performed in the initial delivery round: it is estimated that 42% of orders delivered to consumers with no specific delivery time will cause repeated deliveries (Allen et al., 2017). To address this concern, we enumerate all potential stopovers for customers and consider the overlap of the corresponding time windows when formulating our delivery scenario to ensure that all demands can be met at specific parking spaces.

Wang et al. (2020) integrated the LRPs with MPLs and established a non-linear integer programming model to minimise operating costs. GA was used to simultaneously determine the locations of depots, the number of mobile parcel lockers deployed, and the routes of lockers. The results show that a policy that considers demand aggregation can significantly reduce delivery times while a scheme without demand aggregation would reduce the number of vehicles required.

Finally, J. Li et al. (2021) introduced a two-echelon MPLP, in which locker travel to couriers in the field and transfer parcels between the couriers and the depot, while couriers perform the final delivery without making repeated returns to the depot. A hybrid Clark and Wright heuristic was developed to solve the mixed-integer programming model. The common weakness of the model introduced by J. Li et al. (2021) and Wang et al. (2020) is that all customer demand patterns are assumed to be static in contrast to the stochastic nature
of real-world customer requests, and the proposed method becomes computationally infeasible for larger problem instances.

Based on the reviewed literature of MPLP, we conclude that all existing studies consider deterministic cases in MPLP, where the demand pattern and delivery route are static. The latter shortcoming causes lower optimising performance when stochastic events occur (e.g., demand changes and traffic congestions). However, the predefined route should be dynamically readjusted to respond to those issues. We addressed this issue through the time-consumption-based prediction for the predefined delivery route prior to executing the task, where predictions and route adjustments are carried out by the proposed algorithm in each iteration. We summarise the key features of the above studies in Table 2.

### Table 2. Summary table for the MPLP studies

| Reference | Model/Approach | Network type/size | Objective | Key constraints and features |
|-----------|----------------|-------------------|-----------|-----------------------------|
| Schwerfeger and Boysen (2020) | Dynamic FLP/Domain reduction, MIPs, exact approach | CDPs/Up to 300 customers, 40 locations | Minimise fleet size | Time windows, locker capacity, service radius |
| Orenstein et al. (2019) | CVRP/Saving heuristics and tabu search, LNS | CDPs with alternative delivery points/Up to 50 service points | Minimise fixed and travel costs of lockers, and the penalty of undelivered parcels | Time windows, locker capacity, demand fulfilment, split deliveries are not allowed, fleet size at SP |
| Wang et al. (2020) | CVRP with demand aggregation/MIPs, Multi-layer GA | CDPs with alternative collection points/32 nodes | Minimise transportation and maintenance costs | Time windows, locker capacity, service range, demand fulfilment, fleet size at SP |
| J. Li et al. (2021) | LRHP/Hybrid Clark and Wright heuristic | CDPs with 2-Depots, 200 customers, 4 Echelon/Up to 32 service points | Minimise depot rental, locker deployment and the operational costs | Locker capacity, service radius, demand fulfilment, visiting times at SP |

Notes: 1) FLP=Facility location problem; 2) MIPs=Mixed integer programming; 3) CVRP=Capacitated vehicle routing problem; 4) LNS=Large neighbourhood search; 5) CDP=Collection and delivery problems; 6) SP=Service point

### 2.3 Applications of Reinforcement Learning in Vehicle Routing Problem

RL has recently emerged as a potentially compelling choice in combinatorial mathematical problems due to their ability to better utilise environmental cues when looking for a solution (Cappart et al., 2020; Nazari et al., 2018). In particular, RL methods have been used to solve the VRP, which can be regarded as a subproblem of LRP and MPLP, and are well known to be computationally expensive problems, with practitioners commonly resorting to heuristics and other approximate methods when dealing with large instances in time-sensitive scenarios (Bello et al., 2017). For this reason, this section explores RL solutions for the VRP. We discuss the characteristics of varying RL structures and the design for the key module of the RL framework rather than the detailed implementation procedure to provide insights into the structure design for HQM.

The RL-based methods that have been used to solve VRPs can be classified into two types: i) end-to-end method (e.g., Pointer network, Ptr-Net), and ii) using Deep RL (DRL) to improve iterative search algorithm. End-to-end methods provide a computing efficient and generalised framework for optimising the deep neural network, deriving near-optimal solutions while maintaining well scalability in complex problem instances. However, compared with other combinational optimisation solvers (e.g., LKH3, Google OR Tools, Gurobi, and Concorde), end-to-end methods generate lower solution quality in medium-large problem instances (Wang and Tang, 2021).

DRL-based iterative search frameworks apply a neural network model to learn and select heuristic rules within heuristic algorithms, thereby enhancing the search for solutions and accelerating the speed of new solution generation for classical heuristics in large problem instances. Compared with end-to-end methods, DRL-based iterative search frameworks obtain better solution quality with the expense of generalisation ability.
The trade-off between optimisation ability and computing efficiency in large problem instances is worth considering due to the nature of RL-based methods (Sutton and Barto, 2018).

We notice that most RL studies focus on the simplified training frameworks for the generalised VRPs, and improved iterative methods using DRL have not been widely discussed. Concretely, the typical studies of each classification on RL technique are summarised in Table 3.

| Classification | Study | Model & Training Method | Problem Instance & Performance |
|----------------|-------|--------------------------|-------------------------------|
| End-to-end Method (Pointer Network) | Deudon et al. (2018) | Transformer attention & Critic baseline | 20, 50-TSP: solution obtained better than Bello et al. (2017). 100-TSP: result similar to Bello et al. (2017). |
| | Vaswani et al. (2017) | Transformer attention & Rollout baseline | 100-TSP: solution obtained better than previous research (Bello et al., 2017; Deudon et al., 2018; Nazari et al., 2018; Vinyals et al., 2017). 100-CVRP, 100-SDVRP, 100-OP, 100-PCTSP, SPCTSP: near optimal solved by Gurobi, and better than heuristics algorithms. 20, 50, 250, 500, 1000-TSP: solution obtained better than Bello et al. (2017). 20-TSPTW: solution obtained better than OR-Tools and ant colony optimisation. |
| | Kool et al. (2019) | Graph pointer network; HRL | 100, 150, 200, 500-Multi objective TSP: better than MOEA/D, NSGA-II, and MOGLS. |
| | Ma et al. (2019) | Prr-Net; REINFORCE & Critic baseline & Decompose policy/parameter migration | 100-CVPR: solution obtained better than Vaswani et al. (2017). 100-CVPR: better than multi-heuristics algorithms. 20, 50, 100-CVRP: solution obtained better than Nazari et al. (2018), Vaswani et al. (2017), OR-Tools, and LKH3. Computing speed better than LKH3. |
| DRL-based Iterative Method | Chen and Tian (2019) | Graph attention & PPO | 100-CVPR: solution obtained better than Vaswani et al. (2017). 100-CVPR: better than multi-heuristics algorithms. |
| | Gao et al. (2020) | Transformer attention & REINFORCE | 20, 50, 100-CVRP: solution obtained better than Nazari et al. (2018), Vaswani et al. (2017), OR-Tools, and LKH3. Computing speed better than LKH3. |

Based on the analysis of reviewed literature, we notice that the solution quality of RL techniques will be affected by the performance of sampling, which is determined by the structure of encoder & decoder (Yu et al., 2019) or the interpretation for state (Zhao et al., 2021). Hence, the encoder of our HQM is designed to contain a set of vectors that combine task allocation tuples and task execution sequence tuples, such that provides better initial solution for the main optimisation procedure/loop. Compared with the classic encoder that only uses the coordinates of nodes in traditional pointer networks (Deudon et al., 2018; Vaswani et al., 2017; Vinyals et al., 2017), the HQM’s encoder contains more decision parameters without transforming the original coordinates, improving computing speed and solution quality.

Beyond the above processing in sampling phase, we develop a novel attention-based mechanism to decode the processed vector with better solutions during the state interpretation and transition phase. The optimal strategies are obtained through continuous training with a trial-and-error mechanism. The training process of HQM follows an offline manner and does not rely on historical samples.

Compared with other online training DRL methods with on-policy or off-policy sampling, the offline training in HQM will significantly improve the sample efficiency since the offline data/sample will be generated based on the behaviour policy. To overcome the weakness of solution quality faced by most off-line RL methods, we introduce $\varepsilon - \text{greedy}$, and global and local search mechanisms to balance the exploitation-
3. Problem statement

We propose an MPLP model for parcel deliveries considering customers’ changing locations and time windows. A set of customer nodes $N$ serves as the recipient of parcels from a group of MPLs. For each customer $n \in N$, customer $n$ creates a set of locations $K_n$ during the planning horizon. Each location that a customer visited within the location set ($k \in \{1, 2, \ldots, ||K_n||\}$) is defined by a time interval $[E_{nk}, L_{nk}]$ and a corresponding position $[X_{nk}, Y_{nk}]$. Customer $n$ will pick up parcels from a specific parcel locker if the locker’s current stopover is within the customer’s maximum walking range $\rho_n$.

A set of MPLs is denoted by $M$, which MPLs can change their locations to facilitate parcel pickups by the customer set $N$. The service radius of each MPL $m \in M$ is denoted as $r_m$, and the average travel speed is $v$. We define a set of parking spaces $I$ where $m$ can potentially park to service customers. Each parking space $i \in I$ is represented by a position $[X_i, Y_i]$, and the available parking time windows $[E_i, L_i]$. We use $SV_i$ to denote the time consumption of parking space $i$ for serving each customer. Therefore, for each parking space $i$, the parking time windows $[E_i, L_i]$ can be divided into $||U_i||$ sub-intervals, where $U_i$ represents a set of sub-interval time spans in parking space $i$. Note that $||U_i||$ needs to be rounded up if it is not an integer such that ensure the length of the original available parking time windows $[E_i, L_i]$ is fully considered. Therefore, we can introduce $[e_{ia}, l_{ia}]$ to represent the $a^{th}$ ($a = 1, \ldots, ||U_i||$) sub-interval within $[E_i, L_i]$ in parking space $i$ (i.e., $[e_{ia}, l_{ia}] \subseteq [E_i, L_i]$).

We define a task as the movement of the MPL to a specific parking space $i$, fulfilling the demand within the defined time interval $[e_{ia}, l_{ia}]$ (i.e., Each sub-interval corresponds to one single task). To avoid missed deliveries (e.g., due to traffic congestion) and guarantee feasible parcel handovers at selected stopovers, we define a buffer time $\delta$ to denote the minimum overlap between $[e_{ia}, l_{ia}]$ and $[E_{nk}, L_{nk}]$. By obtaining a set of tasks to be fulfilled at different sub-intervals of all parking spaces $i \in I$, the tasks are assigned to MPLs to perform while considering the execution sequence. The task execution sequence of MPLs, therefore, formatting the VRP. Our proposed MPLP makes the following decisions in sequence:

1) The optimal location of a set of parking spaces $I$, such that customers within the service radius can access MPLs for parcel collection, while the travel distance of customers will not exceed their maximum walking range $\rho_n$.

2) The task assignment scheme for all determined tasks (i.e., by which task to be performed by which MPL), minimising the MPLs deployed.

3) The task execution sequence for all MPLs (i.e., the route of MPLs visiting different parking spaces to perform tasks), considering customers’ time-windows, available parking time, and shorter route length.

Although the customer-to-locker network can be regarded as the second echelon of the model, we do not plan the route between customers and MPLs since each customer will only access the nearest MPL (this only composes a direct shortest path between MPLs and customer locations).

As a result of MPLs’ delay, customers might not be served by MPLs on time. In this case, an additional travel distance arises when modifying predefined delivery routes. This route modification is designed to accommodate the undelivered parcel (the corresponding route adjustment algorithms will be introduced in Section).
Figure 3 illustrates an example of MPLP with three customers and one MPL. The available parking spaces for MPLs are denoted by solid triangles from A to D, and the corresponding available parking time windows are represented by ovals. The three customers are denoted by the solid circle with different numbers from 1 to 3, and their available pickup times at different locations are represented as rectangles. Customers may change location throughout the day as denoted by the dotted arrows. For example, Customer 1 departs from the location covered by Parking Space A to the place covered by Parking Space B, and finally reaches Parking Space D. Customer 2 travels from the location covered by Parking Space C to the destination covered by Parking Space B. Analogously, Customer 3 travels from the location covered by Parking Space C to the destination covered by Parking Space A. The optimal solution in this example is to deploy one MPL, which leaves the depot and visits Parking Space C and Parking Space D in sequence before returning to the start when completion (since the available parking time intervals satisfy both customer time windows and service buffer time).

3.1 Model assumptions
When developing our model, we made the following assumptions:

Assumption 1: It is assumed that the logistics operator will ask customers (via the mobile APP or message) if he/she would like to receive MPL service before parcel deliveries. If the customer accepts, he/she can choose the preferred parking space to collect parcels based on their stopovers within a day. The logistics operator will dispatch the MPL to the predefined parking space to serve the customer according to the optimal dispatch strategy. Since the logistics operators only require the approximate location of the customer prior to the service instead of a real-time and precise trajectory, the data privacy of customers can be well-protected. Other potential incentives of the logistics operators to overcome these concerns, e.g., reduced service fares and a better mechanism for protecting customers’ privacy, is worth considering in practice.

Assumption 2: The time windows of a specific customer at different locations shall never overlap to fit real operations.

Assumption 3: The maximum accepted walking ranges \( \rho_n \) varied from different customers, and we assume \( \rho_n \) is a constant. In addition, the available parking duration varies for different parking spaces due to the different regulations in the region.

Assumption 4: The service charge for parking spaces is not considered. We assume that parking spaces have infinite capacities to accommodate multiple MPLs occupying a parking space. However, the capacity of parking spaces can be adjusted and limited according to real operation.
**Assumption 5:** Each locker is filled only once at the beginning of the planning horizon and will drive back to the depot once the MPL tasks in the current delivery round are completed.

**Assumption 6:** Since parcels are standardised in their maximum dimension and weight by logistics providers, we assume that all parcels fit any compartment of the MPL.

### 3.2 Mathematical model for MPLP

Following the above problem statement and assumptions, the proposed MPLP model consists of three parts (Figure 4). The location selection model determines the optimal location for parking spaces based on the spatial distribution of customers. In this stage, we enumerate all potential stopovers for all customers. We then apply the K-means clustering method to obtain the coordinates of optimal parking spaces and the customer stopovers-parking space pairings, which then serve as part of the inputs for the route planning phase. As the second part of MPLP, the routing planning model aims to determine the delivery routes for MPLs with the objective of minimising the number of lockers deployed, total distance travelled, and service delay while satisfying all customers and meeting time window constraints. To achieve this goal, we first generate the initial feasible solution, which consists of a task assignment scheme and task execution sequence. The generated initial solution will be inputted into the HQM module, where the initial feasible solution will be encapsulated as a state to make it interpretable by the main components of HQM. The main body of HQM includes several crucial optimising components (e.g., encoder & decoder, global & local search mechanism etc.) to obtain better state/solution through iteration. The detailed procedure of the HQM module will be introduced in Section 4.

The notation and the mathematical model for the first two phases are as follows:

**Indices:**

- \( m \): MPL \( m (m \in M) \)
- \( n \): Customer \( n (n \in N) \)
- \( i, j \): Two adjacent parking spaces \((i, j \in I)\)

**Sets:**

- \( N \): Set of customers’ locations
- \( M \): Set of MPLs
- \( I \setminus \{0\} \): Set of parking space (excluding depot 0)
\( k \) The \( k \)th location of a single customer \((k \in K_n)\)
\( a \) The \( a \)th task of parking space \(i\) \((a \in U_i)\)

**Parameters:**

- \( Q \) The capacity of MPLs
- \( r_m \) Service radius of MPL \(m\)
- \( F_m \) Fixed cost of MPL \(m\)
- \( v \) Travel speed of MPLs
- \( h_{nk} \) Coordinates of the \( k \)th location of customer \(n\) \((h_{nk} = [X_{nk}, Y_{nk}])\)
- \( \rho_n \) Maximum walking range of customer \(n\)
- \( c_{ij} \) Unit cost travelling from parking space \(i\) to \(j\)
- \( \phi \) A large indefinite number
- \( SV_i \) Service time of parking space \(i\)
- \( \Psi \) The maximum number of MPLs that can be used

**General Variables:**

- \( A_{ima} \) The time when locker \(m\) arrives at parking space \(i\) to perform the \(a\)th task of parking space \(i\)
- \( q_{ia} \) The total demand to be fulfilled within the \(a\)th sub-interval of parking space \(i\)
- \([e_{ia}, l_{ia}]\) The sub-interval within \([E_i, L_i]\) of the \(a\)th task at parking space \(i\) \((a = 1, \ldots, ||U_i||)\)
- \( d_{ij} \) Distance from parking space \(i\) to \(j\)
- \( d_{in} \) Travel distance for customer \(n\) to parking space \(i\)

**Decision Variables:**

- \( g_i \) Float: coordinates of the parking space \(i\) \((g_i = [X_i, Y_i])\)
- \( b_{ikn} \) Boolean: 1, if the \(k\)th location of customer \(n\) is covered by parking space \(i\); 0, otherwise
- \( x_{m} \) Boolean: 1, if locker \(m\) is used for service; 0, otherwise
- \( \chi_{jm} \) Boolean: 1, if locker \(m\) travel to parking space \(j\) from parking space \(i\); 0, otherwise
- \( z_{ima} \) Boolean: 1, if locker \(m\) fulfils the \(a\)th task of parking space \(i\); 0, otherwise

### 3.2.1 Location model for parking spaces

Following the notation defined in Section 3.2, we establish a location selection model for parking spaces as follows:

\[
\text{Minimise } D\left(\{H_i\}_{i=1}^l\right) = \sum_{i=1}^l \sum_{h_{nk} \in H_i} ||h_{nk} - g_i||^2 
\]

Subject to:

\[
g_i = \frac{\sum_{h_{nk} \in H_i} h_{nk}}{|H_i|} \quad \forall n \in N, i \in I, k \in L_n \tag{1}
\]

\[
\sum_{i=1}^l \sum_{k=1}^{L_n} b_{ikn} = 1 \quad \forall n \in N \tag{2}
\]

\[
\sum_{i=1}^l \sum_{k=1}^{L_n} b_{ikn} = 1 \quad \forall n \in N \tag{3}
\]

\[
d_{in} b_{ikn} \leq r_m \quad \forall n \in N, i \in I, k \in L_n \tag{4}
\]

\[
d_{in} b_{ikn} \leq \rho_n \quad \forall n \in N, i \in I, k \in L_n \tag{5}
\]

\[
d_{in} = ||h_{nk} - g_i|| \quad \forall n \in N, i \in I, k \in L_n \tag{6}
\]

\[
b_{ikn} \in \{0, 1\} \quad \forall n \in N, i \in I, k \in L_n \tag{7}
\]
The objective function (1) minimises the total travel distance between customers and the candidate location of parking spaces, such that obtains the optimal location of parking spaces (the centroid of all customers' locations within the cluster). The constraint (2) ensures that the optimal location of parking spaces within the cluster will be at the centroid of all the customers' locations. Constraint (3) guarantees each stopover of customer \( n \) should be covered by a parking space \( i \) only. Constraint (4) represents the travel distance between customer \( n \) and parking space \( i \) should not exceed the service radius \( r_m \) of MPL if customer \( n \) is covered by parking space \( i \). Constraint (5) represents the travel distance between customer \( n \) and parking space \( i \) should not exceed the maximum walking range of customer \( n \) if customer \( n \) is covered by parking space \( i \). Equation (6) defines the calculation of the travel distance for customer \( n \) to parking space \( i \). Equation (7) and (8) define the domains of the decision variables. Accordingly, the input of the model is the coordinates of all customers \( h_{nk} \), the maximum walking range of customers \( \rho_n \), and the service range of MPL \( r_m \). The output is the coordinates of the optimal parking spaces \( g_i \) and the allocation that each location of customer \( n \) covered by which parking space \( i \).

3.2.2 Route planning model for MPLs
After implementing the location model, the obtained optimal location of parking spaces is used as a part of the input of the route planning model. This allows the routing model to calculate the distance between customers and parking spaces. The route planning model for MPLs is demonstrated as follows:

Minimise  \( f(x_m, y_{ijm}, z_{ima}) = W_1 C_1 + W_2 C_2 + W_3 C_3 \)  

\[
C_1 = F_m \sum_{m=1}^{M} x_m 
\]

\[
C_2 = \sum_{m=1}^{M} \sum_{i=0}^{V} \sum_{j=0}^{V} y_{ijm} \ d_{ij} \ c_{ij} 
\]

\[
C_3 = \sum_{m=1}^{M} \sum_{l=1}^{1} \sum_{a}^{u_i} \ [\max(e_{ia} - A_{ima}, 0) + \max(A_{ima} - l_{ia}, 0)] 
\]

Subject to:

\[
\sum_{m=1}^{M} x_m \leq \Psi 
\]

\[
\sum_{l=1}^{1} \sum_{a=1}^{u_i} z_{ima} \cdot q_{ia} \leq Q \cdot x_m \quad \forall m \in M 
\]

\[
\sum_{m=1}^{M} \sum_{a=1}^{u_i} z_{ima} = 1 \quad \forall i \in I 
\]

\[
\sum_{l=1}^{1} y_{aim} = \sum_{m=1}^{M} x_m 
\]
\[
\sum_{j=1}^{l} y_{jm} = \sum_{m=1}^{M} x_m \quad (17)
\]

\[
A_{0ma} = 0 \quad \forall m \in M, a \in U_i \quad (18)
\]

\[
A_{jma} = A_{ima} + SV_i + d_{ij}/v - \phi (1 - y_{ijm}) \quad i, j \in V; i \neq j; m \in M; a \in U_i \quad (19)
\]

\[
x_m, y_{ijm}, z_{ima} \in \{0, 1\} \quad \forall m \in M; i, j \in V; a \in U_i \quad (20)
\]

The objective function (9) minimises the fleet size, total travel distance and service delay, where \(W_1\), \(W_2\), and \(W_3\) represent the weights of three cost units. Equation (10) to (12) denotes the MPLs’ deployment cost, travel cost, and delay penalty respectively.

Constraint (13) denotes the total quantity of MPLs used should not exceed the predefined value. Constraints (14) represents the total demand fulfilled by locker \(m\) should not exceed the maximum capacity of MPL. Constraint (15) ensures that each task within the parking space \(i\) will be executed only once. Constraints (16) and (17) guarantee MPLs depart from a central depot and will return to the depot when the task is completed, and the number of routes from and back to the depot is equal to the MPL fleet size. Constraint (18) represents the time when MPL \(m\) departs from the central depot is 0. Constraint (19) represents the time relationship of locker \(m\) visit to parking space \(j\) from parking space \(i\). Constraint (20) defines the domains of the decision variables.

4. Methodology

The structure of our proposed solution approach is illustrated in Figure 5. Firstly, we utilise K-means clustering to determine the optimal location of parking spaces and assign different customer locations to the nearest parking space, so that a customer can access MPLs at parking spaces within a day at one of their potential locations. Since locating parking spaces through K-means clustering is not considered to be the main problem in our study, and a number of open-source packages have already been developed for realizing K-means clustering, we will not provide detailed instructions for implementation. Related studies of applying K-means for location selection problems see Kumar and Somasundaram Kumanan (2012) and Pham et al. (2005).

Next, the task generator produces a set of tasks (the total number of tasks equals \(|U_i|\)) for the time windows \([E_i, L_i]\) of each parking space \(i \in I\). MPLs are then allocated to perform each task within the sub-interval \([e_{ia}, l_{ia}] \in [E_i, L_i]\) (as described in Section 3).

Then, the HQM algorithm is developed to determine the following set of decisions: i) the task assignment schemes (denoted as \(x_1\)) that define the task-locker assignment, and ii) the task execution sequence for each MPL (referred as \(x_2\)). The length of \(x_1\) and \(x_2\) corresponds to the total number of tasks (\(|U_i|\)) determined in the task generation phase. A state \(s = (x_1, x_2)\), which consists of a \(x_1, x_2\) pairing, represents a feasible solution of the proposed MPLP. A set of state \(s\) forms an agent \(\pi\). The reward \(R\) of HQM is defined as the reciprocal of the objective function (9) since the HQM is set to maximise the reward. The action \(A_\pi\) is defined as a decision behaviour in which the agent \(\pi\) selects a policy to obtain a better solution in the next timestep \(t\).

The HQM contains an encoder, the global and local search framework, a route generator, and an attention-based decoder. The encoder first produces an agent \(\pi\) through a random generator and permutation pool. To speed up the searching process for the optimal solution, the global search mechanism based on the Q-value matrix and the local search mechanism based on single-state adjustments are used concurrently to generate the new state \(s’\) for the agent \(\pi\).
The updated state $s'$ will then be input into the route generator so that the delivery routes for MPLs can be developed. The decoder is used for calculating the reward $R$ of a state $s$ and decoding the best state $s^*$ such that obtains the optimal task assignment scheme $x_1^*$ and the tasks execution sequence $x_2^*$. The attention layer within the decoder is used to specify the selecting possibility of each element within $x_1$ and $x_2$ that is inherited by the next action $A_{t+1}$. The current best state $s^*$ will then be recorded, and the Q-value matrix will be updated accordingly. Once the convergence condition has been satisfied, the HQM outputs the best state $s^*$.

The encoder generates an agent $π$ that is input to the decoder, which then identifies optimal state $s^*$ for each agent $π$ according to the current knowledge of the problem. In this manner, the optimal task assignment scheme $x_1^*$ and the task execution sequence $x_2^*$ are determined. A description of the detailed structure and essential elements of the HQM approach will be provided in Section 4.3. The following section will describe the detailed mechanisms of the various components within the solution framework in sequential order.

![Solution framework of the model](image)

**Fig. 5** Solution framework of the model

### 4.1 Task Generation

Following the procedure presented in Figure 5, we first demonstrate the process of task generation. The objective of this phase is to identify a set of tasks that satisfy both time-window constraints imposed by customers and parking spaces.
We initially reduce the redundant length of $U_i$ for each parking space $i$. For instance, if a parking space $i$ is available for an MPL from 10:00 to 16:00 ($[E_i, L_i] = [10:00, 16:00]$), but no customer time windows within the range between 14:00 and 16:00 ($[E_{nk}, L_{nk}] = [14:00, 16:00]$), $U_i$ can be reduced since $[E_i, L_i]$ is now only lasts from 10:00 to 14:00. The available time interval of parking space $i$ can be reduced to $U_i'$:

$$U_i' = [E_i, L_i] \cap [E_{nk}, L_{nk}] \quad \forall n \in N; k \in \{1, 2, \cdots, ||L_n||\}; i \in I$$

(21)

To further reduce the range of available time intervals, we determine the earliest time window and the latest time window among all customers within each parking space. The optimal range of available time intervals will be updated as $U_{i}^{\text{opt}}$. By defining $U_{i}^{\text{opt}}$, we obtain a set of sub-intervals (Equation 23) of parking space $i$ when executing $U_{i}^{\text{opt}}$ divided by the service time $S_{V_i}$:

$$U_{i}^{\text{opt}} = \left\{ \left[ \min\{E_{1k}, \cdots, E_{nk}\}, \max\{L_{1k}, \cdots, L_{nk}\}\right] \cap U_i' \quad \forall n \in N; k \in \{1, \cdots, ||L_n||\} \right\}$$

(22)

$$\left[ e_{ia}, l_{ia} \right] = U_{i}^{\text{opt}} / S_{V_i} \quad \forall i \in I; a \in \{1, 2, \cdots, u_i\}; \left[ e_{ia}, l_{ia} \right] \in U_{i}^{\text{opt}}$$

(23)

Thus, the subtask fulfilled by MPL $m$ at parking space $i$ within the $a^{th}$ sub-interval can be denoted by $O_{ima}$ (Equation 24), where $q_{ia}$ denotes the total demand at the $a^{th}$ sub-interval at parking space $i$. The task list of an MPL $m$ is denoted as $O_m = \{O_{1ma}, \cdots, O_{ima}\}$. We use $O (O = \{O_1, \cdots, O_m\}, m \in M)$ presents the task set of all MPLs.

$$O_{ima} = \{(q_{ia}, m, i, [e_{ia}, l_{ia}]) \quad \forall i \in I; m \in M; a \in \{1, 2, \cdots, ||U_i||\} \}$$

(24)

4.2 The structure of a state

By obtaining the tasks list for all MPLs, we can now construct the state of HQM (Figure 6). Each column in the variable $x_1$ and $x_2$ corresponds to a subtask $o \in O$, i.e., the length of either $x_1$ or $x_2$ denotes the total number of tasks $||O||$ to be performed by MPLs. Each element of variable $x_1$ represents the unique identification number (ID) of an MPL $m$ that perform the corresponding subtask. The elements within the variable $x_2$ denote the unique task ID so that forms the task execution sequence. The element of the variable $x_1$ is produce by a random integer generator with the range from 0 to $||M||$, while the elements of the variable $x_2$ is generated within the range from 0 to $||O||$. It is permissible to assign the same MPL to different subtasks $o$, therefore the value of elements can be the same within the variable $x_1$. However, the task ID should never be repeated, as each subtask $o$ is distinct and is executed only once by an MPL $m$.

![Fig. 6 Illustration of a state](image)

4.3 Hybrid Q-Learning network model (HQM)

We introduce an HQM that combines global and local search to generate a set of feasible solutions and obtains the optimal result by updating the Q-value iteratively.

In addition to the state $s = (x_1, x_2) \text{ defined previously, we define } \pi = (s_1, \cdots, s_p) \text{ as an agent that}$

.. $\text{consists of multiple states } s, \text{ which provides a set of feasible solution for the problem. The action } A_{\pi}$

.. $\text{is defined as a decision behaviour in which the agent } \pi \text{ selects a policy to obtain a better solution/state } s \text{ in the next}$

.. $\text{timestep. } A_{\pi} \text{ is sampled from the right-hand side of Equation (26) during training time and obtained by } \varepsilon = \ldots$
greedy search of the Q-Learning and global-local search mechanisms. Our goal is to maximise the reward $R$ (Equation 25) obtained from a set of generated MPL route $G$:

$$R(\pi|G) = \max \left\{ \frac{1}{f_{s_1}(x_m, y_{ijm}, z_{ima})}, \ldots, \frac{1}{f_{s_p}(x_m, y_{ijm}, z_{ima})} \right\} \quad s_p \in \|\pi\|$$

(25)

The proposed HQM is parameterised by $\theta_{x_1}$ and $\theta_{x_2}$, corresponding to $x_1$ and $x_2$ respectively. It updates a strategy $p(\pi|G)$ that promotes agent $\pi$ deriving the optimal solution/state $s^\ast$. It is a stochastic policy and can be factorised as:

$$p_{\theta_{x_1},\theta_{x_2}}(\pi|G) = \prod_{t=1}^{t} p_{\theta_{x_1},\theta_{x_2}}(\pi(t)|\pi(<t), G)$$

(26)

To maximise the objective in Equation (25), we update every component on the right-hand side of Equation (26) based on the Q-value matrix in every timestep $t$. A new/updated state $s'$ in the next timestep $t+1$ will resemble the current state $s$ with the maximum Q-value in the current timestep $t$, indicating that the new state $s'$ retains the elements that contributed significantly to the optimal solution in the current timestep $t$. To accelerate the search for the optimal solution/state $s^\ast$, we generate multiple agents $\pi$ for the global search.

4.3.1 Encoder

We develop an encoder that consists of a random generator and a permutation pool to produce the agent $\pi$ as a normalised input of HQM. As shown in the elements in Figure 7, each element within $x_1$ and $x_2$ is represented by $\theta_x^o$ ($x \in \{x_1, x_2\}$, $o \in \{1, \ldots, O\}$). The value next to the elements of $x_1$ and $x_2$ represents the probability of each current element that is remain selected by the current subtask $o$ in the next timestep $t + 1$, and will be updated by the attention layer embedded within the decoder at each timestep $t$ (will introduce in Attention-based decoder Section).

Fig. 7 Structure of the encoder
4.3.2 Route generation & adjustment strategies

In this section, we describe a route generation mechanism based on the agent $\pi$ produced by the encoder. Specifically, each state $s$ within the agent $\pi$ corresponds to the route for MPLs. However, since the state $s$ is generated randomly, the resulting route may violate time-window requirements of tasks. We then introduce route adjustment strategies to resolve such time-window conflicts that occur during the route generation phase.

**Mechanism of route generation**

The route generation assigns and sequences tasks for MPLs based on the principle of subtask-MPL paring within the state $s$ (as shown in Figure 6). The resulting route will be applied in the global & local search mechanism (Section 4.3.4) for adjustment such that obtain better solution (as shown in the global & local search diagram in Figure 5). To capture the route generation process for MPLs, we modified the lemma developed by Schwerdfeger and Boysen (2020) and prove that for any given subtask $O_{ima}$, the task starting/ending time can be adjusted to execute the task as early as possible such that forms an optimal task $O^*_{ima}$.

**Lemma 1.** For any subtask $O_{ima}$ defined in (24), there is an optimal task starting time $A^*_{ima}$ and ending time $\hat{A}^*_{ima}$ adjusted from $O_{ima}$ to form an optimal task $O^*_{ima}$, where $A_{ima}$ and $\hat{A}_{ima}$ are as follows:

i) task starting time $A^*_{ima} = \max\{\hat{A}^{(i-1)ma} + t_{(i-1)i}, e_{ia}\}$ $\forall m \in M; a \in \{1, \ldots, ||U_i||\}, e_{ia} \in [e_{ia}, l_{ia}]$;

ii) task ending time $\hat{A}^*_{ima} = \max\{\max\{A_{ima}, E_{nk}\} + SV_i\}$ $\forall m \in M; n \in N; a \in \{1, \ldots, ||U_i||\}; E_{nk} \in [E_{nk}, L_{nk}]$.

**Proof:** To simplify the process, we assume that each customer $n$ only has a single location ($K_n = 1$), but our proof is still valid for the multiple-locations situation. Following the nomenclature in Section 3.2, the optimal starting time of MPL $m$ in the first parking space is:

$$A_{1ma} = \max\{e_{1a}, t_{01}\} \quad \forall m \in M; a \in \{1, \ldots, ||U_i||\}; e_{1a} \in [e_{1a}, l_{1a}]$$

(27)

Since the MPL $m$ departs from the depot to the first parking space, the optimal starting time of MPL $m$ is the maximum value between the travel time and the earliest available task starting time in the first parking space.

We assume that the MPL $m$ serves the customers $n$ as early as possible. Thus, MPL $m$ can leave the first parking space as soon as the task is completed:

$$\hat{A}_{1ma} = \{\max\{A_{1ma}, E_{nk}\} + SV_1\} \quad \forall m \in M; n \in N; a \in \{1, \ldots, ||U_i||\}; E_{nk} \in [E_{nk}, L_{nk}]$$

(28)

Similarly, the earliest task starting time in the second parking space are:

$$A_{2ma} = \max\{\hat{A}_{1ma} + t_{12}, e_{2a}\} \quad \forall m \in M; a \in \{1, \ldots, ||U_i||\}, e_{2a} \in [e_{2a}, l_{2a}]$$

(29)

$$\hat{A}_{2ma} = \{\max\{A_{2ma}, E_{nk}\} + SV_2\} \quad \forall m \in M; n \in N; a \in \{1, \ldots, ||U_i||\}; E_{nk} \in [E_{nk}, L_{nk}]$$

(30)

Following the above logic, the earliest task starting $A^*_{ima}$ and ending time $\hat{A}^*_{ima}$ for each sub-task $O_{ima}$ can be represented as:

$$A^*_{ima} = \max\{\hat{A}^{(i-1)ma} + t_{(i-1)i}, e_{ia}\} \quad \forall m \in M; a \in \{1, \ldots, ||U_i||\}, e_{ia} \in [e_{ia}, l_{ia}]$$

(31)

$$\hat{A}^*_{ima} = \max\{\max\{A_{ima}, E_{nk}\} + SV_i\} \quad \forall m \in M; n \in N; a \in \{1, \ldots, ||U_i||\}; E_{nk} \in [E_{nk}, L_{nk}]$$

(32)
Route adjustment strategy and route generation algorithm

The route generator described in the previous section does not consider the capacity constraint and stochastic events (e.g., traffic congestion, service delay) in real implementation, which violates the constraint outlined in Equation (14) and the time-window requirement of a single task, respectively. Thus, a route adjustment strategy is necessary to ensure the feasibility of all MPL routes. Algorithm 1 outlines the procedure of route generations and adjustments.

The route adjustment strategy consists of two algorithms: a back-to-depot (BTD) strategy, and a holding-at-current-parking-space (HCPS) option. The former allows the MPL to reload parcel and postpone deliveries to meet the time-window and capacity constraints. However, this strategy results in an extra distance, impacting the operating costs and delivery efficiency. This process is described in Equation (33) where \( t_{i0} \) and \( t_{0j} \) represent the time consumption of MPL from parking space \( i \) to parking space \( j \) via the central depot.

The latter enables MPLs to wait at their current parking space, postponing the start time of the next task. Thus, the MPL \( m \) will remain at the current parking space \( i \) until the latest ending time \( l_{ia} \) of the \( a \) th sub-interval, as shown in Equation (34). In contrast to BTD, HCPS is only used to resolve time-window conflicts.

\[
A_{jma} = A_{ima} + SV_i + t_{i0} + t_{0j} \quad \forall i, j \in I; m \in M; a \in \{1, 2, \ldots, ||U_i||\} \quad (33)
\]

\[
A_{jma} = l_{ia} + t_{ij} \quad \forall i, j \in I; m \in M; a \in \{1, 2, \ldots, ||U_i||\} \quad (34)
\]

**Algorithm 1**: Route generation and adjustment algorithm

Input: Task set \( O_m \) for \( m \in M \); Locker speed \( v \); Locker capacity \( Q \)

output: The delivery route of MPLs

1. Initialise \( A_{ima}, q_{ia}, e_{ia}, l_{ia}, E_i, L_i, SV_i \)
2. Calculate travel time \( t_{ij} \) between parking space \( i \) and \( j \) for all parking spaces
3. for each subtask \( O_{ima} \) in \( O_m \) do
   4. Add depot to current task list \( O_m \) // Depart from depot
   5. visited_route_list \( \leftarrow \emptyset \) // Record the route of MPLs
   6. for next in \( O_{ima} \) do
      7. current_parking_space \( \leftarrow O_m[O_{ima}] \) // Map the current task of MPL \( m \) to parking space
      8. next_parking_space \( \leftarrow O_m[\text{next}] \) // Map the next task of MPL \( m \) to the next parking space
      9. Add current_parking_space to visited_route_list
      10. Update \( A_{ima}, q_{ia}, e_{ia}, l_{ia} \)
      11. travel_time \( \leftarrow \text{ialg.norm}(\text{current_parking_space} - \text{next_parking_space}) / v \)
      12. if \( A_{ima} + SV_i + t_{ij} < e_{ja} \ and \ l_{ia} + t_{ij} \geq e_{ja} \) then // Violate time-window constraint of tasks
          Adjust task starting and ending time based on Equation (34) // Apply HCPS
          Update \( A_{jma}, q_{ja}, e_{ja}, l_{ja}, E_j, L_j, SV_j \)
          Add next_parking_space to visited_route_list
      13. else if \( A_{jma} < e_{ja} \ or \ q_{ia} > Q \) then // Violate time windows or constraint (14)
          Adjust task starting and ending time based on Equation (34) // Apply BTD
          Update \( A_{jma}, q_{ja}, e_{ja}, l_{ja}, E_j, L_j, SV_j \)
          Add next_parking_space to visited_route_list
      14. else if satisfy constraints then
          Add next_parking_space to visited_route_list
      15. end if
The attention-based decoder is developed to calculate the reward $R$ of the route generated in Mechanism of route generation Section and updated the probability of each current element that is remain selected by the current subtask $o$ in the next timestep $t + 1$ through the Q-value matrix embedded within the attention layer (see Mechanism of attention layer Section). The reward calculation is based on Equation (25), given the route generated.

Figure 8a to 8d demonstrate the decoding procedure for an agent $\pi$. As shown in Figure 8a, the decoder initially decomposes the first state $s_1$ in the timestep $t + 1$ by applying a decoding algorithm (see Appendix A) and calculating the corresponding reward $R$, followed by updating the probability of each element within the state $s_1$. Analogously, the decoding procedure lasts until all states within the agent $\pi$ have been completed (Figure 8b and 8c). Finally, all states within the decoder are sorted according to the reward obtained, and the optimal state $s^*$ is then extracted to determine the optimal delivery route by executing Algorithm 1 (Figure 8d).

Mechanism of attention layer

The state-action space of HQM can be represented by a lookup table (Figure 9). The size of the lookup table equals $O \times T$, where $O$ equals the length of the task list of a state $s$, while $T$ equals the total number of steps/episode of the whole HQM process. Given this property, the size of the lookup table increases with the problem size exponentially.

The attention mechanism is developed to tackle such a curse of dimensionality, in which we decompose the state-action space into multiple low-dimensional state-action combination chains linked by the Q-value matrix. We define two Q-value matrices in Equations (35) and (36), which correspond to $x_1$ and $x_2$ within a state $s$, respectively. Each value within the matrix reflects the contribution of the element to the final reward.

Furthermore, each value within the matrix indicates the interdependence between two adjacent actions, where the larger the value, the closer the two adjacent actions are connected. The new state $s'$ updated in the
next action $A_{t+1}$ inherits the elite elements (the elements that contribute to the greater reward) of the current state $s$, which have greater $Q$-values in the current action $A_t$. Therefore, the solution can be enhanced while inheriting the elite elements of the current state $s$.

$$Q(\theta^o_{x_1}) = \begin{bmatrix} Q_{11}(\theta^1_{x_1}) & \cdots & Q_{1m}(\theta^0_{x_1}) \\ \vdots & \ddots & \vdots \\ Q_{m1}(\theta^1_{x_1}) & \cdots & Q_{mm}(\theta^0_{x_1}) \end{bmatrix} \quad \forall o \in \{1, \ldots, O\}, m \in M \quad (35)$$

$$Q(\theta^o_{x_2}) = \begin{bmatrix} Q_{11}(\theta^1_{x_2}) & \cdots & Q_{1o}(\theta^0_{x_2}) \\ \vdots & \ddots & \vdots \\ Q_{o1}(\theta^1_{x_2}) & \cdots & Q_{oo}(\theta^0_{x_2}) \end{bmatrix} \quad \forall o \in \{1, \ldots, O\} \quad (36)$$

On the basis of the $Q$-value matrices defined above, we can determine the possibility of the current element being selected for the current task $o$ in the following timestep $t + 1$, denote as $p(\theta^o_{x_1})$ and $p(\theta^o_{x_2})$. These are calculated as follows:

$$p(\theta^o_{x_1}) = \frac{\exp(Q(\theta^o_{x_1}))}{\sum_{o=1}^O \exp(Q(\theta^o_{x_1}))} \quad o \in \{1, \ldots, O\} \quad (37)$$

$$p(\theta^o_{x_2}) = \frac{\exp(Q(\theta^o_{x_2}))}{\sum_{o=1}^O \exp(Q(\theta^o_{x_2}))} \quad o \in \{1, \ldots, O\} \quad (38)$$

**Mechanism of Q-value update**

To update the $Q$-value matrix in every timestep $t$, we propose the following $Q$-value function for each element of a state $s$ within an agent $\pi$:

$$Q^\theta_{t+1}(s^x_t, a^x_t) = Q^\theta_t(s^x_t, a^x_t) + \alpha \left[ R_t(s^x_t, s^x_{t+1}, a^x_t) + \gamma \max_{a^x_{t+1} \in A_t} Q^\theta_t(s^x_{t+1}, a^x_t) - Q^\theta_t(s^x_t, a^x_t) \right] \quad (39)$$

where $x_t \in \{x_1, x_2\}$ and $o \in O$, $R_t(s^x_t, s^x_{t+1}, a^x_t)$ represents the reward obtained from state $s^x_t$ by executing action $a^x_t$ towards a state $s^x_{t+1}$ in timestep $t$, $\alpha$ represents the learning rate, and $\gamma$ represents the discount factor.

The update procedure can be regarded as an MDP that derives satisfactory results through updating the $Q$-value between two adjacent actions. For each action $a^x_t \in A_t \quad (t \in \{1, \ldots, max\_time\}, x_t \in \{x_1, x_2\}$ of the variable, there is a corresponding $Q$-value ($Q^\theta_{x_1}$ and $Q^\theta_{x_2}$) associated with action $a^x_t$. We use $A_t$ to denote the action profile that integrates both actions of variable $x_1 (a^x_1) and x_2 (a^x_2)$ in timestep $t$ (i.e., $A_t = a^x_1 \cup a^x_2$).

The action $A_t$ selects the direction of migration that will seek a better solution in the next timestep $t + 1$ based on a $Q$-value such that migrate into the new policy. Once $A_t$ is determined and executed by the agent $\pi$, and the next action $A_{t+1}$ will be selected according to the estimated $Q_{t+1}$ (Figure 9).
4.3.4 Policy for action selection

Q-Learning suffers from the trade-off between exploitation and exploration of the information feedback obtained from previous actions (Sutton and Barto, 2018). A strong emphasis on information exploration may derive the optimal solution but may sacrifice the speed of convergence. While favouring information exploitation can improve convergence speed at the expense of optimality.

As part of our approach, we propose integrating the Q-value-based global search with a local search approach to defining the policy for action selection, leading to a near-optimal solution while improving convergence speed. Concretely, the global search mechanism first updates each element within the state $s$ of the agent $\pi$ to generate a new state $s'$ given a specific action $A_t$, in which the resulting reward is denoted as $R'$. Next, the local search mechanism adjusts the subset of elements within an individual state $s'$ updated by the global mechanism to create another new state $s''$, in which the resulting reward is represented as $R''$. The HQM outputs either $s'$ or $s''$ for the final solution follows $\max \{R', R''\}$.

By implementing a combined global and local search approach, the agent $\pi$ and corresponding Q-value matrices will be updated accordingly. The iteration stops once the error of the Q-value matrices generated by two adjacent actions $A_t$ and $A_{t+1}$ is less than $1 \times 10^{-8}$. The state $s^*$ with the greatest reward $R^*$ among all agent $\pi$ is defined as the optimal solution of MPLP.

In terms of global search mechanism, the migration between the two adjacent actions $A_{t-1}, A_t$ updates the agent $\pi$ based on the Q-value obtained from the previous action $A_{t-1}$. The policy of action selection is based on the $\varepsilon-\text{greedy}$ search which can be defined as follows:

$$a_{t+1}^{s_t} = \begin{cases} \arg\max_{a_t^i \in \Theta_t} Q_t^{a_t^i, s_t} & \tau < \varepsilon \\ a_t^{x_t}, & \tau \geq \varepsilon \end{cases}$$

(40)

Where $\varepsilon$ denotes the greedy factor, $\tau$ denotes the random value between 0 and 1, and $a_t^{x_t'}$ denotes the action that is being selected based on the normalised Q-value matrices within the global scope. The greater the Q-value is, the greater probability of the corresponding element inheriting an elite fragment of the current state $s$.

The local search operation focuses on adjusting the subset of the variable $x_1$ and $x_2$ within an individual state $s_t$ to create a new state $s_t'$:

$$s_t'(x_1, x_2) = s_t(x_1, x_2) + \bar{\omega} \cdot (s_t(x_1, x_2) - s_{\text{neighbour}}(x_1, x_2))$$

(41)
where \( s_{\text{neighbour}} \) represent the neighbour state in \( \{ s_{i+1}, s_{i-1} \} \). \( \overline{\omega} \) is a random vector, where each element \( \theta \) and \( \theta' \) within the two corresponding neighbour state \( s_{\text{neighbour}} \) is assigned to a value based on the following piecewise function. This allows the modification of only subsets of the elements in \( x_1 \) and \( x_2 \).

\[
\overline{\omega} = \begin{cases} 
-1 & \text{if } \theta < \theta' \\
0 & \text{otherwise}
\end{cases}
\]  

(42)

The implementation procedure of HQM is shown in Algorithm 2.

**Algorithm 2:** Hybrid Q-Learning network method (HQM)

Input: Number of MPLs \( |M| \); All task list \( O \); Number of agents \( Agent \); Timestep \( T \)

Output: The optimal state \( s = (x_1, x_2) \)

1. **Initialise learning rate** \( \alpha \), discount factor \( \gamma \), random value \( \tau \), greedy factor \( \epsilon \), Q-value matrices \( Q_{t}^{\theta}, Q_{t}^{\theta'} \)
2. **for** \( i \) in range(\( Agent \)) **do** // Initialise agents
   3. \( x_1 \leftarrow \mathbb{N} \sim U(0, |M|) \)
   4. \( x_2 \leftarrow \text{Permutating task list } O \)
   5. \( s = (x_1, x_2) / / \text{Generate initial state } s \)
   6. Execute **Decoding algorithm, Algorithm 1**
3. **end** for
4. Rank and record all state \( s \) based on the reward obtained
5. **for** \( t \) in range(\( T \)) **do**
   6. Record the current best state \( s^* \) and the corresponding reward \( R^* \) of the agent \( \pi \)
   7. **for** \( o \) in range(\( O \)) **do**
   8. **Normalise** Q-value matrices \( Q_{t}^{\theta}, Q_{t}^{\theta'} \)
   9. **end** for
   10. **for** \( i \) in range(\( Agent \)) **do** // Global search
       11. **for** \( o \) in range(\( O \)) **do**
           12. \( s'(x_1', x_2') \leftarrow \text{Equation (41)} / / \text{Implement global search and obtain new state } s' \)
           13. Execute **Decoding algorithm, Algorithm 1**
           14. **end** for
       15. \( R' \leftarrow \text{Equation (25)} / / \text{Calculate the reward } R' \text{ for the new state } s' \)
       16. **if** \( R' > R^* \) **then**
           17. \( s^*, R^* \leftarrow s', R' / / \text{Update the optimal state } s^* \text{ and reward } R^* \)
           18. **end** if
       19. **end** for
   20. **for** \( i \) in range(\( Agent \)) **do** // Local search
       21. \( s''(x_1'', x_2'') \leftarrow \text{Equation (41)} / / \text{Implement local search and obtain another new state } s'' \)
       22. Execute **Decoding algorithm, Algorithm 1**
       23. \( R'' \leftarrow \text{Equation (25)} / / \text{Calculate the reward } R'' \text{ for the new state } s'' \)
       24. **if** \( R'' > R^* \) **then**
           25. \( s^*, R^* \leftarrow s'', R'' / / \text{Update the optimal state } s^* \text{ and reward } R^* \)
           26. **end** if
       27. **end** for
   28. **end** for
   29. \( Q_{t}^{\theta}, Q_{t}^{\theta'} \leftarrow \text{Equation (39)} / / \text{Update Q-value matrices} \)
30. **end** for
if satisfy convergence constraints then
  break
end if
end for
return $s^*$

5 Results and discussion

In this section, we evaluate the solving effectiveness of the HQM for the proposed MPLP. To validate the proposed mathematical model and identify the solution gap between HQM's derived result and the exact approach counterpart, we first compare HQM's derived solution and computation time with the exact approach. We then apply a Wilcoxon signed-rank to evaluate the result significance between exact approach and HQM. If the results derived by the two methods are significant, the correctness of the model can be asserted. Since the main purpose of our study is to emphasise the advantage of HQM in solving large problem instances compared with meta-heuristic approaches, we next compared the proposed HQM with GA based on the following criteria: a) the performance indicators derived from the objective function (Equation 9), b) the demand fulfilment rate, c) the performance of the solution improvement, d) reward acquisition, e) learning convergence speed and f) the computing efficiency. The HQM vs. GA experiments are all executed under both BTD & HCP strategies are applied.

Beyond the performance comparison between HQM and GA, we further investigate the effect of key factors on influencing different cost units of the objective function. Lastly, we provide managerial insights and fruitful discussions regarding the implementation of our method to benefit academia and the logistics profession.

5.1 Generation of problem instance and parameter setting

Benchmark instances commonly used for VRP experiments (e.g., Solomon problem instances) are not suitable for our MPLP, since the latter involves customers’ trajectory within a day, which is out of the scope of the VRP. On the other hand, since previous MPLP studies (Li et al., 2021; Schwerdfeger and Boysen, 2020) neither consider the dynamic route adjustment technique as ours nor disclose the data set of problem instances and precise optimal solution of different cost units, we cannot compare the derived results obtained by HQM with the existing studies under the same dimension. Therefore, we create our own problem instance and analyse the impact of network size on the deployment and operation of MPLs. For the original data and detailed solutions, see Appendix E & F.

In total, 96 problem instances are evaluated based on 24 networks. Each network has a predefined number of MPL parking locations (ranging from $I_l = 5$ to $I_l = 10$), and a set of customer locations $N_n$ close to these $l_i$ ($N_n = \{5, 10, 15, 20\}$).

The HQM and GA code are implemented in Python 3.8.5 on an Apple M1 (3.20 GHz) processor with 16GB RAM, while the exact approach is implemented on the same machine via Gurobi solver. To ensure that the results are comparable, we configured the two algorithms’ implementation under the same timestep/iteration and agent/population size, and with similar parameters in terms of learning rate/elite rate and mutation rate to the degree possible. To contribute to future MPLP and VRP studies and to ensure the rigour of our research, we provide an open-source package of HQM experiments (https://github.com/Yubin-Liu/Hybrid-Q-Learning-Network-Approach-for-MPLP.git). The corresponding HQM and GA parameters are presented in Table 4. The setup of the problem instance is as follows:

- **Generation of customer demands:** The number of parcels per customer $n \in N$ is defined as a uniform random distribution $U \sim (1, 4)$. 
• **Generation of locations:** The location of the parking spaces $I_i$ are generated randomly within a grid, such that $x_i \sim U(0, x_{\text{max}})$, and $y_i \sim U(0, y_{\text{max}})$. For each location of customer $n \in N$, the coordinates are randomly generated within a specified MPL radius $r_n$ of a parking space $i \in I_i$, in other words $d_{ni}^k \leq r_m$ given that $d_{ni}^k = \sqrt{(x_{nk} - x_i)^2 + (y_{nk} - y_i)^2}$, where $k$ represents the $k$th stopover of the customer $n$.

• **Customer stopover generation:** The number of stopovers $|K_n|$ of a customer $n$ within the planning horizon follows $|K_n| \sim U(1, 3)$. The maximum walking distance $\rho_n$ (km/customer) of customer $n$ is generated by following normal distribution $\rho_n \sim N(0.5, 0.1)$ (Schwendtberger and Boysen, 2020).

• **Generation of time interval:** We generate non-overlapping time intervals for each customer $n$. The time interval $[E_{nk}, L_{nk}]$ of the $k$th locations of a customer $n$ is generated by a random generator, in which $E_{nk}, L_{nk} \in T_{\text{list}} = \{8 \times 60, \ldots, 18 \times 60\}$ (corresponding to working hours from 8:00 to 18:00) with the length of $T_{\text{list}}$ equals 10. Meanwhile, we guarantee that the interval length of time window $|L_{nk} - E_{nk}|$ follows normal distribution $|L_{nk} - E_{nk}| \sim N(60, 5)$, subject to $L_{nk} > E_{nk}$. The generation of parking spaces’ time intervals is similar to the above procedure with an interval length $|L_i - E_i|$ following normal distribution $|L_i - E_i| \sim N (50, 5)$. We ensure that at least one parking space suits each customer.

### Table 4. Parameters setting in model validation

| Parameter in mathematical model | Parameter | Symbol | Value |
|---------------------------------|-----------|--------|-------|
| Service radius of MPLs         | $r_m$     | [km]   | 5     |
| Capacity of MPLs               | $Q$       | [unit] | 20    |
| Travel speed of MPLs           | $v$       | [km/hr]| 40    |
| Fixed cost of locker $m$       | $F_m$     | [$/locker] | 20,000 |
| Unit cost travelling from parking space $i$ to $j$ | $C_{ij}$ | [$/km] | 0.5 |
| A large indefinite number      | $\phi$   | [-]    | $10^8$ |
| Service time of parking space $i$ | $SV_i$ | [min] | 10 |
| Weights of cost units          | $W_1$, $W_2$, $W_3$ | [-] | 10:1:5 |

### Parameters in customer properties

| Parameter | Symbol | Value |
|-----------|--------|-------|
| Maximum walking range $\rho_n$ | $\rho_n$ | [km] | $\rho_n \sim N(0.5, 0.1)$ |
| Length of time interval of customer | $T_s$ | $|L_{nk} - E_{nk}|$ [min] | $T_s \sim N(60, 5)$ |

### Parameters in parking spaces

| Parameter | Symbol | Value |
|-----------|--------|-------|
| Length of time interval of parking spaces | $T_p$ | $|E_i, L_i|$ [min] | $T_p \sim N(50, 5)$ |
| Service buffer time | $\delta$ | [min] | 10 |

### Parameters in HQM

| Parameter | Symbol | Value |
|-----------|--------|-------|
| Number of actions/timestep | $A_t$ | [-] | 1000 |
| Learning rate | $\alpha$ | [-] | $0.9 \times e^{-current\ timestep \cdot \frac{1}{total\ timestep}}$ |
| Discount factor | $\gamma$ | [-] | 0.9 |
| Greedy factor | $\epsilon$ | [-] | $U(0,1)$ |
| Random variables | $\tau$, $\omega$ | [-] | $U(0,1)$ |
| Total states within an agent | $\lambda_{\text{agent}}$ | [-] | 100 |

### Parameters in GA

| Parameter | Symbol | Value |
|-----------|--------|-------|
| Number of iteration/timestep | Iteration | [-] | 1000 |
| Population size | Pop | [-] | 100 |
| Possibility of crossover | $P_{\text{cross}}$ | [%/Pop] | 50 |
| Proportion of elite size | $P_{\text{elite}}$ | [%/Pop] | 5 |
| Mutation rate | $P_{\text{muta}}$ | [%/Pop] | 5 |

**Notes:** MPLs’ fleet size and service delay are given higher weightings because the deployment of MPLs requires high initial costs, while the service delay has a significant effect on the market share for parcel deliveries (Iwan, Kijewska and Lemke, 2016). Nevertheless, the weights of different cost units may be adjusted in accordance with real operations.
5.2 Performance comparison between HQM and exact approach

We utilise the Gurobi solver to obtain solutions from solving the problem instances via the exact approach, and the obtained results are compared with the HQM. The solver was terminated after 2 hours or when it has found an optimal solution. The comparison is evaluated based on two aspects: i) the best value of the objective function, cost units, and the computation time; ii) the significance between the results obtained by Gurobi solver and HQM.

Table 5 shows the best reward obtained by Gurobi solver and HQM. Although the Gurobi solver generates superior solutions for small problem sizes (e.g., nodes \( \leq 100 \)), the computation time is relatively long (average computation time = 3199.74s). While HQM achieve high-quality solutions in large problem instances with significant shorter computation time (average computation time = 609.05s). The average best result between HQM and Gurobi is close with an average gap of 6.21%. Such a gap is influenced by the extreme values caused by HQM obtaining lower rewards in small problem instances (since the solving accuracy of HQM requires larger sample). In real operations, however, we conclude that HQM can achieve better results in shorter computation time than the Gurobi solver based on the results shown in Table 5.

Appendix C shows the best value of different cost units of Gurobi solver and HQM. We notice that the Gurobi solver achieves less MPLs deployment in small problem instances while HQM obtains greater distance savings in large problem instances. Specifically, the difference between the Gurobi solver and HQM in MPLs deployment is within \( \pm 3 \) units. In regard to travel distance, HQM achieves a 9.93% improvement compared with that of the Gurobi solver in general. There is a minor difference between the Gurobi solver and HQM on average service delays (9.07 min of HQM vs. 10.49 min of Gurobi).

| Number of parking spaces \( I_i \) | Customer locations within each parking space \( N_n \) | Gurobi | HQM |
|-----------------------------------|---------------------------------|--------|-----|
|                                   |                                 | Computation Time (Sec) | Best Result \((\times 10^{-3})\) | Computation Time (Sec) | Average Final Reward \((\times 10^{-3})\) | Gap (%) |
| 5                                 | 5                               | 896.5 | 7.247 | 264.3 | 6.481 | -10.57 |
|                                   | 10                              | 1232.4| 4.204 | 384.6 | 3.899 | -7.26  |
|                                   | 15                              | 1381.9| 3.930 | 433.5 | 3.655 | -6.99  |
|                                   | 20                              | 3497.2| 1.096 | 506.4 | 1.158 | 5.66   |
| 6                                 | 5                               | 898.6 | 5.774 | 309   | 5.149 | -10.82 |
|                                   | 10                              | 1371.5| 3.078 | 438.9 | 2.697 | -12.38 |
|                                   | 15                              | 2491.8| 2.162 | 499.5 | 2.024 | -6.38  |
|                                   | 20                              | 3671.6| 2.063 | 580.5 | 2.227 | 7.95   |
| 7                                 | 5                               | 677.1 | 4.672 | 367.2 | 4.281 | -8.37  |
|                                   | 10                              | 1625.6| 0.695 | 551.7 | 0.687 | -1.15  |
|                                   | 15                              | 2884.6| 0.383 | 701.4 | 0.427 | 11.49  |
|                                   | 20                              | 4965.9| 0.408 | 757.8 | 0.456 | 11.76  |
| 8                                 | 5                               | 948.2 | 4.062 | 384.3 | 3.828 | -5.76  |
|                                   | 10                              | 2995.3| 0.675 | 562.5 | 0.717 | 6.22   |
|                                   | 15                              | 4543.7| 0.317 | 734.7 | 0.354 | 11.67  |
|                                   | 20                              | 5779.9| 0.252 | 951.9 | 0.264 | 4.76   |
| 9                                 | 5                               | 971.5 | 3.772 | 411.9 | 3.478 | -7.79  |
|                                   | 10                              | 3126.5| 0.357 | 751.2 | 0.325 | -8.96  |
|                                   | 15                              | 6079.1| 0.639 | 856.8 | 0.709 | 10.96  |
|                                   | 20                              | 7200  | 0.251 | 978.9 | 0.274 | 9.16   |
| 10                                | 5                               | 1324.3| 2.728 | 444.6 | 2.539 | -6.93  |
We conducted statistical tests to evaluate the performance of our proposed HQM statistically. Wilcoxon signed-rank tests were used to determine the significance of the difference between two groups of results obtained by HQM and the Gurobi solver. Wilcoxon signed-rank is a non-parametric statistical test to analyse two related samples (Walpole et al., 2017) and was used for a VRP problem (Yu et al., 2022b). Tables 6 shows the result for Wilcoxon signed-rank tests in terms of the best reward and computation time, respectively. We used the confidence level of alpha = 0.05 to test the hypothesis. It is concluded that the two paired samples differ significantly if the significance level (p-value) is less than alpha.

Based on the result from Table 6, the p-value of the best result comparison is larger than alpha, implying that the performance of the HQM is similar compared with the Gurobi solver in terms of solution quality. According to Table 6, we conclude that the computation time of HQM and the Gurobi solver are significantly different since the p-value is less than alpha. Finally, as a result of Tables 5 and 6, we infer that the HQM significantly outperforms the Gurobi solver in terms of solution quality and computation time for large problem instances.

5.3 Performance comparison between HQM and GA
We next compared the performance of HQM with GA regarding the optimal configuration of MPLs (fleet size, travel distance, and service delay) and optimisation performance (reward acquisition, convergence speed, and computation time). We also apply advanced techniques (e.g., roulette wheel selection, elite strategy) to different operations within GA to ensure good computation performance and rigour in comparisons (the implement procedure of GA see Appendix B).

5.3.1 Optimal configuration of MPLs
We first investigate the implementation results in fleet size and travel distance under different problem instances (see Appendix B). The evaluation is based on two dimensions: i) the performance of algorithms, and ii) the performance of route adjustment strategies. Our results indicate that the maximum difference in achieving optimal fleet size between HQM and GA is ±3 units (the maximum MPL fleet size is 34 units). In fact, the GA achieves a better performance than the HQM in terms of minimising driving distances but causes higher service delay significantly.

Regarding the performance of BTD and HCPS route adjustment strategies, BTD achieves fewer MPL deployment in small-median size problem instances on average ($l_i \leq 8, N_t \leq 15$). However, such fleet size difference is compromised in large networks ($l_i \geq 9$) because HCPS is regarded as a more cost-efficient approach to resolve time window conflicts compared with deploying dense MPLs caused by BTD. With the
increase in parking spaces and customer locations, the HCPS strategy allows for greater driving distance savings than BTD since fewer round trips are required between parking spaces and the central depot.

Figure 10 outlines an example of MPL delivery routes \((I=5, N=5)\), where the routes are indicated by solid lines with arrows of different colours. The central depot is denoted by a square, whereas parking spaces are indicated by pentagons. To avoid the misunderstanding of arrows on MPL's routes in the figure, we include the traceable data and detailed results of small problem instances in Appendix E & F, since MPLs are involved in the round trip between depots and different parking spaces. The route shown in Figure 10 indicates that compared with those non-crossing/closed-loop paths (satisfy triangle property) derived by the classical VRPs without considering time window constraints, MPLP may cause path crossings or repeated visits to the same parking space to satisfy the time window constraint of each task. As such route crossings and repeated visits result from the two proposed route adjustment strategies, we will compare their performance and applicable situations in the following sections.

a) HQM-BTD, Reward = 0.013, Fleet = 1 unit  
Distance = 47.8km, Delay = 0min

b) HQM-HCPS, Reward = 0.011, Fleet = 2 unit  
Distance = 37.5km, Delay = 0min

c) GA-BTD, Reward = 0.011, Fleet = 1 unit  
Distance = 56.6km, Delay = 123.9min

d) GA-HCPS, Reward = 0.001, Fleet = 1 unit  
Distance = 19.2km, Delay = 161.2min

**Fig. 10** Implementation result of a small network instance \((I=5, N=5)\)

In terms of average service delay, the results obtained from the two algorithms are 10.10min (HQM-BTD), 8.91min (HQM-HCPS), 121.03min (GA-BTD), and 113.77min (GA-HCPS), respectively. The average service delay obtained by HQM is therefore significantly lower than the GA’s, contributing to higher reward (Figure 11). GA, however, shows randomness and non-robust performance in optimising service delay across different network sizes, as GA produces high latency even at small problem instances (Figure 11c and 11d),
while HQM enables on-time delivery in small network sizes (Figure 11a and 11b). Comparatively to BTD implemented by HQM and GA, HCPS reduces delays by 11.85% (HQM) and 6.00% (BTD), respectively.

![Graph showing service delay of HQM and GA with different route adjustment policies applied](image)

We further evaluate the demand fulfilment in the first delivery round, i.e., the maximum demand that an MPL can fulfil in a single delivery trip. Operators anticipate a higher first-time delivery rate with smaller fleet sizes and shorter travel distances, leading to better utilisation of delivery fleets and reducing repeated deliveries. Table 7 shows that HQM performs better than the GA, as HQM reduces the return to depot frequency. Concretely, the average first-time delivery fulfilment of HQM ranges from 15.5 units to 19.25 units (the maximum capacity is 20 units per MPL), corresponding to the fulfilment rate from 77.50% to 96.30% respectively. In addition, HCPS strategy obtains higher first-time demand fulfilment that BTD when \( I_d \geq 8 \).
### Table 7. Parcel first-time fulfilment under HQM and GA implementation

| Number of parking spaces $I_i$ | Locations within each parking space $N_r$ | HQM Parcel first-time fulfilment-BTD (units) | HQM Parcel first-time fulfilment-HCPS (units) | GA Parcel first-time fulfilment-BTD (units) | GA Parcel first-time fulfilment-HCPS (units) |
|--------------------------------|-------------------------------------------|---------------------------------------------|---------------------------------------------|---------------------------------------------|---------------------------------------------|
| 5                              | 5                                         | 13                                          | 7                                           | 7                                           | 6                                           |
|                                | 10                                        | 11                                          | 15                                          | 9                                           | 15                                          |
|                                | 15                                        | 18                                          | 20                                          | 18                                          | 18                                          |
|                                | 20                                        | 18                                          | 20                                          | 17                                          | 17                                          |
| Mean value                     |                                            | 15                                          | 15.5                                        | 12.75                                       | 14                                          |
| 6                              | 5                                         | 11                                          | 12                                          | 10                                          | 9                                           |
|                                | 10                                        | 17                                          | 19                                          | 13                                          | 14                                          |
|                                | 15                                        | 19                                          | 20                                          | 18                                          | 18                                          |
|                                | 20                                        | 20                                          | 16                                          | 19                                          | 19                                          |
| Mean value                     |                                            | 16.75                                       | 16.75                                       | 15                                          | 15                                          |
| 7                              | 5                                         | 11                                          | 10                                          | 9                                           | 9                                           |
|                                | 10                                        | 18                                          | 15                                          | 14                                          | 16                                          |
|                                | 15                                        | 19                                          | 14                                          | 19                                          | 20                                          |
|                                | 20                                        | 18                                          | 20                                          | 18                                          | 18                                          |
| Mean value                     |                                            | 16.5                                        | 14.75                                       | 15                                          | 15.75                                       |
| 8                              | 5                                         | 16                                          | 17                                          | 10                                          | 12                                          |
|                                | 10                                        | 14                                          | 17                                          | 18                                          | 18                                          |
|                                | 15                                        | 20                                          | 20                                          | 19                                          | 18                                          |
|                                | 20                                        | 19                                          | 20                                          | 17                                          | 20                                          |
| Mean value                     |                                            | 14.75                                       | 18.5                                        | 16                                          | 17                                          |
| 9                              | 5                                         | 10                                          | 11                                          | 9                                           | 9                                           |
|                                | 10                                        | 18                                          | 20                                          | 16                                          | 18                                          |
|                                | 15                                        | 20                                          | 20                                          | 20                                          | 15                                          |
|                                | 20                                        | 20                                          | 20                                          | 20                                          | 20                                          |
| Mean value                     |                                            | 17                                          | 17.75                                       | 16.25                                       | 15.5                                        |
| 10                             | 5                                         | 20                                          | 14                                          | 14                                          | 15                                          |
|                                | 10                                        | 20                                          | 20                                          | 15                                          | 17                                          |
|                                | 15                                        | 19                                          | 20                                          | 19                                          | 16                                          |
|                                | 20                                        | 18                                          | 20                                          | 20                                          | 20                                          |
| Mean value                     |                                            | 19.25                                       | 18.5                                        | 17                                          | 17                                          |

#### 5.3.2 Optimisation performance of HQM

The optimisation performance of HQM is evaluated from three dimensions: (i) the final reward obtained, (ii) reward acquisition and convergence, and (iii) computation time. The final reward obtained represents the absolute ability to derive a final optimal solution, while the reward acquisition and convergence speed measure the improvement rate throughout the learning process.

**Final reward obtained**

Figure 12 shows the final reward obtained from HQM and GA under different problem instances and route adjustment strategies. We conclude that the proposed HQM combined with HCPS strategy yields the greatest reward with the value of $1.97 \times 10^{-3}$, followed by $1.90 \times 10^{-3}$ (HQM-BTD), $1 \times 10^{-3}$ (GA-HCPS), and $0.97 \times 10^{-3}$ (GA-BTD).
The surface plot in Figure 13 illustrates the reward improvement of HQM and GA, where the bottom layer denotes the initial rewards, and the top layer denotes the final rewards optimised by the two algorithms. The calculation of the improvement based on the following metric: \( \left( R_{t=\text{max}} - R_{t=1} \right) / R_{t=1} \). In Figure 13b, the larger gap between the two layers within the small and medium problem instances \( (l_1 \leq 8, N_r \leq 18) \) indicates that the HQM-HCPS achieves a greater reward improvement than the HQM-BTD (Figure 13a). Contrary to the HQM, the GA is merely good at solving small problem instances \( (l_1 \leq 7, N_r \leq 10) \) and performed poorly when medium and large problem instances are involved (Figure 13c and 13d). Although HQM shows less reward improvement in large problem instances compared with small-medium networks due to the large gap of axis calibration in the figure, the HQM still perform better than GA, which can be proven by the concrete improvement rate shown in Figure 14.

Specifically, HQM-HCPS achieves an average increase of 453.58% in final rewards over initial rewards (Figure 14b), while HQM-BTD achieves an average increase of 433.23% (Figure 14a). By contrast, GA-HCPS achieves an average improvement of 97.31% in final rewards compared to its initial rewards (Figure 14d), while GA-BTD achieves an average improvement of 92.50% (Figure 14c). The optimisation ability of HQM is on average 348.49% greater than that of GA, suggesting that HQM facilitates an efficient search for better solutions and overcomes local optima.
Fig. 13 Optimisation ability of HQM and GA

(a) Solution improvement of HQM-BTD
(b) Solution improvement of HQM-HCPS
(c) Solution improvement of GA-BTD
(d) Solution improvement of GA-HCPS

Fig. 14 Improvement rate of HQM and GA

Reward acquisition and convergence

Figure 15 illustrates the reward acquisition and convergence of HQM and GA. Algorithms are implemented under a large instance setting ($l = 10, N = 20$). The result concludes that GA tends to fall into
a local optimum during the earlier iterations, resulting in a lower reward (Figure 15a). The reason for this can be explained by Figure 15b, which illustrates the convergence performance of the algorithms: we can see that the objective value of GA-BTD and GA-HCPS sharply converges at generation 60 (±20) and generation 200 (±10) respectively, whereas HQM improves solution smoothly and converges at generation 180 (±20), meaning that HQM better exploits the information feedback obtained from the previous action and generate more efficient solution search policy than GA.

(a) Convergence comparison between HQM and GA  
(b) Real-time reward of HQM and GA

Fig. 15 Reward acquisition and convergence comparison

Computing time under different problem instances
In Table 8, we compare the computation time between HQM and GA. The results indicate that the average computation time of HQM is 9.64 times that of GA given the same problem instance. This is due to the fact that the heuristic algorithm does not involve data sampling and knowledge mining, thus reducing the computation time required, but sacrifices satisfactory solutions (the final reward of HQM is greater than GA’s for all problem instances).

Since there are no MPLP benchmark problem instances introduced and solved by RL-based approaches, we cannot compare HQM’s computation time with other RL-based approaches within the same dimension. However, in contrast to other attention-mechanism-based RL approaches for solving VRP with similar network sizes, our proposed HQM provides a shorter computation cycle. We here cite the related research conducted by Kool et al. (2018) for reference which has been proven that the proposed REINFORCE approach achieves better optimisation performance than other RL techniques: the author proposed a model based on attention layers with benefits over the Pointer Network. The REINFORCE is applied to train the model. The computation time under different problem instances with varied sizes is as follows: i) CVRP: 6 min (Node=20), 28 min (Node=50), 2 hr (Node =100); ii) SDVRP: 9 min (Node =20), 42 min (Node =50), 3 hr (Node =100).

Table 8. Comparison of average computation time under different problem instances

| Number of parking spaces $I_i$ | Customer locations within each parking space $N_r$ | HQM | GA |
|-------------------------------|---------------------------------------------|-----|----|
|                               | Computation Time (Sec) | Average Final Reward ($\times 10^{-3}$) | Computation Time (Sec) | Average Final Reward ($\times 10^{-3}$) | Gap (%) |
| 5                             | 5                            | 264.3 | 6.481 | 23.7 | 5.721 | 11.73 |
|                               | 10                           | 384.6 | 3.899 | 48.3 | 1.325 | 66.02 |
|                               | 15                           | 433.5 | 3.655 | 26.7 | 0.710 | 80.57 |
|     | 5  | 10 | 15 | 20 |
|-----|----|----|----|----|
| 6   | 309| 438.9| 499.5| 580.5|
|     | 5.149| 2.697| 2.024| 2.227|
|     | 42  | 46.2 | 57.9 | 48  |
|     | 4.520| 0.568| 0.681| 0.698|
|     | 12.22| 78.94| 66.35| 68.66|
| 7   | 367.2| 551.7| 701.4| 757.8|
|     | 4.281| 0.687| 0.427| 0.456|
|     | 23.4 | 46.8 | 44.7 | 85.8 |
|     | 3.804| 0.252| 0.195| 0.247|
|     | 11.14| 63.22| 54.33| 45.83|
| 8   | 384.3| 562.5| 734.7| 951.9|
|     | 3.828| 0.717| 0.354| 0.264|
|     | 32.4 | 48    | 61.5  | 109.8|
|     | 1.127| 0.289| 0.249| 0.144|
|     | 70.56| 59.69| 29.66| 45.45|
| 9   | 411.9| 751.2| 856.8| 978.9|
|     | 3.478| 0.325| 0.709| 0.274|
|     | 29.7 | 60.9  | 126.3 | 108  |
|     | 1.010| 0.147| 0.258| 0.174|
|     | 70.96| 54.77| 63.61| 36.50|
| 10  | 444.6| 701.4| 914.4| 1129.8|
|     | 2.539| 0.360| 0.275| 0.139|
|     | 47.4 | 48.6  | 157.8 | 117  |
|     | 0.746| 0.161| 0.160| 0.097|
|     | 70.62| 55.28| 41.82| 30.22|

Notes: $\text{Gap} = (\text{Average Final Reward of HQM} - \text{Average Final Reward of GA}) / \text{Average Final Reward of HQM}$

5.4 Effects of critical factors against different cost units

We eventually investigate the effects of the critical factors on three cost units within the objective function (fleet size, travel distance, and service delay) to provide managerial and operational insight to logistics and administration professionals. Factors are classified into four dimensions, depending on their attributes: i) time windows from customers and parking spaces, ii) locker properties, iii) network density, and iv) locker service radius and customer maximum walking range. Sections 5.4.1 to 5.4.3 focus on a parametric discussion of how these factors influence different cost units, while Section 5.4.4 provides managerial implications of the effect of these factors on operations.

5.4.1 Effects of critical factors against fleet size

Figure 16a demonstrates the effect of customer time windows and the available parking time of parking space on fleet size. The length of customer residence time $T_s$ and available parking time $T_p$ are evaluated within the range from 40 to 70 minutes. The result indicates that both customer time windows and the available parking time are negatively correlated to the fleet size. Extending customer residential time provides the potential to reduce time window conflicts, thus resulting in fewer MPLs being required for conflict resolution.

In terms of the effect of MPL properties, we notice that MPL capacity $Q$ has a more significant influence on fleet size than speed $V_i$ once $V_i \geq 40 \text{ km/hr}$ (Figure 16b). However, the fleet size will increase sharply if $V_i \leq 35 \text{ km/hr}$ and $Q \leq 15 \text{ units}$ because more dense contours are within this area. This can be explained by the fact that lower locker speed may not satisfy the time window requirements of customers and parking spaces, which require assigning extra MPLs to resolve such cases. In addition, we notice that a large MPL fleet is required to meet time window constraints for lower MPL capacities. Our result suggests that the MPL capacity $Q$ is expected to be greater than 15 units/MPL in our network settings to reduce MPL fleet size.
Figure 16c shows that both the number of parking spaces and customer locations have conspicuous impacts on service delays, proven by the positive correlation of the contour plot. Compared with the other three factors, the difference in fleet size with varied network density is the most significant (the range of fleet size from 9 to 34 units).

Finally, we conclude that customers walking range $\rho_c$ have negligible impact on MPLs fleet size once the walking range $\rho_c \geq 0.4km$ (Figure 16d), and that MPL service radius has a greater significance on fleet size. In summary, time window constraints, MPL capacity, and network density are the main factors affecting MPLs fleet size.

(a) Effect of time windows  
(b) Effect of locker properties  
(c) Effect of network size  
(d) Effect of service radius and walking range  

**Fig. 16** Effects of critical factors against fleet size

### 5.4.2 Effects of critical factors against travel distance

From Figure 17a, we notice that the travel distance is more sensitive with available parking time $T_p$ compared with customer stay duration $T_s$. The travel distance sharply decreases when available parking time $T_p \geq 55min$, along with fewer fleet required to meet customer demands, thus leading to shorter routes. On the other hand, the travel distance has increased gradually when customer stay time $T_s \leq 60 min$, because some MPLs are scheduled globally to meet the tight delivery time requirements even though the MPLs are far away from the customer to be fulfilled. This issue is even more significant when time windows constraints are tighter (e.g., $57min \leq T_s \leq 63min$, and $T_p \leq 45min$).
Regarding the effect of MPL properties, the result indicates that MPL capacity $Q$ is negatively correlated with travel distance, whereas MPL speed $V_t$ does not. The total route length only has a minor change when the locker capacity $Q > 20 \text{ units/MPL}$ (Figure 17b).

With the increased number of parking spaces and customer locations, the travel distance of MPLs has increased dramatically, especially the number of locations per parking space $N_r \geq 15 \text{ units/parking space}$ and $I_t \geq 8 \text{ units}$ (Figure 17c). Figure 17d demonstrates that the travel distance increased with their service range since more customers are covered by the MPL fleet that results in greater route length.

![Effect of time windows](image1.png)

(a) Effect of time windows  
(b) Effect of locker properties

![Effect of network size](image2.png)

(c) Effect of network size  
(d) Effect of service radius and walking range

**Fig. 17** Effects of critical factors against travel distance

5.4.3 Effects of critical factors against service delay

Figure 18a suggests that the service delay decreases with longer customer stay time $T_s$, meaning that if customers provide more generous time windows, the MPL can be scheduled more flexibly thus reducing service delays. Specifically, the service delay will be significantly reduced once $T_s \geq 50 \text{min}$ or $T_p \geq 45 \text{min}$. However, the service delay will exceed more than 60 min if both customers and parking spaces time windows are tight (e.g., $T_s < 50 \text{min}$ and $T_p < 45 \text{min}$). Furthermore, available parking time $T_p$ has a more significant impact on service delay compared with customer stay time $T_s$.

Figure 18b illustrates that the service delay will be improved as locker capacity $Q$ decreases. The reduced MPL capacity means fewer customers can be fulfilled in a single delivery round, which saves more
travel distance and eliminates the potential delays. However, more MPLs are required to satisfy the remaining demands.

Conversely, the larger capacity $Q$ may promote the algorithm to assign more customers to the same MPL, thus reducing MPL fleet size with the expense of higher service delays. The service delay will be improved as MPL speed increases. However, if the MPL speed is too fast, the travel time between two adjacent parking spaces may be too short to satisfy the available parking time restriction in large problem instances. In this case, the MPL will adopt route adjustment strategies to resolve time window conflicts, which may result in higher delays in the whole system. We conclude that the capacity $Q \leq 25$ units may achieve satisfactory delay improvement since $Q = 25$ is a distinct boundary of the contour plot. This is of practical significance to logistics operators, as it guides operators to adopt MPLs with appropriate capacity to improve service delay in an economic fashion.

Figure 18c shows that the network sizes have positive conspicuous effects on service delay, especially in the case that $I_i \geq 9$ and $N_n \geq 16$. This is because the increase in parking spaces and customer locations leads to more MPL deployments and longer travel distances, resulting in higher delays. Finally, Figure 18d presents that the greater MPL service radius and customer walking distance will cause higher delays since more customers will be served within the service region. Given this concern, $r_m \leq 5km$ and $\rho_c \leq 0.6 km$ achieve lower delays.

In summary, the time window constraints and network density are the crucial factors affecting service delay. This requires coordination between logistics professionals, customers, and urban planners to determine appropriate parameter settings for different factors that improve timeliness economically. Additionally, for carriers, defining a reasonable service radius and effective scheduling policies based on customer preferences is regarded as a means to improve customer satisfaction.

(a) Effect of time windows  
(b) Effect of locker properties
5.4.4 Managerial implications

The managerial implications are discussed based on the attributes of the predefined factors to enable logistics practitioners better deploying MPL operations at scale accordingly in the future.

Customer time windows and available parking time

Considering the tendency shown in Figures 16(a), Figures 17(a) and Figures 18(a), we conclude the ratio $\sigma$ of available parking time $T_p$ to customer stay time $T_s$ ($\sigma = T_p / T_s$) is regarded as a crucial indicator for logistics operators since the larger available parking time $T_p$ enable MPLs process more stopovers, thus reach more customers. For $\sigma \geq 1$, the performance of different cost units has significant improvement subject to $T_s, T_p \geq 60$ min.

Nevertheless, from the perspective of customer stay time $T_s$, it is difficult for customers to offer a more generous time window in practice given that their schedules are often fixed during the weekdays, making it difficult for them to adapt time windows for parcel collection. Logistics operators will therefore require a more generous available parking time to fulfill orders more effectively. One possible solution would be to park MPLs in residential areas (there is less traffic on residential streets, so the MPLs can be parked for longer periods without causing significant congestion). As an alternative, the deployment of MPLs near work sites during peak commute hours would allow more customers to be served within a relatively concentrated time interval. Finally, the application of a track & trace system (e.g., smartphone app) allows customers to monitor MPL's position in real-time, thereby reducing parcel overlap time for parcel handover.

Locker capacity and locker velocity

Intuitively, greater MPL velocity contributes to more agility and can reach more customers by visiting more stops along the way. To achieve this, it would be possible to dispatch MPLs prior to the peak commute period, which would enable them to travel at a faster speed (e.g., $V_l \geq 35$km/h in our case) in the curb-side lane, thereby reducing service delays due to the traffic congestion. However, it is important to note that this policy does not apply to most hours of the day, as there is also an increased demand during off-peak hours, particularly on weekends. Given this concern, increased locker speed on main roads or pavements is not a significant approach to improve delivery efficiency since the average vehicle speed in the urban areas of well-developed cities is limited to 25 km/hr (Kusumaningtyas and Lodewijks, 2008). Therefore, we recall the contour plot in Figure 16b, 16c, and 16d, we suggest that setting the MPL to a medium capacity level (e.g., $Q \in [20,30]$)
units/MPL) can improve the performance of different cost units while avoiding disturbing pedestrians and traffic due to the MPL size.

Parking spaces and customer stopovers
Through the implementation of experiments with different network sizes, we conclude that the location of the customer's stopover and demand density are significant factors influencing the setting of parking spaces. Therefore, an essential prerequisite for the successful and effective diffusion of the MPL concept is the willingness of customers to accept MPL services and announce their approximate potential locations. With this proposition, customers may have concerns about data privacy, as well as inconvenience when filling out the requested data, which might be overcome by additional incentives from the logistics provider (e.g., a lower or free service charge). As an alternative, logistics operators can segment the service options for those customers who do not wish to provide location information, such that stationary lockers or door-to-door delivery services can be used for serving those customers while providing MPL service for the remaining customers that are acceptable for MPLs.

From the perspective of demand density, increasing customer participation reduces the advantage of MPLs' mobility since the travel distance and service delay of the MPL fleet will significantly increase (recall Figure 11a, 11b, and 18c). It becomes easier to reach the maximum capacity of lockers as demand density increases in an urban area, even if the locker remains stationary. Hence, combining stationary lockers with MPL is a feasible option. It is much more likely that MPL will prove applicable and valuable during the start-up phase when the customer base is still small. Having successfully diffused the MPL concept and got more customers on board, more stationary lockers (and thus cheaper) can be added to the existing distribution fleet without jeopardizing customer acceptance.

Service radius and customer maximum walking range
MPL concept can be less beneficial if customers are willing to walk far to their predefined lockers. Empirical studies have revealed that customers will normally accept a walking distance of no more than 0.7 km for parcel collection (De Oliveira et al., 2017; Kedia et al., 2017). Recall that the result shown in Figure 16d, 17d, and 18d, we suggest that the service radius of MPL $r_m$ should not exceed 5 km if there is a requirement to guarantee the cost-savings in different cost units over a maximum walking distance of customer $p_c$ not exceeding 0.7 km. On the other hand, as we generate demand within a square area of 5km in length and width (Appendix E), we determine that by reducing the service radius of each MPL $r_m$ by 1 km, the fleet size and total distance travelled will be reduced by at least 1 unit and 50 km, respectively. Therefore, the break-even point between the service radius of the MPL and the demand density is particularly pertinent for logistics operators, especially under the hypothesis that customers expect short walking distances. With this in mind, synergistic MPLs and other delivery methods (e.g., traditional van-based delivery, and stationary lockers) remain the high-potential efficient and cost-effective delivery scenarios in the area where demand density is high. A group of customers with similar travel habits is more suitable to be served by MPLs in urban areas with high traffic congestion.

5.5 Analysis and discussion
In this section, we discuss the proposed method to deal with a special form of LRP, namely MPLP. The main difference between LRP and MPLP is that the movement trajectory of MPL will be determined based on all potential stopovers of the customer set while the classic LRP considers the static characteristics of the demand pattern.

The mathematical model is initially solved using an exact approach solver, Gurobi, to determine the gap between HQM and optimal solutions. The result (Table 5 and 6) shows that HQM has superior ability in deriving high-quality solutions and computational efficiency compared to the Gurobi solver for large problem instances, while it suffers from insufficient training samples for small-scale problems.
The mathematical model is initially solved using an exact solver, Gurobi, to determine the gap between HQM and optimal solutions. The result (Table 5 and 6) shows that HQM has superior ability in deriving high-quality solutions and computational efficiency compared to the Gurobi solver for large problem instances, while it suffers from insufficient training samples for small-scale problems.

However, the exact approach has a weakness in that it takes a considerable amount of time to compute the result. In real-world scenarios, the magnitude of the problem often makes it impossible to solve the problem accurately within a reasonable timeframe (Yu et al., 2022a). Therefore, the results obtained from the exact approach are only used as a baseline for evaluating the solutions obtained by applying the proposed HQM.

Meta-heuristics, as a typical method used for solving the large problem instances of LRP or VRP, we, therefore, focus on the performance comparison between HQM and GA. The comparison covers two dimensions that provide a comprehensive overview of HQM's performance: i) solution quality and convergence/computation efficiency (e.g., Figure 12-15, Table 8); and ii) cost units and operation indications (e.g., Appendix D, Figure 11, Table 7). We conclude from the obtained results that HQM can provide high-quality solutions within a reasonable computational time when compared to the exact solver and GA solutions. It is pertinent to note, however, that there is no absolutely efficient algorithm, i.e., algorithms will suffer from the dilemma of compromising solution quality and compromising efficiency (Wolpert, 2013). Hence, although the proposed HQM performs well in LRP or VRPs and can provide a novel conceptual RL-based solution framework, further experiments are required to validate its generalisation ability to more complex scenarios.

There are potential limitations to the proposed HQM that may arise in even larger problem instances. First, HQM shows differences in optimisation capability for different cost units within the objective function, even though the results of different cost units are influenced by the weights assigned to them. Nevertheless, information exchange and exploitation among the different variables in within a state will also have an indirect effect on the optimisation of the different cost units, since these variables serve as logical abstractions of various cost units. To tackle this limitation, one potential approach is to design an effective communication mechanism (e.g., intermediate variables, transformer layer) between different variables, to make the information interpretable and utilisable among different variables. Second, RL-based methods still require a more efficient medium to interpret the mathematical model into the RL framework, especially when representing model constraints. A more straightforward approach is to decompose the original model such that processing and implementing the model constraints outside the RL framework, while RL is only responsible for the training of the model. The last limitation is the computation time. Although HQM's computation efficiency is significantly superior to the exact approach, it still falls short of heuristic algorithms on large problem instances due to HQM's sampling procedure. The computational time required by HQM can be determined by analysing the procedure defined in the solution framework (Figure 5) in terms of complexity. The complexity of the proposed solution is in (1) the task generation procedure, (2) the reward calculation within the attention-based decoder, (3) the Q-value update procedure within the attention-based decoder, and (4) the global search and local search procedure. The other remaining modules can be executed in a constant time.

The procedure of task generation has a complexity of $O(|N \times K_n|)$ because we must enumerate all potential stopovers $K_n$ for customer set $N$ to identify a set of tasks that satisfy time-window constraints imposed by customers and parking spaces. The reward calculation within the attention-based decoder has the complexity of $O(|P|)$ since the decoder will calculate the corresponding reward for a state $s (s \in \{1, \cdots, P\})$ within the agent $\pi$. In the case of Q-value updates, the computation complexity increases dramatically. Since there are two Q-value matrixes $Q(\theta_{X1})$ and $Q(\theta_{X2})$ (see Equation 35 and 36) in our case which corresponding to variable $X1$ and $X2$ of a state $s$ respectively, therefore we have to update the Q-value matrix simultaneously. Specifically, the complexity of updating $Q(\theta_{X1})$ is $O(M^2)$ where $M$ represents the total number of MPL, while
the complexity of updating $Q(\theta^o_{x_2})$ is $O(o^2)$ where $o$ represents the total number of tasks. When it comes to the global search procedure, since all elements within the state $s$ will be updated according to Q-value matrixes in each timestep, the complexity is $O(\Gamma)$, where $\Gamma$ represents the total number of elements ($\Gamma = |x_1 + x_2|$) within the state $s$. Finally, the complexity of local search procedure is $O(Y)$, where $Y$ is the number of elements that need to be adjusted/shifted between two neighbourhood states. The complexities discussed above are the primary reason that the HQM's computation time increases with the size of the dataset.

As an alternative to the encoder & attention-based decoder and heuristics introduced in our study, we hereby provide the following potential method for overcoming the limitations of the compromise between solution quality and computation efficiency. The first type of method focuses on generating better initial solution or improving the efficiency of transcription of input samples. An example of the former method is introducing Negatively Corelated Search (NCS) during the training process to improve the diversity of better initial solutions (Tang et al., 2016). The latter method typically involves the use of a transformer framework to increase the efficiency of sampling. Further, a batch reinforcement learning approach may also be considered. Batch reinforcement learning differs from traditional online learning in that the learning agent does not interact directly with the environment. To obtain better solutions, it consists of a baseline agent that collects data, which is then fed into an algorithm to train a new policy (Laroche et al., 2019). Finally, an emerging technique is ‘learning to delegate’ approach, which improves the solution quality and computation efficiency by identifying appropriate subproblems and delegating their improvement to a black box sub-solver (S. Li et al., 2021). The learning-based, iterative framework consists of two components: 1) a sub-solver capable of solving a small problem instance (exactly or approximately), and 2) a learned model for identifying suitable subproblems within a larger problem to delegate to the sub-solver. Thus, the 'learning to delegate' approach combines the generalisation advantage of RL with the excellent solution quality provided by exact solvers.

The proposed mathematical model and HQM can be extended to solve other variant VRP-related problems. To adapt the mathematical model to a specific scenario, existing constraints must be modified, or some new constraints should be added (e.g., considering the parcel pick-up, demand heterogeneity). For the HQM, the encoder and attention-based decoder can be adopted to deal with other VRP variants with some modifications in the proposed solution interpretation, given the excellent generalisation ability of HQM’s framework.

6 Conclusion
This study presents a novel approach for MPLP that accounts for changing customer demand patterns and varied scales, as well as stochastic events. In addition, it offers a novel and effective approach to flexibly scheduling autonomous delivered vehicles in the context of urban logistics in the future. A hybrid Q-Learning network algorithm was developed that combines global and local search mechanisms is presented, which resolve stochastic events that cause delays by employing two route adjusting strategies.

We first evaluate the performance of the HQM by using an exact approach solver (Gurobi) as a benchmark to provide a first-hand insight of the solution quality and computation efficiency. We then measure the significance of the difference between the results obtained by HQM and the Gurobi solver. The result shows that our HQM achieves higher solution quality with significantly shorter computation time than the Gurobi solver in large problem instances. To highlight the advantage of HQM compared with meta-heuristic algorithms that commonly used for solving LRP and VRPs, we exploited a customised GA as baseline cases and conducting pairwise comparison under different network densities (15 to 200 nodes) with different route adjustment strategies applied. Our proposed HQM provides solutions in lower computing times than other comparable learning algorithms and obtains a final reward function twice as greater as our benchmark. We further analysed the operational key performance indicators: fleet size, route length, service delays, and the
parcel first-time fulfilment compared with GA. Our result reveals that HQM achieves significant improvement in service delay and parcel first-time delivery successful rate. Finally, the study investigates the influence of critical factors (such as time windows, locker properties, and network densities) on different cost units within the objective function, providing logistics operators with parameterised and managerial insights for enhancing their delivery service.

In addition to the computational benefits of the HQM model, our results indicate that our proposed HQM strategy combined with HCPS reduces service delays and optimises MPL configuration compared to its BTD counterpart. Although HQM shows a good optimisation ability in our MPLP scenario, further experiments and adaptations are required before migrating it to other LRP and VRP-related problems. To tackle the limitations of sampling efficiency and solution quality that may fall short compared with heuristics, we provide several potential solutions such as applying transformer network, delegate agent, and batch reinforcement learning approaches (see Section 5.5).

We finally provide managerial insight based on the parametric analysis for different critical factors against cost units. The most critical factors that affect the operating cost and on-time delivery rate of different cost units are the time window constraints imposed by parking spaces and customers, locker capacity, and network density. In this regard, one of the biggest challenges facing the industry when integrating MPLs into their last-mile distribution services is balancing deployment costs with increased delivery convenience and punctuality. Further, the result highlight that collaborating MPL with other mainstream delivery methods (e.g., traditional van-based delivery and stationary locker) for last-mile deliveries offer the potential for broader application and diffusion of MPL during start-up phase, especially in cities with high demand density and traffic congestion. As an alternative, dispatching MPLs before peak commuting hours based on customers' travel behaviour can enable more demand to be fulfilled at more concentrated sites (e.g., company, school) and shorter time intervals of the day, thus reducing urban congestion. From the perspective of urban planning and sustainability, the deployment of MPLs in less congested areas (e.g., residential areas) allows lockers to stay longer without disturbing traffic flow while the characteristic of their electric-powered mobility reduces gas emissions. Since the proposed last-mile deliveries integrating customers' daily locations with MPLs is still a conceptual model, more pilot studies and operational details are required for validity, and in particular, the collection and protection of customer information requires further research.

Further work will focus on increasing the complexity of the MPLP by introducing dynamic cases, where more flexible route adjustment approaches are developed in response to demand changes and delay issues. It would also be beneficial to investigate the multi-echelon MPLP, in which transhipment centres or feeder vehicles are introduced to shorten the travel distance between lockers and the central depot. Another extension of the problem is to consider cooperative deliveries in conjunction with MPLs. A possible scenario is using couriers or delivery robots in conjunction with MPLs, while MPLs visit the couriers/robots in the field and transport parcels between the couriers/robots and the central depot. As an alternative, one may consider segmenting the market in which the means of parcel deliveries within a specific area depends on predetermined criteria (e.g., demand density, road condition, traffic flow). From the perspective of algorithmic innovation, more cutting-edge RL-based algorithms for solving MPLP and VRP are forward-looking and valuable. A potential research direction is to implement parallel interoperable interaction between the RL methods and the simulator simultaneously, thus improving the accuracy of the reward prediction with the aim of improving solution quality. Other frontier RL concepts, such as Sim2Real and sub-solver delegation agent, have the potential to narrow the gap between the results obtained from RL optimiser and realistic scenarios. Finally, future research could apply real operational case studies to different scenarios (e.g., urban & rural regions) for the purpose of validating the model.
Albareda-Sambola, M., Díaz, J.A., Fernández, E., 2005. A compact model and tight bounds for a combined location-routing problem. Computers & Operations Research. 32, 407–428. https://doi.org/10.1016/S0305-0548(03)00245-4

Allen, J., Bektaş, T., Chertett, T., Friday, A., McLeod, F., Piecyk, M., Piotrowska, M., Austwick, M.Z., 2017. Enabling a Freight Traffic Controller for Collaborative Multidrop Urban Logistics: Practical and Theoretical Challenges. Transportation Research Record. 2609, 77–84. https://doi.org/10.3141/2609-09

Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S., 2017. Neural Combinatorial Optimization with Reinforcement Learning. https://doi.org/10.48550/arXiv.1611.09940

Cappart, Q., Moisan, T., Rousseau, L.-M., Prémont-Schwarz, I., Cire, A., 2020. Combining Reinforcement Learning and Constraint Programming for Combinatorial Optimization. https://doi.org/10.48550/arXiv.2006.01610

Chen, X., Tian, Y., 2019. Learning to Perform Local Rewriting for Combinatorial Optimization. https://doi.org/10.48550/arXiv.1810.00337

De Oliveira, G.S., Bialek, J., Marcus, R.J., McCarthy, R., 2017. Dose-Ranging Effect of Systemic Diphenhydramine on Postoperative Quality of Recovery After Ambulatory Laparoscopic Surgery: A Randomized, Placebo-Controlled, Double-blinded, Clinical Trial. Survey of Anesthesiology. 61, 34–35. https://doi.org/10.1097/01.sa.0000515837.25453.a2

Deudon, M., Cournut, P., Lacoste, A., Adulyasak, Y., Rousseau, L.-M., 2018. Learning Heuristics for the TSP by Policy Gradient, in: van Hoeve, W.-J. (Ed.), Integration of Constraint Programming, Artificial Intelligence, and Operations Research, Lecture Notes in Computer Science. Springer International Publishing, Cham, pp. 170–181. https://doi.org/10.1007/978-3-319-93031-2_12

Deutsch, Y., Golany, B., 2018. A parcel locker network as a solution to the logistics last mile problem. International Journal of Production Research. 56, 251–261. https://doi.org/10.1080/00207543.2017.1395490

Duhamel, C., Lacomme, P., Prins, C., Prodhon, C., 2010. A GRASP×ELS approach for the capacitated location-routing problem. Computers & Operations Research, Metaheuristics for Logistics and Vehicle Routing. 37, 1912–1923. https://doi.org/10.1016/j.cor.2009.07.004

Enthoven, D.I.J.U., Jargalsaikhan, B., Roodbergen, K.J., uit het Broek, M.A.J., Schrottenboer, A.H., 2020. The two-echelon vehicle routing problem with covering options: City logistics with cargo bikes and parcel lockers. Computers & Operations Research. 118, 104919. https://doi.org/10.1016/j.cor.2020.104919

Escobar, J.W., Linfati, R., Toth, P., 2013. A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. Computers & Operations Research. 40, 70–79. https://doi.org/10.1016/j.cor.2012.05.008

Gao, L., Chen, M., Chen, Q., Luo, G., Zhu, N., Liu, Z., 2020. Learn to Design the Heuristics for Vehicle Routing Problem. https://doi.org/10.48550/arXiv.2002.08539

Guo, F., Zhang, J., Huang, Z., Huang, W., 2022. Simultaneous charging station location-routing problem for electric vehicles: Effect of nonlinear partial charging and battery degradation. Energy. 250, 123724. https://doi.org/10.1016/j.energy.2022.123724

Harper, J., 2020. China’s Singles Day: 3m people, 4,000 planes and cargo ships.
Heidari, A., Imani, D.M., Khalilzadeh, M., Sarbazvatan, M., 2022. Green two-echelon closed and open location-routing problem: application of NSGA-II and MOGWO metaheuristic approaches. *Environ Dev Sustain*. https://doi.org/10.1007/s10668-022-02429-w

Innovation policy as a driver for the development of urban logistics, 2012. *International Transport Forum Leipzig, Leipzig.*

Iwan, S., Kijewska, K., Lemke, J., 2016. Analysis of Parcel Lockers' Efficiency as the Last Mile Delivery Solution – The Results of the Research in Poland. *Transportation Research Procedia*, Tenth International Conference on City Logistics 17-19 June 2015, Tenerife, Spain 12, 644–655. https://doi.org/10.1016/j.trpro.2016.02.018

Jiang, L., Chang, H., Zhao, S., Dong, J., Lu, W., 2019. A Travelling Salesman Problem With Carbon Emission Reduction in the Last Mile Delivery. *IEEE Access*. 7, 61620–61627. https://doi.org/10.1109/ACCESS.2019.2915634

Joerss, M., Schröder, J., Neuhaus, F., Klink, C., Mann, F., 2019. *Parcel delivery: The future of last mile*. McKinsey & Company.

Kaewploy, S., Sindhuchao, S., 2017. *Solving the Location Routing Problem of the Central Rubber Market by Tabu Search*. https://doi.org/10.14416/j.ijast.2017.05.007

Kedia, A., Kusumastuti, D., Nicholson, A., 2017. Acceptability of collection and delivery points from consumers' perspective: A qualitative case study of Christchurch city. *Case Studies on Transport Policy*. 5, 587–595. https://doi.org/10.1016/j.cstp.2017.10.009

Kool, W., van Hoof, H., Welling, M., 2019. *Attention, Learn to Solve Routing Problems!*

Kumar, K., Somasundaram Kumanan, 2012. *Decision Making in Location Selection: An Integrated Approach with Clustering and TOPSIS*.

Kusumaningtyas, I., Lodewijks, G., 2008. Accelerating Moving Walkway: A review of the characteristics and potential application. *Transportation Research Part A: Policy and Practice*. 42, 591–609. https://doi.org/10.1016/j.tra.2008.01.003

Laroche, R., Trichelair, P., Combes, R.T. des, 2019. *Safe Policy Improvement with Baseline Bootstrapping*.

Li, J., Ensafian, H., Bell, M.G.H., Geers, D.G., 2021. Deploying autonomous mobile lockers in a two-echelon parcel operation. *Transportation Research Part C: Emerging Technologies*. 128, 103155. https://doi.org/10.1016/j.trc.2021.103155

Li, S., Yan, Z., Wu, C., 2021. *Learning to Delegate for Large-scale Vehicle Routing*.

Ma, Q., Ge, S., He, D., Thaker, D., Drori, I., 2019. *Combinatorial Optimization by Graph Pointer Networks and Hierarchical Reinforcement Learning*. https://doi.org/10.48550/arXiv.1911.04936

Melechovsky, J., Prins, C., Calvo, R.W., 2005. A Metaheuristic to Solve a Location-Routing Problem with Non-Linear Costs. *Heuristics*. 11, 375–391. https://doi.org/10.1007/s10732-005-3601-1

Myllymaki, J., 2016. *Autonomous delivery platform*. No. 9,256,852 B1.

Nadizadeh, A., Hosseini Nasab, H., 2014. Solving the dynamic capacitated location-routing problem with fuzzy demands by hybrid heuristic algorithm. *European Journal of Operational Research*. 238, 458–470. https://doi.org/10.1016/j.ejor.2014.04.012

Nagy, G., Salhi, S., 2007. Location-routing: Issues, models and methods. *European Journal of Operational Research*. 177, 649–672. https://doi.org/10.1016/j.ejor.2006.04.004
Nazari, M., Oroojlooy, A., Snyder, L.V., Takáč, M., 2018. Reinforcement Learning for Solving the Vehicle Routing Problem. https://doi.org/10.48550/arXiv.1802.04240

Orenstein, I., Raviv, T., Sadan, E., 2019. Flexible parcel delivery to automated parcel lockers: models, solution methods and analysis. EURO Journal on Transportation and Logistics. 8, 683–711. https://doi.org/10.1007/s13676-019-00144-7

Perboli, G., Tadei, R., Vigo, D., 2011. The Two-Echelon Capacitated Vehicle Routing Problem: Models and Math-Based Heuristics. Transportation Science. 45, 364–380.

Pham, D.T., Dimov, S.S., Nguyen, C.D., 2005. Selection of K in K-means clustering. Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science. 219, 103–119. https://doi.org/10.1243/095440605X8298

Prins, C., Prouhon, C., Ruiz, A., Soriano, P., Wolfler Calvo, R., 2007. Solving the Capacitated Location-Routing Problem by a Cooperative Lagrangean Relaxation-Granular Tabu Search Heuristic. Transportation Science. 41, 470–483. https://doi.org/10.1287/trsc.1060.0187

Royal Mail Annual Report and Financial Statements 2020-21, 2022. Royal Mail.

Salhi, S., Rand, G.K., 1989. The effect of ignoring routes when locating depots. European Journal of Operational Research. 39, 150–156. https://doi.org/10.1016/0377-2217(89)90188-4

Savelsbergh, M., Van Woensel, T., 2016. 50th Anniversary Invited Article—City Logistics: Challenges and Opportunities. Transportation Science. 50, 579–590. https://doi.org/10.1287/trsc.2016.0675

Schwerdfeger, S., Boysen, N., 2020. Optimizing the changing locations of mobile parcel lockers in last-mile distribution. European Journal of Operational Research. 285, 1077–1094. https://doi.org/10.1016/j.ejor.2020.02.033

Sutton, R.S., Barto, A.G., 2018. Reinforcement learning: an introduction, Second edition. ed, Adaptive computation and machine learning series. The MIT Press, Cambridge, Massachusetts.

Tang, K., Yang, P., Yao, X., 2016. Negatively Correlated Search. IEEE J. Select. Areas Commun. 34, 542–550. https://doi.org/10.1109/JSAC.2016.2525458

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, I., Polosukhin, I., 2017. Attention Is All You Need.

Vinyals, O., Fortunato, M., Jaitly, N., 2017. Pointer Networks. https://doi.org/10.48550/arXiv.1506.01566

Walpole, R.E., Myers, R.H., Myers, S.L., Ye, K., 2017. Probability & statistics for engineers & scientists: MyStatLab update, Ninth edition. ed. Pearson, Boston.

Wang, Q., Tang, C., 2021. Deep reinforcement learning for transportation network combinatorial optimization: A survey. Knowledge-Based Systems. 233, 107526. https://doi.org/10.1016/j.knosys.2021.107526

Wang, Y., Bi, M., Chen, Y., 2020. A Scheduling Strategy of Mobile Parcel Lockers for the Last Mile Delivery Problem. Promet - Traffic&Transportation. 32, 875–885. https://doi.org/10.7307/ptt.v32i6.3531

Wolpert, D.H., 2013. Ubiquity symposium: Evolutionary computation and the processes of life: what the no free lunch theorems really mean: how to improve search algorithms. Ubiquity. 2013, 1–15. https://doi.org/10.1145/2555235.2555237
Yang, S., Ning, L., Tong, L. (Carol), Shang, P., 2022. Integrated electric logistics vehicle recharging station location–routing problem with mixed backhauls and recharging strategies. *Transportation Research Part C: Emerging Technologies*. 140, 103695. https://doi.org/10.1016/j.trc.2022.103695

Yu, J.J.Q., Yu, W., Gu, J., 2019. Online Vehicle Routing With Neural Combinatorial Optimization and Deep Reinforcement Learning. *IEEE Transaction on Intelligent Transport. Systems*. 20, 3806–3817. https://doi.org/10.1109/TITS.2019.2909109

Yu, V.F., Aloina, G., Susanto, H., Effendi, M.K., Lin, S.-W., 2022a. Regional Location Routing Problem for Waste Collection Using Hybrid Genetic Algorithm-Simulated Annealing. *Mathematics*. 10, 2131. https://doi.org/10.3390/math10122131

Yu, V.F., Jodiawan, P., Hou, M.-L., Gunawan, A., 2021. Design of a two-echelon freight distribution system in last-mile logistics considering covering locations and occasional drivers. *Transportation Research Part E: Logistics and Transportation Review*. 154, 102461. https://doi.org/10.1016/j.tre.2021.102461

Yu, V.F., Lin, S.-W., Lee, W., Ting, C.-J., 2010. A simulated annealing heuristic for the capacitated location routing problem. *Computers & Industrial Engineering, Scheduling in Healthcare and Industrial Systems*. 58, 288–299. https://doi.org/10.1016/j.cie.2009.10.007

Yu, V.F., Susanto, H., Jodiawan, P., Ho, T.-W., Lin, S.-W., Huang, Y.-T., 2022b. A Simulated Annealing Algorithm for the Vehicle Routing Problem With Parcel Lockers. *IEEE Access*. 10, 20764–20782. https://doi.org/10.1109/ACCESS.2022.3152062

Zare Mehrjerdi, Y., Nadizadeh, A., 2013. Using greedy clustering method to solve capacitated location-routing problem with fuzzy demands. *European Journal of Operational Research*. 229, 75–84. https://doi.org/10.1016/j.ejor.2013.02.013

Zhang, H., Zhang, Q., Ma, L., Zhang, Z., Liu, Y., 2019. A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows. *Information Sciences*. 490, 166–190. https://doi.org/10.1016/j.ins.2019.03.070

Zhao, J., Mao, M., Zhao, X., Zou, J., 2021. A Hybrid of Deep Reinforcement Learning and Local Search for the Vehicle Routing Problems. *IEEE Transactions on Intelligent Transportation Systems*. 22, 7208–7218. https://doi.org/10.1109/TITS.2020.3003163

Zhao, Q., Wang, W., De Souza, R., 2018. A heterogeneous fleet two-echelon capacitated location-routing model for joint delivery arising in city logistics. *International Journal of Production Research*. 56, 5062–5080. https://doi.org/10.1080/00207543.2017.1401235
Appendix A. Decoding algorithm for state
The input of the decoding algorithm is a state $s$ obtained from the encoder. The output is an interpretable task set of each MPL $m$. The mechanism of the decoding procedure is to traverse each element of $x_1$ within a state $s$, and assign it to the corresponding MPL following the sequence defined by $x_2$.

**Algorithm:** Decoding algorithm for route generation

Input: State $s = (x_1, x_2)$, Number of MPLs $M$

output: Task set $\tilde{O}_m$ of MPLs $m$

1. $total\_dispatch \leftarrow 0$ // Number of MPLs deployed
2. $\tilde{O}_m \leftarrow \emptyset$
3. $x_1 \leftarrow s[0]$ // Variable of task assignment
4. $x_2 \leftarrow s[1]$ // Variable of task execution sequence
5. for $m$ in $M$ do
6.     $\tilde{O}_m[m] \leftarrow \emptyset$
7.     $index\_x_1 \leftarrow argwhere (x_1 == m)$ // Extract index where the tasks are allocated to MPL $m$
8.     if $index\_x_1$ is not $\emptyset$ then
9.         $index\_x_2 \leftarrow argwhere (x_2 == index\_x_1)$
10.        $index\_x_2 \leftarrow argsort (index\_x_2)$ // Sorting index
11.        $task \leftarrow index\_x_3[index\_x_2]$ // Task of MPL $m$ with execution sequence
12.        $\tilde{O}_m[m] \leftarrow task$ // Task list of MPL $m$
13.        $total\_dispatch += 1$
14.     end if
15. end for
16. return $\tilde{O}_m$
Appendix B. Code listing of Genetic Algorithm (GA)

We design a customised GA for the comparison purpose with HQM in our study. The overall implementation process is as follows. We first create the population, in which the population size equals the number of states within the agent $\pi$ in HQM. The population is then ranked based on the fitness function (the value of Equation 3). Using an elite strategy and roulette wheel selection, individuals with higher fitness values will be selected to produce the next generation. The next step is to exploit the elite individuals to create a mating pool and breed the population. As a final step, we apply mutation operation to the new generation to obtain better solutions. Following the notation and parameter setting mentioned in Table 4, the code listing of GA is as follows:

```
Algorithm: Genetic algorithm

Input: Population size $Pop$; Possibility of crossover $P_{cross}$; Proportion of the elite size $P_{elite}$; Mutation rate $P_{muta}$; Number of MPLs $M$; All task list $O$

output: The optimal individual $\text{individual}^*$

1. $\text{individual} \leftarrow \emptyset$; $\text{population} \leftarrow \emptyset$; $\text{fitness} \leftarrow \emptyset$; $\text{pop}_\text{index} \leftarrow \emptyset$; $\text{pop}_\text{ranked} \leftarrow \emptyset$; $\text{cum}_\text{perc} \leftarrow \emptyset$; $\text{mating}_\text{pool} \leftarrow \emptyset$; $\text{children} \leftarrow \emptyset$

2. for $i$ in range($Pop$) do // Initialise population
3.     $x_1 \leftarrow N \sim U(0, ||M||)$, $x_2 \leftarrow$ Permutating task list $O$
4.     $\text{individual} = (x_1, x_2)$
5.     Add $\text{individual}$ to $\text{population}$
6. end for

7. for $\text{individual}$ in $\text{population}$ do // Rank population
8.     $\text{fitness} \leftarrow$ Calculate the fitness value of $\text{individual}$ according to Equation (3)
9.     $\text{pop}_\text{index} \leftarrow$ Obtain the index of the sorted fitness score in $\text{fitness}$
10. $\text{pop}_\text{ranked} = [\text{population} [i] \text{ for } i \text{ in } \text{pop}_\text{index}]$
11. end for

12. $\text{cum}_\text{perc} \leftarrow$ Calculate the cumulative percentage of the fitness value in $\text{pop}_\text{ranked}$

13. for $i$ in range (len($\text{pop}_\text{ranked}$) - $P_{elite}$, len($\text{pop}_\text{ranked}$)) do // Select elite individuals within population
14.     Add $P_{elite}$ elite individuals to $\text{mating}_\text{pool}$
15. end for

16. for $i$ in range (0, len($\text{pop}_\text{ranked}$) - $P_{elite}$) do // Fill the rest of population via roulette wheel selection
17.     $P_{cross} \leftarrow 100 \times U(0, 1)$ // Generate random probability
18.     for $\text{fitness}_\text{score}$ in $\text{cum}_\text{perc}$ do
19.         if $P_{cross} \leq \text{fitness}_\text{score}$ then
20.             Add the corresponding individual to $\text{mating}_\text{pool}$ // Create the mating pool
21.         end if
22.     end for
23. end for

24. for $i$ in range (0, $P_{elite}$) do // Produce the next generation
25.     Add the top $P_{elite}$ elite individual to $\text{children}$
26. end for

27. for $i$ in range (0, len($\text{mating}_\text{pool}$) - $P_{elite}$) do // Regenerate the rest of individual of $\text{mating}_\text{pool}$
28.     Exchange some chromosome of the two adjacent individuals // Breed the population
29.     Obtain the updated population $\text{population}'$
30. end for
```
for individual in population do // Mutation operation
    for chromosome in individual do
        prob ← 100 * U(0, 1) // Generate random probability for mutation
        if $p_{\text{muta}} \leq \text{prob}$ then
            Exchange the current value of the chromosome with the adjacent chromosome
            Obtain the updated population population''
        end if
    end for
end for
Recalculate and rank the fitness value of the new population population''
Output the best individual individual' with the highest fitness score if convergence condition satisfies
Return individual'
Appendix C. Performance of the Gurobi solver and HQM under different problem instances
This appendix focuses on the implementation result comparison of Gurobi and HQM in terms of MPLs deployment size, total travel distance, and service delays. It should be noticed that the value of different cost units of HQM is the mean value between BTD and HCPS for the corresponding cost unit.

Table C.1 MPLs deployed, travel distance, and service delay of Gurobi and HQM

| Number of parking spaces $I_1$ | Customer locations within each parking space $N_n$ | Gurobi | HQM | Gap$^a$ | Gap$^b$ | Gap$^c$ |
|--------------------------------|---------------------------------|--------|-----|-------|-------|-------|
|                                | MPLs deployed (unit) | Travel distance (km) | Service delay (min) | MPLs deployed (unit) | Travel distance (km) | Service delay (min) |
| 5                              | 8                               | 97.988 | 0   | 9    | 109.519 | 0   | 12.50 | 11.77 | 0     |
| 10                             | 12                              | 177.869 | 0  | 15  | 183.984 | 0  | 20.83 | 3.44  | 0     |
| 15                             | 16                              | 174.453 | 0  | 16  | 194.428 | 0.022 | -3.13 | 11.45 | 0     |
| 20                             | 16                              | 240.409 | 0  | 16  | 221.891 | 7.180 | 0.00  | -7.70 | -     |
| 6                              | 9                               | 128.190 | 0 | 11  | 139.608 | 0  | 22.22 | 8.91  | 0     |
| 10                             | 15                              | 171.281 | 0.721 | 16  | 224.479 | 0.965 | 3.33  | 31.06 | 33.84 |
| 15                             | 18                              | 254.115 | 5.684 | 18  | 264.484 | 2.583 | -2.78 | 4.08  | -54.56 |
| 20                             | 22                              | 297.026 | 2.341 | 22 | 278.925 | 0.606 | -2.27 | -6.09 | -74.13 |
| 7                              | 12                              | 154.041 | 0 | 12  | 173.707 | 0  | 0.00  | 12.77 | 0     |
| 10                             | 19                              | 244.694 | 5.076 | 20  | 278.588 | 11.076 | 2.63  | 13.84 | 118.20 |
| 15                             | 25                              | 451.235 | 20.958 | 24  | 341.547 | 15.759 | -4.00 | -24.31 | -24.81 |
| 20                             | 28                              | 408.578 | 25.618 | 26  | 423.333 | 13.996 | -8.93 | 3.61  | -45.37 |
| 8                              | 15                              | 171.184 | 0 | 15  | 186.306 | 0  | 0.00  | 8.83  | 0     |
| 10                             | 20                              | 401.430 | 15.438 | 19  | 309.036 | 10.524 | -5.00 | -23.02 | -31.83 |
| 15                             | 25                              | 566.556 | 14.359 | 23  | 371.916 | 21.252 | -10.00 | -34.35 | 48.00 |
| 20                             | 33                              | 707.538 | 20.113 | 33  | 456.919 | 19.589 | -1.52 | -35.42 | -2.61 |
| 9                              | 14                              | 195.111 | 0 | 15  | 204.009 | 0.12  | 7.14  | 4.56  | -     |
| 10                             | 22                              | 348.313 | 14.293 | 22  | 383.603 | 23.471 | 0.00  | 10.13 | 64.21 |
| 15                             | 33                              | 515.278 | 20.439 | 30  | 421.291 | 5.613  | -9.09 | -18.24 | -72.54 |
| 20                             | 33                              | 575.891 | 19.872 | 31  | 470.912 | 19.691 | -6.06 | -18.23 | -9.91 |
| 10                             | 18                              | 222.569 | 0 | 19  | 252.073 | 0.606 | 2.78  | 13.26 | -     |
| 20                             | 26                              | 339.106 | 31.611 | 24  | 402.434 | 19.333 | -9.62 | 18.67 | -38.84 |
| 15                             | 34                              | 519.177 | 19.243 | 31  | 476.686 | 19.437 | -8.82 | -8.18 | 1.01  |
| 20                             | 34                              | 776.938 | 35.987 | 33  | 561.443 | 25.866 | -2.94 | -27.74 | -28.12 |

**Notes:** i) Gap$^a$ = (MPLs deployed of HQM - MPLs deployed of Gurobi)/ MPLs deployed of Gurobi; ii) Gap$^b$ = (Travel distance of HQM - Travel distance of Gurobi)/ Travel distance of Gurobi; iii) Gap$^c$ = (Service delay of HQM – Service delay of Gurobi)/ Service delay of Gurobi;
Appendix D. Performance of HQM and GA under different problem instances
This appendix focuses on the implementation result comparison of HQM and GA in terms of MPLs deployment size and the total travel distance. The comparison is based on two perspectives: 1) HQM vs. GA, and 2) BTD vs. HCPS.

| Table D.1 MPLs deployed and travel distance of HQM and GA |
|----------------------------------------------------------|
| Parking spaces $I_t$ | Customers per parking space $N_n$ | Route adjustment policy | HQM | GA | Gap (HQM vs. GA) |
|---------------------|---------------------------------|------------------------|-----|----|-----------------|
|                     |                                 |                        | MPLs deployed (unit) | Travel distance (km) | MPLs deployed (unit) | Travel distance (km) |   |
|                     |                                 |                        | MPLs deployed (unit) | Travel distance (km) | MPLs deployed (unit) | Travel distance (km) |   |
| 5                   | 5                               | BTD                    | 9   | 115.196 | 12 | 107.922 | -3 | 7.274 |
|                     |                                 | HCPS                   | 9   | 103.842 | 15 | 125.609 | -6 | -21.767 |
|                     |                                 | Gap (BTD vs. HCPS)     | 0   | 11.354  | -3 | -17.687 | - | - |
| 10                  | 10                              | BTD                    | 13  | 188.929 | 16 | 168.477 | -3 | 20.452 |
|                     |                                 | HCPS                   | 16  | 179.039 | 16 | 175.058 | 0  | 3.981 |
|                     |                                 | Gap (BTD vs. HCPS)     | -3  | 9.89    | 0  | -6.581  | - | - |
| 15                  | 15                              | BTD                    | 15  | 203.999 | 16 | 178.342 | -1 | 25.657 |
|                     |                                 | HCPS                   | 16  | 184.857 | 14 | 162.803 | 2  | 22.054 |
|                     |                                 | Gap (BTD vs. HCPS)     | -1  | 19.142  | 2  | 15.539  | - | - |
| 20                  | 20                              | BTD                    | 17  | 230.168 | 16 | 199.032 | 1  | 31.136 |
|                     |                                 | HCPS                   | 15  | 213.614 | 15 | 216.142 | 0  | -2.528 |
|                     |                                 | Gap (BTD vs. HCPS)     | 2   | 16.554  | 1  | -17.11  | - | - |
| 5                   | 5                               | BTD                    | 11  | 148.131 | 11 | 135.441 | 0  | 12.69 |
|                     |                                 | HCPS                   | 11  | 131.085 | 11 | 127.910 | 0  | 3.175 |
|                     |                                 | Gap (BTD vs. HCPS)     | 0   | 17.046  | 0  | 7.531   | - | - |
| 10                  | 10                              | BTD                    | 15  | 214.336 | 17 | 205.476 | -2 | 8.86 |
|                     |                                 | HCPS                   | 16  | 234.621 | 14 | 196.877 | 2  | 37.744 |
|                     |                                 | Gap (BTD vs. HCPS)     | -1  | -20.285 | 3  | 8.599   | - | - |
| 15                  | 15                              | BTD                    | 18  | 268.043 | 17 | 227.272 | 1  | 40.771 |
|                     |                                 | HCPS                   | 17  | 260.924 | 17 | 214.830 | 0  | 46.094 |
|                     |                                 | Gap (BTD vs. HCPS)     | 1   | 7.119   | 0  | 12.442  | - | - |
| 20                  | 20                              | BTD                    | 23  | 274.511 | 21 | 277.209 | 2  | -2.698 |
|                     |                                 | HCPS                   | 20  | 283.338 | 21 | 261.223 | -1 | 22.115 |
|                     |                                 | Gap (BTD vs. HCPS)     | 3   | -8.827  | 0  | 15.986  | - | - |
| 7                   | 5                               | BTD                    | 12  | 179.269 | 12 | 157.566 | 0  | 21.703 |
|                     |                                 | HCPS                   | 12  | 168.145 | 15 | 170.794 | -3 | -2.649 |
|      | BTD | HCPS | Gap (BTD vs. HCPS) |
|------|-----|------|-------------------|
| 10   | 20  | 19   | 11.124            |
|      |     |      | 236.719           |
|      |     |      | 1                 |
|      |     |      | 35.219            |
|      | 19  | 18   | -13.228           |
|      |     |      | 224.509           |
|      |     |      | 1                 |
|      |     |      | 60.668            |
|      | 1   | 1    | 12.21             |
|      |     |      | 12.21             |
|      |     |      | -                  |
|      |     |      | -                  |
| 15   | 23  | 25   | -1                 |
|      |     |      | 279.518           |
|      |     |      | 2                 |
|      |     |      | 58.466            |
|      | -2  | 0    | 7.126             |
|      |     |      | 2.08              |
|      |     |      | -                  |
|      |     |      | -                  |
| 20   | 23  | 25   | 13.239            |
|      |     |      | -13.239           |
|      |     |      | -12.21            |
|      |     |      | -12.21            |
|      |     |      | -                  |
|      | 28  | 24   | 5.08              |
|      |     |      | 5.08              |
|      |     |      | -                  |
|      |     |      | -                  |
| 5    | 15  | 16   | 188.846           |
|      |     |      | 163.958           |
|      |     |      | -1               |
|      |     |      | 24.888           |
|      | 15  | 15   | 183.766           |
|      |     |      | 174.367           |
|      |     |      | 0                 |
|      |     |      | 9.399            |
| 10   | 19  | 19   | -12.21            |
|      |     |      | -12.21           |
|      |     |      | -                  |
|      |     |      | -                  |
| 8    | 22  | 24   | 315.499           |
|      |     |      | 248.248           |
|      |     |      | 0                 |
|      |     |      | 67.251            |
|      | 23  | 23   | 302.573           |
|      |     |      | 251.514           |
|      |     |      | 0                 |
|      |     |      | 51.059            |
|      | -1  | 1    | 15.388           |
|      |     |      | 27.651            |
|      |     |      | 1                  |
|      |     |      | -                  |
|      |     |      | -                  |
| 15   | 32  | 30   | 459.328           |
|      |     |      | 414.339           |
|      |     |      | 2                 |
|      |     |      | 44.989            |
|      | 33  | 31   | 454.509           |
|      |     |      | 406.337           |
|      |     |      | 2                 |
|      |     |      | 48.172            |
|      | -1  | -1   | 4.819             |
|      |     |      | 8.002             |
|      |     |      | -                 |
|      |     |      | -                 |
| 20   | 15  | 15   | 202.043           |
|      |     |      | 192.197           |
|      |     |      | 0                 |
|      |     |      | 9.846            |
|      | 15  | 16   | 205.974           |
|      |     |      | 194.847           |
|      |     |      | -1               |
|      |     |      | 11.127            |
|      | 0   | -1   | -3.931            |
|      |     |      | -2.65             |
|      |     |      | -                  |
|      |     |      | -                  |
| 10   | 22  | 22   | 315.499           |
|      |     |      | 248.248           |
|      |     |      | 0                 |
|      |     |      | 67.251            |
|      | 23  | 23   | 302.573           |
|      |     |      | 251.514           |
|      |     |      | 0                 |
|      |     |      | 51.059            |
|      | -1  | 1    | 15.388           |
|      |     |      | 27.651            |
|      |     |      | 1                  |
|      |     |      | -                  |
|      |     |      | -                  |
| 9    | 29  | 30   | 432.421           |
|      |     |      | 411.984           |
|      |     |      | -1               |
|      |     |      | 20.437            |
|      | 31  | 28   | 410.161           |
|      |     |      | 387.925           |
|      |     |      | 3                 |
|      |     |      | 22.236            |
|      | -2  | 2    | 22.26            |
|      |     |      | 24.059            |
|      |     |      | -                 |
|      |     |      | -                 |
| 15   | 32  | 32   | 480.981           |
|      |     |      | 450.828           |
|      |     |      | 0                 |
|      |     |      | 30.153            |
|      | 30  | 30   | 460.842           |
|      |     |      | 419.316           |
|      |     |      | 0                 |
|      |     |      | 41.526            |
|      | -2  | 2    | 20.139            |
|      |     |      | 31.512            |
|      |     |      | -                 |
|      |     |      | -                 |
| No. | BTD  | HCPS  | Gap (BTD vs. HCPS) |
|-----|------|-------|-------------------|
| 5   | 18   | 254.706 | 16             | 204.258 | 2 | 50.448 |
|     | 19   | 249.439 | 16             | 217.141 | 3 | 32.298 |
|     | -1   | 5.267   | 0              | -12.883 | - | -     |
| 10  | 24   | 420.902 | 24             | 345.720 | 0 | 75.182 |
|     | 23   | 383.965 | 23             | 335.836 | 0 | 48.129 |
|     | 1    | 36.937  | 1              | 9.884   | - | -     |
|     | 31   | 491.675 | 29             | 435.426 | 2 | 56.249 |
|     | 31   | 461.697 | 30             | 446.390 | 1 | 15.307 |
|     | 0    | 29.978  | -1             | -10.964 | - | -     |
| 15  | 34   | 580.228 | 31             | 507.270 | 3 | 72.958 |
|     | 32   | 542.657 | 32             | 515.494 | 0 | 27.163 |
|     | 2    | 37.571  | -1             | -8.224  | - | -     |

**Notes:** 1) Gap (HQM vs. GA) = HQM – GA; 2) Gap (BTD vs. HCPS) = BTD - HCPS
Appendix E. Problem instances (small-size network)
The original data of different sizes of problem instances consists of three parts: i) the data of the parking spaces, which includes the coordinates of the locations and the available parking time. ii) the data of customers, which includes the coordinates of the locations, time-windows of the current location, demand, customers’ ID it belongs to, and the parking spaces’ ID it belongs to. iii) task list obtained from the task generator, which includes a set of sub-tasks determined, the coordinates of the parking space that will perform the current task, the sub-time interval of the task, the total demands of the sub-task, and the corresponding service time. Due to the page limitations, we provide the data of four small-size problem instances ($I_l = 3$, $N_n=5$; $I_l = 4$, $N_n=10$, $I_l = 5$, $N_n=5$, and $I_l = 6$, $N_n=10$).

Table E.1 The original data of the parking spaces ($I_l = 3$, $N_n=5$)

| Parking spaces ID | $X_l$ | $Y_l$ | Available parking time (min) |
|-------------------|------|------|-------------------------------|
| 0                 | 2.521287 | 0.912737 | 265                          |
| 1                 | 3.205659 | 3.703741 | 247                          |
| 2                 | 3.830499 | 0.193539 | 231                          |

Table E.2 The original data of customers ($I_l = 3$, $N_n=5$)

| Locations | $X_{nk}$ | $Y_{nk}$ | Earliest time $E_{nk}$ | Latest time $L_{nk}$ | Demand | Customers’ ID | Parking spaces’ ID |
|-----------|----------|----------|------------------------|----------------------|--------|--------------|-------------------|
| 0         | 4.001592 | -0.08038 | 620                    | 645                  | 3      | 0            | 2                 |
| 1         | 3.627509 | 0.206915 | 725                    | 752                  | 1      | 0            | 2                 |
| 2         | 3.863388 | 0.277737 | 540                    | 588                  | 2      | 1            | 2                 |
| 3         | 3.919225 | 0.227169 | 1008                   | 1044                 | 4      | 1            | 2                 |
| 4         | 3.74078  | 0.336257 | 780                    | 815                  | 2      | 1            | 2                 |
| 5         | 3.231472 | 3.2122   | 925                    | 958                  | 4      | 1            | 1                 |
| 6         | 2.979996 | 4.002823 | 790                    | 850                  | 4      | 2            | 1                 |
| 7         | 3.081792 | 3.645949 | 710                    | 763                  | 1      | 3            | 1                 |
| 8         | 3.343271 | 3.763546 | 1013                   | 1050                 | 3      | 3            | 1                 |
| 9         | 3.391767 | 3.894187 | 870                    | 918                  | 4      | 3            | 1                 |
| 10        | 2.52824  | 1.798009 | 968                    | 1032                 | 4      | 3            | 0                 |
| 11        | 2.194383 | 0.448486 | 590                    | 616                  | 3      | 4            | 0                 |
| 12        | 2.163809 | 0.487288 | 986                    | 1062                 | 1      | 4            | 0                 |
Notes: 1) \([E_{nk}, L_{nk}]\) represents the duration of staying at the current location; 2) Customers’ ID represents the current location/whereabout belongs to which customer; 2) Parking spaces’ ID represents the current location/whereabout to be fulfilled by which parking space if the customer will be served at this location.

Table E.3 The task list obtained from the task generator \((I_t = 3, N_n=5)\)

| Tasks ID | \(X_{nk}\)     | \(Y_{nk}\)     | Earliest time of task execution \(e_{ia}\) | Latest time of task execution \(l_{ia}\) | Total demands to be fulfilled (units) | Service time \(SV_t\) (mins) |
|----------|----------------|----------------|------------------------------------------|------------------------------------------|--------------------------------------|-----------------------------|
| 0        | 2.521287       | 0.912737       | 590                                      | 855                                      | 5                                    | 17.097                      |
| 1        | 2.521287       | 0.912737       | 968                                      | 1233                                     | 9                                    | 21.066                      |
| 2        | 3.205659       | 3.703741       | 710                                      | 957                                      | 9                                    | 14.683                      |
| 3        | 3.205659       | 3.703741       | 925                                      | 1172                                     | 7                                    | 16.153                      |
| 4        | 3.830499       | 0.193539       | 540                                      | 771                                      | 6                                    | 14.037                      |
| 5        | 3.830499       | 0.193539       | 1011                                     |                                          | 6                                    | 12.107                      |

Notes: 1) \([X_{nk}, Y_{nk}]\) represents the location of the parking space that will serve the current task;

Table E.4 The original data of the parking spaces \((I_l = 4, N_n=10)\)

| Parking spaces ID | \(X_l\)     | \(Y_l\)     | Available parking time (min) |
|-------------------|-------------|-------------|------------------------------|
| 0                 | 3.83562     | 0.182881    | 246                          |
| 1                 | 0.989961    | 3.786384    | 263                          |
| 2                 | 2.502348    | 1.084103    | 238                          |
| 3                 | 3.051987    | 3.740668    | 250                          |

Table E.5 The original data of customers \((I_l = 4, N_n=10)\)

| Locations | \(X_{nk}\)     | \(Y_{nk}\)     | Earliest time \(E_{nk}\) | Latest time \(L_{nk}\) | Demand | Customers’ ID | Parking spaces’ ID |
|-----------|----------------|----------------|--------------------------|------------------------|--------|--------------|-------------------|
| 0         | 3.648962       | 0.197255       | 970                      | 1039                   | 1      | 0            | 0                 |
| 1         | 3.862753       | 0.261446       | 580                      | 653                    | 2      | 1            | 0                 |
|   |   |   |   |   |
|---|---|---|---|---|
| 2 | 3.913361 | 0.215613 | 773 | 817 | 2 |
| 3 | 3.751626 | 0.314485 | 1037 | 1089 | 3 |
| 4 | 3.964835 | 0.125219 | 540 | 604 | 3 |
| 5 | 3.676114 | 0.039873 | 1070 | 1090 | 3 |
| 6 | 3.944436 | 0.07442 | 730 | 790 | 1 |
| 7 | 3.82332 | 0.163161 | 740 | 920 | 1 |
| 8 | 3.86323 | 0.17962 | 920 | 963 | 1 |
| 9 | 3.907567 | 0.257714 | 1020 | 1063 | 1 |
| 10 | 3.63678 | 3.531836 | 830 | 863 | 3 |
| 11 | 2.701789 | 3.517024 | 863 | 887 | 4 |
| 12 | 2.652172 | 3.887856 | 1057 | 1103 | 4 |
| 13 | 3.22244 | 4.107676 | 810 | 841 | 1 |
| 14 | 2.825353 | 4.150459 | 1071 | 1118 | 4 |
| 15 | 2.999279 | 4.034054 | 740 | 816 | 2 |
| 16 | 2.935849 | 4.18131 | 816 | 885 | 4 |
| 17 | 2.960505 | 3.362484 | 980 | 1036 | 2 |
| 18 | 3.045836 | 3.280985 | 830 | 889 | 3 |
| 19 | 3.533969 | 3.346694 | 1019 | 1043 | 1 |
| 20 | 2.453585 | 1.414691 | 600 | 676 | 3 |
| 21 | 2.270908 | 1.210781 | 936 | 1011 | 1 |
| 22 | 2.851622 | 1.154788 | 1051 | 1076 | 4 |
| 23 | 2.572941 | 0.713296 | 1060 | 1086 | 3 |
| 24 | 2.365226 | 0.749248 | 990 | 1015 | 4 |
| 25 | 2.319121 | 0.973085 | 1045 | 1084 | 1 |
| 26 | 2.429684 | 0.959008 | 720 | 748 | 1 |
| 27 | 2.614204 | 0.987016 | 910 | 989 | 2 |
| 28 | 2.727189 | 1.461021 | 1019 | 1052 | 2 |
| Tasks ID | $X_{nk}$ | $Y_{nk}$ | Earliest time of task execution $e_{la}$ | Latest time of task execution $l_{la}$ | Total demands to be fulfilled (units) | Service time $SV_t$ (min) |
|----------|---------|---------|---------------------------------|---------------------------------|----------------------------------|-------------------|
| 0        | 3.83562 | 0.182881| 540                            | 786                            | 8                               | 11.92             |
| 1        | 3.83562 | 0.182881| 773                            | 1019                           | 4                               | 12.34             |
| 2        | 3.83562 | 0.182881| 1020                           | 1266                           | 7                               | 12.678            |
| 3        | 0.989961| 3.786384| 640                            | 903                            | 7                               | 13,428            |
| 4        | 0.989961| 3.786384| 700                            | 963                            | 10                              | 16,072            |
| 5        | 0.989961| 3.786384| 960                            | 1223                           | 10                              | 16,738            |
| 6        | 0.989961| 3.786384| 1072                           | 1335                           | 7                               | 15,144            |
| 7        | 2.502348| 1.084103| 550                            | 788                            | 8                               | 14,177            |
| 8        | 2.502348| 1.084103| 910                            | 1148                           | 10                              | 15,486            |
| 9        | 2.502348| 1.084103| 1051                           | 1289                           | 7                               | 14,718            |

Notes: 1) $[E_{nk}, L_{nk}]$ represents the duration of staying at the current location; 2) Customers’ ID represents the current location/whereabout belongs to which customer; 3) Parking spaces’ ID represents the current location/whereabout to be fulfilled by which parking space if the customer will be served at this location.
Notes: 1) \([X_{nk}, Y_{nk}]\) represents the location of the parking space that will serve the current task;

| Parking spaces ID | \(X_i\)       | \(Y_i\)    | Available parking time (min) |
|-------------------|---------------|------------|------------------------------|
| 0                 | 3.429408      | 3.712662   | 235                          |
| 1                 | 0.814192      | 0.131808   | 232                          |
| 2                 | 2.55478       | 1.052008   | 241                          |
| 3                 | 1.137501      | 3.555553   | 243                          |
| 4                 | 3.791613      | 0.352651   | 254                          |

| Locations | \(X_{nk}\)    | \(Y_{nk}\)   | Earliest time \(E_{nk}\) | Latest time \(L_{nk}\) | Demand | Customers' ID | Parking spaces' ID |
|-----------|---------------|---------------|--------------------------|------------------------|--------|--------------|-------------------|
| 0         | 3.874083      | 0.551948      | 580                      | 629                    | 2      | 0            | 4                 |
| 1         | 4.017926      | 0.421679      | 1039                     | 1106                   | 1      | 0            | 4                 |
| 2         | 3.558227      | 0.702701      | 970                      | 1030                   | 3      | 0            | 4                 |
| 3         | 4.164229      | 0.164753      | 780                      | 830                    | 1      | 1            | 4                 |
| 4         | 3.343601      | -0.07782      | 1020                     | 1063                   | 3      | 2            | 4                 |
| 5         | 3.086026      | 3.650752      | 820                      | 880                    | 3      | 2            | 0                 |
| 6         | 3.334698      | 3.76259       | 930                      | 1005                   | 3      | 2            | 0                 |
| 7         | 3.380819      | 3.886832      | 1075                     | 1151                   | 1      | 2            | 0                 |
| 8         | 3.599656      | 3.774616      | 740                      | 790                    | 1      | 3            | 0                 |
| 9         | 3.745842      | 3.48852       | 920                      | 953                    | 1      | 3            | 0                 |
| 10        | 2.420224      | 0.983926      | 1020                     | 1063                   | 1      | 4            | 2                 |
| Tasks ID | $X_{nk}$ | $Y_{nk}$ | Earliest time of task execution $e_{i \alpha}$ | Latest time of task execution $l_{i \alpha}$ | Total demands to be fulfilled (units) | Service time $SV_i$ (min) |
|----------|----------|----------|---------------------------------|---------------------------------|-------------------------------------|------------------------|
| 0        | 3.429408 | 3.712662 | 740                             | 975                             | 8                                    | 14.847                 |
| 1        | 3.429408 | 3.712662 | 1075                            | 1310                            | 1                                    | 12.26                  |
| 2        | 0.814192 | 0.131808 | 600                             | 832                             | 3                                    | 20.008                 |
| 3        | 0.814192 | 0.131808 | 936                             | 1168                            | 9                                    | 17.659                 |
| 4        | 2.55478  | 1.052008 | 810                             | 1051                            | 9                                    | 11.885                 |
| 5        | 2.55478  | 1.052008 | 1057                            | 1298                            | 4                                    | 10.87                  |
| 6        | 1.137501 | 3.555553 | 740                             | 983                             | 9                                    | 19.32                  |

Notes: 1) $[E_{nk}, L_{nk}]$ represents the duration of staying at the current location; 2) Customers' ID represents the current location/whereabout belongs to which customer; 2) Parking spaces' ID represents the current location/whereabout to be fulfilled by which parking space if the customer will be served at this location.
7 1.137501 3.555553 980 1223 6 23.769
8 3.791613 0.352651 580 834 3 15.216
9 3.791613 0.352651 970 1224 7 17.766

Notes: 1) \([X_{nk}, Y_{nk}]\) represents the location of the parking space that will serve the current task;

### Table E.10 The original data of the parking spaces \((l_t = 6, N_n=10)\)

| Parking spaces ID | \(X_t\)   | \(Y_t\)   | Available parking time (min) |
|-------------------|-----------|-----------|-------------------------------|
| 0                 | 3.412416  | 4.807321  | 259                           |
| 1                 | 2.514307  | 1.035503  | 256                           |
| 2                 | 0.989961  | 3.786384  | 257                           |
| 3                 | 3.856962  | 0.140871  | 254                           |
| 4                 | 0.800079  | 0.221703  | 266                           |
| 5                 | 3.048543  | 3.790156  | 252                           |

### Table E.11 The original data of customers \((l_t = 6, N_n=10)\)

| Locations | \(X_{nk}\)   | \(Y_{nk}\)   | Earliest time \(E_{nk}\) | Latest time \(L_{nk}\) | Demand | Customers’ ID | Parking spaces’ ID |
|-----------|----------------|----------------|--------------------------|--------------------------|--------|---------------|-------------------|
| 0         | 3.859128       | 0.168506       | 610                      | 675                      | 3      | 0             | 3                 |
| 1         | 3.879908       | 0.149687       | 1075                     | 1104                     | 1      | 0             | 3                 |
| 2         | 3.813499       | 0.190284       | 590                      | 625                      | 2      | 0             | 3                 |
| 3         | 3.901043       | 0.112571       | 735                      | 787                      | 2      | 0             | 3                 |
| 4         | 3.782494       | 0.077528       | 670                      | 722                      | 2      | 1             | 3                 |
| 5         | 3.892667       | 0.091713       | 1052                     | 1097                     | 4      | 1             | 3                 |
| 6         | 3.842937       | 0.12815        | 650                      | 700                      | 3      | 1             | 3                 |
| 7         | 3.859324       | 0.134908       | 1050                     | 1095                     | 2      | 1             | 3                 |
| 8         | 3.877529       | 0.166974       | 1060                     | 1085                     | 3      | 2             | 3                 |
|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 9 | 3.861086 | 0.188394 | 980 | 1031 | 3 |
| 10 | 2.851001 | 3.589637 | 750 | 795 | 3 |
| 11 | 2.817256 | 3.841845 | 880 | 958 | 1 |
| 12 | 3.205103 | 3.991347 | 988 | 1017 | 1 |
| 13 | 2.935038 | 4.020445 | 1047 | 1075 | 4 |
| 14 | 3.053328 | 3.941276 | 690 | 720 | 3 |
| 15 | 3.010188 | 4.041426 | 600 | 649 | 1 |
| 16 | 3.033078 | 3.484532 | 779 | 799 | 2 |
| 17 | 3.084992 | 3.429104 | 919 | 947 | 1 |
| 18 | 3.416978 | 3.473793 | 930 | 991 | 2 |
| 19 | 3.078473 | 4.088155 | 1011 | 1035 | 1 |
| 20 | 2.406117 | 1.768965 | 570 | 623 | 3 |
| 21 | 2.000821 | 1.316558 | 680 | 755 | 2 |
| 22 | 3.289226 | 1.192329 | 975 | 1053 | 1 |
| 23 | 2.670929 | 0.212809 | 620 | 668 | 2 |
| 24 | 2.21008 | 0.292573 | 1068 | 1094 | 3 |
| 25 | 2.107788 | 0.789192 | 990 | 1015 | 4 |
| 26 | 2.35309 | 0.757959 | 1045 | 1084 | 1 |
| 27 | 2.762477 | 0.8201 | 720 | 748 | 1 |
| 28 | 3.013153 | 1.871755 | 910 | 989 | 2 |
| 29 | 2.329391 | 1.332787 | 1019 | 1052 | 2 |
| 30 | 1.100306 | 3.535327 | 550 | 586 | 4 |
| 31 | 0.768909 | 3.512685 | 700 | 772 | 3 |
| 32 | 1.055046 | 3.876299 | 1072 | 1147 | 3 |
| 33 | 0.962357 | 3.569709 | 640 | 700 | 4 |
|    |        |        |        |      |    | |
|----|--------|--------|--------|-----|----|---|
| 34 | 0.810107 | 3.797087 | 760    | 800  | 2  | 11 |
| 35 | 1.526668 | 3.736069 | 1070   | 1132 | 3  | 11 |
| 36 | 0.786009 | 3.82491  | 960    | 1018 | 3  | 11 |
| 37 | 1.078158 | 4.20056  | 700    | 779  | 4  | 12 |
| 38 | 0.504752 | 3.762778 | 969    | 1038 | 4  | 12 |
| 39 | 1.307293 | 4.048416 | 850    | 916  | 4  | 12 |
| 40 | 0.264682 | 0.254297 | 1076   | 1140 | 4  | 12 |
| 41 | 0.924848 | 0.190392 | 770    | 811  | 1  | 13 |
| 42 | 0.861032 | 0.659372 | 1061   | 1105 | 2  | 13 |
| 43 | 0.642039 | -0.05012 | 740    | 781  | 3  | 13 |
| 44 | 0.676609 | 0.0888   | 670    | 745  | 1  | 14 |
| 45 | 1.162396 | -0.0516  | 815    | 884  | 1  | 14 |
| 46 | 1.152492 | 0.403784 | 924    | 986  | 3  | 14 |
| 47 | 0.968849 | -0.00048 | 750    | 789  | 3  | 15 |
| 48 | 0.449069 | 0.623741 | 550    | 574  | 4  | 16 |
| 49 | 0.89877  | 0.098843 | 600    | 663  | 4  | 17 |
| 50 | 3.485976 | 4.79664  | 973    | 1036 | 2  | 17 |
| 51 | 3.491542 | 5.025621 | 780    | 809  | 1  | 18 |
| 52 | 3.481372 | 5.10149  | 929    | 1006 | 2  | 18 |
| 53 | 3.501442 | 4.952121 | 1046   | 1124 | 2  | 18 |
| 54 | 3.11521  | 4.703535 | 760    | 832  | 4  | 18 |
| 55 | 3.598691 | 4.416446 | 630    | 670  | 2  | 19 |
| 56 | 3.373257 | 4.471041 | 820    | 872  | 3  | 20 |
| 57 | 3.432986 | 5.049805 | 1002   | 1068 | 2  | 20 |
| 58 | 3.484536 | 4.846177 | 1010   | 1081 | 4  | 21 |
Notes: 1) \([E_{nk}, L_{nk}]\) represents the duration of staying at the current location; 2) Customers’ ID represents the current location/whereabout belongs to which customer; 2) Parking spaces’ ID represents the current location/whereabout to be fulfilled by which parking space if the customer will be served at this location.

Table E.12 The task list obtained from the task generator \((l_i = 6, N_n =10)\)

| Tasks ID | \(X_{nk}\) | \(Y_{nk}\) | Earliest time of task execution \(e_{ia}\) | Latest time of task execution \(l_{ia}\) | Total demands to be fulfilled (units) | Service time \(SV_i\) (min) |
|----------|------------|------------|-----------------|-----------------|----------------------------------|------------------|
| 0        | 3.412416   | 4.807321   | 630             | 889             | 7                                | 15.412           |
| 1        | 3.412416   | 4.807321   | 820             | 1079            | 11                               | 14.232           |
| 2        | 3.412416   | 4.807321   | 1002            | 1261            | 8                                | 13.042           |
| 3        | 2.514307   | 1.035503   | 570             | 826             | 8                                | 20.468           |
| 4        | 2.514307   | 1.035503   | 910             | 1166            | 10                               | 22.172           |
| 5        | 2.514307   | 1.035503   | 1068            | 1324            | 3                                | 20.035           |
| 6        | 0.989961   | 3.786384   | 550             | 807             | 8                                | 13.428           |
| 7        | 0.989961   | 3.786384   | 700             | 957             | 9                                | 15.293           |
| 8        | 0.989961   | 3.786384   | 850             | 1107            | 11                               | 16.072           |
| 9        | 0.989961   | 3.786384   | 1070            | 1327            | 6                                | 16.738           |
| 10       | 3.856962   | 0.140871   | 590             | 844             | 8                                | 10.823           |
| 11       | 3.856962   | 0.140871   | 670             | 924             | 4                                | 11.222           |
| 12       | 3.856962   | 0.140871   | 980             | 1234            | 12                               | 10.759           |
| 13       | 3.856962   | 0.140871   | 1075            | 1329            | 1                                | 10.307           |
| 14       | 0.800079   | 0.221703   | 550             | 816             | 9                                | 16.671           |
| 15       | 0.800079   | 0.221703   | 740             | 1006            | 11                               | 15.673           |
| 16       | 0.800079   | 0.221703   | 1061            | 1327            | 6                                | 16.705           |
| 17       | 3.048543   | 3.790156   | 600             | 852             | 9                                | 13.825           |
| 18       | 3.048543   | 3.790156   | 880             | 1132            | 10                               | 16.07            |
Notes: 1) $[X_{nk}, Y_{nk}]$ represents the location of the parking space that will serve the current task;
Appendix F. Implementation results for small-size networks

We now provide detailed and traceable routes for the problem instances presented in the Appendix C. For each delivery route of MPL, the result consists of the task execution sequence, arriving time and leaving time of MPLs against parking space, service delay of each MPL at different parking spaces, and the real-time demand fulfilment of MPLs.

Table F.1 Implementation result of the problem instance with BTD applied ($I_t = 3, N_r = 5$)

| Delivery Route | Task execution sequence (Task’ ID) |
|----------------|-------------------------------------|
| MPL’s ID       | [6, 0, 4, 6, 3, 2, 6, 1, 5, 6]      |

| Arriving Time When Executing Task | Starting Time of the Corresponding Task |
|-----------------------------------|-----------------------------------------|
| MPL’s ID                         | [420, 590.0, 609.2, 628.7, 925, 941.2, 962.8, 968, 991.2, 1008.8] |

| Leaving Time When Task Completed | Ending Time of the Corresponding Task |
|----------------------------------|--------------------------------------|
| MPL’s ID                         | [586.2, 607.1, 623.3, 918.0, 941.2, 955.8, 964.2, 989.1, 1003.3, 0] |

| Service Delays | Delays of the Corresponding Task (min) |
|----------------|----------------------------------------|
| MPL’s ID       | [0, 0.0, 0.0, 0, 0, 0.0, 0, 0.0, 0, 0] |

| Real-time Demand Fulfilment | Accumulated Number of Parcels that have been fulfilled (units) |
|-----------------------------|---------------------------------------------------------------|
| MPL’s ID                    | [0, 5, 11, 0, 7, 16, 0, 9, 15, 0]                              |

Notes: 1) MPLs depart from the depot and will return to the depot once all tasks are completed. Therefore, ‘6’ represent the depot here. Other numbering represents the task ID. 2) The MPLs may return to the depot during a delivery round to meet time windows or demand constraints. 3) The start time of executing task 0 is 590.0, and the task is completed at 607.1. Following this logic to understand the duration of performing other tasks. 4) If MPLs have returned to the depot, the real-time demand fulfilment of the corresponding task will reset to 0, and will recalculate once MPLs start to perform the remaining tasks.
### Table F.2 Implementation result of the problem instance with HCPS applied ($I_t = 3, N_n=5$)

| MPL’s ID | Delivery Route | Task execution sequence (Task’ ID) |
|----------|----------------|-----------------------------------|
| 0        |                | [6, 1, 5, 6, 4, 0, 6, 3, 2, 6]    |

**Arriving Time When Executing Task**

| MPL’s ID | Arrival Time When Executing Task |
|----------|---------------------------------|
| 0        | [420, 968.0, 991.2, 1008.8, 540, 590.0, 610.9, 925, 941.2, 962.8] |

**Leaving Time When Task Completed**

| MPL’s ID | Leaving Time When Task Completed |
|----------|---------------------------------|
| 0        | [964.2, 989.1, 1003.3, 534.5, 554.0, 607.1, 918.0, 941.2, 955.8, 0] |

**Service Delays**

| MPL’s ID | Delays of the Corresponding Task (min) |
|----------|----------------------------------------|
| 0        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]          |

**Real-time Demand Fulfilment**

| MPL’s ID | Accumulated Number of Parcels that have been fulfilled (units) |
|----------|--------------------------------------------------------------|
| 0        | [0, 9, 15, 0, 6, 11, 0, 7, 16, 0]                            |

### Table F.3 Implementation result of the problem instance with BTD applied ($I_t = 4, N_n=10$)

| MPL’s ID | Delivery Route | Task execution sequence (Task’ ID) |
|----------|----------------|-----------------------------------|
| 0        |                | [13, 0, 13, 1, 13, 2, 8, 13, 7, 13, 10, 3, 13, 12, 13, 11, 13, 4, 13, 9, 13, 6, 5, 13] |

**Arriving Time When Executing Task**

| MPL’s ID | Arrival Time When Executing Task |
|----------|---------------------------------|
| 0        | [420, 540.0, 557.4, 773, 790.8, 1020, 1035.0, 1054.4, 550, 568.1, 740, 758.6, 777.7, 1019, 1043.7, 830, 854.6, 700, 721.7, 1051, 1069.6, 1072, 1087.1, 1109.5] |
| MPL's ID | Ending Time of the Corresponding Task |
|----------|---------------------------------------|
| 0        | [534.5, 551.9, 767.5, 785.3, 1014.5, 1032.7, 1050.5, 546.1, 564.2, 733.1, 755.7, 772.1, 1012.1, 1036.8, 823.1, 847.7, 694.4, 716.1, 1047.1, 1065.7, 1066.4, 1087.1, 1103.9, 0] |

**Service Delays**

| MPL's ID | Delays of the Corresponding Task (min) |
|----------|----------------------------------------|
| 0        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |

**Real-time Demand Fulfilment**

| MPL's ID | Accumulated Number of Parcels that have been fulfilled (units) |
|----------|---------------------------------------------------------------|
| 0        | [0, 8, 0, 4, 0, 7, 17, 0, 8, 0, 7, 14, 0, 9, 0, 12, 0, 10, 0, 7, 0, 7, 17, 0] |

**Notes:** Number ‘13’ represents the central depot in this case.

**Table F.4** Implementation result of the problem instance with HCPS applied ($I_t=4, N_t=10$)

**Delivery Route**

| MPL’s ID | Task execution sequence (Task’ ID) |
|----------|------------------------------------|
| 0        | [13, 2, 8, 13, 10, 11, 13, 4, 3, 13, 9, 13] |
| 1        | [13, 0, 13, 5, 12, 13] |
| 2        | [13, 1, 7, 13, 6, 13] |

**Arriving Time When Executing Task**

| MPL’s ID | Starting Time of the Corresponding Task |
|----------|----------------------------------------|
| 0        | [420, 1020.0, 1035.0, 1054.4, 740, 830.0, 854.6, 700, 716.1, 735.1, 1051, 1069.6] |
| 1        | [420, 540.0, 557.4, 960, 1019.0, 1043.7] |
| 2        | [420, 773.0, 787.6, 805.7, 1072, 1092.7] |

**Leaving Time When Task Completed**

| MPL’s ID | Ending Time of the Corresponding Task |
|----------|---------------------------------------|
| 0        | [1014.5, 1032.7, 1050.5, 733.1, 755.7, 847.7, 694.4, 716.1, 729.5, 1047.1, 1065.7, 0] |
### Service Delays

| MPL’s ID | Delays of the Corresponding Task (min) |
|----------|----------------------------------------|
| 0        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]           |
| 1        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]           |
| 2        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]           |

### Real-time Demand Fulfilment

| MPL’s ID | Accumulated Number of Parcels that have been fulfilled (units) |
|----------|---------------------------------------------------------------|
| 0        | [0, 7, 17, 0, 7, 19, 0, 10, 17, 0, 7, 0]                      |
| 1        | [0, 8, 0, 10, 19, 0]                                          |
| 2        | [0, 4, 12, 0, 7, 0]                                           |

**Notes:** Number ‘13’ represents the central depot in this case.

**Table F.5** Implementation result of the problem instance with BTD applied \((I = 5, N_a=5)\)

### Delivery Route

| MPL’s ID | Task execution sequence (Task’ ID) |
|----------|------------------------------------|
| 0        | [10, 6, 8, 0, 10, 3, 4, 10, 2, 10, 9, 10, 5, 10, 1, 7, 10] |

### Arriving Time When Executing Task

| MPL’s ID | Starting Time of the Corresponding Task |
|----------|-----------------------------------------|
| 0        | [420, 740.0, 765.3, 785.3, 807.4, 936, 956.5, 972.3, 600, 621.2, 970, 993.2, 1057, 1071.8, 1075, 1090.5, 1119.6] |

### Leaving Time When Task Completed

| MPL’s ID | Ending Time of the Corresponding Task |
|----------|---------------------------------------|
| 0        | [734.7, 759.3, 780.5, 800.2, 934.8, 953.7, 968.4, 598.8, 620.0, 964.6, 987.8, 1053.1, 1067.9, 1067.8, 1087.3, 1114.3, 0] |

### Service Delays
| MPL’s ID | Delays of the Corresponding Task (min) |
|----------|----------------------------------------|
| 0        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]          |

**Real-time Demand Fulfilment**

| MPL’s ID | Accumulated Number of Parcels that have been fulfilled (units) |
|----------|---------------------------------------------------------------|
| 0        | [0, 9, 12, 20, 0, 9, 18, 0, 3, 0, 7, 0, 4, 0, 1, 7, 0]          |

**Notes:** Number ‘10’ represents the central depot in this case.

**Table F.6** Implementation result of the problem instance with HCPS applied ($I_i = 5, N_{n}=5$)

### Delivery Route

| MPL’s ID | Task execution sequence (Task’ ID) |
|----------|-----------------------------------|
| 0        | [10, 8, 2, 0, 10, 6, 7, 1, 5, 10]  |
| 1        | [10, 9, 4, 10, 3, 10]             |

### Arriving Time When Executing Task

| MPL’s ID | Starting Time of the Corresponding Task |
|----------|----------------------------------------|
| 0        | [420, 580.0, 600.0, 740.0, 762.1, 740, 980.0, 1075.0, 1091.3, 1106.1] |
| 1        | [420, 970.0, 989.8, 1005.6, 934.8, 953.7, 0] |

### Leaving Time When Task Completed

| MPL’s ID | Ending Time of the Corresponding Task |
|----------|--------------------------------------|
| 0        | [574.6, 595.2, 620.0, 754.8, 734.7, 759.3, 1003.8, 1087.3, 1102.1, 0] |
| 1        | [964.6, 987.8, 1001.7, 934.8, 953.7, 0] |

### Service Delays

| MPL’s ID | Delays of the Corresponding Task (min) |
|----------|----------------------------------------|
| 0        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]          |
| 1        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]          |

**Real-time Demand Fulfilment**
| MPL’s ID | Accumulated Number of Parcels that have been fulfilled (units) |
|----------|-------------------------------------------------------------|
| 0        | [0, 3, 6, 14, 0, 9, 15, 16, 20, 0]                          |
| 1        | [0, 7, 16, 0, 9, 0]                                        |

Notes: Number ‘10’ represents the central depot in this case.

Table F.7 Implementation result of the problem instance with BTD applied ($I_f = 6$, $N_i=10$)

| Delivery Route |
|----------------|
| MPL’s ID       | Task execution sequence (Task’ ID)                      |
| 0              | [19, 11, 17, 19, 15, 14, 19, 7, 6, 19, 13, 18, 19, 12, 19] |
| 1              | [19, 0, 19, 8, 19]                                       |
| 2              | [19, 10, 3, 19]                                          |
| 3              | [19, 1, 19, 16, 5, 19]                                   |
| 4              | [19, 2, 19, 9, 4, 19]                                    |

| Arriving Time When Executing Task |
|-----------------------------------|
| MPL’s ID       | Starting Time of the Corresponding Task                  |
| 0              | [420, 670.0, 686.6, 707.3, 740, 755.7, 773.5, 700, 715.3, 734.3, 1075, 1090.6, 1113.7, 980, 996.3] |
| 1              | [420, 630.0, 653.8, 850, 871.7]                           |
| 2              | [420, 590.0, 603.1, 627.5]                                |
| 3              | [420, 820.0, 842.7, 1061, 1080.4, 1104.3]                 |
| 4              | [420, 1002.0, 1023.5, 1070, 1091.2, 1117.3]               |

| Leaving Time When Task Completed |
|----------------------------------|
| MPL’s ID       | Ending Time of the Corresponding Task                     |
| 0              | [664.5, 681.2, 700.4, 738.8, 755.7, 772.3, 694.4, 715.3, 728.7, 1069.5, 1085.3, 1106.7, 974.5, 990.8, 0] |
| 1              | [621.6, 645.4, 844.4, 866.1, 0]                           |
| 2              | [584.5, 600.8, 623.6, 0]                                  |
| 3              | [811.6, 834.2, 1059.8, 1077.7, 1100.5, 0]                |

73
### Service Delays

| MPL’s ID | Delays of the Corresponding Task (min) |
|----------|---------------------------------------|
| 0        | [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| 1        | [0, 0, 0, 0, 0] |
| 3        | [0, 0, 0, 0, 0] |
| 3        | [0, 0, 0, 0, 0] |
| 4        | [0, 0, 0, 0, 0] |

### Real-time Demand Fulfilment

| MPL’s ID | Accumulated Number of Parcels that have been fulfilled (units) |
|----------|---------------------------------------------------------------|
| 0        | [0, 4, 13, 0, 11, 20, 0, 9, 17, 0, 1, 11, 0, 12, 0] |
| 1        | [0, 7, 0, 11, 0] |
| 2        | [0, 8, 16, 0] |
| 3        | [0, 11, 0, 6, 9, 0] |
| 4        | [0, 8, 0, 6, 16, 0] |

**Notes:** Number ‘19’ represents the central depot in this case.

### Table F.8 Implementation result of the problem instance with HCPS applied ($I_i = 6, N_n = 10$)

#### Delivery Route

| MPL’s ID | Task execution sequence (Task’ ID) |
|----------|-----------------------------------|
| 0        | [19, 0, 11, 19, 15, 3, 19, 10, 7, 19, 14, 19, 16, 8, 19, 18, 5, 19] |
| 1        | [19, 17, 6, 19] |
| 2        | [19, 12, 19] |
| 3        | [19, 9, 19] |
| 4        | [19, 1, 2, 13, 19, 4, 19] |

**Arriving Time When Executing Task**
| MPL’s ID | Starting Time of the Corresponding Task |
|----------|----------------------------------------|
| 0        | [420, 630.0, 670.0, 686.7, 740, 758.4, 782.7, 590, 700.0, 720.9, 550, 567.9, 1061, 1082.8, 1104.5, 880, 1068.0, 1091.9] |
| 1        | [420, 600.0, 616.8, 635.8] |
| 2        | [420, 980.0, 996.3] |
| 3        | [420, 1070.0, 1092.3] |
| 4        | [420, 820.0, 1002.0, 1075.0, 1090.8, 910, 936.1] |

| Leaving Time When Task Completed |
|----------------------------------|
| MPL’s ID | Ending Time of the Corresponding Task |
|----------|----------------------------------------|
| 0        | [621.6, 645.4, 681.2, 738.8, 755.7, 778.9, 584.5, 600.8, 715.3, 548.8, 566.7, 1059.8, 1077.7, 1098.9, 873.1, 896.1, 1088.0, 0] |
| 1        | [593.1, 613.8, 630.2, 0] |
| 2        | [974.5, 990.8, 0] |
| 3        | [1064.4, 1086.7, 0] |
| 4        | [811.6, 834.2, 1015.0, 1085.3, 906.1, 932.2, 0] |

| Service Delays |
|----------------|
| MPL’s ID | Delays of the Corresponding Task (min) |
|----------|----------------------------------------|
| 0        | [0, 0.0, 0.0, 0, 0, 0.0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] |
| 1        | [0, 0.0, 0.0, 0] |
| 3        | [0.0, 0] |
| 3        | [0, 0.0, 0] |
| 4        | [0, 0.0, 0.0, 0, 0, 0, 0] |

| Real-time Demand Fulfilment |
|-----------------------------|
| MPL’s ID | Accumulated Number of Parcels that have been fulfilled (units) |
|----------|-------------------------------------------------------------|
| 0        | [0, 7, 11, 0, 11, 19, 0, 8, 17, 0, 9, 0, 6, 17, 0, 10, 13, 0] |
| 1        | [0, 9, 17, 0] |
| 2        | [0, 12, 0] |
| 3        | [0, 6, 0] |
Notes: Number ‘19’ represents the central depot in this case.