Optimization models and methods for tour planning in smart urban logistics

Shaohua Yu

Thèse de doctorat

NNT: 2020UPAST045

Optimization models and methods for tour planning in smart urban logistics

Thèse de doctorat de l'Université Paris-Saclay et de Northwestern Polytechnical University

École doctorale n° 573 interfaces: approches interdisciplinaires, fondements, applications et innovation (Interfaces)
Spécialité de doctorat: Informatique
Unité de recherche: Université Paris-Saclay, CentraleSupélec, Laboratoire Génie Industriel, 91190, Gif-sur-Yvette, France.
Référent: CentraleSupélec

Thèse présentée et soutenue en visioconférence totale, le 18 décembre 2020, par

Shaohua Yu

Composition du jury:

Dominique Barth Professeur, Université de Versailles	Président
Chengbin Chu Professeur, Université Paris-Est	Rapporteur & Examinateur
Guanghui Zhou Professeur, Xi'an Jiaotong University	Rapporteur & Examinateur
Caroline Prodhon Professeur associé, Université de Technologie Troyes	Examinatrice
Jakob Puchinger Professeur, Université Paris Saclay	Directeur
Shudong Sun Professeur, Northwestern Polytechnical University	Codirecteur

UNIVERSITÉ PARIS-SACLAY AND
NORTHWESTERN POLYTECHNICAL UNIVERSITY

DOCTORAL THESIS

Optimization models and methods for tour planning in smart urban logistics

---

Author:
Shaohua YU

Supervisor:
Prof. Jakob PUCHINGER
Prof. Shudong SUN

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

Operations Managment Team
Department of Industrial Engineering

January 14, 2021

Declaration

I, Shaohua YU, declare that this thesis titled, "Optimization models and methods for tour planning in smart urban logistics" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

Abstract

Urban distribution refers to the distribution activities of goods serving urban areas and suburbs. Recent technological advances in unmanned distribution fields, as well as new regulations limiting the use of combustion engine vehicles, will significantly change urban goods distribution. Besides, the new e-commerce business mode also brings new opportunities and challenges to urban goods distribution. Combining the above three new trends, this thesis designs and studies a van-based robot distribution system to provide better services for urban logistics distribution.

We first introduce a van-based robot urban delivery system. This problem typically applies on delivering parcels or other small commodities to pedestrianized areas such as campuses or residential clusters. To model the proposed vehicle routing problem, we introduce a mixed-integer programming and adopt column generation to get a better lower bound. We further propose construction heuristics and a hybrid metaheuristic approach with backtracking for solving larger instances. A sensitivity analysis for vehicle speed combinations reveals that increasing robot speeds has only very limited cost effects. We therefore recommend to keep robot speeds rather low because of a more pedestrian-friendly environment in practical implementations.

Then, we incorporate electric vehicles, en-route charging, and reverse charging technology into the van-based robot urban delivery system in logistics operations. The time during which electric vans are carrying robots can be used effectively to recharge the robots, thereby increasing distribution systems' efficiency. To model the proposed system, we present a mixed-integer programming. We note that the energy transfer from a van to its robot needs time and will cause the available travel distance of a van to decrease and that of a robot to increase. Focusing on the new time-distance-energy trade-off problem, which increases the difficulty of checking any given route's feasibility, we further propose a greedy route evaluation approach and a linear programming-based route evaluation method. An adaptive large neighborhood search algorithm is presented for solving larger instances. A sensitivity analysis for vehicle charging modes, maximum battery capacities, and charging rate shows that using en-route charging while appropriately increasing battery level and charging rate can have useful cost effects.

Finally, we incorporate hybrid pickup and delivery operations into the van-based

robot urban distribution system to adapt to the new business model of e-commerce enterprises. Five van/robot pickup and delivery modes are introduced, according to the van/robot's roles in the process of pickup/delivery and whether the van helps to transport its robot. To model the proposed problem, we introduce a mixed-integer programming in detail, especially for the new freight flow constraints and position constraints that emerged from the hybrid pickup and delivery operation in the van-based robot system. We further propose an adaptive large neighborhood search algorithm to solve larger instances and a capacity feasibility test approach for a single route. Then we assess the influence of parking node density on model output. A case study based on a realistic city scene is introduced. A sensitivity analysis on the robot's travel cost rate and maximum travel distances, as well as compared the van no-go area's effect on our model is conducted. Two classical models have been compared with our model, and results show the 2E-VRHPD model is competitive in appropriate scenarios settings.

Overall, through this thesis's research, we have established a framework that van-based robot distribution system. We have developed the methodologies to solve this kind of problem. Furthermore, we have put forward some suggestions for enterprises to use the van-based robot distribution system after our computational study experiments.

Keywords: Innovative last mile distribution; Mothership based robot services; Electric vehicle routing; En-route charging; Hybrid pickup and delivery

Résumé

La distribution urbaine fait référence aux activités de distribution des biens qui desservent les zones urbaines et les banlieues. Les progrès technologiques récents dans champs de distribution sans pilote et la limite d'utilisation des véhicules à moteur thermique par les nouvelles réglementations modifieront considérablement la distribution des biens urbains. Un nouveau modèle commercial de site e-commerce apporte également de nouvelles opportunités et défis à la distribution de biens urbains. Combinant les trois nouvelles tendances ci-dessus, cette thèse conçoit et étudie un système de distribution robotisé basé sur des fourgons pour fournir de meilleurs services pour la distribution de la logistique urbaine.

Nous introduisons d'abord un système de livraison urbaine de robot qui est basé sur une fourgonnette. Ce système pose des problèmes qui s'appliquent généralement à la livraison de colis ou d'autres petits produits aux zones piétonnières telles que les campus ou les groupes d'habitations. Pour modéliser le problème de routage de véhicule proposé, nous introduisons un programme d'entiers mixtes et adoptons la génération de colonnes pour obtenir une meilleure borne inférieure. Nous proposons en outre des constructions heuristiques et une approche métaheuristique hybride avec un retour en arrière pour résoudre des cas plus importants. Une analyse pour la sensibilité des combinaisons de vitesse de véhicule révèle que l'augmentation de la vitesse du robot n'a que des effets de coût très limités. Nous recommandons de maintenir la vitesse du robot à un niveau bas en raison d'un environnement plus convivial pour les piétons dans les implémentations pratiques.

Ensuite, nous intégrons les véhicules électriques, la recharge en route et la technologie de recharge inversée dans le système de livraison urbaine robotisé par la fourgonnette dans les opérations logistiques. Pendant le temps pour lequel les fourgons électriques transportent des robots peut être utilisé efficacement sur les recharges des robots, ainsi fait l'augmentation de l'efficacité des systèmes de distribution. Pour modéliser le système proposé, nous présentons un programme d'entiers mixtes. On constate que le transfert d'énergie d'une fourgonnette vers son robot nécessite du temps et entraînera une diminution de la distance de déplacement disponible d'un fourgon, par contre celui entraine une augmentation de déplacement d'un robot. Nous concentrons sur le nouveau problème de compromis temps-distance-énergie, qui augmente la difficulté de vérifier la faisabilité d'un itinéraire donné, nous proposons une approche

d'évaluation d'itinéraire gourmande et une méthode d'évaluation d'itinéraire basée sur la programmation linéaire. Un algorithme de recherche adaptatif de grand voisinage est présenté pour résoudre des instances plus importantes. Une analyse de sensibilité pour les modes de charge des véhicules, les capacités maximales de la batterie et le taux de charge montre que l'utilisation de la charge en route permet d'augmenter de manière appropriée le niveau de la batterie et le taux de charge, qui conduisent des effets bénéficiaires de coût.

Enfin, Pour s'adapter au nouveau modèle économique des entreprises de commerce électronique, nous intégrons des opérations de ramassage et de livraison hybrides dans le système de distribution urbaine de robots basés sur des fourgons. Selon les rôles du fourgon/robot dans le processus de ramassage/livraison et si le fourgon aide à transporter son robot. Cinq modes de ramassage et de livraison de fourgon/robot sont introduits. Pour modéliser le problème proposé, nous avons détaillé un programme d'entiers mixtes, en particulier pour les nouvelles contraintes de flux de marchandises et les contraintes de position qui ont émergé de l'opération hybride de ramassage et de livraison dans le système de robot basé sur fourgonnette. Nous proposons un algorithme de recherche adaptatif de grand voisinage pour résoudre des instances plus importantes et une approche de test de faisabilité de capacité pour un seul itinéraire, puis nous évaluons l'influence de la densité des nœuds de stationnement sur la sortie du modèle. Une étude basée sur une scène de ville réaliste est introduite. Une analyse de sensibilité sur le taux de coût de voyage du robot et les distances de déplacement maximales, ainsi que la comparaison de l'effet de la zone de no-go des fourgons sur notre modèle est réalisée. Deux modèles classiques ont été comparés à notre modèle, et les résultats montrent que le modèle 2E-VRHPD est compétitif dans des scénarios appropriés.

Dans l'ensemble, grâce à cette recherche de thèse, nous avons établi un cadre de système de distribution de robot basé sur fourgonnette. Nous avons développé les méthodologies pour résoudre ce genre de problème. En outre, nous avons proposé aux entreprises d'utiliser le système de distribution de robots basés sur des fourgons après nos expériences d'études informatiques.

Mots-clés: Distribution innovante du dernier kilomètre; Services de robots basés sur le vaisseau mère; Routage de véhicules électriques; Recharge en route; Ramassage et livraison hybrides

Acknowledgements

I am delighted to have this chance to express my heartful gratitude to those who helped and supported me during my four-years' PhD study.

This is a joint PhD training program between CentraleSupélec, Université Paris-Saclay (France), and the Northwestern Polytechnical University (NWPU) (China). Thanks to the Laboratoire Génie Industriel (LGI) at CentraleSupélec and the School of Mechanical Engineering at the NWPU to provide me with this excellent training program and outstanding working conditions. I would also like to thank the China Scholarship Council for financial support during my academic study in France.

I am particularly grateful to my two supervisors, Prof. Jakob Puchinger, at CentraleSupélec and Prof. Shudong Sun, at the NWPU. They have taught me many professional knowledge and scientific research experiences, which are helpful to my research and meaningful for my future career.

My deepest appreciation goes to all the jury members: They are Professor Chengbin Chu, Professor Guanghui Zhou, Professor Dominique Barth, and Professor Caroline Prodhon. Thank them for participating in the thesis defense and giving valuable opinions.

I would like to express my thankfulness to other teachers who helped me during my PhD study: Profs Bernard Yannou, Vincent Mousseau, Yiping Fang, Zhiguo Zeng, Flore Vallet, Marija Jankovic, Delphine Martin, Carole Stoll, Matthieu Tournadre, Tiancheng Li, Shubin Si, Ziqiang Cai etc.

I would also like to thank my friends and colleagues in LGI and the School of Mechanical Engineering at the NWPU who help me. Particularly, Abood Mourad, Ouail Al Maghraoui, Réza Vosooghi, Daogui Tang, Hongping Wang, Jinduo Xing, Gustavo Santamaria-Acevedo, Yue Su, Zigao Wu, Shichang Xiao, Xuedong Wang, Kai An, Yaqiong Liu, Jinlun Dai, Shengbo Chang. I would like to thank them for their support and help whenever I needed help. It has been an enjoyment to work with such wonderful persons.

Special thanks go to my family for their unlimited love and support. Their love and encouragement have always been my driving force to carry on.

Contents

List of Figures

List of Tables

List of Abbreviations

VRP	Vehicle Routing Problem
2E-VRP	Two-Echelon Vehicle Routing Problem
2E-LRP	Two-Echelon Location Routing Problem
TTRP	Truck and Trailer Routing Problem
OVRP	Open Vehicle Routing Problem
EVRP	Electric Vehicle Routing Problem
GPDP	General Pickup and Delivery Routing Problem
E2EVRP	Electric Two-Echelon Routing Problem
VRPPD	Pickup and Delivery Problem
VRPCB	Vehicle Routing Problem with Clustered Backhauls
VRPMB	Vehicle Routing Problem with Mixed Linehauls and Backhauls
VRPSPD	Vehicle Routing Problem with Simultaneous Pick-ups and Deliveries
VRPDDP	Vehicle Routing Problem with Divisible Deliveries and Pickups
PDPTW	Pickup and Delivery Problem with Time Window
VRP-D	Vehicle Routing Problem with Drone
TSP-D	Travel Sale-man Problem with Drone
2E-VRRP	(Two-Echelon) Van-Based Robot Routing Problem
2E-VREC	Electric Van-Based Robot Deliveries with En-Route Charging
2E-VRHPD	Van-Based Robot Hybrid Pickup and Delivery Routing Problem

Chapter 1

Introduction

In this introduction, we present the motivation of this work, and outline the structure of this thesis.

1.1 Motivation

Over 50% of the world's population is now living in cities. In Europe, 75% of European citizens lived in urban areas (European Commission, 2014). This percentage is expected to increase to over 80% by 2050. In some countries, the urbanization rate will rise to over 90%. Urban mobility accounts for 40% of all CO2 emissions from road transport and up to 70% of other pollutants from transport (European Commission, 2015). Optimizing urban transport has become an essential part of transport management as it will affect the quality of life of increasingly more urban citizens.

Traditional urban logistics is a critical component of urban transport, but it is criticized for causing traffic jams and urban pollution today. The rapid development of e-commerce is driving a steady yet steep rise in parcel pickup/delivery demands. Global B2C e-commerce turnover forecasted to reach over $2 trillion in 2019, and continues growing steadily at around 11% (SAP, 2019). The rise of e-commerce has led to the explosive growth of urban logistics. OECD predicts the urban freight road activity will double from the year 2015 to the year 2050 (OECD, 2017). In parallel, rising labor costs and restrictions on delivery-staff workhours are challenging logistic companies' efforts to provide customers with affordable, efficient and round-the-clock courier services. Cities thus need to identify new delivery strategies to increase the quality of life of their citizens while keeping traffic smooth and environmental pollution under control.

The developments of the communication technologies, sensor capabilities, the Internet of Things (IoT), and artificial intelligence are making drones/robots smarter. By leveraging drones and robots, many companies are developing new logistic systems that can change the competition landscape (Tang and Veelenturf, 2019). We observed interest in freight distribution with drones/robots has surged in the past five

years, producing a number of studies and tests on new automated logistics and distribution tools. For example, UPS tested home delivery via drones in Florida (Lithia, 2017). JD.com launched city delivery by robots in several Chinese universities and districts in Beijing (Li, 2017; Gu, 2018). French start-up TwinswHeel is testing an unmanned delivery robot with the ability to climb a certain sidewalk height to complete last-mile delivery (Chevallier, 2017). Starship Technologies (Andrew, 2019) have implemented autonomous distribution robots on campuses and in some residential areas. Kiwi Coldewey, 2019 had used food delivery bots to deliver foods. Note that most small fully-automated robots are electrically driven, helping reduce local emissions in cities.

Drones are usually not subject to crowded traffic infrastructure and can carry limited goods serving limited ranges with fast speed (Bakach et al., 2020). However, in the real world, the application of drones is restricted by a lot of factors. (i) Could lead to more severe accidents. For example, when a fast-flying drone stalls, it will cause serious accidents in a city. (ii) Making more noise than cars (Christian and Cabell, 2017). (iii) Drones are also often not allowed to visit no-fly zones, such as airports.

Robot delivery can overcome the above defects of drone delivery. Robots make less noise than cars. Compared to the dangers of distribution with the drone to a slowly moving robot, accidents caused by stalled robots are controllable. And robots can always deliver most of the areas. Hence, the delivery robot seems more pragmatic and safer than drones. This thesis focuses on robot delivery in urban areas.

Also, public acceptance of the delivery robot is very high and positive. United States Postal Service surveyed public opinion on delivery robots (USPS, 2018), and 70% of interviewees would consider receiving deliveries from robots. Figure 1.1 is the public opinion about delivery robots (USPS, 2018). Figure 1.1 shows that more than 50% of interviewees believe delivery robots could offer a more flexible delivery experience, and nearly 30% of interviewees would be willing to pay slightly more to receive those benefits.

Most robots are electrically powered in real-world applications because electric robots have no local pollutant emissions (Bektaş et al., 2019). However, in the foreseeable future, the robots' applications are limited by their limited travel distances and speed for technical and safety reasons (Daimler, 2017).

Research question 1: How to overcome the shortages of robot delivery?

Research question 2: How to make the robot charging more efficiently?

With the continuous development of e-commerce, the business modes of e-commerce companies have begun to diversify, and their distribution modes have also diversified. The hybrid distribution business mode requires logistics companies to perform one-to-one or one-to-many-to-one pickup and/or delivery services simultaneously, which

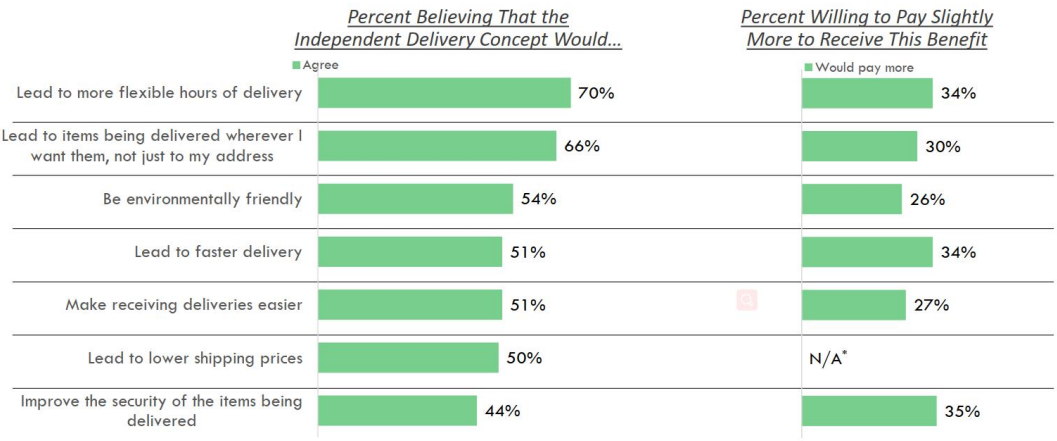

FIGURE 1.1: Public opinion about delivery robots

poses new challenges to the design and optimization of the distribution system.

Research question 3: How to integrate multiple modes of pickup and delivery operations to make the logistics distribution system more efficient?

We note the following technology development trend and business model distribution trend of enterprises may help a lot in solving the three research questions. (i) UPS tested home delivery via truck and robot in Florida as a move towards a more automated delivery process (Lithia, 2017) (ii) Automotive companies have begun to study mobile charging vans. NIO, an electric car company in China, is courting Tesla owners with mobile charging stations inside electric vans (Fred, 2018). (iii) JD.com, an e-commerce company in China, should deliver and pick up self-products and help individuals or individual stores (bricks and mortars or local supermarkets) deliver goods in real business operations. For the latter, a request consists of picking up goods from individuals or individual stores, and transporting them to corresponding customers needs to be completed.

This thesis will integrate these new technological and business concepts for reorganizing urban distribution. The new urban distribution should provide a more efficient choice of distribution model that can adopt new technologies and services and, on the other hand, limit emissions and energy consumption. Specifically speaking, this thesis mainly aims at these three strategic and operational levels:

- include new means of distribution (Van-based robot urban delivery)
- include new means of charging strategy (Van-based robot urban delivery with en-route charging)
- include new means of the business model (Van-based robot routing with hybrid pickup and delivery)

1.2 Structure of the thesis

The rest of this thesis is organized as follows.

Chapter 2 first introduces the state of the art of the van-based robot routing problems and the similarities and differences between them. Then Chapter 2 reviews the vehicle routing problems (VRP) related to the van-based robot routing problem: two-echelon vehicle routing problem, two-echelon location routing problem, truck and trailer routing problem, open vehicle routing problem, electric two-echelon routing problem, and pickup and delivery problem.

Chapter 3 - Chapter 5 study the two-echelon van-based robot routing problem (2E-VRRP), electric two-echelon van-based robot deliveries with en-route charging (2E-VREC), and two-echelon van-based robot hybrid pickups and deliveries routing problem (2E-VRHPD) respectively.

Chapter 3 investigates an innovative two-echelon van-based multiple robots urban delivery problem that contains the time window constraint, capacity constraint, and maximum travel time constraint. This innovative distribution model can eliminate a lot of real estates and human resources compared to the traditional two-echelon delivery model, which provides a new choice for logistics enterprises. We found that simply improving the robot's speed leads to only limited cost reduction during our experiments. Therefore we recommend keeping robot speeds rather low because of a more pedestrian-friendly environment in practical implementations.

Chapter 4 focuses on incorporating en-route charging and reserve charging technologies into van-based robot delivery problem. We use the time during which vans are carrying robots effectively to recharge the robots, thereby increasing distribution systems' efficiency. Because of the difficulty in processing the trade-off between energy, distance and time for checking the feasibility of a given 2E-VREC route, we further propose a greedy route evaluation approach and an LP-based route evaluation method. We recommend that logistics companies use en-route charging technology if the new technology's fixed cost is controllable, based on our computational study.

Chapter 5 considers incorporating multiple modes of pickup and delivery operations into the van-based robot logistic distribution system to adapt to the new business modes and needs of logistics enterprises. We detailed introduce five pickup and delivery cases in the 2E-VRHPD problems and model the new freight flow and position constraints, which never occur so far. We have a case study based on realistic scenarios. We conduct a sensitivity analysis on the robot's travel cost rate and maximum travel distances, as well as compare the van no-go area's effect on the 2E-VRHPD model. Besides, we compare the 2E-VRHPD model with two classical models. We recommend

that companies consider the 2E-VRHPD model for distribution in appropriate scenarios to improve the system's efficiency.

Chapter 6 concludes the thesis and points out potential future works.

The chapters of the thesis are based on the following papers:

Chapter 3 Yu, S., Puchinger, J., & Sun, S. (2020). Two-echelon urban deliveries using autonomous vehicles. Transportation Research Part E-Logistics and Transportation Review, 141, 102018.

Chapter 4 Yu, S., Puchinger, J., & Sun, S. (2020). Electric van-based robot deliveries with en-route charging. Transportation Research Part C-Emerging Technologies, Under Review.

Chapter 5 Yu, S., Puchinger, J., & Sun, S. (2020). Van-based robots hybrid pickups and deliveries routing problem. European Journal of Operational Research, prepare to submit.

In addition, the different elements of the research conducted in this thesis are presented in the following conferences:

Puchinger, J., & Yu, S. (2018). A two-echelon vehicle routing problem with unmanned ground vehicles for city logistics. EURO 2018. 29th European Conference On Operational Research.

Yu, S., Puchinger, J., & Sun, S. (2020). Urban deliveries using robots in a two-echeleon system. 21ème Congrès Annuel de La Société Française de Recherche Opérationnelle et d'Aide à La Décision (ROADEF).

Chapter 2

Literature Review

In this chapter, we first give a brief introduction of the van-based robot routing problem. We then introduce some types of vehicle routing problems relative to the van-based robot routing problems this thesis considered.

2.1 A brief introduction of van-based robot routing problem

Van-based robot routing problem (2E-VRRP), see Figure 2.1, has drawn much attention (Otto et al., 2018). In the van-based robot routing problem, the van can transport robots, and along the route, it can drop off and pick up the robot at different positions. The advantage of van-based robot deliveries is that goods can be delivered in parallel by the van and its robot, thereby increasing the efficiency of distribution systems compared to VRP-based systems. Murray and Chu, 2015 first presented the 2E-VRRP, and farther-reaching problems have since been studied. We give a van-based robot routing similarities and differences comparison table (Table 2.1) with recent related research articles.

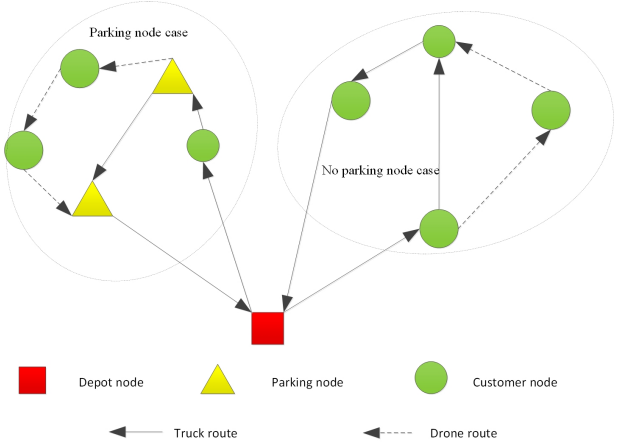

FIGURE 2.1: Van-based robot routing problem

The comparison focuses on how many vans and robots are considered and the objective of the problem, the number of customers a robot can visit in the course of one trip (VMC), whether there are time window constraints for customers (CTW), whether both pickup and delivery are allowed in the model (PD), whether the van can serve customers directly (VD), whether charging is possible in the model (Charging), and the contribution of the work (Contribution).

TABLE 2.1: Similarities and differences between van-based robot routing problems

Reference	Vans	Robots	Objective	VMC	CTW	PD	VD	Charging	Contribution
Murray and Chu, 2015	1	1	time	1	no	no	yes	no	MILP, heuristic
Poikonen et al., 2017	n	m	time	1	no	no	yes	no	Theoretical insights
Wang et al., 2017	n	m	time	1	no	no	yes	no	Theoretical insights
Pugliese and Guerriero, 2017	n	m	cost	1	yes	no	yes	no	MILP
Luo et al., 2017	n	1	time	m	no	no	no	no	MILP, heuristic
Carlsson and Song, 2017	1	1	time	1	no	no	no	no	Heuristic
Agatz et al., 2018	1	1	time	1	no	no	yes	no	IP, heuristic
Bouman et al., 2018	1	1	cost	1	no	no	yes	no	DP
Schermer et al., 2019a	n	m	time	1	no	no	yes	no	MILP, VNS, matheuristic
Schermer et al., 2019b	n	m	time	1	no	no	yes	no	MILP, VNS, matheuristic
Karak and Abdelghany, 2019	1	m	cost	m	no	yes	no	no	MIP, heuristic
Sacramento et al., 2019	n	1	cost	1	no	no	yes	no	MILP, ALNS
Wang and Sheu, 2019	n	m	cost	m	no	no	yes	no	MIP, branch-and-price
Poikonen et al., 2019	1	1	time	1	no	no	yes	no	Branch-and-bound, heuristic
Murray and Raj, 2020	1	m	time	1	no	no	yes	no	MILP, heuristic
Poikonen and Golden, 2020b	1	1	time	1	no	no	yes	no	Branch-and-bound, heuristic
Poikonen and Golden, 2020a	1	m	time	m	no	no	no	no	ILP, heuristic
Kitjacharoenchai et al., 2020	n	m	time	m	yes	no	yes	no	MIP, heuristic, LNS
Moshref-Javadi et al., 2020a	1	m	waiting time	1	no	no	yes	no	MIP, heuristic
Moshref-Javadi et al., 2020b	1	m	waiting time	1	no	no	yes	no	MIP, hybrid TS-SA
Dayarian et al., 2020	1	1	maximize orders	m	no	no	yes	no	Heuristics
Gonzalez-R et al., 2020	1	1	time	m	no	no	yes	no	MIP, Iterated greedy heuristic
Salama and Srinivas, 2020	1	m	time,cost	1	no	no	yes	no	MIP, Heuristic
Chapter 3	n	m	cost	m	yes	no	no	no	MILP, hybrid metaheuristic
Chapter 4	n	m	cost	m	yes	no	yes	yes	MILP, ALNS
Chapter 5	n	m	cost	m	yes	yes	yes	no	MIP, ALNS

We introduce these papers in detail in terms of solution methods from exact methods, matheuristics, and heuristics (include metaheuristics).

2.1.1 2E-VRRP with exact methods

Wang et al., 2017 studied the vehicle routing problem with drones (TSP-D). They posed several situations to explore the maximum savings from cooperatively using van and drones rather than van only in delivering freights and then derive several worst-case results. Poikonen et al., 2017 extended models of (Wang et al., 2017) further.

Bouman et al., 2018 presented dynamic-programming-based approaches for the TSP-D. Their approaches can solve larger problems than the mathematical programming approaches that have been presented so far in the experimental comparison study. They found restrictions on the number of locations the van can visit while the drone is away can significantly reduce the solution times. Besides, the limits have relatively little impact on the overall solution quality.

Poikonen and Golden, 2020b introduced a new type of TSP-D problem, called mothership (carrier) and drone routing problem. The new problem allows the van to move in continuous space, allowing second-order cone programs to be used throughout as subroutines in solution methods. They presented a branch and bound method to solve small instances and presented several heuristics to address large cases.

Wang and Sheu, 2019 proposed an arc-based and path-based integer programming model for vehicle routing problems with drones (VRP-D) and docking node. They further develop a branch-and-price algorithm to solve this kind of problem, and the proposed algorithm performs well.

2.1.2 2E-VRRP with matheuristics

Pugliese and Guerriero, 2017 extended the VRP-D by considering customer time windows to perform last-mile deliveries operations. They presented a MIP model and investigated the advantages and disadvantages of using drones. Computational study results show the higher the van transportation cost, the higher the number of deliveries performed by the drones.

Agatz et al., 2018 modeled TSP-D as an integer program and develop several fast route-first, cluster-second heuristics based on local search and dynamic programming. The worst-case approximation ratios for the heuristics are proved. Computational study shows the algorithm's high efficiency and shows that substantial savings are possible with this concept compared to van-only delivery.

Poikonen et al., 2019 proposed four heuristic methods based on the branch-and-bound algorithm for the TSP-D variant (the truck may remain stationary while the drone makes a delivery) of Agatz et al., 2018.

2.1.3 2E-VRRP with heuristics

Luo et al., 2017 considered a two-echelon cooperated routing problem for a van and its drone. There are specific parking lots for the vehicle to drop off and pick up the drone. They developed a 0-1 integer programming model and proposed two heuristics to solve the model.

Carlsson and Song, 2017 studied a TSP-D problem with the drones that can be launched from several points (not only customer locations). Heuristic methods were presented to analyze the benefit of using van and drone delivery systems and describe how much improvement can be realized by involving drones to deliver packages. They have concluded that the improvement in efficiency due to augmenting a delivery truck with a drone is related to the square root of the ratio of the truck's speeds and the drone.

Schermer et al., 2019a considered the VRP-D problem with the en-route operations by allowing drones to be dropped off and picked up at some discrete locations on each arc. They used MILP to model the new problem and propose a heuristic that combines the variable neighborhood search and tabu search. The computational study showed that allowing en-route operations can lead to more efficient solutions due to the time-objective's potential reduction. Schermer et al., 2019b also extended the VRP-D problem by allowing the execution of cyclic operations for drones. The authors further introduced several additional valid inequalities and discussed the benefits of using the new MILP formulation. In addition, a matheuristic that decomposes the problem into an allocation subproblem and a sequencing subproblem is proposed. Computational study shows the benefits of using drones to minimize the makespan and conclude that their heuristic is effective and efficient.

Karak and Abdelghany, 2019 first studied the hybrid vehicle-drone routing problem for pickup and delivery services, in which the node has goods pickup and delivery demands simultaneously. The MIP formulation was presented. A novel solution methodology is developed which extends the classic Clarke and Wright algorithm. Sacramento et al., 2019 considered the VRP-D problem with time limit constraints, and designed an adaptive large neighborhood search metaheuristic for the problem.

Murray and Raj, 2020 extended their research in 2015 by considering parcel delivery with multiple drones. They presented a heuristic solution approach that leverages a subset of subproblems is proposed. They gave some critical insights into the truck-drone system design according to their computational study. (1) Although it is generally faster to reach a customer with a drone rather than a truck, it is rarely beneficial to serve all drone-eligible customers via drone. (2) UAVs with high-speed and long-range offer more incredible benefits in larger geographic regions, where customers are distributed over a larger area. (3) Adding more UAVs to an existing fleet tends to have diminishing marginal makespan improvements, with UAVs offering a more significant benefit in instances involving many customers. (4) Problem instances involving densely-distributed customers tend to benefit the most. (5) Automation within both the truck and the depot result in time savings, with depot automation providing the greatest savings.

Poikonen and Golden, 2020a studied the multi-visit drone routing problem. Poikonen and Golden, 2020a allows the energy drain function to be any non-decreasing function of weight for each location pair and decouples the set of launch/landing vertices from the set of customer locations. Computational study indicates that objective functions are highly sensitive to drone speed. The number of drones was also very impactful on the objective value.

Moshref-Javadi et al., 2020a and Moshref-Javadi et al., 2020b studied a truck and

drones delivery model for last-mile delivery to minimizes the waiting time of customers in the system. Moshref-Javadi et al., 2020a presented a mathematical formulation and a heuristic solution approach for the optimal planning of delivery routes in a multi-modal system combining truck and drone operations. Moshref-Javadi et al., 2020b extended the model of Moshref-Javadi et al., 2020a by allowing launch drones multiple times from each stop location to serve customers, like Schermer et al., 2019b did. They developed a hybrid tabu search-simulated annealing algorithm to address the problem. The computational study showed that the new delivery system could reduce the customer waiting time for a wide range of model parameters compared to the classical delivery models.

Dayarian et al., 2020 introduced a novel way to exploit drones in same-day home delivery settings: drone resupply. They considered a home delivery system in which drones regularly resupply delivery trucks. They developed different algorithms and compared their performance. Computational study shows the potential benefits of drone resupply.

2.2 Related vehicle routing problem

Here we introduced some vehicle routing problems related to the van-based robot routing problems this paper studied. Section 2.2.1 - 2.2.4 are the problems related to the proposed basic 2E-VRRP model. Section 2.2.5 introduces the electric two-echelon routing problem, which is related to the 2E-VREC problem. Section 2.2.6 introduces the pickup and delivery problem, which is related to the 2E-VRHPD problem.

2.2.1 Two-echelon vehicle routing problem (2E-VRP)

The two-echelon vehicle routing problem (2E-VRP), which is shown in Figure 2.2, is a well-known variant of the classic VRP. It involves determining a set of optimal routes for a two-level freight distribution system, where goods are delivered from a depot to a subset of intermediary satellites in the first echelon, and from the satellites to customers in the second echelon. For cost effectiveness reasons, delivery tasks in the first echelon are usually accomplished by large identical trucks while delivery tasks in the second echelon are usually accomplished by small identical vehicles (Baldacci et al., 2013; Dellaert et al., 2018; Liu et al., 2018). In 2E-VRP, the intermediary satellite can be seen as a transfer station. The freight is the primary connection between truck and small identical vehicle. In the 2E-VRRP model, the intermediary satellite is just like a rendezvous node and does not possess any goods storage function. Furthermore, the connection between van and robot is closer than that of truck and small vehicle in the

2E-VRP problem. The van not only carries the robots but also sends them out and picks them up at same/different rendezvous nodes.

2.2.2 Two-echelon location routing problem (2E-LRP)

The two-echelon location routing problem (2E-LRP), which is shown in Figure 2.3, is similar to the 2E-VRP. The main difference is that the 2E-LRP incorporates the location decision problem into the VRP while the 2E-VRP does not (Crainic et al., 2011; Cuda et al., 2015; Bala et al., 2017; Wang et al., 2018). The 2E-VRRP model is like the 2E-LRP model in that we need to decide which vans/robots need to visit which rendezvous nodes, but the models differ on the earliest time that a vehicle is allowed to leave the pick-up rendezvous node. In 2E-LRP, the earliest leaving time of a vehicle is equal to the time when the vehicle arrived at the satellites, whereas in the 2E-VRRP model it depends on the time when its robots arrive at the pick-up node.

2.2.3 Truck and trailer routing problem (TTRP)

The truck and trailer routing problem (TTRP), as shown in Figure 2.4, describes a fleet of truck and trailer combinations with known capacity serving a set of customers with predetermined demands and locations. In this problem, a vehicle may be a truck pulling a trailer, called a complete vehicle, or a single truck, called a pure truck. Some customers must be served by a truck while others can be served either by a truck or a complete vehicle (Chao, 2002; Li et al., 2016; Parragh and Cordeau, 2017; Rothenbächer et al., 2018). In the 2E-VRRP model, both the van and the robot can move by themselves, whereas the trailer in TTPR model stays stationary while the truck is not pulling.

2.2.4 Open vehicle routing problem (OVRP)

The open vehicle routing problem (OVRP), in which a vehicle does not necessarily return to the start node after servicing the last customer on its route, performs as a Hamiltonian path (or Hamiltonian circuit if the vehicle chooses to come back to the start node) (Li et al., 2007; Repoussis et al., 2007; Brandão, 2018). The OVRP has attracted less attention than the VRP, largely due to its limited application environment: the main application background cited in the academic study is a company using hired vehicles for goods delivery, where the vehicle does not need to return to the depot after finishing serving the last customer. Note that in the second-level 2E-VRRP delivery scenario, the robot route could be regarded as a Hamiltonian path since the van can drop off and pick up a robot at different rendezvous nodes. Consequently, the OVRP

can be expected to receive more attention as automated distribution such as automated vehicle and/or drone delivery gains currency. The OVRP is shown in Figure 2.5.

2.2.5 Electric two-echelon routing problem (E2EVRP)

The electric two-echelon vehicle routing problem (E2EVRP), as shown in Figure 2.6, have begun to be studied by researches for application to city scenarios in recent years. And the E2EVRP is related to the 2E-VREC problem we studied in Chapter 4.

Breunig et al., 2019 extended the two-echelon vehicle routing problem where electric vehicles are used on the second echelon with full charging technology. They proposed a large neighborhood search and an exact mathematical programming algorithm, which uses decomposition techniques to enumerate promising first-level solutions in conjunction with bounding functions and route enumeration for the second-level routes.

Jie et al., 2019 and Wang et al., 2019 studied the two-echelon vehicle routing problem with battery swap technology. Jie et al., 2019 considered a two-echelon capacitated electric vehicle routing problem with battery swapping stations. In their problem setting, the first and second echelon vehicles could all swap their batteries in the battery swapping stations. They proposed an integer programming formulation and a hybrid algorithm that combines a column generation and an adaptive large neighborhood search to solve the problem. Wang et al., 2019 addressed a two-echelon vehicle routing problem involving electric vehicles considering time windows and battery swapping stations. They formulated the mathematical model and minimized the total of shipping cost, handling cost, battery swapping cost, fixed cost of vehicles, and penalty cost due to tardiness.

In the E2EVRP model, the vehicle (robot) can only be recharged at the charging station. However, the robot can also be recharged by other vehicles in the 2E-VREC model.

2.2.6 Pickup and delivery problem (VRPPD)

The vehicle routing problem with pickup-and-delivery (VRPPD), depicted in Figure 2.7, is an essential family of routing problems in which freight or passengers have to be transported between different origins and destinations (Paolo and Vigo, 2014). The pickup and delivery problem is related to the 2E-VRHPD problem we studied in Chapter 5, not only about picking up and delivering freights, but also robots.

The VRPPD can basically be subdivided into four subclasses (Polat, 2017). In the VRP with clustered backhauls (VRPCB), mixed linehauls and backhauls (VRPMB), and simultaneous pick-ups and deliveries (VRPSPD), the vehicles only visit each customer once for pickups and deliveries, i.e. loads cannot be split. In the VRP with divisible

deliveries and pickups (VRPDDP), the vehicles can arrive at each customer twice. Nagy et al., 2013 formulated the VRPDDP as a mixed-integer linear programming problem and presented an exact and heuristic algorithm to implement the problem. Polat, 2017 presented an efficient parallel approach based on variable neighborhood search to solve the VRPDDP that significantly improved the best solutions available in the literature. The classification of pickup and delivery problems can also be seen in Battarra et al., 2014, and Ko et al., 2020.

Time window constraints are usually needed in urban pickup and delivery. Dumas et al., 1991 were the first to use column generation for solving pickup and delivery problem with time window (PDPTW). They propose a branch-and-bound method that can handle problems with up to 55 requests. Ropke and Pisinger, 2006 presented an adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows.

In 2E-VRHPD model, van and robot can be seen as heterogeneous vehicles. Qu and Bard, 2013 studied a heterogeneous pickup and delivery problem in which each vehicle's capacity can be modified by reconfiguring its interior to satisfy different types of customer demands. The number of participants and support equipment that a van can accommodate depends on how it is configured. They developed a two-phase heuristic that uses ideas from greedy randomized adaptive search procedures with multiple starts to solve this problem. Avci and Topaloglu, 2016 studied a heterogeneous vehicle routing problem with simultaneous pickup and delivery problems. They developed a hybrid local search algorithm in which a non-monotone threshold adjusting strategy is integrated with tabu search. They indicated that the developed approach could produce efficient and effective solutions. Sun et al., 2019 presented the formulation and exact solution for heterogeneous vehicle pickup and delivery problems, intending to minimize carbon emissions of pickups and deliveries by a fleet of heterogeneous vehicles. Their exact algorithm is based on a set partitioning model and the key characteristics of its optimal solution, which can rapidly find the largest-scale instance's optimal solution. Drexl, 2020 studied a one-to-one pickup-and-delivery problem with time windows and trailers. The discussed extension consists in consideration of a heterogeneous vehicle fleet comprising lorries with detachable trailers. Trailers are advantageous as they increase the overall vehicle capacity. However, some locations may be accessible only by lorries.

In the 2E-VRHPD model, the pickup and delivery operation is also reflected in the pickup and delivery robots. The van picks up and delivers robots, while the van in the VRPPD model picks up and delivers freights. Besides, the van must first launch the robot before recovering it. Whereas the vehicle has no priority for delivery and pickup in the VPRPD, they could be delivered first and pickup second; mixed pickups and

deliveries; simultaneous pickups and deliveries. Besides, in the 2E-VRHPD model, there is a minimum time interval between releasing and recovering the same robot. This interval equals the shortest completion time for the robot to finish delivery to all its customers from origins to destinations. Note that the pickup and delivery operation (target is the robot) also exist in the 2E-VRRP and 2E-VREC model.

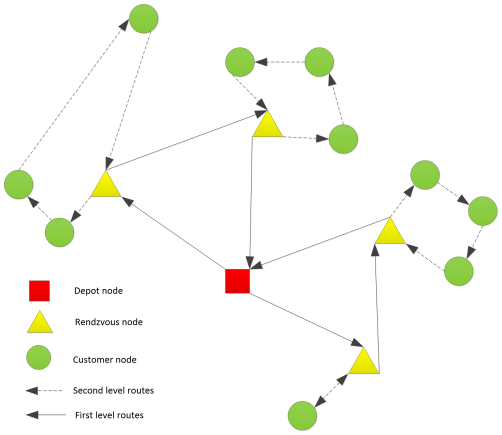

FIGURE 2.2: 2E-VRP

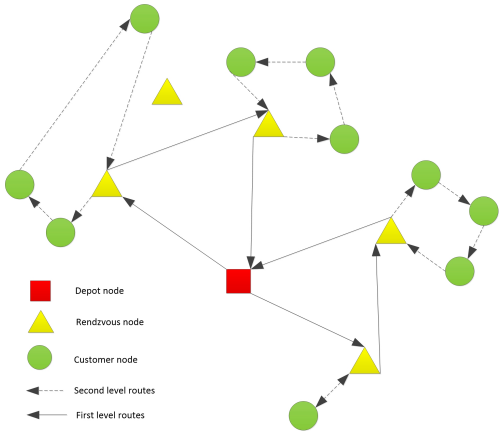

FIGURE 2.3: 2E-LRP

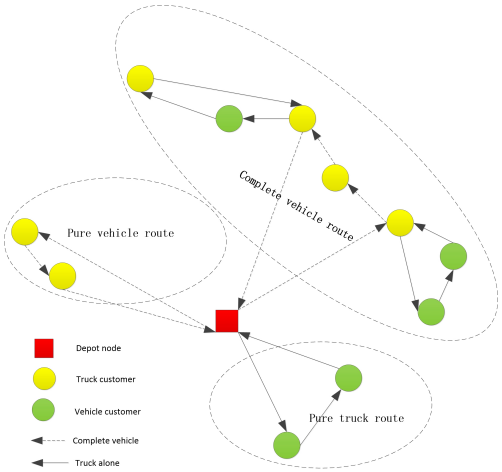

FIGURE 2.4: TTRP

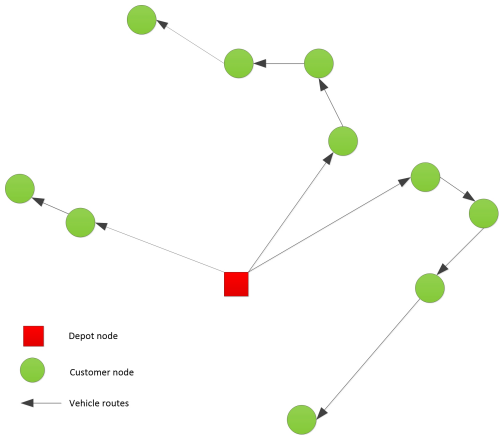

FIGURE 2.5: OVRP

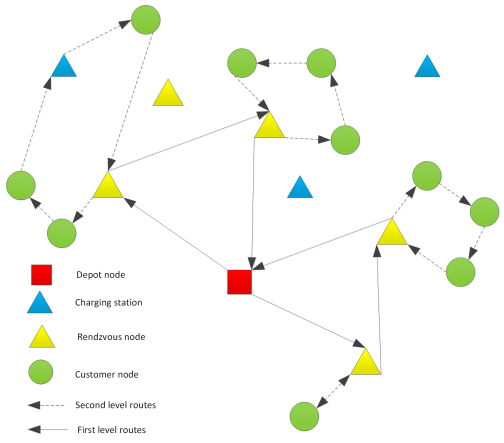

FIGURE 2.6: E2EVRP

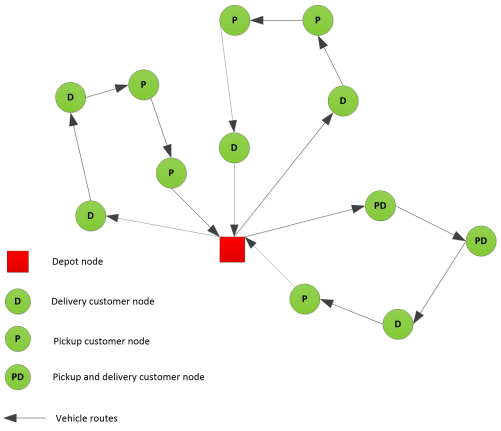

FIGURE 2.7: VRPPD

Chapter 3

Van-based robot urban deliveries problem

Abstract

We introduce a two-echelon urban delivery problem with second-level robots. This problem typically applies for delivering parcels or other small commodities to pedestrianized areas such as campuses or residential clusters. To model the proposed vehicle routing problem, we introduce a mixed-integer program. We further propose construction heuristics and a hybrid metaheuristic approach with backtracking for solving larger instances. A sensitivity analysis for vehicle (van/robot) speed combinations reveals that increasing robot speeds has only very limited effects on cost. We therefore recommend to keep robot speeds rather low because of a more pedestrian friendly environment in practical implementations.

3.1 Background

Autonomous delivery services can be realized in certain areas as of today. Researchers have begun to study autonomous vehicle delivery problems under restricted conditions, for example Scherr et al., 2019 designed an urban service network with mixed autonomous fleets.

However, in the foreseeable future, the applications of the robot are limited by its travel distances and speed for technical and safety reasons. Hence, one innovative city logistics delivery concept was presented by combining traditional vehicles and robots. Boysen et al., 2018 considered using vans to transport and drop off small robots. They presented a decentralized robot depot within the city center, only used for storing delivery robots. Mercedes-Benz also presented the idea and works closely together with Starship Technologies to develop the "mothership approach" (van carries robots) in their Future Transportation unit. They believe that the carrier system can avoid the drawback of robot-only delivery (Daimler, 2017).

Based on the mothership concept, we propose a 2E-VRRP model where multiple vans cooperate with their associated robots. The van carries the robots, sending out and picking up the robot in rendezvous nodes while the robot manages customer service. The 2E-VRRP model has the following advantages:

(1) The 2E-VRRP model can avoid some drawbacks of robot-only delivery. For example, the robot will only have a very limited range. Logically, the range is limited by battery capacity, besides, the robots move at walking speed for safety reasons, so that their application for long distances is not efficient (Daimler, 2017). If we adopt the 2E-VRRP model, the van can carry robots to implement long distances and high-speed transportation. (2) The 2E-VRRP model can greatly reduce the cost of the enterprise in operational aspects when compared to the traditional two echelon delivery model. Since the 2E-VRRP model does not need suitable real city estate, also without a lot of manpower cost. Note that satellites (intermediate warehouse) and labor costs are two big expenses of logistics enterprises. (3) The number of rendezvous nodes is always larger than that of satellites, which means vans can be more flexible in choosing transfer nodes. Moreover, the second level route in the 2E-VRRP model is an open route, the robot can choose being picked up from a different node than the one they have been dropped off, which often brings about a lower cost at the second level of delivery. (4) Compared to the truck-based drone model, we believe the 2E-VRRP model is safer in urban delivery. For example, when a fast-flying drone stalls, it will cause serious accidents in a city. However, to a slowly moving robot, accidents caused by stalled robots are controllable.

The contributions of this chapter are as follows. We consider a new two-echelon urban delivery concept with time windows relying on autonomous robots, in which the *2nd-level route* is an open route. First we introduce the problem and propose a mathematical formulation. A column generation procedure is presented for trying to get a tighter lower bound (compared with the low bound got by CPLEX) for the MIP model. Next we propose a construction heuristic for the newly introduced problem, which is useful for quickly generating feasible initial solutions and providing a first upper bound. We then propose a multi-start hybrid metaheuristic approach based iterative local search and backtracking. The backtracking procedure accurately connects the chosen robot routes to the van route. Furthermore, we analyze how van/robot speed combinations influence the objective value, which can provide a reference for real-world implementation of such a service.

The remainder of the chapter is structured as follows. Section 3.2 describes the problem and model and Section 3.3 elaborates on the heuristic approaches. A computational study is presented in Section 3.4 and Section 3.5 concludes the chapter.

3.2 Problem description and model

3.2.1 Problem statement

We consider a two-echelon urban delivery problem using robots for *2nd-level route* delivery. The van carries the robots on the *1st-level route* and drops off and picks up them in the rendezvous nodes, while the robot handles customer service on the *2nd-level route*. The research developed in this chapter considers using the van only for carrying robots, and so no direct shipping from vans to customers is allowed. This setting is reasonable, since the robot can only deliver parcels or other small commodities to pedestrianized areas such as campuses or residential clusters with current technologies. Also, our target customers are in these pedestrianized areas, where vans are often banned. Hence, we assume the van cannot serve customers directly.

Since campuses and residential clusters have multiple entrances and exits, the van can drop off and pick up a robot at different positions. In other words, the van can move to other rendezvous nodes after making a drop-off operation without having to wait for its robot to come back to the same rendezvous location. Besides, the robot is not forced to return back to the rendezvous node it departed from, which means the *2nd-level route* is an open route.

Each pick-up or drop-off (rendezvous) node can only be visited at most once by the same vehicle. This setting is reasonable, as a van can release all its associated robots immediately at a drop-off node and has no need to reach that node again. Likewise, a van can arrive at a pick-up node when all its associated robots have arrived. Hence, a drop-off/pick-up node does not have to be visited twice by the same van. Each customer node must be visited by just one robot exactly once. In addition, customer nodes and depot have their time windows based on real demand in city logistics. Our model accommodates waiting at all locations without cost. Moreover, we allow a robot to visit multiple customers during a dispatch rather than only visit one customer, since the capacity for a robot is usually larger than that of a drone. The travel range of a van is infinite. Nevertheless, the total travel time of a robot cannot exceed a predetermined value on the *2nd-level route*, as the robot tends to be small-sized and thus equipped with limited fuel/battery capacity.

We present three (simplifying) assumptions. In order to simplify the model and reduce the complexity of our problem, a robot is dropped off and picked up by a single van. Furthermore, we assume constant operation times, and we can therefore integrate it into the travel time from and to the rendezvous nodes (Grangier et al., 2016). In the 2E-VRRP model proposed here, freight can only be loaded when rounds begin at the depot. In other words, the van carries a sufficient number of robots full of freight

instead of carrying some robots and freight and then taking on freight replenishment during the van route.

Figure 3.1 shows an example of the 2E-VRRP model. Triangles represent rendezvous nodes, the square represents the depot, and circles correspond to customer nodes. Solid lines correspond to *1st-level routes* (Van routes) and dotted lines correspond to *2nd-level routes* (robot routes).

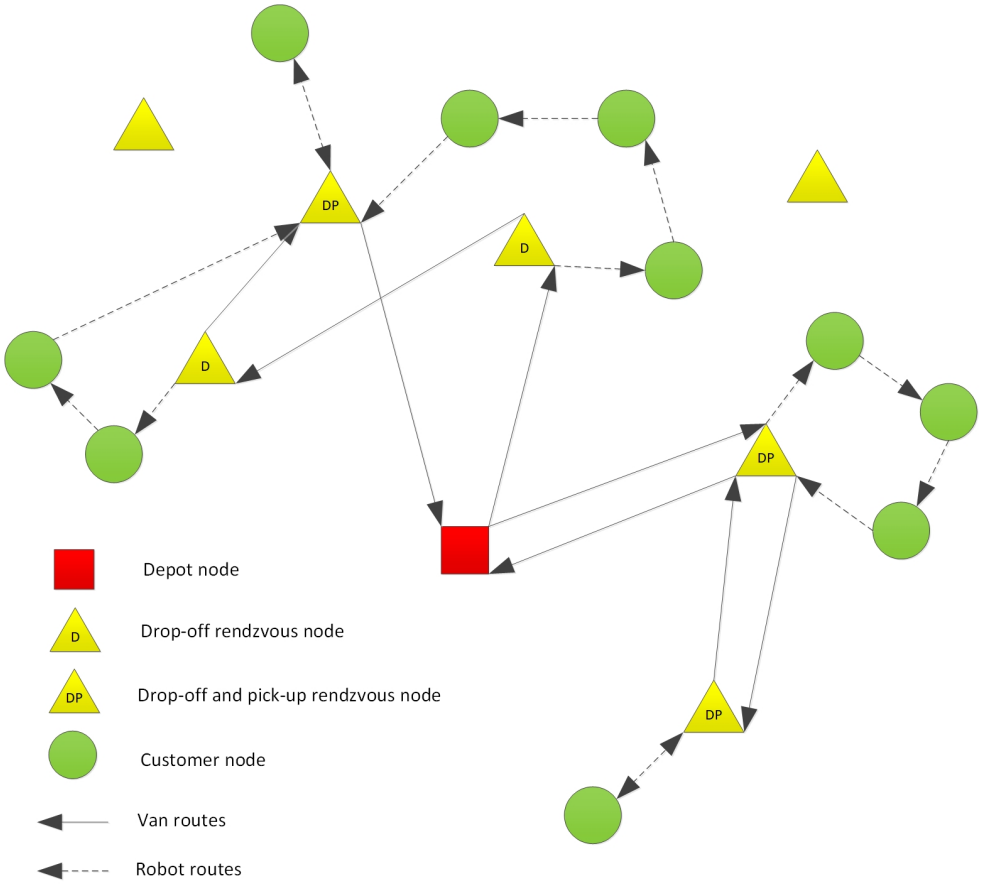

FIGURE 3.1: Delivery system with van/robot

3.2.2 Mixed Linear Integer Programming Model

The problem is defined on a directed graph $G = (V, A)$, where the depot V_0 is represented by two nodes 0 and $0'$. Let $V_r = \{1, 2, ..., m, m+1, ..., 2m\}$ be rendezvous nodes where vans drop off and pick up their robots. The node $m + i$ in pick-up nodes set $V_p = \{m+1, m+2, ..., 2m\}$ is a copy of the drop-off node i (physical rendezvous node) in set $V_d = \{1, 2, ..., m\}$. Note that node i and node $m + i$ correspond to a same physical rendezvous node. We use different names to distinguish drop-off and pick-up operations of vans. Furthermore, $V_c = \{2m+1, 2m+2, ..., 2m+n\}$ represents the customer nodes set. Moreover, we define $V_{dc} = V_d \cup V_c$ and $V_{pc} = V_p \cup V_c$. Also,

$V_r^0 = V_r \cup \{0\}$ and $V_c^0 = V_c \cup \{0\}$ while $V_r^{'} = V_r \cup \{0^{'}\}$ and $V_c^{'} = V_c \cup \{0^{'}\}$. Let $A_1 = \{(i,j) \mid i \in \{0\}; j \in V_d\} \cup \{(i,j) \mid i,j \in V_r, i \neq j\} \cup \{(i,j) \mid i \in V_p; j \in \{0^{'}\}\}$ be the *1st-level route* (Van routes) and $A_2 = \{(i,j) \mid i \in V_d; j \in V_c\} \cup \{(i,j) \mid i,j \in V_c, i \neq j\} \cup \{(i,j) \mid i \in V_c; j \in V_p\}$ be the *2nd-level route* (robot routes).

For each edge, let $t_{i,j} > 0$ be the associated travel time and $c_{i,j}$ be the associated travel cost. The freight must be delivered from depot $\{0\}$ to customer node i with the demand d_i and serving time s_i. The time window of the customer nodes $i \in V_c$ is $[a_i, b_i]$, which is the time interval that the service at node i is allowed to start. Furthermore, let $[a_0, b_0] = [a_{0'}, b_{0'}]$, where $a_0/a_{0'}$ represents the earliest possible departure time from the depot $0/0'$ and $b_0/b_{0'}$ is the latest possible arrival time at the depot $0/0'$. These time windows are hard constraints. $F_T = \{1, 2, ..., K\}$ is the set of vans and $F_D^k = \{(k,1), (k,2), ..., (k,L)\}$ is the set of robots belonging to the kth van, where K is the number of vans and L is a maximum number of robots assignable to a van. $F_D = F_D^1 \cup F_D^2 \cup, ..., \cup F_D^K = \{(1,1), (1,2), ..., (k,l), ..., (K,L)\}$ denotes the set of all robots. Let C, T, and M be capacity, maximum total travel time for a robot, and an arbitrary large constant number, respectively. In addition, we further introduce the following decision variables:

- Let $x_{i,j,k}$ equal to 1 if arc (i,j) in A_1 is traveled by the kth van, 0 otherwise.
- Let $y_{i,j}^{k,l}$ equal to 1 if arc (i,j) in A_2 is traveled by the lth robot belonging to kth van, 0 otherwise.
- Let $Q_{i,j,k}$ be the robot flow carried by kth van passing through arc (i,j) in A_1, i.e., the number of robots.
- Let W_i^k be the arrival time of kth van (or the robot belonging to the kth van) at node i. Note that W_i^k represents the last arrival time of kth van and the robots belonging to the kth van at node i.

The 2E-VRRP problem is modeled as the following MIP:

$$\text{Lex-min}\Big(\sum_{k \in F_T} \sum_{j \in V_d} x_{0,j,k}, \sum_{k \in F_T} \sum_{(i,j) \in A_1} c_{i,j} x_{i,j,k} + \sum_{(k,l) \in F_D} \sum_{(i,j) \in A_2} c_{i,j} y_{i,j}^{k,l}\Big) \tag{3.1}$$

$$\sum_{(i,j)\in A_1} x_{i,j,k} \leq 1, \qquad \forall j \in V_r^{'}, k \in F_T \tag{3.2}$$

$$\sum_{i\in V_p} x_{i,0',k} = \sum_{j\in V_d} x_{0,j,k}, \qquad \forall k \in F_T \tag{3.3}$$

$$\sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(j,i)\in A_1} x_{j,i,k} = 0, \qquad \forall j \in V_r, k \in F_T \tag{3.4}$$

$$\sum_{(k,l)\in F_D} \sum_{i\in V_{dc}} y_{i,j}^{k,l} = 1, \qquad \forall j \in V_c \tag{3.5}$$

$$\sum_{i\in V_{dc}} y_{i,j}^{k,l} - \sum_{i\in V_{pc}} y_{j,i}^{k,l} = 0, \qquad \forall j \in V_c, (k,l) \in F_D \tag{3.6}$$

$$\sum_{i\in V_p} Q_{i,0',k} = \sum_{j\in V_d} Q_{0,j,k}, \qquad \forall k \in F_T \tag{3.7}$$

$$\sum_{j\in V_d} Q_{0,j,k} = \sum_{l\in F_D^k} \sum_{i\in V_c} \sum_{j\in V_d} y_{j,i}^{k,l}, \qquad \forall k \in F_T \tag{3.8}$$

$$\sum_{i\in V_r^0} Q_{i,j,k} - \sum_{i\in V_r} Q_{j,i,k} = \sum_{l\in F_D^k} \sum_{i\in V_c} y_{j,i}^{k,l}, \qquad \forall j \in V_d, k \in F_T \tag{3.9}$$

$$\sum_{i\in V_r} Q_{i,j,k} - \sum_{i\in V_r^{'}} Q_{j,i,k} = -\sum_{l\in F_D^k} \sum_{i\in V_c} y_{i,j}^{k,l}, \qquad \forall j \in V_p, k \in F_T \tag{3.10}$$

$$0 \leq Q_{i,j,k} \leq x_{i,j,k} * L, \qquad \forall (i,j) \in A_1, k \in F_T \tag{3.11}$$

$$W_i^k + t_{i,j} - W_j^k \leq M(1 - x_{i,j,k}), \qquad \forall (i,j) \in A_1, k \in F_T \tag{3.12}$$

$$W_i^k + t_{i,j} - W_j^k \leq M(1 - y_{i,j}^{k,l}), \qquad \forall i \in V_d, j \in V_c, (k,l) \in F_D \tag{3.13}$$

$$W_i^k + t_{i,j} + s_i - W_j^k \leq M(1 - y_{i,j}^{k,l}), \qquad \forall i \in V_c, j \in V_{pc}, (k,l) \in F_D \tag{3.14}$$

$$a_i \leq W_i^k, \qquad \forall i \in V_c^0, k \in F_T \tag{3.15}$$

$$W_i^k \leq b_i, \qquad \forall i \in V_c^{'}, k \in F_T \tag{3.16}$$

$$\sum_{i\in V_c} d_i \sum_{j\in V_{pc}} y_{i,j}^{k,l} \leq C, \qquad \forall (k,l) \in F_D \tag{3.17}$$

$$\sum_{(i,j)\in A_2} y_{i,j}^{k,l} t_{i,j} \leq T, \qquad \forall (k,l) \in F_D \tag{3.18}$$

$$x_{i,j,k} \in \{0,1\}, \qquad \forall (i,j) \in A_1, k \in F_T \tag{3.19}$$

$$y_{i,j}^{k,l} \in \{0,1\}, \qquad \forall (i,j) \in A_2, (k,l) \in F_D \tag{3.20}$$

The lexicographic objective function (3.1) minimizes the number of vans first and then minimizes the total transportation cost of *1st-level* and *2nd-level* routes. Constraint (3.2) implies that each van departs from (or arrives at) the rendezvous node no more than once, and each van arrives at the depot no more than once. Constraints (3.3-3.4) ensures the number of arrivals is equal to the number of departures for the van at

the depot/rendezvous node. Constraints (3.5-3.6) indicate that each customer node is visited exactly once. Constraints (3.7-3.8) assure the number of robots (carried by the *kth* van) departing from/arriving at the depot is equal to the total number of robots (carried by the *kth* van) departing from the drop-off nodes. Constraints (3.9-3.10) link the number of robots carried by van to the number of robots departing from the drop-off node/arriving at the pick-up node. Constraint (3.11) guarantees that the number of robots carried by a given van cannot exceed its capacity on *1st-level route*.

Constraint (3.12) is the time-flow constraint for the vans. Constraints (3.13-3.14) are time-flow constraints for the robots. Service time does not need to be considered when a robot departs from drop-off nodes to customer nodes, but need to be considered when a robot is traveling between customer nodes or from customer nodes to pick-up nodes. Constraint (3.12-3.14) can also eliminate the subtour of *1st-level route* and *2nd-level route*. Constraints (3.15-3.16) are the time window constraints for the customer nodes. Constraint (3.17) ensures the demand of every customer is met. Constraint (3.18) forces the maximum travel time of each robot. Constraints (3.19-3.20) are the constraints on variables.

3.3 Methodology

This section introduces approximate solution methods for the 2E-VRRP problem. Section 3.3.1 proposes a construction heuristic to obtain a feasible solution quickly. The construction heuristic is applied to provide an upper bound during optimization of the primary objective of the MIP model, and also to generate multiple initial solutions for a hybrid metaheuristic approach. In Section 3.3.2 we propose a hybrid multi-start metaheuristic including destroy and repair operators together with a backtracking component.

3.3.1 Construction heuristic

The general structure of the construction heuristic is sketched out in Algorithm 1. Let S_2 be the set of rendezvous nodes and the depot. The sequence *GetStartCustomerNode*, *GetDropOffNode* and *ModifiedNearestNeighbor* is repeated to construct multiple robot routes (*2nd-level route*) until no customer nodes are left in the unvisited customer nodes set S_1 (lines 2-8). Meanwhile, set S_1 and the complete robot route set S_3 are updated (lines 6-7). Afterwards, a simple connection heuristic (*SimpleConnectionHeuristic*) is applied to construct multiple van routes (*1st-level route*) to get the final solution S (line 9).

Algorithm 1 *Construction_Heuristic*(S_1, S_2)

1: **Initialization:** $S_3 \leftarrow \varnothing$
2: **while** $S_1 \neq \varnothing$ **do**
3: $Node_1 \leftarrow GetStartCustomerNode(S_1)$
4: $Node_2 \leftarrow GetDropOffNode(Node_1, S_2)$
5: $RobotRoute \leftarrow ModifiedNearestNeighbor(Node_1, Node_2, S_1, S_2)$
6: Remove Customer nodes in $RobotRoute$ from S_1
7: $S_3.add(RobotRoute)$
8: **end while**
9: $S \leftarrow SimpleConnectionHeuristic(S_3)$
10: **Output:** S

Note: S_1: unvisited customer nodes, S_2: rendezvous nodes and depot, S_3: complete robot routes, S: final solution.

First, we introduce how to construct the *2nd-level route*. The *GetStartCustomerNode* procedure is run to select a customer node for the second level route construction algorithm (lines 2-8) to start in Algorithm 1. Here, we introduce the deterministic/random start customer node selection strategies that can be used in *GetStartCustomerNode*. The deterministic start customer node selection strategy is the one we obtain the start customer node with the minimum latest service time (similar to (Li et al., 2016)), while in the random start customer node selection strategy, the start visiting customer node from the customer nodes set is chosen randomly to maintain the diversity of solutions.

After choosing a start customer node, we select a drop-off node to connect the depot, drop-off node and start customer node. We propose optimal/nearest drop-off node selection strategies that can be implemented in *GetDropOffNode*. The optimal drop-off node selection strategy chooses the drop-off node for which distance to depot plus distance to start customer node is the smallest, whereas the nearest drop-off node selection strategy picks the drop-off node with the nearest distance to the start customer node.

It is important to note here that the optimal drop-off node selection strategy tends to select a drop-off node close to the depot; whereas the nearest drop-off node selection strategy tends to choose a drop-off node close to customer nodes. These two different selection strategies thus yield a significant difference in initial solutions.

Next, we apply *ModifiedNearestNeighbor* to select the remaining customer nodes and a pick-up node sequentially for constructing the robot route. *ModifiedNearestNeighbor* not only uses the nearest distance information like the nearest neighbor search does, but it also considers the start time window information at the chosen customer node. The total nearest of *ModifiedNearestNeighbor* is described as the robot start serving time earliest. In other words, we choose the customer node with the earliest time at which the robot can start to provide services.

The specific operation of *ModifiedNearestNeighbor* chooses the total nearest customer

node based on the current chosen point subject to all constraints. The constraints involve the time window constraints for the robot and van, the capacity constraint, and the maximum travel time constraint for the robot. The procedure iterates until there is no customer node that satisfies the constraints, then selects the nearest pick-up node to finally make a complete robot route.

Second, we introduce how the *SimpleConnectionHeuristic* constructs the *1st-level route*. The general structure of the simple connection heuristic is sketched out in Algorithm 2, which is based on a distance nearest neighbor search to connect the given robot routes with vans. The sequence of *GetStartRendezvousNode*, *GetRobotRoute* and *ConnectionCheck* procedure is repeated to construct one complete 2E-VRRP route until the number of chosen robot routes in set S_2 is equal to the capacity of the van (Van_{max}) or until all unconnected robot routes in set S_3 (a copy of unconnected robot routes set S_1) have been tried (lines 5-15). The process iterates until all robot routes are connected (lines 2-18). Finally, we output final solution S.

Algorithm 2 $Simple\_Connection\_Heuristic(S_1)$

1: **Initialization:** $S_2 \leftarrow \varnothing$, $S \leftarrow \varnothing$, Van_{max}
2: **while** $S_1 \neq \varnothing$ **do**
3: $S_2 \leftarrow \varnothing$
4: $S_3 \leftarrow Copy(S_1)$
5: **while** $S_3 \neq \varnothing$ **do**
6: $Node \leftarrow GetStartRendezvousNode(S_2, S_3)$
7: $RobotRoute \leftarrow GetRobotRoute(Node, S_2, S_3)$
8: $S_3.remove(RobotRoute)$
9: **if** $!(VanRobotRoute \leftarrow ConnectionCheck(S_2, RobotRoute))$ **then**
10: $S_2.add(RobotRoute)$
11: **end if**
12: **if** $len(S_2) = Van_{max}$ or $S_3 = \varnothing$ **then**
13: $S.add(VanRobotRoute)$, break while loop
14: **end if**
15: **end while**
16: $S_1.remove(S_2)$
17: **end while**
18: **Output:** S

Note: S_1: unconnected robot routes set, S_2: chosen robot routes, S_3: a copy of S_1, S: final solution, Van_{max}: capacity of van.

The *GetStartRendezvousNode* procedure is applied to choose a drop-off node for Algorithm 2 to start constructing one 2E-VRRP route. If it is the first drop-off node we will choose, then we select the node with the maximum number of robot routes departing from it. Otherwise, we select the drop-off node based on the nearest distance criterion. Furthermore, a drop-off node can be selected more than once if there are robot routes from this node not chosen. The *GetRobotRoute* procedure is implemented to choose a

robot route based on the selected drop-off node. We classify the procedure into two types involving four sub-scenarios as described in Table 3.1. The *ConnectionCheck* procedure is executed to connect the chosen robot routes. In order to simplify the connection process, *ConnectionCheck* accesses all the drop-off nodes first according to the chosen sequence. Afterward, *ConnectionCheck* visits the pick-up nodes according to the selected order. The procedure checks the constraints during the whole process.

TABLE 3.1: Get Robot Route procedure

Scenario	Sub-Scenario	Operator
The drop-off node is the first node a van visited in one 2E-VRRP route	There is no previous robot route connected	Choose the corresponding robot route with the smallest robot arrival time at its pick-up node
	There is previous robot route connected	Choose the node with nearest distance from the chosen robot route's pick-up node to the previous robot route's pick-up node
The drop-off node is not the first node a van visited in one 2E-VRRP route	For the robot routes departing from current drop-off node, there exist corresponding pick-up nodes the same as the pick-up nodes of already chosen robot routes	Select the robot route with the same pick-up node to the already chosen robot routes
	For the robot routes departing from current drop-off node, there not exist corresponding pick-up nodes the same as the pick-up nodes of already chosen robot routes	Select the robot route with the nearest distance between its pick-up node and the previous robot route's pick-up node

Third, we give a simple example of how the simple connection heuristic performs in Figure 3.2. The triangles represent the rendezvous nodes, the square represents the depot, and the circles are the customer nodes. The solid line with the arrow is the van route, and the dotted lines with the arrow are robot routes. In order to simplify the example and focus on the specific selection process, we assume that the van route 0-A-D-B-C-0 is feasible in advance. Also, we assume robot route 1 has the shortest robot arrival time at its pick-up node.

- Step 1: First we choose drop-off node A as it has 3 robots departing from it, which is larger than the number for drop-off node D. Then we select robot route 1 since it has the smallest robot arrival time at its pick-up node. Afterward, we check van route 0-A-B-0 and find it feasible.

- Step 2: We select drop-off node A since it is closest to the drop-off node of the previously chosen route 1. Next, we choose robot route 2 since its pick-up node

B is nearest to the pick-up node for the previously selected route 1. Finally, van route 0-A-B-0 is connected, and it is feasible.

- Step 3: We select the drop-off node A and robot route 3 based on the nearest distance criterion, as done in step 2. Afterward, we use *ConnectionCheck* to check and connect van route 0-A-B-C-0.

- Step 4: First we find drop-off node D as it has the nearest distance to the drop-off node of the chosen route 3. Afterward, we choose robot route 4 since its pick-up node C has already been chosen before. Van route 0-A-D-B-C-0 is feasible, and we find the number of robots that the van carried is up to its capacity. One complete 2E-VRRP route is successfully constructed.

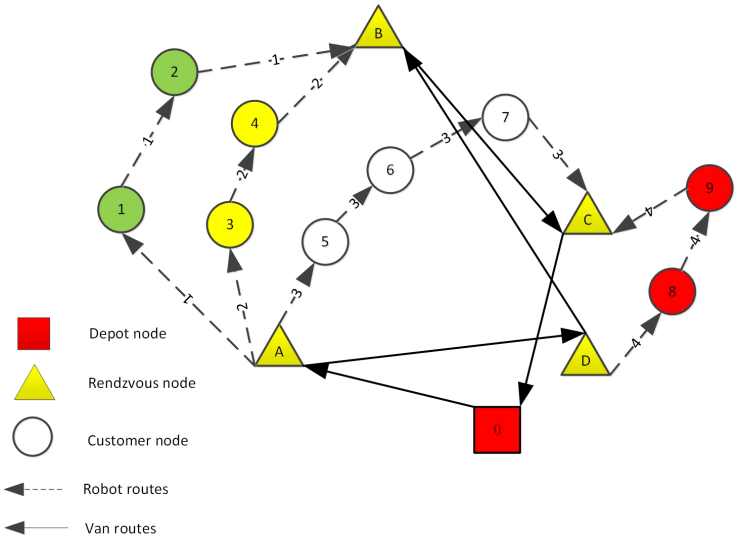

FIGURE 3.2: Simple Connection Heuristic example

3.3.2 Hybrid metaheuristic

Hemmelmayr et al., 2012 and Breunig et al., 2016 proposed a large neighborhood search-based approach for handling general two-echelon routing problems in which they used a destroy and repair operator with local search phase. Here, we draw on some of their ideas and then present a hybrid metaheuristic approach to improve the construction heuristic solutions. One way to achieve diversification is to re-start the procedure from a new solution (Martí et al., 2019). It is clear that if the iteration number is fixed, the initial solution has a significant impact on quality of the final solution. Hence, we also borrow some of the ideas used in multi-start heuristics (Nguyen et al., 2012; Han and Chu, 2016) to design the hybrid metaheuristic.

The general structure of the hybrid metaheuristic is depicted in Figure 3.3. The destroy-repair loop aims to minimize the primary objective while the iterated local search (ILS) loop optimizes the primary and secondary objectives simultaneously. A multi-start loop restarts the algorithm by starting from a new initial solution clearly distinct from the previous one. Furthermore, we use maximum iteration numbers as the acceptance criteria 1,2,3,4.

A more detailed algorithm in pseudocode is given in Algorithm 3. The construction heuristics are used to generate multiple initial solutions from $S_1^{initial}$ to $S_{max}^{initial}$ (line 2). The destroy-repair loop with an iteration number $IterN2_{max}$ (lines 5-13) is performed to obtain solution S^{dr} (line 14). In which the destroy and repair procedures are run until there is no customer node in the chosen customer node set S_{node} (lines 8-11). If we get S^{dr} for the first time or if the number of complete 2E-VRRP routes in S^{dr} is smaller than the previous one, the local search and ILS procedures are implemented (lines 16-27). Otherwise, the algorithm breaks out of the repair-destroy-ILS loop (lines 3- 32). The ILS loop (lines 17-26) is run to update and obtain best solution S^{best} (line 22). In which the perturbation and local search repeat until the iteration number is equal to $IterN3_{max}$. If the perturbation is feasible for the current best solution S^{best}, the local search is conducted to get solution S^{end} (lines 19-20). If S^{end} is better than S^{best}, then update S^{best} (lines 21-22). These procedures restart using the multi-start loop (lines 2-34), and every best solution S^{best} will be saved in solution set S (line 33).

For two-echelon routing problems, the number of satellites is generally much less than the number of customer nodes since a satellite is essentially like a small depot and maintaining its normal operation requires funds. Hence, using an exact algorithm to build the *1st-level route* is a viable method if the number of rendezvous nodes is small. In this chapter, we first construct the *2nd-level route* and then use vans to connect them. If we have several robots belonging to one van, we can employ a backtracking algorithm to connect the robot routes with the van route. The method is applied in the repair and ILS procedures. After a customer node has been inserted into a complete 2E-VRRP route or after performing moves in the ILS, the backtracking algorithm is applied to check whether the *1st-level route* can be successfully connected, and then the backtracking outputs the solution with the minimum travel cost of *1st-level route* if needed.

Destroy and Repair

A feasible and straightforward way to reduce the number of vans used is to choose a complete 2E-VRRP route to destroy it and then insert its customer nodes into other 2E-VRRP routes.

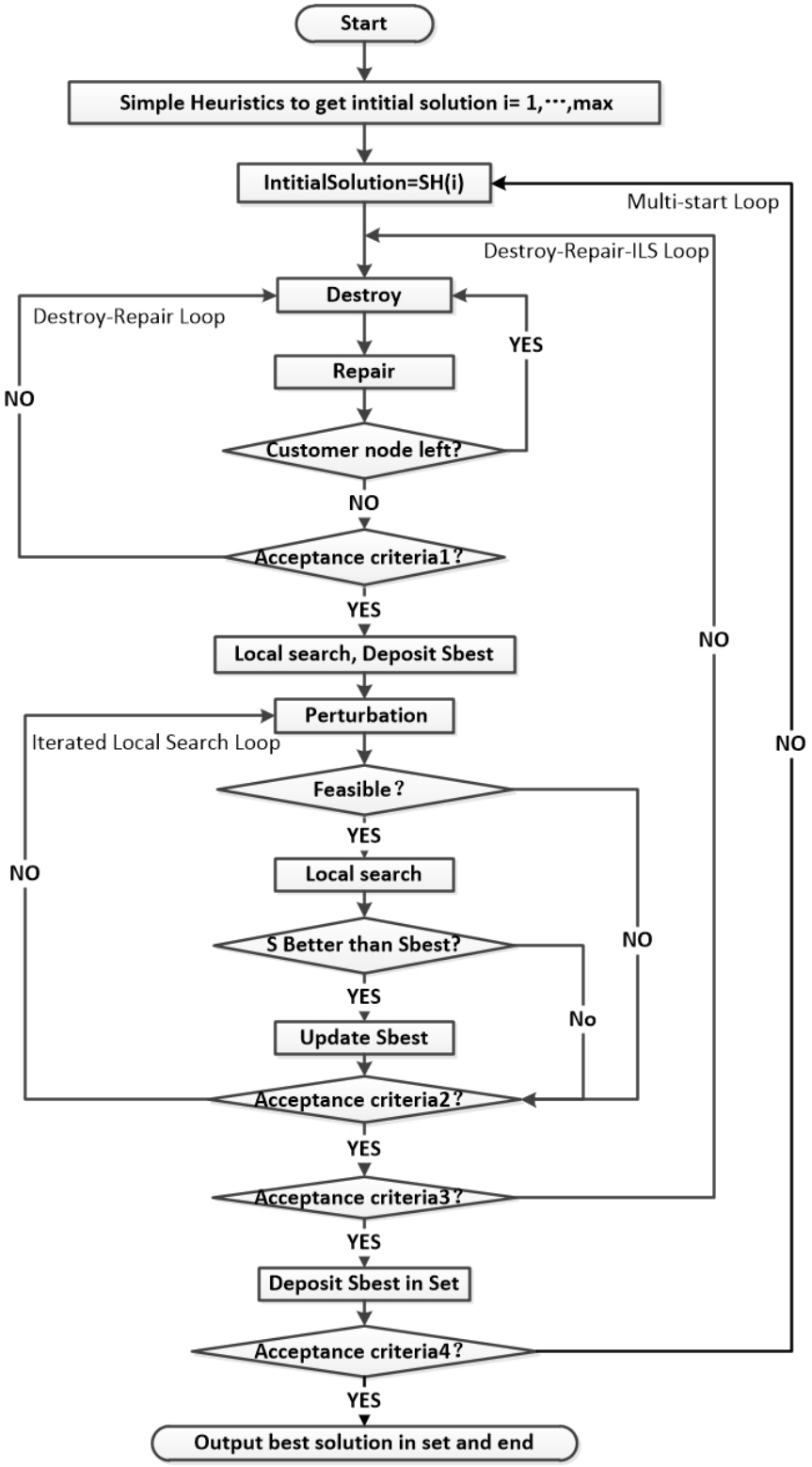

FIGURE 3.3: Hybrid metaheuristic-General flowchart

Algorithm 3 *Hybrid_metaheuristic*($IterN1_{max}$, $IterN2_{max}$, $IterN3_{max}$)

$S_1^{initial}$,..., $S_{max}^{initial} \leftarrow ConsrtuctionHeuristic()$, The solution set $S \leftarrow \varnothing$
for $S_1^{initial}$ to $S_{max}^{initial}$ **do**
  Initialization: $IterN1 \leftarrow 0$
  while $IterN1 < IterN1_{max}$ **do**
    Initialization: The temp solution set $S^{temp} \leftarrow \varnothing$, $IterN2 \leftarrow 0$
    while $IterN2 < IterN2_{max}$ **do**
      $S_{node} \leftarrow \varnothing$, $S^{temp1} \leftarrow S_i^{initial}$
      while S_{node} is empty **do**
        $S^{temp2} \leftarrow S^{temp1}$
        $(S_{node}, S^{temp1}) \leftarrow Repair(Destroy(S^{temp1}))$
      end while
      $S^{temp}.add(S^{temp2})$, $IterN2 \leftarrow IterN2 + 1$
    end while
    Get S^{dr} by choosing the solution with minimum primary objective value in set S^{temp}
    if we get the S^{dr} for the first time or the number of complete 2E-VRRP route in S^{dr} is smaller than the previous one **then**
      $S^{best} \leftarrow Localsearch(S^{dr})$
      Initialization: $IterN3 \leftarrow 0$
      while $IterN3 < IterN3_{max}$ **do**
        if $Perturbation(S^{best})$ is feasible **then**
          $S^{end} \leftarrow Localsearch(Perturbation(S^{best}))$
          if S^{end} better than S^{best} **then**
            $S^{best} \leftarrow S^{end}$
          end if
        end if
        $IterN3 \leftarrow IterN3 + 1$
      end while
      $S_i^{initial} \leftarrow S^{best}$
    else
      break while loop
    end if
    $IterN1 \leftarrow IterN1 + 1$
  end while
  $S.add(S^{best})$
end for
Output: Output the best solution in S

The destroy procedure destroys a chosen complete 2E-VRRP route into the list of nodes for re-inserting. In this chapter, we consider randomly choosing a 2E-VRRP route to be destroyed, which guarantees a diversity of solutions. At each repair phase, we randomly insert the lists of nodes obtained by the destroy procedure to other complete 2E-VRRP routes in random order. In addition, the repair procedure inserts each customer node at its first feasible position or at the position with the largest positive saving of the secondary objective. If there are still nodes that cannot be successfully inserted at the end, the insertion fails.

Local search

In this chapter, we exploit well-known moves such as insertion, swap, and 2-opt. Furthermore, the moves we called 'change-satellites' are also used to change the drop-off and pick-up nodes of a robot route.

The 2-opt moves are executed inside the robot route. Details on 2-opt can be found in (Croes, 1958). The insertion and swap operators are applied in three different route configurations that are inside a robot route, inside a complete 2E-VRRP route, and between complete 2E-VRRP routes, respectively. The insertion moves searches the intercalation of one node after one of the neighbor nodes while the swap moves explore swapping one node with one of the neighbor nodes. For one robot route, the change-satellites operator involves changing the drop-off and pick-up node individually or simultaneously. Specific insertion, swap, and change-satellites operations are introduced in greater depth below.

Figure 3.4 describes three neighborhood structures, and each contains several sub-neighborhood structures. The insertion operators N1, N2, and N3 all consist of removing a customer node from a position i and inserting it after a position j: N1 removes and inserts customer nodes in the same robot route, N2 removes and inserts customer node in different robot routes from a same van, and N3 removes and inserts customer node in different robot routes between different 2E-VRRP routes. Swap operators N4, N5, and N6 all involve swapping customer-node positions. N4 swaps two customer nodes in the same robot route whereas N6 swaps two customer nodes in different robot routes incident to different 2E-VRRP routes. N5 swaps customer nodes in two robot routes within a same van route. The change-satellites operators N7, N8, and N9 change pick-up node only, drop-off node only, and both drop-off and pick-up nodes, respectively.

Since the insertion moves make it possible to reduce the number of van routes, the local search procedure can simultaneously optimize both the primary and secondary objectives. In the insertion process, we multiply a significant penalty by the number of van routes in the cost function. Hence, once a solution emerges that reduces the primary objective, it can readily become the current optimal solution and can be saved.

In this chapter, we apply the sequential Variable Neighbor Descent (VND) with first improvement (Duarte et al., 2018) to conduct the local search. Note that the neighbor sequence is randomly predetermined before starting the VND in order to promote a diversity of solutions.

Perturbation

We present the perturbation procedure involving three perturbations that bring a significant change to the structure of the solution, which is good for escaping local optima.

The first perturbation *Change-Multisatellites* is an enhanced version of the change-satellites operator which allows simultaneous change of two rendezvous nodes in a 2E-VRRP route. The second perturbation *Change-Robotroute* determines that robot routes can swap between different 2E-VRRP routes. The last perturbation *Destroy-Repair-Reconstruction* randomly destroys a 2E-VRRP route and inserts its customer nodes into other 2E-VRRP routes. If there are nodes left, the route is reconstructed by construction heuristics.

For the reconstruction procedure, we use the *GetStartCustomerNode* procedure by choosing the random start customer node selection strategy. Moreover, we randomly choose a drop-off node selection strategy in the *GetDropOffNode* procedure, which is used in the multi-start procedure of the hybrid metaheuristic. These two node selection strategies guarantee the diversity of solutions.

Backtracking algorithm for connection

In general, one van carries several robots not exceeding its capacity. Furthermore, each robot route starts from a drop-off node and ends at a pick-up node. However, different robot routes may start from the same drop-off node or end at the same pick-up node. If the number of robots carried by a van is larger than the number of physical rendezvous nodes, in other words, $L > m$, there are $2m$ nodes that will be visited at most by one van. Otherwise, there are at most $2L$ nodes that will be visited by one van. Overall, a maximum of $2 * min(L, m)$ different rendezvous nodes will be visited by one van. Hence, there are less than $A_{2*min(L,m)}^{2*min(L,m)}$ cases that need to be examined in a full enumeration method. Furthermore, as there are priority constraints between visiting drop-off and pick-up nodes belonging to the same robot route, for example a van should visit the drop-off node first before it can access the pick-up node on the same robot route, then the possible combinations of visiting sequences will be significantly reduced.

If one of the two values L or m is small, we can connect the robot routes belonging to one van entirely through an exact algorithm. For the instances studied here, $L = 4, m =$

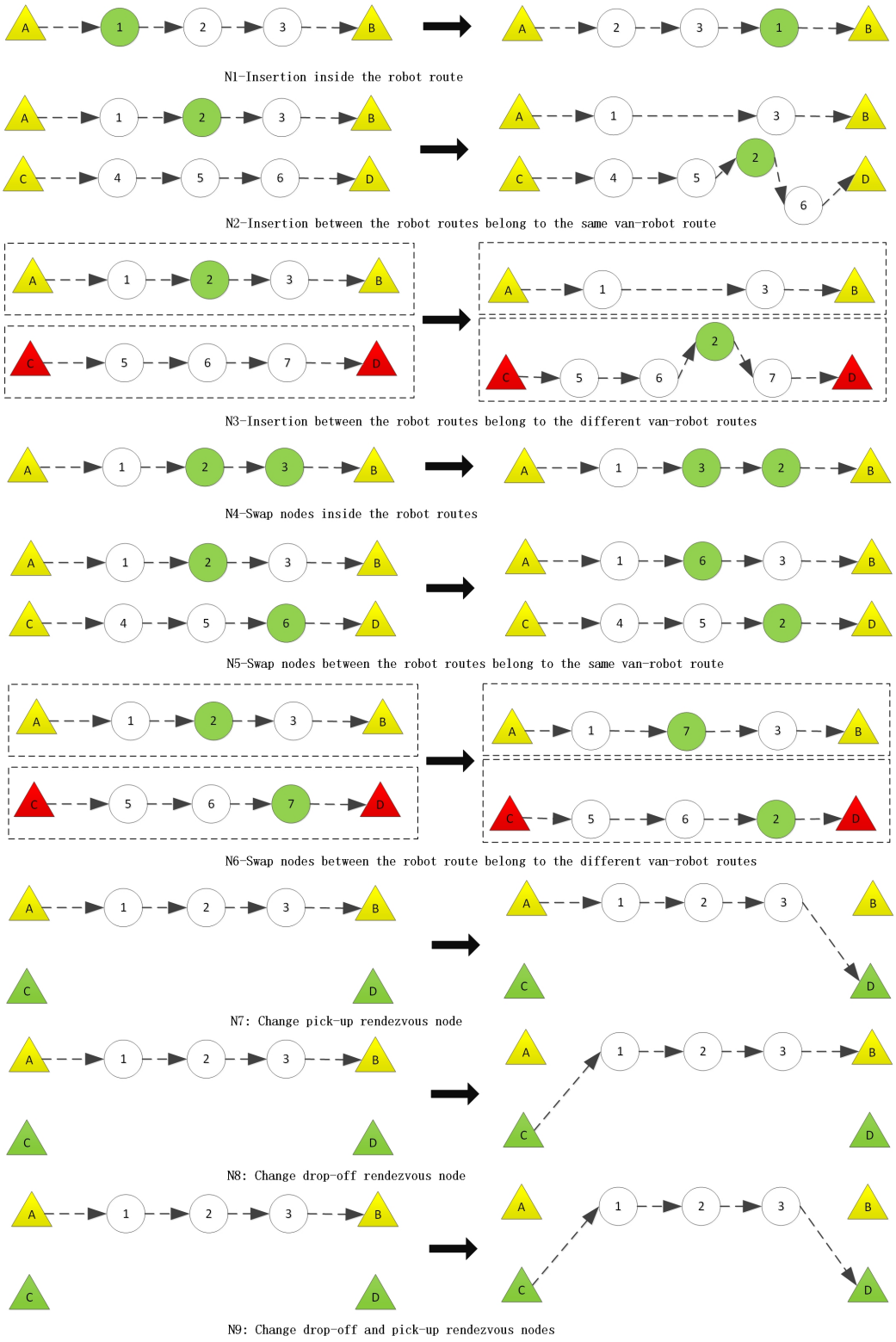

FIGURE 3.4: Local search operator

3, 4, 5. Hence, we choose a simple and adaptable backtracking algorithm to address the connection between a van and its robots.

The backtracking algorithm involves a depth-first search scheme and recursive invocation. After the van visits a rendezvous node, the collection of accessible rendezvous nodes is synchronously updated. Pruning is also implemented if the current arrival time of the van is greater than the pre-calculated latest allowed arrival time of the van at the rendezvous node. Note that the general constraints are checked during the whole flow of the algorithm for pruning. The algorithm records and outputs the best solution finally.

The general structure of the backtracking connection algorithm is depicted in Algorithm 4. If current accessible points set S_2 is empty (line 1), check and update the best solution (Line 2-8). Return the status $true/false$, best link S^{best} and best value V. If S_2 is not empty (line 1), execute (lines 10-22). *JudgeTimeWindow* and *UpdateCollection* procedures are run sequentially if connecting node i does not violate the *JudgeTimeWindow* constraints (lines 12-14). *Backtracking_Connection* calls itself to obtain the new S^{best} and V if its status is *true* (line 15). If V is larger than a predetermined value that the 2E-VRRP route value cannot reach, return status as *false* (lines 20-22).

We introduce the *UpdateCollection* and *JudgeTimeWindow* procedures as follows.

UpdateCollection: For the robot routes belonging to one van, there are priority constraints between the drop-off and pick-up nodes. For example, there may be several viable robot routes starting from the same drop-off node but ending at different pick-up nodes. These pick-up nodes have to be visited after the drop-off node has been visited. Similarly, there may be several viable robot routes ending at the same pick-up node but starting from different drop-off nodes. This pick-up node should be visited after all its relative drop-off nodes have been visited. During the *Backtracking_Connection* algorithm, we continually update the accessible point collection (*UpdateCollection*) synchronously after one rendezvous node has been visited.

The *UpdateCollection* procedure is addressed at each rendezvous node during the backtracking algorithm. In the beginning, only drop-off nodes are allowed to be visited. Once a drop-off node has been visited, we remove it from the accessible point collection. For the current chosen drop-off node, there may be several robot routes departing from it and ending at one or more than one pick-up nodes. For these pick-up nodes, if there is the node whose corresponding drop-off nodes have all been visited, we put it into the accessible point collection. In addition, if the algorithm visited one pick-up node, we just remove it from the accessible point collection.

JudgeTimeWindow: In order to speed up the backtracking, we calculate the latest allowed drop-off nodes arrival times for a van as a pruning standard. If a van arrived at the drop-off node later than its latest allowed arrival time, the route is unfeasible. We

Algorithm 4 *Backtracking_Connection*$(S_1, S_2, S^{best}, T, V, P)$

if S_2 is empty **then**
  if the value of S_1 is large than V **then**
    $V \leftarrow value(S_1)$
    $S^{best} \leftarrow S_1$
    return($true, S^{best}, V$**)**
  else
    return($false, S^{best}, V$**)**
  end if
else
  for i to S_2 **do**
    $S_1^{temp}, S_2^{temp}, T^{temp} \leftarrow Copy(S_1, S_2, T)$
    if $!(JudgeTimeWindow(i, S_1^{temp}, T^{temp}, P))$ **then**
      $T^{temp} \leftarrow JudgeTimeWindow(i, S_1^{temp}, T^{temp}, P)$
      $(S_1^{temp}, S_2^{temp}) \leftarrow UpdateCollection(i, S_1^{temp}, S_2^{temp})$
      $(S_1^{best}, V) \leftarrow !(Backtracking\_Connection(S_1^{temp}, S_2^{temp}, S^{best}, T^{temp}, V, P))$
    else
      return($false, S_1^{best}, V$**)**
    end if
  end for
  if V large than predetermined largeValue **then**
    return($false, S_1^{best}, V$**)**
  end if
end if
return($true, S_1^{best}, V$**)**

Note: S_1: already visited node list, S_2: current could visited node set, S^{best}: best result, T: current time, V: value. P: chosen robot routes.

can quickly exclude permutations that are obviously not feasible.

We calculate the latest allowed drop-off node arrival time of a robot route from back to front. First, we calculate the latest allowed arrival time of the relative pick-up node, which equals the end of the depot time window minus the travel time between depot and pick-up node. Second, we add the latest allowed arrival time of the previous node to the travel time between the previous node and the current node. Then we compare the value obtained before to the end of the customer node time window and take the largest one. Third, we count the latest allowed arrival time of the drop-off node considering the travel time from the previous node. Note that if several robot routes depart from one drop-off node, we choose the smallest latest allowed arrival time among them.

The specific operation process calculates the latest allowed drop-off arrival time before the *Backtracking_Connection* algorithm executes. Then we make a dictionary, including the drop-off nodes and their corresponding latest allowed arrival times, to facilitate the query during *JudgeTimeWindow* procedure.

The *JudgeTimeWindow* procedure is applied to check the feasibility of the van visiting each rendezvous node and output the departure time from that node. If the judged node is a drop-off node, query the dictionary then see if we need to prune this node or not. If the node does not need to be pruned, calculate and output the departure time of the judged node. If the judged node is the pick-up node, judge the time window constraints for the van and robot routes then output the departure time for the judged nodes if feasible.

3.4 Computational study

We performed three types of computational experiments. First, we use CPLEX to provide benchmarks for small instances and estimate the scale of the problem that the solver can manage. Second, we evaluate the performance of the multi-start heuristic, iteration number, different moves and perturbations in the hybrid metaheuristic, and then compare the hybrid metaheuristic results against the CPLEX results to analyze its performance. Third, we implement a sensitivity analysis on the van/robot speed ratio to see how the related speed influences the objective of the 2E-VRRP model.

The mathematical programming algorithm is coded in OPL. CPLEX 12.8 is used to solve the MIP model. The hybrid metaheuristic is coded in python version 3.6.5. Both CPLEX and python are executed on an Intel(R) Core(TM) 3.6GHz processor with 32 GB memory running under Windows 10. Note that python is run with single-threading.

3.4.1 Instance generation

Here we borrow the instances of (Van Woensel, 2018) adapted to testing the 2EVRPTW problem. The specific instance generation approach can be found in (Dellaert et al., 2018). There are 12 types of instances with 3/4/5 rendezvous nodes and 15/30/50/100 customer nodes, respectively. Each type of instance counts 20 instances that can be divided into four cases according to the different time window and demand generation methods:

- For customer i of an instance of category CA. randomly generate $20 \leq a_i \leq 260$ and $a_i \leq b_i \leq a_i + 20$ and $d_z = 10$ or 20
- For customer i of an instance of category CB. randomly generate $20 \leq a_i \leq 260$ and $a_i \leq b_i \leq a_i + 20$ and $5 \leq d_z \leq 25$
- For customer i of an instance of category CC. randomly generate $60 \leq a_i \leq 360$ and $a_i \leq b_i \leq a_i + 90$ and $d_z = 10$ or 20
- For customer i of an instance of category CD. randomly generate $60 \leq a_i \leq 360$ and $a_i \leq b_i \leq a_i + 20$ and $d_z = 10$ or 20

We decreased the number of depots in the instances, as there is only one depot rather than multi-depots in 2E-VRRP model. In our test setting, the first depot in the instances is always chosen. In the algorithm design verification phase, we assume that speed of the van and robot and cost of the van and robot per meter are identical. Note that we propose a speed sensitivity analysis in section 3.4.3. Here, we assume that one van can carry 4 robots at most, that the capacity of one robot equals 50, that the largest travel time for one robot is 200, and that the time window of the depot is [0, 450].

After modifying the instances, there are several unfeasible instances for our experiments. The infeasibility mainly manifested in violating the time windows of the customer and the depot even in the most optimistic situation. In order to ensure the feasibility of our instances, we maintained the absolute value width of the time window unchanged, and then adjust the time window to just not violate the constraints.

3.4.2 CPLEX experiment

The total computation time for CPLEX 12.8 in each instance to minimize the primary and secondary objective is limited to 5 hours (18000s), respectively. In order to speed up the search, the upper bound of the primary objective value is obtained from the construction heuristic. We also try to compute a better lower bound of the secondary objective by column generation procedure (Desaulniers et al., 2006) within 5 hours. If the column generation procedure is not completed in 5 hours, the best of a valid

lower bound each iteration generated will be chosen as the column generation lower bound. We finally choose an enhanced lower bound, a better one between CPLEX lower bound and column generation lower bound, to assess the performance of our hybrid metaheuristics. Note that the column generation procedure is coded in DOcplex by calling CPLEX 12.8. The detailed column generation procedure is in A.1.

For each instance, CPLEX12.8 runs with default settings until finding an optimal solution or exhausting the predetermined maximum computation time. The computational results of all instances are presented in Table 3.2-3.3. Column 1 indicates the test instances, column 2 and column 4 show the solving time for primary and secondary objective. Column 3 gives a minimum number of vans required to be used while the columns 5, 6, and 7 are the upper bound and the lower bound of the secondary objective and the achieved gap, respectively. Column 8 and column 9 are the enhanced lower bound got by column generation and the achieved gap between CPLEX upper bound and the enhanced lower bound.

TABLE 3.2: CPLEX results for 15 customer instances

	3-15								4-15								5-15							
	TK(s)	K	TC(s)	UB	LB	Cgap(%)	ELB	Egap(%)	TK(s)	K	TC(s)	UB	LB	Cgap(%)	ELB	Egap(%)	TK(s)	K	TC(s)	UB	LB	Cgap(%)	ELB	Egap(%)
CA1	0.5	2.0	5.0	383.9	383.9	0.0	383.9	0.0	0.6	2.0	29.0	411.3	411.3	0.0	411.3	0.0	2.2	2.0	8.8	403.7	403.7	0.0	403.7	0.0
CA2	0.7	2.0	1.1	378.5	378.5	0.0	378.5	0.0	0.5	2.0	11.7	407.8	407.8	0.0	407.8	0.0	0.5	2.0	90.3	420.5	420.4	0.0	420.4	0.0
CA3	0.7	2.0	1.9	387.8	387.8	0.0	387.8	0.0	0.6	2.0	10.9	444.7	444.7	0.0	444.7	0.0	0.8	2.0	65.7	360.2	360.2	0.0	360.2	0.0
CA4	0.4	2.0	2.6	400.9	400.9	0.0	400.9	0.0	1.4	1.0	93.7	397.8	397.8	0.0	397.8	0.0	1.7	2.0	108.1	345.9	345.9	0.0	345.9	0.0
CA5	2.6	2.0	4.5	382.8	382.8	0.0	382.8	0.0	0.4	1.0	1.5	308.7	308.7	0.0	308.7	0.0	2.3	2.0	25.5	386.7	386.7	0.0	386.7	0.0
CB1	1.7	2.0	2.2	391.8	391.7	0.0	391.8	0.0	1.3	2.0	11.3	414.5	414.4	0.0	414.5	0.0	253.0	1.0	5623.8	530.0	530.0	0.0	530.0	0.0
CB2	0.5	2.0	1.8	410.4	410.4	0.0	410.4	0.0	0.6	2.0	28.6	446.9	446.9	0.0	446.9	0.0	2.6	2.0	47.1	406.9	406.8	0.0	406.9	0.0
CB3	0.9	2.0	61.6	448.4	448.4	0.0	448.4	0.0	1.4	2.0	30.9	438.5	438.5	0.0	438.5	0.0	2.3	2.0	67.0	366.2	366.2	0.0	366.2	0.0
CB4	1.1	2.0	2.4	377.6	377.5	0.0	377.6	0.0	0.8	2.0	52.6	393.6	393.5	0.0	393.6	0.0	8.2	1.0	124.5	335.2	335.2	0.0	335.2	0.0
CB5	2.6	1.0	4.3	408.1	408.1	0.0	408.1	0.0	1.2	1.0	2.0	361.6	361.6	0.0	361.6	0.0	2.4	2.0	54.9	396.6	396.6	0.0	396.6	0.0
CC1	2064.6	2.0	31.8	361.9	361.8	0.0	361.8	0.0	1038.3	2.0	18000.0	419.9	346.4	17.5	415.5	1.0	0.6	1.0	4321.8	340.3	340.3	0.0	340.3	0.0
CC2	18000.0	2.0	182.1	344.8	344.7	0.0	344.8	0.0	770.0	2.0	530.7	391.5	391.5	0.0	391.5	0.0	2.4	1.0	18000.0	359.8	308.0	14.4	350.7	2.5
CC3	448.6	2.0	97.8	380.9	380.9	0.0	380.9	0.0	18000.0	2.0	18000.0	435.9	369.8	15.2	419.8	3.7	1764.7	2.0	18000.0	354.5	284.6	19.7	337.9	4.7
CC4	1225.0	2.0	2633.8	379.3	379.3	0.0	379.3	0.0	17.2	2.0	18000.0	377.6	326.4	13.6	364.4	3.5	18000.0	2.0	13593.3	347.3	347.2	0.0	347.2	0.0
CC5	0.4	1.0	1107.2	286.5	286.4	0.0	286.5	0.0	0.8	1.0	29.6	301.4	301.4	0.0	301.4	0.0	300.5	2.0	18000.0	392.8	320.4	18.4	374.4	4.7
CD1	1.5	2.0	1.8	370.8	370.8	0.0	370.8	0.0	0.8	2.0	21.5	408.4	408.4	0.0	408.4	0.0	1.8	2.0	9.5	385.4	385.4	0.0	385.4	0.0
CD2	0.2	1.0	1.6	348.8	348.8	0.0	348.8	0.0	2.0	2.0	6.8	396.9	396.9	0.0	396.9	0.0	0.6	1.0	6.0	384.2	384.2	0.0	384.2	0.0
CD3	0.2	1.0	1.4	382.1	382.1	0.0	382.1	0.0	1.5	2.0	33.0	451.2	451.2	0.0	451.2	0.0	11.8	2.0	50.1	351.2	351.2	0.0	351.2	0.0
CD4	0.5	2.0	6.2	381.6	381.6	0.0	381.6	0.0	1.2	2.0	18.5	355.4	355.3	0.0	355.4	0.0	4.0	2.0	42.5	339.5	339.4	0.0	339.4	0.0
CD5	0.5	2.0	2.2	368.9	368.9	0.0	368.9	0.0	0.7	1.0	29.4	301.4	301.4	0.0	301.4	0.0	1.6	2.0	13.7	404.3	404.2	0.0	404.2	0.0

TABLE 3.3: CPLEX results for 30 customer instances

	3-30								4-30								5-30							
	TK(s)	K	TC(s)	UB	LB	Cgap(%)	ELB	Egap(%)	TK(s)	K	TC(s)	UB	LB	Cgap(%)	ELB	Egap(%)	TK(s)	K	TC(s)	UB	LB	Cgap(%)	ELB	Egap(%)
CA1	54.9	3.0	18000.0	756.6	687.2	9.2	752.7	0.5	123.7	3.0	18000.0	702.3	594.0	15.4	693.7	1.2	16.0	3.0	18000.0	624.4	558.6	10.5	602.2	3.6
CA2	55.8	3.0	18000.0	653.1	618.9	5.2	653.1	0.0	19.5	3.0	18000.0	687.1	613.5	10.7	676.8	1.5	18.5	3.0	18000.0	635.7	552.4	13.1	624.8	1.7
CA3	26.6	3.0	18000.0	688.5	616.0	10.5	680.3	1.2	141.1	3.0	18000.0	649.1	556.2	14.3	639.5	1.5	71.3	3.0	2050.1	602.6	602.5	0.0	602.5	0.0
CA4	34.2	3.0	2100.4	579.5	579.4	0.0	579.4	0.0	24.2	3.0	18000.0	688.8	590.2	14.3	676.3	1.8	18.8	3.0	18000.0	662.2	578.1	12.7	578.1	12.7
CA5	30.8	3.0	18000.0	650.1	577.2	11.2	650.1	0.0	17.8	3.0	18000.0	585.1	542.5	7.3	585.1	0.0	6845.4	3.0	18000.0	649.4	488.1	24.8	488.1	24.8
CB1	192.3	3.0	18000.0	768.8	706.3	8.1	758.2	1.4	41.1	3.0	18000.0	713.6	563.7	21.0	673.1	5.7	180.9	3.0	18000.0	644.3	512.6	20.4	580.1	10.0
CB2	6.4	3.0	14885.8	632.2	632.2	0.0	632.2	0.0	161.5	3.0	18000.0	647.5	584.2	9.8	642.9	0.7	6844.3	3.0	18000.0	640.0	531.0	17.0	588.1	8.1
CB3	130.8	3.0	18000.0	691.3	606.4	12.3	679.9	1.6	46.8	3.0	18000.0	651.7	564.5	13.4	623.1	4.4	572.4	3.0	18000.0	637.8	534.5	16.2	534.5	16.2
CB4	17.0	3.0	18000.0	613.0	557.4	9.1	609.2	0.6	121.6	3.0	18000.0	689.7	561.9	18.5	654.1	5.2	18000.0	3.0	18000.0	681.6	572.2	16.1	572.2	16.1
CB5	30.4	2.0	2809.0	623.6	623.5	0.0	623.5	0.0	175.8	3.0	18000.0	615.5	553.9	10.0	601.9	2.2	18000.0	3.0	18000.0	690.5	498.2	27.8	498.2	27.8
CC1	18000.0	2.0	18000.0	658.9	436.8	33.7	436.8	33.7	18000.0	3.0	18000.0	713.4	418.0	41.4	503.3	29.5	18000.0	3.0	18000.0	685.4	407.1	40.6	407.1	40.6
CC2	18000.0	3.0	18000.0	695.8	381.1	45.2	381.1	45.2	18000.0	3.0	18000.0	647.5	466.0	28.0	466.0	28.0	18000.0	3.0	18000.0	611.9	399.4	34.7	399.4	34.7
CC3	18000.0	3.0	18000.0	651.8	336.6	48.4	450.8	30.8	18000.0	3.0	18000.0	671.3	436.5	35.0	436.5	35.0	18000.0	3.0	18000.0	670.0	460.4	31.3	460.4	31.3
CC4	18000.0	3.0	18000.0	599.9	397.9	33.7	442.0	26.3	18000.0	3.0	18000.0	627.5	440.9	29.7	453.8	27.7	18000.0	3.0	18000.0	682.1	431.6	36.7	431.6	36.7
CC5	18000.0	3.0	18000.0	651.0	381.7	41.4	432.7	33.5	18000.0	3.0	18000.0	595.3	418.3	29.7	418.3	29.7	18000.0	2.0	18000.0	628.4	340.6	45.8	340.6	45.8
CD1	47.4	2.0	977.0	619.7	619.7	0.0	619.7	0.0	17.8	3.0	18000.0	688.7	624.0	9.4	671.2	2.5	19.2	3.0	18000.0	656.4	522.4	20.4	624.5	4.9
CD2	31.3	3.0	15046.2	628.3	628.2	0.0	628.2	0.0	1039.3	3.0	18000.0	647.4	548.5	15.3	643.9	0.5	16.7	3.0	18000.0	635.0	538.4	15.2	612.7	3.5
CD3	315.3	3.0	12249.1	648.6	648.6	0.0	648.6	0.0	1143.2	3.0	18000.0	614.0	575.6	6.3	606.5	1.2	28.0	3.0	6316.7	590.1	590.0	0.0	590.0	0.0
CD4	31.2	3.0	516.0	583.9	583.9	0.0	583.9	0.0	69.0	3.0	18000.0	664.5	564.1	15.1	661.7	0.4	349.8	3.0	18000.0	673.0	570.1	15.3	630.1	6.4
CD5	12.4	3.0	18000.0	647.9	595.9	8.0	641.9	0.9	41.8	3.0	18000.0	640.6	539.0	15.9	630.5	1.6	312.3	3.0	18000.0	647.4	487.8	24.6	487.8	24.6

It is easy to observe from the results that CPLEX could solve most instances of 15 customers with 3, 4, and 5 rendezvous nodes. 3/6 instances cannot be solved in

5 hours for the primary/secondary objective. For 30 customers instances, 17/60 instances cannot be solved within the specified time. While the secondary objective of most 30 customer instances cannot be solved within 5 hours, especially for 4-30 and 5-30 rendezvous nodes-customer cases. In general, data suggest that the increase in the number of rendezvous nodes increases the difficulty of the problem for the primary and secondary objective.

In addition, The Egap is very lower for most 15 customers instances. For instances with larger Cgap, The value of Egap decreased significantly compared with the value of Cgap. Such as instance CC1-4-15, CC3-4-15, CC4-4-15, CC2-5-15, CC3-5-15, and CC5-5-15. For 30 customers instances, the value of Egap is lower than that of the Cgap for a lot of instances. In other words, the column generation lower bound is better than the CPLEX lower bound, and the gap improvement is significant for some instances. However, there still are instances with big Egap, which also reflects the difficulty of solving the problem.

3.4.3 Hybrid metaheuristic experiment

We tested 60 instances of 15 customer nodes with 3/4/5 rendezvous nodes. The hybrid metaheuristic results were compared against the CPLEX results to evaluate the multi-start heuristic, iteration number, different moves, and perturbations performances in the hybrid metaheuristic. After preliminary experiments, we selected an initial set of algorithmic components to investigate the contribution of the different algorithmic components. In which $IterN1_{max} = 5$, $IterN2_{max} = 10$, $IterN3_{max} = 50$ and the two-start heuristic contains the *optimal* and *nearest* drop-off nodes selection strategy in Algorithm 3 are chosen. Furthermore, the four moves involve swap, insertion, 2-opt, and change-satellites, as well as the two perturbations *Change-Multisatellites*, *Destroy-Repair-Reconstruction*, are used.

In Tables 3.4-3.7, the AT(s) row is the average runtime for the hybrid metaheuristic, the gap0(%) row represents the gap between the average hybrid metaheuristic result and the CPLEX upper bound (baseline) for the secondary objective value. The average gap0 and runtime were obtained by solving each instance 10 times.

Based on the performance evaluation of the different algorithmic components, we designed an efficient hybrid metaheuristic. We compared the results obtained using the hybrid metaheuristic against the results with CPLEX to see how the algorithm performs: See Section 3.4.3.

Multi-start, iteration number, moves and perturbations performance

First, we investigated whether the multi-start procedure performs better than the one-start procedure. Section 3.3.1 provided optimal and nearest drop-off node selection strategies for construction heuristics. Here we evaluate the performance of one-start (optimal/nearest) and two-start strategies at a fixed runtime and see how they perform. Note that the one-start or two-start strategy are both adopted in the construction heuristic and reconstruction.

The two-start and one-start strategies were compared with the runtime of the hybrid metaheuristic set to 70s. The results are shown in Table 3.4. The table quickly shows that the gap0 of the two-start strategy at gap0 1.655% is better than both one-start strategies with gap0 1.781% and gap0 2.143%. We suspect that in cases where the runtime is predetermined, presetting two distinct initial solutions make the search space larger. Note that the drop-off node selection strategies used in the multi-start procedure are also involved in the reconstruction phase of perturbation.

TABLE 3.4: Comparison of multi-start and one-start strategies

	Optimal-Nearest	Optimal	Nearest
gap0(%)	1.655	1.781	2.143

Second, we investigated how the quality of the solution changes as number of $IterN3_{max}$ increases. The experimental results are reported in Table 3.5, where row 1 represents the iteration number from 25 to 400. It is easy to observe from Table 3.5 that as the iteration number increases, the gap0 is trends downward while the AT is on an approximately linear upward trend. It seems that the algorithm can give a fairly good solution as long as it allowed for longer computation times.

The gap0 decreased significantly from 3.259% at iteration number 25 down to 2.050% at iteration number 50. From iteration number 50 to 200, the gap0 fell by nearly 0.9% as runtime increased from 47.186s to 185.061s. From 200 to 400 iterations, the gap0 slowly declined from 1.189% to 0.844% as runtime increased rapidly from 185.016s to 362.556s. We can choose the appropriate $IterN3_{max}$ based on the runtime limit and accuracy requirements.

TABLE 3.5: Sensitivity analysis on iteration number

	25	50	75	100	200	300	400
gap0(%)	3.259	2.050	1.732	1.527	1.189	1.031	0.844
AT(s)	25.012	47.186	69.984	94.633	185.016	277.287	362.556

Third, we compared the performances of the moves. Table 3.6 reports the results of our sensitivity analysis on the moves. Row 1 indicates the different combinations of moves in which the basemoves contain the swap, insertion, 2-opt, and change-satellites, and the next four columns represent the combinations of the 3 moves respectively. From the simulation results, we know that the change-satellites operator is the most time-consuming one. If we remove the change-satellites operator, the average runtime reduces to 11.572s compared with 47.186s for the basemoves. However, without the change-satellites operator, the gap0 drops rapidly from 2.050% to 5.189%. Removing the swap or insertion operator reduces the quality of the solution to a similar extent as removing the change-satellites operator, but less time gets saved. Overall, removing swap, insertion or change-satellites increases the gap0 to over 5%, which makes the quality of the solution challenging to accept.

The 2-opt operator had almost no effect on the quality of the solution, giving a gap0 of 1.978% versus 2.050% for basemoves. Also, removing the 2-opt operation did not significantly reduce the solution time. The cause for this phenomenon may come from our instances as there were fewer than 5 customer nodes visited in each robot route, which limited the space for the 2-opt operation. We keep this move and hope it could be useful to the problem where each robot route contains a more significant number of customer points.

TABLE 3.6: Sensitivity analysis on the moves

	basemoves	no swap	no insertion	no 2-opt	no change-satellites
gap0(%)	2.050	6.315	5.430	1.978	5.189
AT(s)	47.186	34.167	41.007	46.843	11.572

Fourth, we analyzed the quality of each perturbation. Table 3.7 reports our sensitivity analysis on perturbations. Experiments kept the $IterN3_{max} = 50$ unchanged then employed the different perturbations combinations shown in row 1. CMS is for *Change-Multisatellites*, CR is for *Change-RobotRoute* and DRR if for *Destroy-Repair-Reconstruction*.

Table 3.7 shows that DRR has the best gap0 compared to CMS and CR. CR performed the worst but also had the shortest runtime. We also found that using the CMS-DRR combination gave the best quality of solution, with a 2.050% gap0 among all combinations. Compared to several other well-behaved perturbation combinations, like CMS-CR-DRR with 2.523% gap0 or DRR with 2.668% gap0, CMS-DRR gave better performance at a comparably similar runtime. We thus removed the CR perturbation from the final hybrid metaheuristic.

TABLE 3.7: Sensitivity analysis on the perturbations

	CMS-CR-DRR	CMS-CR	CR-DRR	CMS-DRR	CMS	CR	DRR
gap0(%)	2.523	4.766	2.937	2.050	5.832	8.755	2.668
AT(s)	44.065	39.645	39.262	47.186	46.625	28.959	50.842

Hybrid metaheuristic results

The hybrid metaheuristic was run for all the instances of 15/30/50/100 customer nodes. We set $IterN3_{max} = 200$ for instances with 15/30/50 customers as it guarantees good-quality solutions at acceptable runtimes. For instances with 100 customers, we set $IterN3_{max} = 50$ to ensure that the average runtime of the hardest instances with 5 rendezvous nodes and 100 customers can be solved in 1 hour.

The 120 instances with 15/30 customers were compared to the results obtained by CPLEX. We found that the hybrid metaheuristic has an acceptable runtime and always reaches optimal solutions (when known) computed using CPLEX.

Tables 3.8-3.11 report the results for the hybrid metaheuristic. BK and AK are the best and average primary objective value. BC and AC are the best and average secondary objective value. SDK and SDC represent the standard deviation of the primary and secondary objective value. AT(s) is the average runtime of the hybrid metaheuristic. Tables 3.8-3.9 also feature gap1(%) and gap2(%), which are the gap between best hybrid metaheuristic result and CPLEX upper bound (baseline) and the gap between average hybrid metaheuristic result and CPLEX upper bound (baseline) for secondary objective value, respectively. Besides, the gap3(%) and gap4(%) are the gap between best hybrid metaheuristic result and enhanced lower bound (baseline) and the gap between average hybrid metaheuristic result and enhanced lower bound (baseline) for secondary objective value, respectively.

For the instances CB5-3-15, CB5-4-15, and CB1-5-15, the best/average primary objective values obtained from the hybrid metaheuristic are different from the CPLEX values, and so this data was eliminated when calculating the average gap1, gap2, gap3, and gap4. For clarity, we put a star behind the value where there is a difference between CPLEX and our algorithm in the primary objective. All gap1, gap2, gap3, and gap4 values ≤ 0 are highlighted in bold (except those with a star). In addition, the instances with Egap(%) larger than 10% in Tables 3.2-3.3 are highlighted in italics in Tables 3.8-3.9.

First, we compared the hybrid metaheuristic results of small and medium-sized instances to CPLEX results. Tables 3.8-3.9 report comparative results. For the primary objective value, the hybrid metaheuristic performs almost as well as the CPLEX results (in Tables 3.2-3.3). Two CPLEX results are better (CB5-3-15 and CB1-5-15), while other

values of the 118 instances are the same. Note that for these two instances with different primary objective values, we will eliminate them when comparing the secondary objective value since such comparisons do not make sense. For why the better secondary objective solutions are derived by our hybrid metaheuristic for these two instances, we think increasing the number of vans used may lead the total travel distances of the 2E-VRRP model decrease in some cases.

For the secondary objective value, more than two thirds of the hybrid metaheuristic's best results are the same as or even better than the CPLEX upper bounds (gap1 $\leq$ 0). Furthermore, the average gap2 is below 0.9% for 15/30 customers instances. Note that for the instances with 5 rendezvous nodes and 30 customers, the hybrid metaheuristic performs much better than CPLEX upper bound.

Besides, more than two-thirds of the hybrid metaheuristic's best results are the same as the enhanced lower bound (gap3 $=$ 0), and the average gap4 is nearly 1.43% for 15 customers instances. For 30 customers instances, some best results of hybrid metaheuristic have a larger gap with the enhanced lower bounds. The main reason may be the enhanced lower bounds we got are still weak, and the enhanced lower bound may have a larger gap with the optimal value. If we remove the instances of 30 customers with Egap $>$ 10% (highlighted in italics in Tables 3.9), the average gap3 and gap4 are acceptable, 2.2% and 4.2% respectively.

The results for 50 and 100 customers are shown in Tables 3.10-3.11, which show that as the number of rendezvous nodes increases, the average secondary objective value trends downward. Hence, a reasonable increase in rendezvous nodes is useful for saving transportation costs.

For all the instances, category CC instances with wide time windows had the longest runtimes. We suspect that the wider time windows result in a larger solution space, and our hybrid metaheuristic contains backtracking algorithms that need an exact search of the solution space, so category CC instances cause long runtimes.

3.4.4 Sensitivity analysis on related speed

As speed of the vehicles may affect the model-computed outputs, we analyzed the impact of vehicle speed changes on the results.

The speed of robots is commonly around walking speed (5km/h) in real-world applications. For the robot, a speed faster than 10km/h seems to be unrealistic in pedestrianized and high density urban areas. Hence, we assume the speed of the robot to be between 5 and 10 km/h in our experiments. Besides, the van speed can be assumed to be usually less than 30km/h as imposed by speed limits in city centers. We assume the speed of vans to be between 15 and 25 km/h in our experiments.

TABLE 3.8: Hybrid metaheuristic for 15 customer instances (Iteration 200)

	3-15											4-15										5-15									
	BK	BC	AK	AC	SDK	SDC	AT	gap1	gap2	gap3	gap4	BK	BC	AK	AC	SDK	SDC	AT	gap1	gap2	gap3	gap4	BK	BC	AK	AC	SDK	SDC	AT	gap1	gap2	gap3	gap4
CA1	2.0	383.9	2.0	383.9	0.0	0.1	79.7	**0.0**	**0.0**	**0.0**	**0.0**	2.0	411.3	2.0	413.0	0.0	1.9	126.6	**0.0**	0.4	**0.0**	0.4	2.0	403.7	2.0	405.8	0.0	1.8	202.9	**0.0**	0.5	**0.0**	0.5
CA2	2.0	378.5	2.0	378.5	0.0	0.0	125.3	**0.0**	**0.0**	**0.0**	**0.0**	2.0	408.1	2.0	415.1	0.0	4.9	197.5	0.1	1.7	0.1	1.8	2.0	420.5	2.0	420.5	0.0	0.2	314.1	**0.0**	**0.0**	**0.0**	**0.0**
CA3	2.0	390.8	2.0	391.4	0.0	1.2	54.7	0.8	0.9	0.8	0.9	2.0	446.8	2.0	447.7	0.0	1.0	81.5	0.5	0.7	0.5	0.7	2.0	360.2	2.0	360.4	0.0	0.3	94.9	**0.0**	0.1	**0.0**	0.1
CA4	2.0	400.9	2.0	407.5	0.0	7.8	75.3	**0.0**	1.6	**0.0**	1.6	1.0	397.8	1.0	399.9	0.0	1.7	107.2	**0.0**	0.5	**0.0**	0.5	2.0	346.8	2.0	347.6	0.0	0.4	99.6	0.3	0.5	0.3	0.5
CA5	2.0	382.8	2.0	383.1	0.0	0.4	53.7	**0.0**	0.1	**0.0**	0.1	1.0	309.8	1.0	311.8	0.0	2.4	56.6	0.4	1.0	0.4	1.0	2.0	386.7	2.0	386.7	0.0	0.1	158.7	**0.0**	**0.0**	**0.0**	**0.0**
CB1	2.0	391.8	2.0	392.2	0.0	0.2	75.3	**0.0**	0.1	**0.0**	0.1	2.0	414.5	2.0	424.2	0.0	14.6	178.5	**0.0**	2.3	**0.0**	2.3	2.0*	386.2	2.0*	400.0	0.0	9.3	106.2	-37.2	-32.5	-37.2	-32.5
CB2	2.0	410.4	2.0	422.1	0.0	6.2	53.5	**0.0**	2.8	**0.0**	2.8	2.0	446.9	2.0	450.8	0.0	7.6	140.9	**0.0**	0.8	**0.0**	0.8	2.0	406.9	2.0	414.6	0.0	5.6	170.9	**0.0**	1.9	**0.0**	1.9
CB3	2.0	448.4	2.0	454.1	0.0	3.4	76.7	**0.0**	1.2	**0.0**	1.3	2.0	438.5	2.0	453.2	0.0	10.9	198.4	**0.0**	3.2	**0.0**	3.2	2.0	366.2	2.0	366.2	0.0	0.0	72.5	**0.0**	**0.0**	**0.0**	**0.0**
CB4	2.0	377.6	2.0	384.7	0.0	4.2	75.9	**0.0**	1.9	**0.0**	1.9	2.0	393.6	2.0	401.4	0.0	9.0	112.8	**0.0**	1.9	**0.0**	1.9	1.0	335.9	1.0	349.8	0.0	6.0	151.3	0.2	4.2	0.2	4.2
CB5	2.0*	355.5	2.0*	355.5	0.0	0.0	58.8	-14.8	-14.8	-14.8	-14.8	1.0	361.6	1.9*	351.8	0.3	3.5	104.5	**0.0**	-2.8	**0.0**	-2.8	2.0	396.6	2.0	398.7	0.0	2.2	201.4	**0.0**	0.5	**0.0**	0.5
CC1	2.0	361.9	2.0	362.5	0.0	1.3	150.5	**0.0**	0.2	**0.0**	0.2	2.0	419.9	2.0	425.8	0.0	3.3	360.2	**0.0**	1.4	1.0	2.4	1.0	340.3	1.0	340.3	0.0	0.0	875.5	**0.0**	**0.0**	**0.0**	**0.0**
CC2	2.0	345.8	2.0	346.9	0.0	0.4	223.8	0.3	0.6	0.3	0.6	2.0	396.1	2.0	407.0	0.0	8.4	360.2	1.2	3.8	1.2	3.8	1.0	359.8	1.0	365.2	0.0	3.0	634.3	**0.0**	1.5	2.5	4.0
CC3	2.0	380.9	2.0	390.8	0.0	5.9	133.8	**0.0**	2.5	**0.0**	2.5	2.0	435.9	2.0	448.5	0.0	6.6	340.3	**0.0**	2.8	3.7	6.4	2.0	354.5	2.0	354.5	0.0	0.0	267.5	**0.0**	**0.0**	4.7	4.7
CC4	2.0	379.3	2.0	380.7	0.0	3.4	147.2	**0.0**	0.4	**0.0**	0.4	2.0	371.6	2.0	374.1	0.0	7.9	289.8	**-1.6**	**-0.9**	1.9	2.6	2.0	348.0	2.0	349.5	0.0	0.8	474.2	0.2	0.6	0.2	0.6
CC5	1.0	286.5	1.0	286.5	0.0	0.0	150.5	**0.0**	**0.0**	**0.0**	**0.0**	1.0	301.4	1.0	308.4	0.0	3.2	253.9	**0.0**	2.3	**0.0**	2.3	2.0	392.8	2.0	397.2	0.0	4.7	541.3	**0.0**	1.1	4.7	5.7
CD1	2.0	370.8	2.0	371.9	0.0	1.7	99.2	**0.0**	0.3	**0.0**	0.3	2.0	408.4	2.0	408.8	0.0	1.3	213.5	**0.0**	0.1	**0.0**	0.1	2.0	385.4	2.0	392.6	0.0	11.1	319.1	**0.0**	1.8	**0.0**	1.8
CD2	1.0	348.8	1.0	362.3	0.0	9.3	68.1	**0.0**	3.7	**0.0**	3.7	2.0	402.3	2.0	404.3	0.0	2.6	148.4	1.3	1.8	1.3	1.8	1.0	384.2	1.0	387.6	0.0	4.8	217.4	**0.0**	0.9	**0.0**	0.9
CD3	1.0	386.9	1.0	398.9	0.0	7.7	49.5	1.2	4.2	1.2	4.2	2.0	451.2	2.0	452.7	0.0	1.6	166.6	**0.0**	0.3	**0.0**	0.3	2.0	351.2	2.0	351.2	0.0	0.0	160.2	**0.0**	**0.0**	**0.0**	**0.0**
CD4	2.0	381.6	2.0	385.6	0.0	8.5	110.3	**0.0**	1.0	**0.0**	1.0	2.0	355.4	2.0	355.4	0.0	0.0	187.3	**0.0**	**0.0**	**0.0**	**0.0**	2.0	339.5	2.0	342.8	0.0	1.2	163.3	**0.0**	1.0	**0.0**	1.0
CD5	2.0	368.9	2.0	369.6	0.0	2.2	57.2	**0.0**	0.2	**0.0**	0.2	1.0	301.4	1.0	308.1	0.0	2.7	246.6	**0.0**	2.2	**0.0**	2.2	2.0	404.3	2.0	406.5	0.0	2.6	275.8	**0.0**	0.6	**0.0**	0.6
Aver	1.9	376.6	1.9	380.4	0.0	3.2	95.9	0.1	1.1	0.1	1.1	1.8	393.6	1.8	398.1	0.0	4.8	193.6	0.1	1.4	0.5	1.8	1.8	373.5	1.8	376.9	0.0	2.7	275.1	0.0	0.8	0.7	1.4

TABLE 3.9: Hybrid metaheuristic for 30 customer instances (Iteration 200)

	3-30											4-30										5-30									
	BK	BC	AK	AC	SDK	SDC	AT	gap1	gap2	gap3	gap4	BK	BC	AK	AC	SDK	SDC	AT	gap1	gap2	gap3	gap4	BK	BC	AK	AC	SDK	SDC	AT	gap1	gap2	gap3	gap4
CA1	3.0	760.8	3.0	776.6	0.0	8.9	199.8	0.6	2.6	1.1	3.1	3.0	705.7	3.0	711.1	0.0	3.7	363.2	0.5	1.2	1.7	2.4	3.0	632.5	3.0	658.5	0.0	17.9	563.6	1.3	5.2	4.8	8.6
CA2	3.0	656.7	3.0	666.3	0.0	4.6	222.5	0.5	2.0	0.5	2.0	3.0	686.5	3.0	696.0	0.0	10.7	421.2	**-0.1**	1.3	1.4	2.8	3.0	640.7	3.0	651.6	0.0	10.6	583.3	0.8	2.4	2.5	4.1
CA3	3.0	688.5	3.0	697.5	0.0	5.9	267.4	**0.0**	1.3	1.2	2.5	3.0	653.2	3.0	668.9	0.0	10.6	385.9	0.6	3.0	2.1	4.4	3.0	602.6	3.0	604.8	0.0	2.3	596.6	**0.0**	0.4	**0.0**	0.4
CA4	3.0	585.6	3.0	596.6	0.0	9.4	198.6	1.1	2.9	1.1	2.9	3.0	684.8	3.0	702.3	0.0	10.7	307.0	**-0.6**	1.9	1.2	3.7	3.0	664.1	3.0	674.1	0.0	8.1	368.4	*0.3*	*1.8*	*12.9*	*14.2*
CA5	3.0	659.9	3.0	672.7	0.0	8.8	261.0	1.5	3.4	1.5	3.4	3.0	586.9	3.0	616.4	0.0	18.6	405.6	0.3	5.1	0.3	5.1	3.0	648.4	3.0	663.0	0.0	15.4	522.7	***-0.1***	*2.0*	*24.7*	*26.4*
CB1	3.0	768.8	3.0	778.0	0.0	5.0	189.9	**0.0**	1.2	1.4	2.5	3.0	720.0	3.0	726.6	0.0	8.1	389.3	0.9	1.8	6.5	7.4	3.0	646.8	3.0	660.5	0.0	8.8	1095.6	*0.4*	*2.5*	*10.3*	*12.2*
CB2	3.0	642.5	3.0	662.4	0.0	11.4	175.7	1.6	4.6	1.6	4.6	3.0	658.0	3.0	667.2	0.0	7.6	263.7	1.6	3.0	2.3	3.6	3.0	642.9	3.0	654.9	0.0	6.2	748.6	0.5	2.3	8.5	10.2
CB3	3.0	698.3	3.0	704.4	0.0	6.1	250.6	1.0	1.9	2.6	3.5	3.0	645.9	3.0	664.7	0.0	12.5	467.3	**-0.9**	2.0	3.5	6.3	3.0	630.9	3.0	640.4	0.0	6.8	489.5	***-1.1***	*0.4*	*15.3*	*16.5*
CB4	3.0	613.0	3.0	623.1	0.0	11.2	227.8	**0.0**	1.6	0.6	2.2	3.0	672.9	3.0	690.4	0.0	10.0	351.7	**-2.5**	0.1	2.8	5.3	3.0	683.5	3.0	692.2	0.0	5.8	278.3	*0.3*	*1.5*	*16.3*	*17.3*
CB5	2.0	651.1	2.0	670.4	0.0	12.4	100.4	4.2	7.0	4.2	7.0	3.0	612.6	3.0	632.8	0.0	13.2	479.3	**-0.5**	2.7	1.7	4.9	3.0	673.0	3.0	685.6	0.0	12.4	1158.0	***-2.6***	***-0.7***	*26.0*	*27.3*
CC1	2.0	615.9	2.0	631.6	0.0	14.0	234.2	***-7.0***	***-4.3***	*29.1*	*30.8*	3.0	648.8	3.0	658.2	0.0	10.6	822.2	***-10.0***	***-8.4***	*22.4*	*23.5*	3.0	623.4	3.0	635.8	0.0	16.6	3632.6	***-9.9***	***-7.8***	*34.7*	*36.0*
CC2	3.0	624.3	3.0	632.1	0.0	7.4	502.0	***-11.5***	***-10.1***	*39.0*	*39.7*	3.0	645.0	3.0	651.3	0.0	5.7	885.5	***-0.4***	*0.6*	*27.8*	*28.5*	3.0	605.9	3.0	622.0	0.0	14.6	1575.5	***-1.0***	*1.6*	*34.1*	*35.8*
CC3	3.0	640.3	3.0	650.3	0.0	10.4	579.0	***-1.8***	***-0.2***	*29.6*	*30.7*	3.0	616.1	3.0	630.3	0.0	13.6	777.5	***-8.9***	***-6.5***	*29.2*	*30.7*	3.0	624.3	3.0	636.0	0.0	8.0	1199.9	***-7.3***	***-5.3***	*26.2*	*27.6*
CC4	3.0	568.7	3.0	574.3	0.0	6.0	438.8	***-5.5***	***-4.5***	*22.3*	*23.0*	3.0	614.9	3.0	628.2	0.0	9.4	943.3	***-2.0***	*0.1*	*26.2*	*27.8*	3.0	647.4	3.0	653.4	0.0	4.2	1175.1	***-5.4***	***-4.4***	*33.3*	*34.0*
CC5	3.0	633.1	3.0	639.9	0.0	7.1	469.0	***-2.8***	***-1.7***	*31.7*	*32.4*	3.0	573.7	3.0	588.1	0.0	8.2	932.0	***-3.8***	***-1.2***	*27.1*	*28.9*	2.0	546.9	2.0	558.2	0.0	8.6	2279.1	***-14.9***	***-12.6***	*37.7*	*39.0*
CD1	2.0	632.3	2.0	670.2	0.0	18.7	133.7	2.0	7.5	2.0	7.5	3.0	692.5	3.0	699.4	0.0	7.5	553.2	0.5	1.5	3.1	4.0	3.0	651.4	3.0	661.4	0.0	9.3	2194.6	**-0.8**	0.8	4.1	5.6
CD2	3.0	636.2	3.0	645.7	0.0	7.3	332.4	1.2	2.7	1.3	2.7	3.0	646.4	3.0	661.0	0.0	11.5	516.5	**-0.2**	2.1	0.4	2.6	3.0	634.8	3.0	647.1	0.0	7.8	939.4	**0.0**	1.9	3.5	5.3
CD3	3.0	650.3	3.0	667.7	0.0	11.1	380.2	0.3	2.9	0.3	2.9	3.0	621.8	3.0	634.2	0.0	9.6	557.7	1.2	3.2	2.5	4.4	3.0	598.6	3.0	608.4	0.0	9.6	1021.7	1.4	3.0	1.4	3.0
CD4	3.0	583.9	3.0	597.7	0.0	12.7	367.3	**0.0**	2.3	**0.0**	2.3	3.0	667.2	3.0	684.0	0.0	18.5	567.2	0.4	2.9	0.8	3.3	3.0	666.8	3.0	673.2	0.0	2.9	584.0	**-0.9**	**0.0**	5.5	6.4
CD5	3.0	648.4	3.0	661.3	0.0	13.3	338.6	0.1	2.0	1.0	2.9	3.0	637.0	3.0	645.6	0.0	8.7	783.8	**-0.6**	0.8	1.0	2.4	3.0	621.9	3.0	645.6	0.0	11.9	928.6	***-4.1***	***-0.3***	*21.6*	*24.4*
Aver	2.9	647.9	2.9	660.9	0.0	9.6	293.4	-0.7	1.2	8.6	10.4	3.0	649.5	3.0	662.8	0.0	10.4	558.7	-1.2	0.9	8.2	10.1	3.0	634.3	3.0	646.3	0.0	9.4	1096.8	-2.2	-0.3	16.2	17.7

TABLE 3.10: Hybrid metaheuristic for 50 customer instances (Iteration 200)

	3-50							4-50							5-50						
	BK	BC	AK	AC	SDK	SDC	AT(S)	BK	BC	AK	AC	SDK	SDC	AT(S)	BK	BC	AK	AC	SDK	SDC	AT(S)
CA1	4.0	1029.5	4.0	1057.6	0.0	19.4	587.4	4.0	965.6	4.0	984.3	0.0	12.3	1214.1	4.0	926.5	4.0	961.7	0.0	35.7	1611.3
CA2	4.0	1066.8	4.0	1076.7	0.0	8.8	774.4	4.0	983.2	4.0	996.5	0.0	10.8	1959.4	4.0	913.7	4.0	930.3	0.0	11.6	1276.4
CA3	4.0	1012.7	4.0	1034.7	0.0	17.2	1064.6	4.0	969.6	4.0	982.9	0.0	7.2	1178.8	4.0	889.9	4.0	904.2	0.0	9.8	2520.1
CA4	4.0	993.2	4.0	1000.2	0.0	5.8	1028.6	4.0	840.5	4.0	854.8	0.0	13.1	1395.4	4.0	920.6	4.0	935.9	0.0	10.8	2452.0
CA5	4.0	952.4	4.0	964.3	0.0	7.5	881.8	4.0	943.8	4.0	964.1	0.0	21.7	1382.3	4.0	927.3	4.0	951.8	0.0	15.9	2609.6
CB1	4.0	994.0	4.0	1007.1	0.0	10.1	1068.1	4.0	965.7	4.0	1004.3	0.0	17.4	940.4	4.0	922.5	4.0	945.7	0.0	21.4	1451.0
CB2	4.0	1000.0	4.0	1016.0	0.0	12.6	791.2	4.0	965.7	4.0	1004.3	0.0	17.4	940.4	4.0	971.8	4.0	1006.9	0.0	28.0	952.8
CB3	4.0	1022.4	4.0	1053.9	0.0	23.1	742.0	4.0	986.6	4.0	1024.8	0.0	30.3	1114.2	4.0	1047.0	4.9	1044.2	0.3	11.8	1765.6
CB4	4.0	983.8	4.0	994.8	0.0	8.6	714.9	4.0	888.4	4.0	919.7	0.0	18.6	1060.1	4.0	896.1	4.0	945.5	0.0	29.2	1586.6
CB5	4.0	982.1	4.0	1003.6	0.0	13.7	1056.8	4.0	994.0	4.0	1020.0	0.0	19.3	1313.1	4.0	908.8	4.0	935.0	0.0	14.9	2689.2
CC1	4.0	940.8	4.0	946.6	0.0	5.2	2384.6	4.0	866.5	4.0	879.6	0.0	10.6	3386.5	4.0	853.7	4.0	876.3	0.0	14.0	5573.0
CC2	4.0	990.4	4.0	1003.5	0.0	12.1	2017.8	4.0	886.3	4.0	909.4	0.0	12.9	3278.9	4.0	847.2	4.0	879.2	0.0	23.3	7529.1
CC3	4.0	902.6	4.0	920.6	0.0	11.0	2071.6	4.0	898.4	4.0	909.5	0.0	9.0	3246.4	4.0	876.3	4.0	886.9	0.0	5.1	4812.3
CC4	4.0	941.4	4.0	947.3	0.0	4.8	1796.6	4.0	792.4	4.0	807.2	0.0	11.6	3420.2	4.0	901.6	4.2	920.9	0.4	17.0	3594.1
CC5	4.0	1001.6	4.0	1008.9	0.0	5.4	1861.8	5.0	996.9	5.0	1021.4	0.0	16.8	3449.4	4.0	889.2	4.0	911.3	0.0	23.5	5779.2
CD1	4.0	993.9	4.0	1006.3	0.0	11.0	1580.8	4.0	913.8	4.0	935.0	0.0	13.4	2391.0	4.0	871.3	4.0	889.8	0.0	13.5	3628.7
CD2	4.0	1067.8	4.0	1080.9	0.0	10.8	1029.8	4.0	943.4	4.0	951.7	0.0	6.4	1867.2	4.0	873.6	4.0	886.0	0.0	9.6	4995.0
CD3	4.0	960.3	4.0	980.2	0.0	17.9	1352.9	4.0	988.6	4.0	1007.8	0.0	14.3	1448.6	4.0	910.3	4.0	931.4	0.0	10.8	3166.7
CD4	4.0	977.4	4.0	993.9	0.0	13.3	1371.3	4.0	822.5	4.0	845.6	0.0	15.8	2018.1	4.0	863.7	4.0	889.8	0.0	20.1	3783.1
CD5	4.0	1011.0	4.0	1042.4	0.0	27.1	818.3	4.0	968.2	4.0	986.0	0.0	11.8	2091.0	4.0	901.9	4.0	926.0	0.0	14.5	3909.1
Aver	4.0	991.2	4.0	1007.0	0.0	12.3	1249.8	4.1	929.0	4.1	950.4	0.0	14.5	1954.8	4.0	905.6	4.1	927.9	0.0	17.0	3284.3

TABLE 3.11: Hybrid metaheuristic for 100 customer instances (Iteration 50)

	3-100							4-100							5-100						
	BK	BC	AK	AC	SDK	SDC	AT(S)	BK	BC	AK	AC	SDK	SDC	AT(S)	BK	BC	AK	AC	SDK	SDC	AT(S)
CA1	8.0	1925.3	8.0	1977.6	0.0	29.6	1234.0	8.0	1891.8	8.0	1927.7	0.0	29.9	1485.4	8.0	1812.6	8.0	1855.4	0.0	34.9	2296.4
CA2	8.0	1958.4	8.0	2033.9	0.0	61.1	1046.9	7.0	1810.2	7.0	1864.2	0.0	68.3	886.5	8.0	1699.8	8.0	1758.0	0.0	34.7	2018.1
CA3	8.0	1990.1	8.0	2035.5	0.0	29.8	1111.3	8.0	2069.1	8.0	2142.5	0.0	45.8	1377.7	8.0	1705.4	8.0	1777.1	0.0	39.1	1657.8
CA4	8.0	2315.4	8.0	2380.8	0.0	38.2	1077.1	7.0	1700.8	7.0	1764.3	0.0	35.1	1317.1	8.0	1798.1	8.0	1868.6	0.0	40.1	2266.2
CA5	8.0	1855.0	8.0	1916.0	0.0	33.6	1669.4	8.0	1817.0	8.0	1867.0	0.0	35.8	2476.8	8.0	1761.4	8.0	1842.0	0.0	46.3	2213.5
CB1	7.0	1852.0	7.0	1924.5	0.0	43.8	839.4	7.0	1899.4	7.0	1986.6	0.0	61.0	774.4	8.0	1823.6	8.0	1904.3	0.0	50.5	2129.1
CB2	8.0	1964.6	8.0	2027.9	0.0	45.2	1040.4	8.0	1892.5	8.0	1929.6	0.0	27.9	1607.0	8.0	1711.7	8.0	1797.2	0.0	43.5	1886.0
CB3	8.0	1970.2	8.0	2026.8	0.0	32.7	1156.2	8.0	2090.3	8.0	2158.7	0.0	40.6	1318.7	8.0	1767.4	8.0	1847.4	0.0	35.8	1854.0
CB4	8.0	2301.5	8.0	2365.4	0.0	38.2	828.2	8.0	2122.7	8.9	2058.1	0.3	42.4	1385.4	8.0	1960.6	8.4	2019.5	0.5	65.0	1466.6
CB5	8.0	1902.0	8.0	1941.2	0.0	27.0	1181.1	8.0	1890.6	8.0	1976.2	0.0	47.2	1144.2	8.0	1910.7	8.0	2047.3	0.0	98.5	1294.1
CC1	8.0	1847.9	8.0	1891.1	0.0	28.7	3557.8	8.0	1737.3	8.0	1813.6	0.0	38.1	3153.3	8.0	1695.2	8.0	1764.4	0.0	42.7	6271.8
CC2	8.0	1742.4	8.0	1780.9	0.0	30.2	2483.4	8.0	1771.7	8.0	1828.4	0.0	33.1	4161.9	8.0	1600.9	8.0	1641.2	0.0	31.6	4970.2
CC3	8.0	1875.0	8.0	1914.0	0.0	35.1	1994.1	7.0	1768.1	7.0	1849.7	0.0	37.0	5149.8	8.0	1677.1	8.0	1721.8	0.0	28.7	5113.4
CC4	8.0	2185.3	8.0	2237.0	0.0	31.4	2139.8	8.0	1741.4	8.0	1786.8	0.0	29.3	4083.3	8.0	1653.7	8.0	1728.1	0.0	35.9	5625.3
CC5	8.0	1810.1	8.0	1839.2	0.0	24.7	2439.4	8.0	1774.6	8.0	1822.2	0.0	27.7	3436.2	8.0	1720.4	8.0	1791.0	0.0	36.4	4613.2
CD1	8.0	1801.2	8.0	1846.5	0.0	35.6	1615.9	8.0	1885.6	8.0	1917.3	0.0	14.4	2173.0	8.0	3282.0	8.0	3347.5	0.0	52.6	1966.1
CD2	8.0	1927.9	8.1	2017.9	0.3	48.7	905.9	8.0	1767.5	8.0	1858.9	0.0	39.0	1825.0	8.0	1650.3	8.0	1696.1	0.0	38.0	3638.4
CD3	8.0	1931.2	8.0	1964.5	0.0	26.1	976.0	7.0	1851.5	7.0	1919.9	0.0	36.8	1673.8	8.0	1700.5	8.0	1762.6	0.0	50.3	3604.8
CD4	8.0	2201.7	8.0	2261.1	0.0	40.1	1019.2	8.0	1838.5	8.0	1905.2	0.0	33.7	1972.8	8.0	1742.0	8.0	1796.1	0.0	40.7	3604.1
CD5	8.0	1832.0	8.0	1901.4	0.0	48.8	1529.0	8.0	1827.6	8.0	1868.9	0.0	23.9	1841.5	8.0	1777.9	8.0	1841.2	0.0	38.2	3808.2
Aver	8.0	1959.5	8.0	2014.2	0.0	36.4	1492.2	7.8	1857.4	7.8	1912.3	0.0	37.4	2162.2	8.0	1822.6	8.0	1890.3	0.0	44.2	3114.9

We chose simple instances to test different van/robot speed combinations by CPLEX to see how vehicle speeds influence the results. We set 5km/h as the baseline speed.

For the CPLEX experiments, we generated the different van/robot speed combinations via the following step. First, we set the speed of robot equals to 5km/h. Then, we kept robot speed unchanged but gradually increased van speed from 15km/h, 20km/h, up to 25km/h. Next, we kept van speed unchanged but increased robot speed from 5km/h up to 10km/h.

We chose 15 simple instances with 3 rendezvous nodes and 15 customers to implement the CPLEX experiments. Tables 3.12 shows the instances used in column 1. For the primary objective value, all results for van/robot speed combinations were identical. The secondary objective results are reported in Table 3.12, where row 1 gives the different combinations of the van/robot speed. Bold-type shows that results are different from the previous column.

TABLE 3.12: Sensitivity analysis on related speed by CPLEX

3-15	(15km/h,5km/h)	(20km/h,5km/h)	(25km/h,5km/h)	(25km/h,10km/h)
CA1	383.86	383.86	383.86	**383.49**
CA2	378.54	378.54	378.54	378.54
CA3	385.91	385.91	385.91	385.91
CA4	400.92	400.92	400.92	400.92
CA5	382.76	382.76	382.76	**380.72**
CB1	391.76	391.76	391.76	391.76
CB2	363.70	363.70	363.70	**351.87**
CB3	412.51	412.51	412.51	**399.63**
CB4	377.55	377.55	377.55	377.55
CB5	408.05	408.05	408.05	**353.84**
CD1	370.84	370.84	370.84	**359.50**
CD2	348.82	348.82	348.82	348.82
CD3	382.10	382.10	382.10	382.10
CD4	381.56	381.56	381.56	381.56
CD5	368.87	368.87	368.87	**352.81**
Aver	382.52	382.52	382.52	**375.27**

In experiments with simple instances, we found that increasing the van speed from 15km/h to 25km/h has little effect on reducing the secondary objective function. However, increasing the robot speed helps reduce average costs. For example, increasing robot speed from 5km/h to 10km/h reduces the secondary objective value from 382.52 to 375.27. We assumed that the bottleneck of the 2E-VRRP delivery system may be caused by the speed of the robot.

Hence, we implemented an in-depth experimental analysis on the speed of robot. We conducted the hybrid metaheuristic to test all instances with 15 and 30 customers. The van/robot speed combination (25km/h,5km/h) was chosen as the control group. We fixed the speed of the van and let the speed of the robots be 6.25km/h, 7.5km/h,...,

10km/h, as the experimental groups. There are 120 instances, and every instance is run 10 times with *Iter3=50* for each type of van/robot speed combination.

For the primary objective, the best results in the experimental group are all equal to the control group results in all the experiments. This comes from two factors: First, the number of vans used depends on a variety of constraints, not only the speed of the vehicles but also on vehicle capacity constraints, maximum travel time constraints, and so on. As long as one type of constraint is limiting, the number of vans used cannot be reduced. Second, in our instances, the value of the primary objective is usually 1,2 or 3. The need to cause changes in vehicle numbers often requires a significant change in conditions.

Table 3.13 shows the sensitivity analysis on the secondary objective. Row 1 represents the different combinations of van/robot speed. Row 2 and row 3 are the average best results for the 15 customers and 30 customers instances, in which the control group result (baseline) is the average best secondary objective value and the quality of the experimental group solution is reported as an average percentage gap from the baseline.

Table 3.13 shows that as robot speed grows, the quality of solutions for the secondary objective value increases. However, the total improvement is limited. Increasing the speed of the robot from 5km/h to 10km/h reduces the cost by less than 0.6% of the cost for our 15 customers and 30 customers instances. We suspect that the improvement of the objective value is affected by many factors (constraints). Simply improving the speed of robot can improve the solution, but improvements remains limited. Also, the objective functions where no term is related to travel times on edges or the number of robots used, therefore, increasing the speed of robots can only make the time window constraints easier to be satisfied and has marginal effect on the objective values.

We therefore recommend to keep robot speeds rather low because of increased safety and a more convenient environment for pedestrians in practical implementations.

TABLE 3.13: Sensitivity analysis on speed with the hybrid metaheuristic

	baseline(25km/h,5km/h)	(25km/h,6.25km/h)	(25km/h,7.5km/h)	(25km/h,8.75km/h)	(25km/h,10km/h)
15 cust best cost	380.13	-0.23%	-0.34%	-0.50%	-0.58%
30 cust best cost	648.20	-0.21%	-0.42%	-0.50%	-0.50%

3.5 Conclusions

This chapter provides an efficient transportation delivery model for the logistics distribution of autonomous robots in the city. We investigate an innovative two-echelon urban delivery problem using autonomous robots that contains the time window constraint, capacity constraint and maximum travel time constraint. The van carries the

robots on the *1st-level route* and drops off and picks up them in the rendezvous nodes, while the robot handles customer service on the *2nd-level route*. In addition, our model allows each van carries multiple robots, and each robot can visit multiple customers during a trip. This innovation distribution model can eliminate a lot of real estates and manpower compared to the traditional two-echelon delivery model. Which provides a new choice for logistics enterprises.

We first define this problem and present its mixed linear integer programming formulation, and present a column generation procedure to try to get a better lower bound. Since CPLEX cannot solve medium-sized and large-scale problems, we introduce construction heuristics that can provide a nice upper bound for the CPLEX solver and quickly generate the feasible initial solutions. The construction heuristic first uses a modified nearest neighbor approach to construct multiple robot routes (*1st-level route*), then adopts a simple connection heuristic to construct multiple van routes (*1st-level route*) to get the final solution. In addition, a hybrid metaheuristic that involves multi-start, destroy, repair, iterative local search, and backtracking is presented to quickly solve the medium-sized and large-scale problems that CPLEX cannot address. And a performance evaluation of the different algorithmic components for hybrid metaheuristic are implemented. Besides, we used the CPLEX upper bound and enhanced lower bound to assess the hybrid metaheuristic for the small-sized problem. Assessment results show that our hybrid metaheuristic is competitive. Furthermore, we have provided a benchmark solution for the problem along with a sensitivity analysis for the van/robot speed combinations. In general, increasing the speed of the robot rather than the speed of van can reduce the objective function value in real-word applications. However, simply improving the speed of the robot leads to only limited reduction in cost. We therefore recommend to keep robot speeds rather low because of a more pedestrian friendly environment in practical implementations.

Chapter 4

Electric van-based robot deliveries with en-route charging

Abstract

We present a two-echelon electric van-based robot delivery system with en-route charging for last-mile delivery in logistics operations. Robots can visit areas with van access restrictions, such as pedestrianized areas or university campuses. The time during which electric vans are carrying robots can be used effectively to recharge the robots, thereby increasing the efficiency of distribution systems.

To model the proposed system, we present a mixed-integer program. We note that the energy transfer from a van to its robot needs time and will cause the available travel distance of a van to decrease and that of a robot to increase. Focusing on the new time-distance-energy trade-off problem, which increases the difficulty checking the feasibility of any given route, we further propose a greedy route evaluation approach and a linear programming-based route evaluation method. An adaptive large neighborhood search algorithm is presented for solving larger instances. A sensitivity analysis for vehicle charging modes, battery capacities, and charging rate shows that using en-route charging, while appropriately increasing battery capacity and charging rate can have useful effects on cost.

4.1 Background

Current research tends to use traditional vehicles with robots (mothership concept) for collaborative distribution. Chapter 3 presented a two-echelon vehicle routing problem with robots for *2nd-level route* delivery, the van carrying the robot along a *1st-level route* for dropping off and picking up at parking nodes. We note that the robot still needs to be recharged and replenished if it is to be used multiple times.

To take van-based robot delivery research further, we present electric van-based robot deliveries as prototypical problems, where vans and robots are all-electric vehicles. In addition, the time during which the vans are carrying the robots can be used

effectively to recharge the robots, thereby increasing the efficiency of the distribution system.

Current technological and research developments can also support our concept: (i) Automotive companies have begun to study mobile charging vans. For example, NIO, an electric car company in China, is courting Tesla owners with mobile charging stations inside electric vans (Fred, 2018). (ii) Mobile charging platforms have also been studied in robot research. For example, Mathew et al., 2015 and Yu et al., 2019 considered and tested joint planning of mobile recharging platforms and drone routes to enable the drone to successfully visit specified nodes. This prior work suggests that a van, stationary or moving, can act as a mobile charging station to charge its robots en-route.

Given that the battery accounts for about 40% of the price of electric vehicles, we focus here on fast-charging rather than battery swapping, since fast-charging technology can reduce a company's fixed costs. In addition, we use partial recharging (Keskin and Çatay, 2016) to improve distribution efficiency.

We describe here a two-echelon electric van-based robot delivery system with en-route charging (2E-VREC). Larger electric vans carry small electric robots along the *1st-level route*. The robots travel along the *2nd-level open route*. The van and robot can both serve customers directly, but some constrained customers can be visited only by the delivery robots. The van stops at parking nodes for dropping off and/or picking up its robot, and to recharge and replenish its robot if needed. The van can charge its onboard robot on the move or at nodes, and the van can be recharged at the parking nodes.

Our 2E-VREC model offers the following advantages over the vehicle routing problem with drones (VRPD) model (Otto et al., 2018): (i) in our 2E-VREC model, we extend the VRPD model to cover electric vehicle aspects (E-VRPD), (ii) the 2E-VREC model brings two new technologies (en-route charging and reverse charging) into the E-VRPD model, (iii) the 2E-VREC model makes full use of the time during which vans are carrying robots on the road to charge them, shortening the extra waiting time spent by robots at parking stations for recharging, and thereby improving the effectiveness of the distribution system (in other words, transmission of electric power between vehicles is carried out in the 2E-VREC model, and efficiency improved), and (iv) autonomous electric vehicles are generally safer than other VRPD delivery models in cities; for example, when a fast-flying drone stalls, it can cause serious accidents in a city, whereas for a slowly moving robot, accident risks from stalled vehicles are controllable (Yu et al., 2020).

In theoretical terms, the new model adds a new time-distance-energy trade-off to electric vehicle routing. In the classical electric vehicle routing problem (EVRP), the

decrease in a vehicle's energy is caused only by the increase in the distance the vehicle travels, and these two variables are usually negatively correlated. By contrast, in the 2E-VREC model, the decrease in the energy of a van can be caused by the increase in the distance the van has traveled, but also by the amount of energy it transfers to its robot. The energy transfer is negatively correlated with the available travel distance of a van, but positively correlated with the available travel distance of its robot (i.e. the energy transfer will cause the available travel distance of a van to decrease and that of a robot to increase).

In addition, energy transfer needs time. Generally speaking, charging a robot en-route can reduce extra waiting time for charging, but it reduces the available travel distance of the van. Whether to charge the robot en-route and how much energy to transfer is therefore a difficult choice. To the best of our knowledge, these characteristics have not been considered, modeled, and solved in research to date. The work reported here makes the following contributions:

(1) We describe a new two-echelon van-based robot routing problem with en-route charging, which has the potential to improve the efficiency of the distribution system.

(2) We introduce the 2E-VREC model and propose a mathematical formulation.

(3) We provide a high-accuracy heuristic for single route feasibility checking of the 2E-VREC problem with time-distance-energy trade-off.

(4) We propose an ALNS algorithm for the newly presented problem.

(5) We conduct a sensitivity analysis for vehicle charging modes, battery capacities, and charging rate.

In what follows, Section 4.2 describes the problem and model, Sections 4.3 and 4.4 deal with the route evaluation and meta-heuristic approaches, respectively, a computational study is presented in Section 4.5, and Section 4.6 concludes.

4.2 Problem description and formulation

4.2.1 Problem statement

In the real world, there are customers located in narrow streets, on campuses, or in other communities where the entry of vans is restricted. We therefore distinguish two kinds of customers. One kind can be visited by either the van or the robot: we call these customers van customers. The other kind can only be visited by the robot: we call these robot customers. This implies that the robot can serve all the customers, but the van can only serve some of the customers.

We consider a two-echelon routing problem with van/robot delivery. Vans or vans carrying robots move along a *1st-level route*, serve van customers, or drop off/pick up,

and replenish their robots at parking nodes. Robots handle customer services along *2nd-level open routes*: in other words, the robots need not return to the parking node from where they set out.

We assume there are specific parking nodes for the vehicles, used for the van to rendezvous, replenish and recharge its robot, rather than performing these operations at customer nodes. This setting is reasonable, especially for electric vehicles, since it is often impossible for vans/robots to be recharged at customer nodes, and parking nodes can be visited multiple times by the vans/robots.

For replenishing, the freight carried by the robot can be loaded by the van at parking nodes, while the freight carried by the van must be loaded at the beginning of the tours at the depot. There are capacities for both the van and the robot. We also allow a robot to visit multiple customers during a dispatch instead of only one customer, since the capacity for a robot is usually greater than that of a drone.

For recharging, we consider partial and en-route charging. Recharging takes time for both the van and the robot, depending on the recharging rate and level. The van can be recharged at parking nodes. The robot can be recharged by the van while moving or at parking nodes and at customer nodes if it is on board the van. In addition, there are battery capacities for both the van and the robot. We note that the battery capacity of the robot corresponds to its maximum individual travel distance. The van's battery not only provides its own power, but is also needed to recharge the robots. When the van charges the robot, the power of the van decreases, while the power of the robot increases.

We do not consider the case where the robot charges the van, nor the case where the robot replenishes the van. Nor do we consider the case where the van replenishes other vans. Here we assume that each customer node must be visited by exactly one van/robot once. Customer nodes and the depot have time windows. Our model allows costless waiting at all locations.

We make the following assumptions: (i) each van can only carry one robot, (ii) the robot cannot leave the depot to serve customers directly, (iii) a robot dropped off by a van must be picked up by the same van, and (iv) the operating time spent dropping off, picking up, replenishing and preparing for recharging is ignored.

An example illustrating the problem is presented in Figure 4.1. Triangles represent parking nodes, the square represents the depot, and circles correspond to robot customer nodes/van customer nodes. Solid lines correspond to *1st-level routes* and dotted lines to *2nd-level routes*.

In addition, we define five types of route in the 2E-VREC model for the sake of illustration. The van route is the *1st-level-route*. The robot route is the *2nd-level-route*. In other words, the robot route is the route where a robot moves under its own power. The route

along which the van carries its robot is termed the van-robot route, the route where the van moves independently is termed the independent-van route, and the route traveled by the robot is called the whole-robot route.

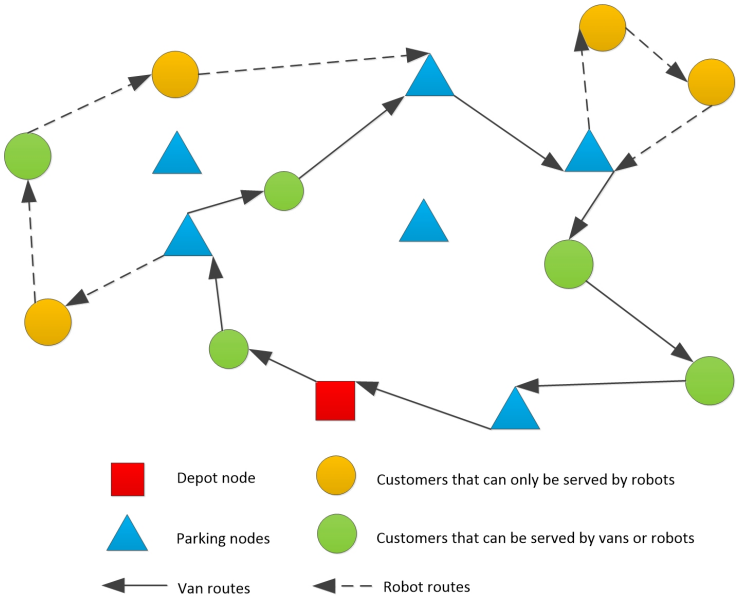

FIGURE 4.1: 2E-VREC model

4.2.2 Variable and Parameter Definitions

The problem is defined on a directed graph $G = (V, A)$, where the depot V_0 is represented by two nodes 0 and $0^{'}$. Every van route starts at 0 and ends at $0^{'}$. Let V_r be the set of parking nodes where vans can drop off, pick up, replenish and recharge their robots; the van can also be recharged at parking nodes. We note that a parking node may be visited more than once, we use dummy nodes in the model. V_{c1} denotes the van customer nodes set and V_c the customer nodes set. We have $V_{c1} \in V_c$, and $V_{c2} = V_c \setminus V_{c1}$ is the robot customer nodes set. The set $V_\alpha^0, V_\alpha^{'}, V_\alpha^{''}$ is the nodes set V_α union the depot 0, $0^{'}$, $0 \cup 0^{'}$ respectively, and α can be expressed in a variety of different combinations of customer nodes and parking nodes sets. Let $A_1 = \{(i,j) \mid i \in \{0\}; j \in V_{rc1}\} \cup \{(i,j) \mid i,j \in V_{rc1}, i \neq j\} \cup \{(i,j) \mid i \in V_{rc1}; j \in \{0^{'}\}\}$ be the *1st-level route* (van routes) and let $A_2 = \{(i,j) \mid i \in V_r; j \in V_c\} \cup \{(i,j) \mid i,j \in V_c, i \neq j\} \cup \{(i,j) \mid i \in V_c; j \in V_r\}$ be the *2nd-level route* (robot routes). We also let $A_3 = A_1 \cup A_2$ be the complete possible robot routes and $A_4 = A_3 \setminus A_1$ be the routes that the van cannot reach. For each edge, $d_{i,j}$ is the associated travel distance, while $h_1 * d_{i,j}$ ($h_2 * d_{i,j}$) is the associated travel cost and $d_{i,j}/v_1$ ($d_{i,j}/v_2$) is the associated travel time for the van (robot). The freight must be delivered from the depot $\{0\}$ to customer nodes i, with the demand d_i and serving time s_i. The time window of the customer nodes $i \in V_c$ is $[a_i, b_i]$, is the time interval that the

service at node i is permitted to start. Let $[a_0, b_0] = [a_{0'}, b_{0'}]$, where $a_0/a_{0'}$ represents the earliest possible departure time from the depot $0/0'$ and $b_0/b_{0'}$ is the latest possible arrival time at the depot $0/0'$. These time windows are hard. $F_T = \{1, 2, .., k, .., K\}$ presents the set of vans, where K is the number of vans. $k \in F_T$ represents the robot belonging to the kth van. Let M be an arbitrary large constant number. Let C_1 be the capacity for the van and robot and C_2 the capacity for the robot. Let G_1 and G_2 be the battery capacity for the van and robot respectively. Let g_1 and g_2 be the recharging rates of the van and the robot, and let h_1 and h_2 be the charge consumption rate of the van and the robot. In addition, we introduce the following decision variables.

Let $x_{i,j,k}$ be equal to 1 if arc (i, j) in A_1 is traveled by the $kth - van$, 0 otherwise.

Let $y_{i,j,k}$ be equal to 1 if arc (i, j) in A_3 is traveled by the $kth - robot$, 0 otherwise.

Let $Q_{i,j,k}$ be equal to 1 if arc (i, j) in A_1 is traveled by the $kth - van$ with its robot on board, 0 otherwise.

Let z_i^k represent the recharging amount of the robot at the parking node i, $i \in V_r$.

Let Z_i^k represent the recharging amount of of the van at the parking node i, $i \in V_r$.

Let $z_{i,j}^k$ represent the recharging amount of the robot by the van during arc (i, j), $(i, j) \in A_1$.

Let E_i^k and $e_i^{k,l}$ be the remaining battery level of the van and the robot at node i on arrival.

Let W_i^k and w_i^k be the arrival time for the van and robot at node i.

Let $p_{i,j,k}$ be the freight flow of the robot in arc (i, j) in A_2.

Let u_i^k be the dummy variable for node V_r.

To illustrate which route type is allowed and which is not allowed, we show the 2E-VREC route cases in Figure 4.2. For the customer nodes, cases (1)–(3) show that customers can only be reached and then left by van, robot, or van carrying robot, respectively. Cases (4)–(6) forbid customers being reached by both van and robot independently or left by both van and robot independently. For the parking nodes, case (7) allows a van to arrive and depart from a parking node. Case (8) forbids a robot from accessing the parking node independently unless there is a van also accessing this node – cases (9)–(12). For the depot, only the van carrying the robot can arrive and leave the depot – case (15) and case (19), while cases (13), (14), (16), (17), (18), (20) are forbidden.

4.2.3 Mixed Integer Programming Model

Electric van-based robot delivery with en-route charging problem can be modeled as follows.

Objective

$$\min\Big(\sum_{k\in F_T}\sum_{(i,j)\in A_1} h_1 * d_{i,j} * x_{i,j,k} + \sum_{k\in F_T}\sum_{(i,j)\in A_3} h_2 * d_{i,j} * y_{i,j,k} - \sum_{k\in F_T}\sum_{(i,j)\in A_1} h_2 * d_{i,j} * Q_{i,j,k}\Big) \tag{4.1}$$

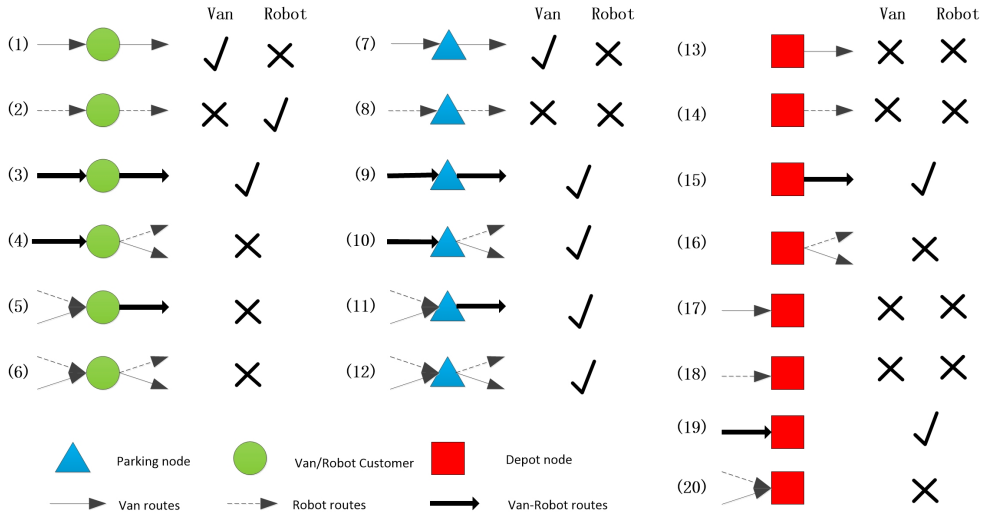

FIGURE 4.2: 2E-VREC route cases of permission and prohibition

Subject to
Arc constraints

$$\sum_{(i,j)\in A_1} x_{i,j,k} \leq 1, \forall j \in V_{rc1}, k \in F_T \tag{4.2}$$

$$\sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(j,i)\in A_1} x_{j,i,k} = 0, \forall j \in V_{rc1}, k \in F_T \tag{4.3}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} \leq 1, \forall j \in V_{rc}, k \in F_T \tag{4.4}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} - \sum_{(j,i)\in A_3} y_{j,i,k} = 0, \forall j \in V_{rc}, k \in F_T \tag{4.5}$$

$$\sum_{i\in V_{rc1}} x_{i,0',k} = \sum_{j\in V_{rc1}} x_{0,j,k} = \sum_{i\in V_{rc1}} y_{i,0',k} = \sum_{j\in V_{rc1}} y_{0,j,k} \leq 1, \forall k \in F_T \tag{4.6}$$

$$\sum_{k\in F_T} (\sum_{(i,j)\in A_3} y_{i,j,k} + \sum_{(i,j)\in A_3} x_{i,j,k} - \sum_{(i,j)\in A_3} Q_{i,j,k}) = 1, \forall j \in V_c \tag{4.7}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} + \sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(i,j)\in A_1} Q_{i,j,k} \leq 1, \forall i \in V^0_{c1}, k \in F_T \tag{4.8}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} + \sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(i,j)\in A_1} Q_{i,j,k} \leq 1, \forall j \in V^{'}_{c1}, k \in F_T \tag{4.9}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} \leq \sum_{(i,j)\in A_1} x_{i,j,k}, \forall i \in V_r, k \in F_T \tag{4.10}$$

$$2 * Q_{i,j,k} \leq x_{i,j,k} + y_{i,j,k} \leq 2 * Q_{i,j,k} + 1, \forall (i,j) \in A_1, k \in F_T \tag{4.11}$$

$$x_{i,j,k}, y_{i,j,k}, Q_{i,j,k} \in \{0,1\}, \forall (i,j) \in A_3, k \in F_T \tag{4.12}$$

$$Q_{i,j,k} = 0, \forall (i,j) \in A_4, k \in F_T \tag{4.13}$$

$$x_{i,j,k} = 0, \forall (i,j) \in A_4, k \in F_T \tag{4.14}$$

Time and energy constraints

$$max(W_i^k + Z_i^k/g_1, W_i^k + z_i^k/g_2, w_i^k + z_i^k/g_2) + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_r^0 | (i,j) \in A_1\}, k \in F_T \tag{4.15}$$

$$max(W_i^k, w_i^k) + z_i^k/g_2 + d_{i,j}/v_2 - w_j^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{i \in V_r | (i,j) \in A_2\}, k \in F_T \tag{4.16}$$

$$W_i^k + max(s_i, z_i^k/g_2) + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_{c1} | (i,j) \in A_1\}, k \in F_T \tag{4.17}$$

$$w_i^k + s_i + d_{i,j}/v_2 - w_j^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{i \in V_c | (i,j) \in A_2\}, k \in F_T \tag{4.18}$$

$$w_i^k - w_j^k \leq M(1 - y_{i,j,k}), \forall\{i \in V_{rc}^0 | (i,j) \in A_3\}, k \in F_T \tag{4.19}$$

$$|W_j^k - w_j^k| \leq M(1 - \sum_{(i,j)\in A_1} Q_{i,j,k}), \forall\{j \in V_{rc1}^{'}\}, k \in F_T \tag{4.20}$$

$$a_i \leq W_i^k, w_i^k \leq b_i, \forall i \in V_c^{''}, k \in F_T \tag{4.21}$$

$$E_j^k + h_1 * d_{i,j} - Z_i^k + z_i^k + z_{i,j}^k - E_i^k \leq M(1 - x_{i,j,k}), \forall\{(i,j) \in A_1 | i \in V_r^0\}, k \in F_T \tag{4.22}$$

$$E_j^k + h_1 * d_{i,j} + z_i^k + z_{i,j}^k - E_i^k \leq M(1 - x_{i,j,k}), \forall\{(i,j) \in A_1 | i \in V_{c1}\}, k \in F_T \tag{4.23}$$

$$e_j^k - z_i^k - z_{i,j}^k - e_i^k \leq M(1 - Q_{i,j,k}), \forall\{(i,j) \in A_1 | i \in V_{rc1}^0\}, k \in F_T \tag{4.24}$$

$$e_j^k + h_2 * d_{i,j} - z_i^k - e_i^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{(i,j) \in A_2 | i \in V_r\}, k \in F_T \tag{4.25}$$

$$e_j^k + h_2 * d_{i,j} - e_i^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{(i,j) \in A_2 | i \in V_c\}, k \in F_T \tag{4.26}$$

$$0 \leq z_i^k \leq G_2 * \sum_{(i,j)\in A_1} y_{i,j,k}, \forall i \in V_r, k \in F_T \tag{4.27}$$

$$0 \leq z_i^k \leq G_2 * \sum_{(i,j)\in A_1} Q_{i,j,k}, \forall i \in V_{c1}^0, k \in F_T \tag{4.28}$$

$$0 \leq Z_i^k \leq (G_1 + G_2) * \sum_{(i,j)\in A_1} x_{i,j,k}, \forall i \in V_r^0, k \in F_T \tag{4.29}$$

$$0 \leq z_{i,j}^k \leq G_2 * Q_{i,j,k}, \forall (i,j) \in A_1, k \in F_T \tag{4.30}$$

$$0 \leq E_i^k + Z_i^k - z_i^k \leq G_1, \forall i \in V_r^0, k \in F_T \tag{4.31}$$

$$0 \leq e_i^k + z_i^k \leq G_2, \forall i \in V_r^0, k \in F_T \tag{4.32}$$

$$0 \leq E_i^k \leq G_1, \forall i \in V_{rc1}^{''}, k \in F_T \tag{4.33}$$

$$0 \leq e_i^k \leq G_2, \forall i \in V_{rc}^{''}, k \in F_T \tag{4.34}$$

Freight constraints

$$\sum_{(i,j)\in A_2} p_{i,j,k} - \sum_{(j,i)\in A_2} p_{j,i,k} = d_j * \sum_{(i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}), \forall j \in V_c, k \in F_T \tag{4.35}$$

$$\sum_{i\in V_c} d_i \sum_{j\in V_{rc}^{'} | (i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}) + \sum_{i\in V_{c1}} d_i \sum_{j\in V_{rc1}^{'} | (i,j)\in A_1} x_{i,j,k} \leq C_1, \forall k \in F_T \tag{4.36}$$

$$0 \leq p_{i,j,k} \leq C_2 * (y_{i,j,k} - Q_{i,j,k}), \forall (i,j) \in A_2, k \in F_T \tag{4.37}$$

Extra subtour elimination constraints

$$u_i^k - u_j^k + 1 \leq M * (1 - x_{i,j,k}), \forall i \in V_r, j \in V_r, (i,j) \in A_1, k \in F_T \tag{4.38}$$

$$0 \leq u_i^k \leq M, \forall i \in V_r, k \in F_T \tag{4.39}$$

The objective function (4.1) minimizes the total travel cost. It corresponds to the van routes cost plus the whole-robot routes cost, minus the van-robot routes cost.

Constraints (4.2)-(4.14) are pure van/robot arc constraints. Constraints (4.2)-(4.6) ensure every node is visited by a van/robot at most once, and the times of departures are equal to the times of arrivals. Constraints (4.6) force the number of a van leaving the depot to be equal to that of its robot leaving the depot, and equal to the number of van/robot coming back to the depot. Constraints (4.7) ensure that every customer node is visited by vans or robots exactly once. Constraints (4.8)-(4.9) enforce for customer nodes and depot, that a van and its robot cannot visit the same node unless the robot is on board the van. Constraints (4.10) ensure a robot cannot visit a parking node unless its van visited this node. Constraints (4.11) let $Q_{i,j,k}$ be equal to 1 only if $x_{i,j,k}$ and $y_{i,j,k}$ are all equal to 1. Constraints (4.12) are the binary variable constraints. Constraints (4.13)-(4.14) force the arc variable to be equal to 0 where they are not allowed to visit.

Constraints (4.15)-(4.32) are the time and energy constraints. Constraints (4.15)-(4.16) are time flow constraints describing vehicle departure from the parking node or depot. Constraints (4.15) are the time flow of the van routes, and constraints (4.16) are the time flow of robot routes. Constraints (4.17)-(4.18) are time flow constraints, ensuring that the van and robot depart from the customer node. Constraints (4.17) are the time flow of the van routes, and constraints (4.18) are the time flow of the robot routes. Constraints (4.19) are the subtour elimination constraints on the whole-robot routes. Constraints (4.20) force the arrival time of the van at a node to be equal to that of the robot if its van is carrying the robot. Constraints (4.21) are time window constraints.

Constraints (4.22)-(4.23) define the battery level of van route in A_1. Constraints (4.22) consider that a van departs from a parking node or depot, while constraints (4.23) consider that a van departs from a van customer node. Constraints (4.24)-(4.26) define the energy flow of the robot. Constraints (4.24) ensure the robot energy flow when it is carried. Constraints (4.25)-(4.26) ensure the energy flow when the robot moves on its own. Constraints (4.27)-(4.34) represent the variable constraints. Constraints (4.27)-(4.28) define that only when both the van and the robot reach the same node, can the charging amount of robot be greater than 0. Constraints (4.27) represent the case where the van and robot rendezvous at the parking node while constraints (4.28) are the case where the van carries the robot to the parking node. Constraints (4.29) ensure that only when a van reaches a parking node/depot can the charging amount of the van be

greater than 0. Constraints (4.30) ensure that only when a van carries its robot can the charging amount of the robot en-route be greater than 0. Constraints (4.31)-(4.32) ensure that when a van or a robot leaves a parking node, the battery amount of van/robot is below its battery capacity. Constraints (4.33)-(4.34) are variable constraints.

Constraints (4.35)-(4.37) are freight flow constraints. Constraints (4.35) are the freight flow along *2nd-level routes* (robot routes). Constraints (4.36) guarantee the total capacity of a van and its robot. Constraints (4.37) are the variable constraints.

Constraints (4.38)-(4.39) are Miller-Tucker-Zemlin constraints to eliminate subtours in dummy stations.

Note: Some constraints are used to allow or forbid the cases in Figure 4.2. Constraints (4.6) forbid cases (13), (14), (17), (18). Constraints (4.8)-(4.9) forbid cases (4), (5), (6), (16), (20). Constraints (4.10) forbid case (8). In addition, nonlinear constraints in the MIP model can be linearized as follows.

Constraints (4.15-4.17) can be linearized as follows:

$$W_i^k + Z_i^k/g_1 + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_r^0 | (i,j) \in A_1\}, k \in F_T \tag{4.40}$$

$$W_i^k + z_i^k/g_2 + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_r^0 | (i,j) \in A_1\}, k \in F_T \tag{4.41}$$

$$w_i^k + z_i^k/g_2 + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_r^0 | (i,j) \in A_1\}, k \in F_T \tag{4.42}$$

$$W_i^k + z_i^k/g_2 + d_{i,j}/v_2 - w_j^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{i \in V_r | (i,j) \in A_2\}, k \in F_T \tag{4.43}$$

$$w_i^k + z_i^k/g_2 + d_{i,j}/v_2 - w_j^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{i \in V_r | (i,j) \in A_2\}, k \in F_T \tag{4.44}$$

$$W_i^k + s_i + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_{c1} | (i,j) \in A_1\}, k \in F_T \tag{4.45}$$

$$W_i^k + z_i^k/g_2 + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_{c1} | (i,j) \in A_1\}, k \in F_T \tag{4.46}$$

Constraints (4.20) can be linearized as follows:

$$-M(1 - \sum_{(i,j)\in A_1} Q_{i,j,k}) \leq W_j^k - w_j^k \leq M(1 - \sum_{(i,j)\in A_1} Q_{i,j,k}), \forall\{j \in V_{rc1}^{'}\}, k \in F_T \tag{4.47}$$

4.3 Single Route Evaluation and Feasibility

We will propose a local search based metaheuristic for solving the problem presented. Before developing the metaheuristic, we need to introduce methods required for evaluating and checking the feasibility of a given 2E-VREC route. For example, Gschwind, 2019 and Gschwind and Drexl, 2019 designed new route feasibility checking technology for pickup and delivery problems.

Move procedures in metaheuristics for solving vehicle routing problems aim to determine which node to visit in what order. However, other essential decisions need to

be made first during move procedures. For example, we need to decide the departure time of a vehicle from a node, and the charging time at a charging station in EVRP.

In the classical VRP problem, a strategy that lets vehicles leave nodes as early as possible is usually adopted. This is a best choice from a feasibility point of view (Savelsbergh and Martin, 1992). For the partial charging EVRP problem, the decision strategy is more complicated. Hiermann et al., 2016; Hiermann et al., 2019 presented a greedy route evaluation approach to make a charging decision, and prove that this greedy approach leads to an optimal decision.

However, research in VRP has been extended to more complex fields, including VRPD and 2EVRP problems in recent years. Sometimes we need to consider the temporal, spatial, and energy synchronization between vehicles. In our problem, vans and robots are all-electric vehicles. The van can charge the robot when the van is carrying its robot, even during the arc. We need to make decisions about: (i) who charges, (ii) how much, (iii) where, and (iv) when. We also need to consider synchronization in the mothership system, such as pickup and delivery, time windows, and other constraints. Hence making a suitable decision involves a broad variety of constraints that are not straightforward in the problem we are addressing.

Because of the increasing difficulty met when making complex decisions for a given route, sometimes even checking the feasibility of a given single route is a hard, possibly even an NP-hard problem. Ideas for solving such cases have been studied. Masson et al., 2017 adopted heuristics to check the feasibility of a given route in the ALNS since they believe there is no polynomial-time algorithm to solve the rolling containers assignment problem (feasibility checking problem). Hunsaker and Savelsbergh, 2002 presented a linear-time complexity heuristic to check the feasibility of a given route in a dial-a-ride problem.

In this section, we first propose a greedy route checking approach to solving our route feasibility checking problem. We then present a linear programming (LP) checking approach to accurately check the feasibility of a given single route. We also compare the two route checking approaches in Section 4.4. Checking the freight of a given route is straightforward, and so will not be dealt with here.

4.3.1 Greedy route checking approach

We draw on the idea of a greedy route evaluation approach (Hiermann et al., 2019) to solve our 2E-VREC route feasibility checking problem. Here we apply a greedy recharging policy with a simple rule, which is to give priority to charging the van battery over the robot battery: the robot needs the van to carry and charge it, so prioritizing the van battery is reasonable.

We define and calculate the time warp between two parking stations, to calculate how much free time can be used for recharging at the departure station, without increasing the time to reach the arrival station. The time warp can be calculated as follows. For the sequence from node i to node j, we calculate the earliest arrival time at node j, and then the latest starting time from node i subject to the earliest arrival time at node j not being changed. The time warp between node i and node j is equal to the latest start time at node i minus the earliest start time at node i. The pseudocode for calculating the time warp is given in B.1.

Based on the greedy recharging policy, we propose a four-step heuristic procedure to charge the vehicles. A 2E-VREC route example in Figure 4.3 is used to introduce our heuristic. In the four-step heuristic, the robot-battery may be recharged to low-level, high-level, or max-level. Low-level charging means the robot battery is charged to the level that makes sure the robot can independently finish its next trip, e.g. tour 4-5-4 in Figure 4.3. High-level charging means the robot battery is charged to the level that ensures the robot can independently finish the tours before its van leaves the parking station, e.g. tour 4-5-4 and 4-6-8 in Figure 4.3. Max-level charging means the robot battery is charged to its maximum level. The van battery may be recharged to low-level or max-level. Low-level charging of a van means the van battery is charged to the level that ensures that the van can reach the next parking station. Max-level charging of a van means the van battery is charged to its maximum level.

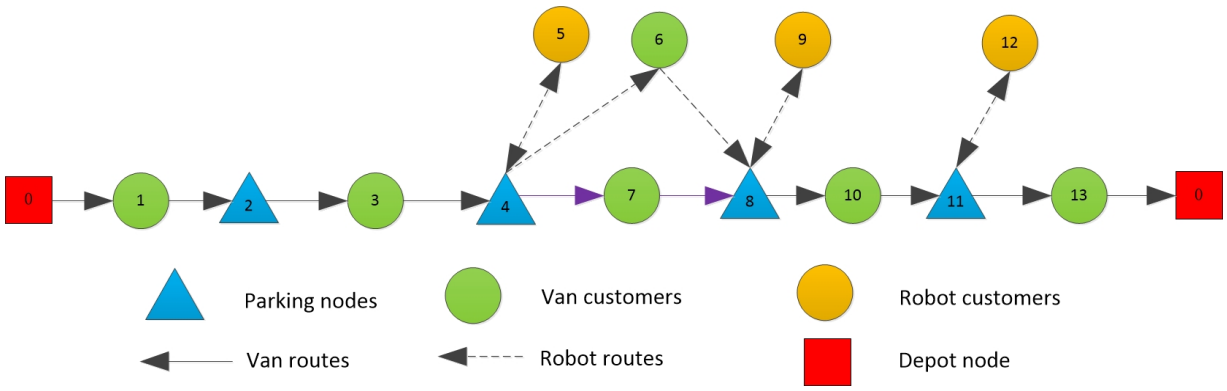

FIGURE 4.3: A 2E-VREC route example

The four-step (Priority) heuristic procedure is as follows.

1. We ensure the current van battery level is enough to reach the next parking station.

    - **Example** We must ensure the van battery is charged enough to travel from node 2 to node 4 in Figure 4.3, even if the charging time is longer than the calculated time warp.

2. We charge the current robot battery to high-level.

   - **Subject to** Priority 1 and time warp.
   - **Special case 1** If the parking station where the vehicles are now is at the node where this robot is to be dropped off, we must charge the robot battery to low-level even if the charging time is longer than the calculated time warp.
   - **Special case 2** When we used the whole calculated time warp to charge the robot, but the robot battery is still at less than its high-level, we perform en-route charging subject to the van battery being charged enough to reach the next parking station.
   - **Example** In station node 2, we charge the robot battery so it can finish route 4-5-4 and route 4-6-8 in Figure 4.3. If we used the whole calculated time warp to charge the robot, but the robot battery is still at less than its high-level, we conduct en-route charging on arc 2-3-4 subject to the van battery being charged enough to reach station 4. In station node 4, we must ensure that the robot can finish tour 4-5-4 even if the charging time is longer than the calculated time warp.

3. We charge the current battery of the van to its max-level

   - **Subject to** Priority 1, 2 and time warp.

4. We charge the current battery of the robot to its max-level.

   - **Subject to** Priority 1, 2, 3 and time warp.

After we have defined the priority charging strategy, we can further optimize the charging by appropriately setting charging time. The target for the further optimal charging approach is to minimize the recharging end time of vehicles. An example of further optimizing the charging is shown in Figure 4.4. First, we charge the van battery to its low-level in case (A). Second, we charge the robot-battery to its high-level in case (B). We can then optimize the charging by properly setting the charging time of the van and the robot, by bringing forward the charging time of the robot to the starting recharging time of the van, as in case (C). We note that if the robot charging time is longer than the charging time of the van, we charge the van battery until its charging time reaches the robot charging time or charge the van-battery up to its max-level. Third, we charge the van battery to its max-level in case (D), and then charge the robot-battery to its max-level in the fourth step in case (E). We can further optimize the charging by bringing forward the charging time of the robot to the recharging start time of the van again, as in case (F).

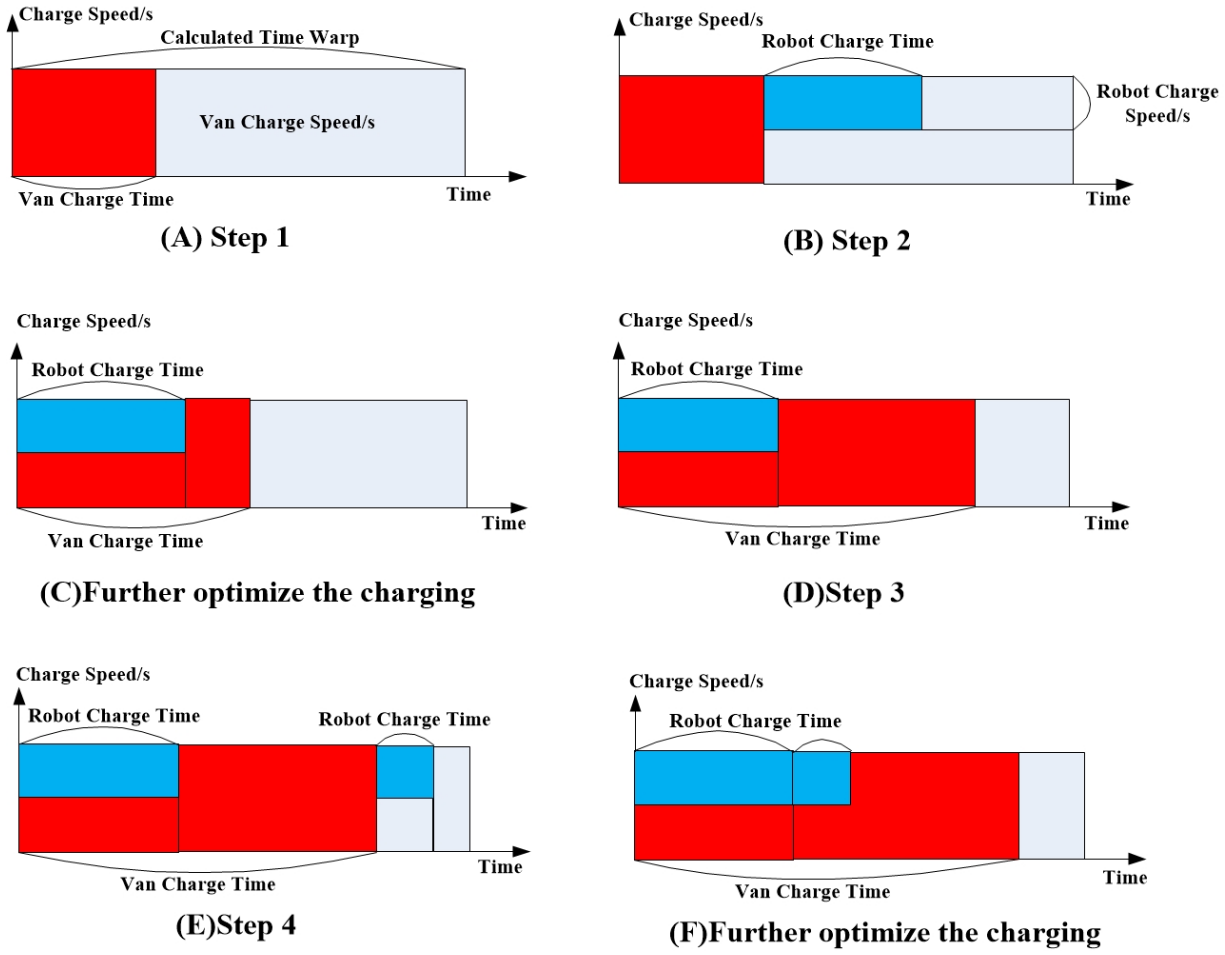

FIGURE 4.4: Example of further optimizing the charging

4.3.2 Linear programming model for route checking

We present an LP model for route evaluation. It is used to accurately check the feasibility of time, distance, and energy for a given single route. The input of the LP model is a 2E-VREC route, and the output is the feasibility of the given route. The details of the formulation and explanation of the LP model are given in B.2.

4.4 Adaptive Large Neighborhood Search

In this section, we adopt an adaptive large neighborhood search (ALNS) algorithm to solve the 2E-VREC problem. Since our problem considers electric vehicles in a two-echelon concept, we draw on some operations proposed in Hemmelmayr et al., 2012, Keskin and Çatay, 2016, and Sacramento et al., 2019. The specific structure of our problem lies in parking stations that have a variety of functions, unlike other similar situations that have been solved by ALNS. For example, in our problem, parking stations can be used for dropping off and picking up robots (vehicle transshipment), and can also be used as charging stations or stations where a van can top up its robot. The parking station is an independent station and can be visited multiple times. Table 4.1 gives the characteristics of stations in different problem types.

TABLE 4.1: Characteristics of station in different problems

	charging	vehicle transshipment	freight transshipment	multi-visit station	independent station
EVRP	yes	no	no	yes	yes
2EVRP	yes	no	yes	no	yes
VRPD	no	yes	yes	no	no
This work	yes	yes	yes	yes	yes

After the initial solution has been constructed, the proposed ALNS algorithm seeks to improve it iteratively until a stopping condition is satisfied. We adopt a maximum iteration number (MaxIteNum) limit to terminate the algorithm. At each iteration, the existing feasible solution is destroyed by removing some nodes or routes by a destroy algorithm. The resulting partial solution is then repaired using a repair algorithm which heuristically repairs the existing route or constructs a new route if needed, for the purpose of obtaining a better solution than the previous one. We also set a number of non-improving iterations (MaxNonImp) before restarting the ALNS algorithm from a new initial solution as in Breunig et al., 2019.

During repair operations, the route evaluation method is used to check the feasibility of the routes. We adopt simulated annealing as acceptance criterion, which allows the algorithm to accept poor solutions during iterations. In addition, each of our

destroy-and-repair operations is assigned a weight that controls how often the operation is selected. The weights are adjusted dynamically as the search progresses by a weight-adjusting method. The algorithm thus adapts to the instance at hand and to the state of the search. We draw on simulated annealing and weight-adjusting proposed by Pisinger and Ropke, 2019. The details of simulated annealing and weight-adjusting used here can be found in B.3.

Our proposed ALNS algorithm is described in Algorithm 5. The best solution is recorded in *recordset* (Line 1). The algorithm starts with an initial solution x and it is also the current best solution x^b (Line 2). The value of $NumNunImp$ is then initialized to 0 (Line 3), where $NumNunImp$ records the number of iterations until $NumNunImp$ becomes larger than $MaxNunImp$. Next, a destroy-and-repair operation is performed until the stopping criterion ($MaxIteNum$) is met (Line 4-19). The $restart()$ function is used to restart the ALNS from Line 2 but keeps the iteration number unchanged.

Algorithm 5 *Adapt_Large_Neighborhood_Search*()

1: $recordset = \{\}$
2: **Initialization:** use RR move to get an initial solution x, let $x^b = x$.
3: $NumNunImp = 0$
4: **while** stopping criterion is not met **do**
5: select destroy and repair methods from set.
6: $x^t = r(d(x))$
7: **if** simulated annealing criterion accepts the solution **then**
8: $x = x^t$
9: **end if**
10: **if** $c(x) < c(x^b)$ **then**
11: $x^b = x$, $NumNunImp = 0$
12: **else**
13: $NumNunImp+ = 1$
14: **if** restart conditions is met **then**
15: $recordset.append(x^b)$ and then $restart()$
16: **end if**
17: **end if**
18: update weights and selection parameters
19: **end while**
20: **return:** output best x^b from *recordset*

4.4.1 Initial solution

At the beginning of the algorithm, all customers are inserted into an un-assigned customer list in random order. We then repeat the following two steps until all the customer nodes are assigned. In the first step, we randomly choose a customer node in the un-assigned customer list. If the chosen node is a robot customer node, then we

construct a 2E-VREC route with van routes and robot routes. Otherwise, we decide whether to construct a 2E-VREC route with van routes and robot routes or to construct a 2E-VREC route only including van routes, according to roulette-wheel selection. The second step randomly sorts the order of the un-assigned customer list, and then chooses a customer to insert into the random-feasible position of the constructed route, until no customer node can be inserted.

4.4.2 Destroy moves and operations

A destroy operation destroys a chosen 2E-VREC route by removing nodes or routes. We define four destroy operations: customer removal, station removal, route closure, and route destruction. The ALNS algorithm destroys a part of the current solution in each iteration. We mainly adopt a random and a greedy destroy strategy.

In our problem setting, we denote a parking station used for charging only a charging station. Otherwise, it is denoted a connection station. We distinguish between removing two kinds of stations in station removal operations, since removing a connection station dramatically changes the solution, while removing a charging station does not.

The four destroy operations and the moves used are as follows:

1. **Customer removal operation**: The customer removal operation removes customers from a 2E-VREC route with a probability $\beta \in [0, 1]$. We adopt *random customer removal* and *greedy customer removal* moves.

    - *Random customer removal* randomly removes a customer node.
    - *Greedy customer removal* removes the customer that can yield the largest cost reduction for a given route.

2. **Station removal operation**: The station removal operation removes a parking station from a 2E-VREC route. Here we adopt *station-route removal*, *charging station removal* and *redundant charging station removal* moves. After we remove a station, the route may become battery-unfeasible. We remove a customer before or after the removed station in the van route, or we insert a station different from the removed station in the van route to make the route feasible if needed, since visiting fewer customers or visiting more stations could make a battery-unfeasible route feasible.

    - *Station-route removal* randomly removes a connection station and the robot routes depart from and arrive at this station.
    - *Charging station removal* randomly removes a charging station.

- *Redundant charging station removal* removes a redundant charging station if one exists.

3. **Route closure operation**: The route closure operation closes a van or robot route in a 2E-VREC route. We use *van/robot route closure* moves, and we adopt *random/greedy* strategies.
   - *Random van/robot route closure* removes all customers from a randomly chosen van/robot route.
   - *Greedy van/robot route closure* removes the route that can yield the largest cost reduction.
4. **Route destruction operation**: The route destruction operation chooses a 2E-VREC route and then destroys it. All customer nodes in this 2E-VREC route are put in the un-assigned customer list. We adopt *random/greedy route destruction* moves.
   - *Random route destruction* randomly chooses a 2E-VREC route in the solution to destroy.
   - *Greedy route destruction* chooses a 2E-VREC route, with a minimum number of customer nodes to destroy.

4.4.3 Repair moves and operations

Route repair operations repair the existing routes or construct a new route if needed, using three kinds of repair operations: route reconstruction, customer insertion, and route structure change first / customer insert second. The repair operations do not allow unfeasible solutions but allow worse solutions through simulated annealing evaluation and calibration. Note that in the repair process, we consider opening a new van/robot route from a connection station, or changing the role of charging stations to connection stations and then opening a robot route to considerably change the solution.

The three kinds of repair operation are as follows.

1. **Route reconstruction operation**: The route reconstruction operation ensures that we always obtain a viable solution. We adopt the same method as we used to obtain the initial solution in Section 4.4.1.
2. **Customer insertion operation**: The customer insertion operation is used to insert customers in a 2E-VREC route. First, we use *random/greedy customer insertion* moves to insert customer nodes. If a customer cannot be inserted because of battery outage, we try to use *nearest station insertion* moves to make the route

feasible. If all the possible positions have been tried and the customer still cannot be inserted, we try to open a new robot route, where the start node and end node of the route are the same, and then insert the customer node. Finally, if there still are unassigned customer nodes, we use route reconstruction-based repair to reconstruct a new 2E-VREC route.

- *Random customer insertion* sorts the order of the un-served customers list randomly and chooses a customer to insert into the random-feasible position of the 2E-VREC route until all customers have been tried.
- *Greedy customer insertion* sorts the order of un-served customers list randomly, and then chooses a customer to insert into the 2E-VREC route with the least cost increases.
- *Nearest station insertion* is used after we have inserted a customer node and found the route is unfeasible: we try to insert a station before or after the inserted customer position.

3. **Route structure change first / customer insert second operation**: We present two kinds of route structure change approach: (i) we use *van/robot route open* moves to begin a new van/robot route from a station; the van/robot route includes one customer node, and the ending station can be the same one or a different one; we then perform a customer insertion operation, (ii) we first use *random/greedy station insertion* moves to insert a station into a 2E-VREC route and then follow route open moves at this station if the station is not inserted in an independent van route; next, we conduct customer insertion- based repair moves.

   - *Van route open* to begin a new van route from a connection station.
   - *Robot route open* to begin a new robot route from a station; *robot route open* can convert a charging station into a connection station or increase the robot route from a connection station. The ending station can be the same as or different from the starting station.
   - *Random station insertion* randomly chooses a position in the 2E-VREC route to insert a station.
   - *Greedy station insertion* chooses the best insertion position with feasible and least total travel cost increases to insert a station.

4.5 Computational study

We performed four types of computational experiment. First, we used the parameter tuning approach to determine the parameters. Second, we evaluated the performance

of two proposed route evaluation approaches. Third, we conducted the ALNS experiment to see the overall performance of the proposed ALNS algorithm. Fourth, we performed a sensitivity analysis to determine the impact of related charging modes, battery capacities, and charging rate. In the four types of computational experiment, we ran our algorithms 10 times for every instance.

The mathematical programming model was coded in OPL. CPLEX 12.8 was used to solve the model. The ALNS was coded in python version 3.6.8. Both CPLEX and python were executed on an Intel(R) Core(TM) 3.2GHz processor with 8 GB memory running under Windows 10. Python was run with single-threading.

4.5.1 Instance generation

We used the instances of Dellaert et al., 2018 and Yu et al., 2020 as a basis. The specific instance generation approach is described in those two articles. We chose eight types of instances with 3/5 parking nodes and 15/30/50/100 customer nodes, respectively. Each type includes 20 instances that could be divided into four cases according to the different time window and demand generation methods.

We set the speed of the van at 2, and the speed of the robot at 1. The van's energy consumption rate was 2 units per distance, and the robot's energy consumption rate was 1 unit per distance. The recharging rate was 10 units per time for a van and 4 units per time for a robot. The capacity of a robot was 50, and the capacity of a van was 200. The battery capacity of a robot was 120, and the battery capacity of a van was 400. The depot time window was [0,500]. Vans could access two-thirds of the total customer nodes in each instance. If the calculated number of van customer nodes was fractional, we rounded up. The first two-thirds of customers in the instance are the van customers.

4.5.2 Parameter tuning

We drew on the parameter tuning concept (Jie et al., 2019) to determine the parameters. The parameter tuning approach started with a set of initial parameter values. Parameter tuning conducted the ALNS algorithm with the parameter values in the search interval. The parameter value that had the best solution was chosen to perform the following calculation. The process was repeated until all the parameters in the parameter set had been tuned. We chose eight instances with 3-15 satellite-customers to conduct the parameter tuning procedure: CA1-3-15, CA2-3-15, CB1-3-15, CB2-3-15, CC1-3-15, CC2-3-15, CD1-3-15, and CD2-3-15. The parameter notations, descriptions, and values used here can be found in B.4.

4.5.3 Comparison of greedy route evaluation and LP-based evaluation

Since the greedy route evaluation (GE) approach presented is an approximate method, we compared it with the LP-based evaluation approach to assess its accuracy and speed. We also conducted additional experiments to evaluate an approach combining the GE-based and LP-based evaluation (GE-LP-based evaluation) to exactly check the feasibility of a given route. The GE-LP-based evaluation first uses a GE-based evaluation to test the feasibility of the route. If the GE-based evaluation finds the route feasible, it is deemed feasible, and the evaluation procedure ends. Otherwise, the evaluation procedure takes the LP-based evaluation approach to finally check the feasibility of the route.

The instances used in this experiment were the same as those in the turning parameter section. Table 4.2 gives the results for the comparison of GE-based, LP-based, and GE-LP-based evaluations. Column 1 indicates the instances tested. Our num1 is the average number of times the output of the GE-based route evaluation differs from the output of the LP-based route evaluation approach. Our num2 is the average total number of times that the route evaluation procedure is performed. The LP time, GE time, and GE-LP time are the average evaluation times of LP-based, GE-based, and GE-LP-based evaluation approaches in 10 experiments, respectively. The runtime (LP time, GE time, GE-LP time) was the time for the ALNS to solve the instance using different evaluation approaches.

TABLE 4.2: Comparison of greedy route evaluation and linear programming based evaluation

3-15	num1/num2	LP time(s)	GE time(s)	GE-LP time(s)
CA1	0/284603	2953.2	44.0	2895.1
CA2	11/317831	3146.9	61.9	3070.2
CB1	6/259734	2955.8	48.6	2535.5
CB2	3/353319	3422.8	59.6	3206.3
CC1	54/175804	1955.1	35.1	1608.4
CC2	57/183887	2167.2	40.9	1690.8
CD1	90/291439	3065.8	51.3	2604.9
CD2	8/326793	3692.7	54.0	3329.8
AVER	229/2193410	2919.9	49.4	2617.6

It is obvious from the results that the estimation error (num1/num2) of the GE-based evaluation approach was around one in a thousand. The GE-based evaluation ran 59 times faster than the LP-based evaluation method. The GE-LP-based evaluation also ran faster than the LP-based evaluation approach. However, the runtime reduction was only about 10%. This may be because many unfeasible routes that need to be

checked using the LP-based evaluation method were generated in the process of running the ALNS algorithm. In any case, if we need to check the feasibility of a route exactly, the GE-LP-based evaluation approach is faster than the evaluation method using only the LP-based approach.

4.5.4 ALNS experiment

First, we compared the GE-based ALNS results with the GE-LP-based ALNS results to analyze the GE-based ALNS algorithm's performance. Second, we compared the ALNS results with the CPLEX results in very small-scale instances. Third, we used the ALNS algorithm to solve larger instances and see the overall performance of the proposed ALNS algorithm.

Comparison the GE-based with GE-LP-based ALNS results

In Subsection 4.5.3, we analyzed the estimation error of the GE-based evaluation approach, which was about one in a thousand. Here we ran the GE-based ALNS algorithm and the GE-LP-based ALNS algorithm on 3–15 satellite-customer instances to see whether a one thousandth estimation error greatly changed the results.

Table 4.3 gives a comparison of the GE-based ALNS and GE-LP-based ALNS results. AC and BC are the averages and best objective values. SDC is the standard deviation of the objective value, and AT(s) is the average runtime of the GE-based(GE-LP-based) ALNS algorithm.

Table 4.3 shows that the average performance difference between the two algorithms was not significant, and the average solving time of the GE-based ALNS was much shorter than that of the GE-LP-based ALNS algorithm. We therefore used the GE-based ALNS algorithm in the subsequent ALNS computation study.

Comparison of the ALNS results with the CPLEX results

The VRPD model is hard to solve for general-purpose MIP solvers. For example, Wang and Sheu, 2019 showed the commercial solver can only address the VRPD problem with 10 nodes (contains parking nodes and customer nodes), and even the branch and price algorithm they provided can only solve the problem with 12 points. In our problem, we replicated the parking nodes for multiple visiting. Dummy nodes increase the number of nodes in the model, making the problem more difficult to solve, and reducing the problem size we can address.

We tested 20 instances with five customer nodes and three parking nodes. Theoretically, each parking node can be reached up to 5+1 times. However, using too many

TABLE 4.3: GE-based and GE-LP-based ALNS algorithm comparison

	GE-based ALNS				GE-LP-base ALNS			
3-15	AC	BC	SDC	AT	AC	BC	SDC	AT
CA1	532.2	532.2	0.0	40.6	532.2	532.2	0.0	2546.5
CA2	612.3	598.3	11.4	52.5	615.6	598.3	7.6	2938.7
CA3	586.1	574.8	9.7	45.2	583.4	574.8	7.8	3002.7
CA4	590.5	573.2	18.9	46.3	583.1	573.2	11.9	2789.2
CA5	584.8	582.9	4.0	47.0	585.0	582.9	3.7	2557.7
CB1	570.0	557.4	6.7	45.6	573.6	569.8	4.0	2653.5
CB2	607.8	588.9	17.0	51.2	596.4	568.0	20.3	3188.2
CB3	663.5	650.6	6.7	41.6	668.4	659.7	5.2	2363.7
CB4	541.4	515.7	13.1	44.4	546.4	515.7	14.4	2370.4
CB5	540.6	537.2	4.4	40.4	540.0	537.2	4.0	2379.2
CC1	445.9	426.2	11.4	31.0	446.7	426.2	10.5	1553.4
CC2	460.4	457.6	1.0	36.1	461.1	459.5	2.0	1690.2
CC3	516.1	513.4	2.6	32.0	517.3	513.4	4.5	1576.6
CC4	481.7	466.5	7.6	30.6	476.3	450.6	11.3	1577.8
CC5	391.4	391.4	0.0	34.9	391.4	391.4	0.0	2227.2
CD1	526.4	520.0	6.7	45.5	529.4	520.0	8.7	2646.6
CD2	530.0	516.9	15.8	49.7	536.6	516.9	14.2	3125.7
CD3	604.6	595.2	7.1	47.8	602.6	597.5	4.8	3248.7
CD4	522.6	516.1	7.8	34.0	521.6	520.7	2.9	2187.0
CD5	513.8	513.8	0.0	46.3	513.8	513.8	0.0	2695.3
AVER	541.1	531.4	7.6	42.1	541.0	531.1	6.9	2465.9

dummy parking nodes usually makes the problem difficult to solve. We therefore iteratively solved our model by CPLEX, gradually increasing the number of dummy nodes. The increase in the number of dummy nodes stops when no improvement in the solution cost is found. However, if the solution is still worse than the ALNS solution, we continue to increase the number of dummy nodes until the CPLEX solution is better than or equal to the ALNS solution. The iterative process will also stop if the procedure runs out of solving time or out of memory.

For each instance, CPLEX 12.8 runs with default settings until it finds an optimal solution, exhausting the predetermined maximum computation time (7200s), or until the program runs out of memory. The computational results for all instances are presented in Table 4.4, in which K represents the number of times a vehicle can visit a parking node, and UB and LB are the upper and lower bounds of CPLEX solutions. E1 is the CPLEX gap between CPLEX upper bound (baseline) and CPLEX lower bound, and TIME is the CPLEX solving time. BC and AT are the best solutions for ALNS in 10 tests and average solving time of ALNS. E2 is the gap between the ALNS best solution and CPLEX upper bound (baseline). The symbol $*$ means CPLEX is run out of memory. The solution selected at the end of each instance is in bold.

Table 4.4 shows that our ALNS can always reach the best solutions in very small instances. For case CB2, ALNS performed better than the CPLEX solver, which ran out of memory in $K = 3$.

TABLE 4.4: CPLEX and ALNS results comparison (5 customer nodes and 3 parking nodes)

	K=2				*K*=3				*K*=4				*K*=5				ALNS				
	UB	LB	GAP1	TIME	UB	LB	GAP1	TIME	UB	LB	GAP1	TIME	UB	LB	GAP1	TIME	BC	AC	SDC	AT	GAP2
CA1	291.6	291.5	0.0	22.1	286.8	286.7	0.0	372.0	**284.2**	**284.1**	**0.0**	**5058.0**	286.8	217.0	24.3%	7200	284.2	284.2	6.0	14.6	0.0
CA2	360.3	360.3	0.0	18.2	**360.3**	**360.3**	**0.0**	**2753.6**	/	/	/	/	/	/	/	/	360.3	367.2	8.9	12.4	0.0
CA3	275.2	275.2	0.0	1.5	**275.2**	**275.2**	**0.0**	**29.3**	/	/	/	/	/	/	/	/	275.2	275.2	0.0	17.0	0.0
CA4	261.2	261.2	0.0	1.3	**261.2**	**261.2**	**0.0**	**17.7**	/	/	/	/	/	/	/	/	261.2	261.2	0.0	14.6	0.0
CA5	192.2	192.2	0.0	1.0	167.7	167.7	0.0	4.4	**167.7**	**167.7**	**0.0**	**100.7**	/	/	/	/	167.7	167.7	0.0	14.8	0.0
CB1	288.7	288.7	0.0	8.6	**288.7**	**288.6**	**0.0**	**485.6**	/	/	/	/	/	/	/	/	288.7	288.7	0.0	15.0	0.0
CB2	**406.6**	**406.6**	**0.0**	**33.2**	*	*	*	*	/	/	/	/	/	/	/	/	360.3	360.3	0.0	24.4	-11.4%
CB3	320.5	320.5	0.0	6.8	**314.7**	**314.6**	**0.0**	**118.5**	314.7	302.1	0.0	7200.5	/	/	/	/	314.7	314.7	0.0	20.0	0.0
CB4	256.3	256.3	0.0	1.3	256.3	256.3	0.0	15.6	/	/	/	/	/	/	/	/	256.3	256.3	0.0	14.2	0.0
CB5	190.7	190.7	0.0	0.7	170.6	170.6	0.0	4.5	**170.6**	**170.6**	**0.0**	**129.8**	/	/	/	/	170.6	170.6	0.0	14.7	0.0
CC1	199.9	199.9	0.0	0.3	195.7	195.7	0.0	1.4	**195.7**	**195.7**	**0.0**	**23.328**	/	/	/	/	195.7	195.7	0.0	12.9	0.0
CC2	297.2	297.2	0.0	8.0	**297.2**	**297.2**	**0.0**	**227.7**	/	/	/	/	/	/	/	/	297.4	297.4	0.0	15.5	0.0
CC3	306.0	306.0	0.0	6.6	**262.6**	**262.6**	**0.0**	**95.6**	306.0	249.1	0.2	7200	/	/	/	/	262.6	262.6	0.0	13.6	0.0
CC4	270.6	270.6	0.0	2.8	**270.6**	**270.6**	**0.0**	**64.4**	/	/	/	/	/	/	/	/	270.6	270.6	0.0	14.4	0.0
CC5	180.4	180.4	0.0	2.7	157.8	157.8	0.0	5.8	**157.8**	**157.8**	**0.0**	**67.4**	/	/	/	/	157.8	157.8	0.0	12.2	0.0
CD1	295.2	295.2	0.0	5.8	**295.2**	**295.2**	**0.0**	**331.5**	/	/	/	/	/	/	/	/	295.2	295.2	0.0	19.0	0.0
CD2	321.3	321.3	0.0	2.4	**321.3**	**321.3**	**0.0**	**170.9**	/	/	/	/	/	/	/	/	321.3	321.3	0.0	19.2	0.0
CD3	270.7	270.7	0.0	4.3	249.1	249.1	0.0	23.4	**249.1**	**249.1**	**0.0**	**750.1**	/	/	/	/	249.1	249.1	0.0	16.7	0.0
CD4	281.0	281.0	0.0	7.1	**281.0**	**281.0**	**0.0**	**97.3**	/	/	/	/	/	/	/	/	281.0	281.0	0.0	16.8	0.0
CD5	190.7	190.7	0.0	1.0	170.6	170.6	0.0	8.2	**170.6**	**170.6**	**0.0**	**160.0**	/	/	/	/	170.6	170.6	0.0	12.1	0.0

ALNS to solve larger instances

We used the ALNS algorithm for all the instances with 15/30/50/100 customer nodes and 3/5 parking nodes to evaluate the performance of our ALNS algorithm.

The results of ALNS for 15, 30, 50, and 100 customers are shown in Tables 4.5-4.6. AC and BC are the averages and best objective values. The SDC represents the standard deviation of the objective value, and the AT(s) is the average runtime of the ALNS.

TABLE 4.5: ALNS (15 and 30 customers)

	3-15				5-15				3-30				5-30			
	AC	BC	SDC	AT	AC	BC	SDC	AT	AC	BC	SDC	AT	AC	BC	SDC	AT
CA1	532.2	532.2	0.0	40.6	683.0	672.4	7.3	44.7	1189.2	1159.1	21.4	134.3	1063.6	1041.7	14.7	143.0
CA2	612.3	598.3	11.4	52.5	717.0	708.1	8.8	40.4	1027.4	972.1	31.9	138.8	1004.4	965.1	21.4	127.5
CA3	586.1	574.8	9.7	45.2	485.9	485.9	0.0	41.8	1004.8	979.0	16.8	150.5	1045.6	992.5	26.7	144.2
CA4	590.5	573.2	18.9	46.3	490.0	486.4	11.1	43.5	826.7	802.7	16.5	128.5	1268.3	1229.3	23.8	130.9
CA5	584.8	582.9	4.0	47.0	620.6	605.0	9.5	46.5	971.9	958.3	9.1	132.2	1047.1	1019.4	18.2	140.9
CB1	570.0	557.4	6.7	45.6	639.5	622.3	10.5	46.5	1216.2	1177.4	24.4	144.1	1073.0	1060.9	12.8	141.7
CB2	607.8	588.9	17.0	51.2	646.9	643.7	3.4	49.1	935.8	914.9	17.3	134.0	1017.0	1002.4	11.8	129.8
CB3	663.5	650.6	6.7	41.6	512.9	512.9	0.0	41.6	973.0	948.7	20.7	133.5	1008.8	960.7	20.6	148.0
CB4	541.4	515.7	13.1	44.4	523.3	522.7	1.3	37.4	876.0	862.4	7.7	142.7	1295.2	1264.2	24.6	142.7
CB5	540.6	537.2	4.4	40.4	608.8	608.8	0.0	50.0	961.0	940.9	14.0	193.8	1032.8	1018.5	14.4	191.9
CC1	445.9	426.2	11.4	31.0	474.0	463.0	12.6	43.6	946.9	919.5	19.6	126.1	939.8	904.8	22.9	131.8
CC2	460.4	457.6	1.0	36.1	518.2	501.6	10.7	37.2	796.5	771.4	15.5	98.7	823.0	813.5	6.7	91.9
CC3	516.1	513.4	2.6	32.0	431.6	431.2	1.1	30.5	831.8	789.7	23.5	124.9	878.5	858.1	13.7	162.4
CC4	481.7	466.5	7.6	30.6	466.9	452.0	18.5	38.6	770.4	719.4	23.8	130.8	1000.6	944.2	38.5	114.6
CC5	391.4	391.4	0.0	34.9	560.4	557.9	5.8	40.9	846.5	824.1	13.5	111.8	802.1	757.4	28.9	113.4
CD1	526.4	520.0	6.7	45.5	585.9	580.5	11.4	50.1	1137.7	1113.3	21.3	130.6	1118.3	1064.2	31.6	153.2
CD2	530.0	516.9	15.8	49.7	625.9	618.3	3.0	47.6	996.0	955.0	27.4	139.3	1017.5	973.0	24.7	137.0
CD3	604.6	595.2	7.1	47.8	476.4	475.2	1.6	49.8	1014.2	982.4	15.6	125.3	965.5	933.7	22.9	163.8
CD4	522.6	516.1	7.8	34.0	529.9	526.9	2.1	48.8	877.9	858.6	13.7	153.8	1123.5	1086.8	23.0	152.2
CD5	513.8	513.8	0.0	46.3	634.3	631.6	5.5	43.9	954.7	926.6	27.0	156.7	965.3	932.9	27.7	144.1
AVER	541.1	531.4	7.6	42.1	561.6	555.3	6.2	43.6	957.7	928.8	19.0	136.5	1024.5	991.2	21.5	140.2

The results show that the average solving time for the hardest instances with five parking nodes and 100 customers was less than 900s. It also shows that our algorithm has the potential to solve larger problems quickly.

TABLE 4.6: ALNS (50 and 100 customers)

	3-50				5-50				3-100				5-100			
	AC	BC	SDC	AT	AC	BC	SDC	AT	AC	BC	SDC	AT	AC	BC	SDC	AT
CA1	1722.5	1674.0	31.4	284.6	1608.5	1550.5	36.7	287.5	2839.8	2748.6	44.3	1061.3	2814.0	2745.9	55.5	1014.1
CA2	1743.2	1684.7	34.0	309.4	1541.7	1481.2	36.1	312.2	3153.5	3103.6	41.3	1258.0	2735.0	2648.3	47.5	1152.4
CA3	1676.3	1597.9	36.7	313.5	1558.5	481.7	68.0	298.4	3225.9	3096.4	83.4	1002.0	2972.0	2865.3	74.5	1083.7
CA4	1630.7	1550.1	41.3	293.4	1493.0	1451.8	27.1	299.4	3971.1	3799.8	107.8	830.6	3052.6	2952.7	71.8	916.2
CA5	1562.2	1468.6	45.8	370.0	1539.3	1485.9	37.6	376.9	2981.8	2886.4	51.2	979.4	2861.0	2816.3	39.7	855.3
CB1	1707.9	1632.8	41.7	368.3	1683.5	1629.9	38.4	359.9	2635.9	2567.8	54.0	808.3	2896.6	2780.4	69.4	808.4
CB2	1633.7	1577.7	31.0	296.2	1581.7	1539.9	31.9	331.5	2931.8	2782.7	89.4	898.8	2869.9	2791.6	41.4	891.1
CB3	1738.0	1669.6	35.1	337.4	1625.7	1558.5	30.8	313.8	3196.1	3042.3	70.7	849.1	2896.9	2757.5	97.5	822.1
CB4	1601.9	1544.6	32.3	323.1	1453.1	1431.6	19.9	322.3	3977.5	3756.2	97.8	848.8	2991.9	2872.0	80.4	896.3
CB5	1590.7	1499.1	43.0	266.3	1566.2	1534.8	20.5	269.2	2913.6	2837.4	43.3	853.9	2916.9	2735.4	84.0	818.8
CC1	1380.5	1334.3	25.8	199.5	1290.0	1238.5	31.6	217.7	2414.3	2300.6	64.5	700.1	2443.0	2406.7	33.7	690.2
CC2	1414.5	1380.0	20.6	203.8	1238.0	1203.5	21.1	214.7	2518.9	2411.2	59.1	697.3	2265.6	2157.4	69.3	679.9
CC3	1387.6	1325.2	35.3	209.8	1351.0	1302.9	36.0	227.6	2551.9	2387.5	87.9	787.1	2541.3	2498.5	31.2	743.9
CC4	1348.6	1301.3	31.3	208.4	1321.8	1274.1	30.0	204.4	3049.3	2958.4	54.9	756.9	2437.1	2316.6	75.5	803.5
CC5	1511.8	1474.6	21.4	219.5	1307.9	1248.4	32.8	192.4	2460.0	2381.4	57.1	710.7	2385.6	2343.4	31.9	640.2
CD1	1630.7	1559.5	49.2	251.4	1518.4	1465.0	30.8	242.8	2791.5	2617.1	78.4	844.6	5483.2	4990.8	227.9	1268.1
CD2	1702.3	1617.1	48.1	249.9	1537.6	1477.3	35.6	247.8	3083.4	2995.9	49.3	865.8	2783.9	2684.6	51.3	878.5
CD3	1684.1	1629.9	35.9	252.9	1522.7	1481.6	30.9	242.0	3195.2	3114.0	76.3	856.3	3080.9	2944.5	78.4	963.0
CD4	1589.1	1515.2	38.4	253.7	1431.9	1382.5	35.5	252.8	3813.6	3690.6	82.3	809.5	2957.5	2901.9	40.3	901.4
CD5	1602.0	1544.3	33.9	245.2	1534.8	1478.0	32.8	261.6	2884.5	2801.4	58.2	945.9	2774.8	2646.0	61.7	846.7
AVER	1592.9	1529.0	35.6	272.8	1485.3	1384.9	33.2	273.7	3029.5	2914.0	67.6	868.2	2907.9	2792.8	68.1	883.7

4.5.5 Sensitivity analysis on impact of charging modes, battery capacities, and charge speeds

We analyzed the impact of the en-route-charge, static-charge, and no-charge modes of vehicles on the mode-computed outputs. The en-route-charge mode is the mode we are studying here. The static-charge mode does not allow a van to charge its robot en-route, and the van can only recharge its robot at charging stations. The no-charge mode only allows the van and robot to be recharged at the depot only. The battery capacity of vehicles and the charging rate of vehicles may also affect the efficiency of the distribution system. Hence, we also analyzed the impact of battery capacity / charging rate changes on the results.

We reset the battery capacity for each instance in the sensitivity analysis section, to better distinguish the impact of charging modes, battery capacity, and charging rate changes. The baseline battery capacity of the van (G_1) and robot (G_2) for each instance were set as follows.

Step 1: We let a van carry its robot to provide services for a customer with a minimum total van and robot travel distance and then let the van and robot return from the original road.

Step 2: We recorded the maximum van travel distance / maximum robot travel distance in Set 1/Set 2, and then repeated Step 1 until all the customers were served.

Step 3: We chose the maximum van/robot travel distance in Set 1/Set 2. We set the maximum van travel distance to twice the maximum robot travel distance if the selected van robot travel distance was less than twice that of the robot. We then multiplied the maximum van/robot travel distance by $\theta_1 * h_1/\theta_2 * h_2$ as the battery capacity

of van/robot for each instance, respectively, where the θ_1/θ_2 was the multiplication factor for the battery capacity of the van/robot.

We chose 20 instances with three parking nodes and 15 customers to conduct the sensitivity analysis experiments.

First, we performed a comparison experiment between different charging modes. We set the en-route-charge mode as baseline, with a value of 575.69. The experimental results show that en-route-charge and static-charge can effectively reduce the cost of the 2E-VREC model. The average best cost fell by nearly 21% compared with the no-charge mode. Adopting the en-route-charge mode instead of static-charge mode reduced the average best cost by 0.35%. Furthermore, we doubled or halved the customer time window to see the efficiency improvement using en-route-charge mode rather than static-charge-mode in different customer time window settings. However, the improvements were still 0.3%-0.4% in our 2E-VREC model settings.

Even if using the en-route-charge mode only reduced best cost by nearly 0.35% on average compared with using the static-charge mode, we recommend unmanned logistics companies to use an en-route-charge mode if the fixed cost of the new technology is controllable, since the reduction of 0.35% in cost is a valuable saving in a distribution system.

For the sensitivity analysis on the impact of battery capacity experiments, we generated the different van/robot battery capacity combinations by multiplying the van/robot's battery capacity by a different multiplication factor. Table 4.7 gives the sensitivity analysis on battery capacities. Row 1 shows the different battery capacity combinations, in which θ_1/θ_2 is the multiplication factor. Row 2 shows the average best result for the instances in which the control group result (baseline) and the quality of the experimental group solution are reported as an average percentage gap.

TABLE 4.7: Comparison of battery capacities

(θ_1/θ_2=1.0/1.2)	(θ_1/θ_2=1.0/1.1)	baseline(θ_1/θ_2=1.0/1.0)	(θ_1/θ_2=1.1/1.0)	(θ_1/θ_2=1.2/1.0)
-5.59%	-3.65%	575.69	-0.02%	-0.05%

Table 4.7 shows that increasing the battery capacity of the robot 1.1 and 1.2 times can reduce the cost by 3.65% and 5.59%, respectively. Increasing the battery capacity of the van 1.1 and 1.2 times produces only a tiny increase in efficiency.

For the sensitivity analysis on the impact of charging rate experiments, we set the charging rate used in the en-route-charge mode as the baseline. We then set the charging rate at 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2, and 2 times the baseline. The comparison of the charging rate is shown in Figure 4.5.

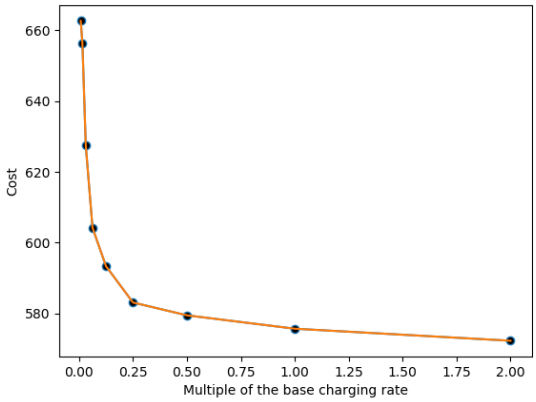

FIGURE 4.5: Comparison of charging rate

Figure 4.5 shows that increasing charging rate can significantly reduce cost. However, the yield curve on increasing the charging rate tends to flatten as the charging rate increases.

4.6 Conclusions

The van-based robot delivery concept is beginning to gain acceptance by logistic companies, such as JD.COM, Amazon, etc., since robots can visit pedestrianized areas or university campuses that vans cannot reach. Recent demonstrations by NIO have shown the potential of electric vans for mobile charging platforms.

This chapter notes that the time during which vans are carrying robots can be used effectively to recharge the robots, thereby increasing the efficiency of distribution systems. Here we present a novel transportation delivery model that incorporates en-route-charging for autonomous electric vehicle logistics in cities. The model extends the VRPD model to electric vehicle aspects and brings en-route-charge technologies into the E-VRPD model, which provides a new option for logistics operators.

In the 2E-VREC model, the vans carry small robots along the *1st-level route*, and the robot itself travels along the *2nd-level open route*. The van and robot can both serve customers directly, but some constrained customer sets can be visited only by robots. The van stops at parking nodes for dropping off and/or picking up its robot, and for recharging and replenishing its robot if needed. The van can charge its robot (if it is on board the van) during its trip. The van can be recharged at the parking nodes.

To model the proposed problem, we introduce a mixed-integer program. The 2E-VREC model brings a new time-distance-energy trade-off to electric vehicle routing. The energy transfer is negatively correlated with the van's available travel distance, but positively correlated with the robot's available travel distance. Energy transfer also

needs time. Owing to the difficulty processing the trade-off for checking the feasibility of a given 2E-VREC route, we further propose a greedy route evaluation approach and an LP-based route evaluation method. A comparison of greedy route evaluation and LP-based evaluation showed that the GE-based evaluation method has high accuracy and runs faster.

An adaptive large neighborhood search algorithm is presented for solving larger instances. A sensitivity analysis for vehicle charging modes, battery capacities, and charging rate reveals that using en-route-charge technologies, with appropriate increases in battery capacity and charging rate, has useful effects on cost. We recommend that logistics companies use en-route-charge technology if the fixed cost of the new technology is controllable.

Chapter 5

Van-based robots hybrid pickups and deliveries routing problem

Abstract

We present a two-echelon van-based robot last-mile pickup and delivery system in an urban area. Robots can visit areas with van access restrictions, such as pedestrianized areas or university campuses. The van stops at parking nodes to drop off and/or pick up its robot, and to replenish its robot and/or swap its robot's battery if needed. Five van/robot pickup and delivery cases are considered, according to the roles the van/robot plays in the process of pickup/delivery and whether the van helps to transport its robot.

To model the proposed problem, we introduce a mixed-integer program including time, freight, and energy. We further propose an adaptive large neighborhood search algorithm to solve larger instances and a capacity feasibility test approach for a single route. We then assess the influence of parking node density on model output. A case study based on a realistic city scene is introduced, A sensitivity analysis is performed on the robot's travel cost rate and maximum travel distances, and examines the van no-go area's effect. Two classical models are compared with ours, and results show our model is competitive in appropriate scenario settings. We therefore advocate using the two-echelon van-based robot last-mile pickup and delivery system in urban areas.

5.1 Background

Pickup-and-delivery problems (PDPs) are an essential family of routing problems in which goods or passengers have to be transported from different origins to different destinations (Battarra et al., 2014).

Urban logistics and transportation companies have to conduct goods pickup and delivery operations. Supposing an e-commerce company has surplus logistics distribution capacity after completing its e-commerce logistics distribution task. In such a case, it can make gainful use of its surplus logistics capacity to generate additional earnings

by helping other individuals deliver freight. For example, JD.com delivers and picks up JD-Mall's own products and helps individuals or individual stores (offline store or local hypermarket) deliver goods in real business operations. For the latter, a request needs to be made to pick up goods from individuals or individual stores and transport them to corresponding customers. A hybrid distribution model requires logistics companies to perform one-to-one or one-to-many-to-one pickup and/or delivery services simultaneously, raising new challenges for the design and optimization of the distribution system.

Bergmann et al., 2020 analyzed the route efficiency trade-offs that emerge from combining first-mile pickup and last-mile delivery operations in an urban logistics network. They show that combining deliveries and pickups on a single route can significantly improve the distribution system efficiency. Integrating e-commerce proprietary logistics into one system instead of executing them separately may thus improve logistics distribution efficiency.

As the e-commerce market has grown, technology companies and traditional logistics providers have been experimenting with robot delivery (Kitjacharoenchai et al., 2020). The van-based robot service models studied in Chapter 3 and Chapter 4 are innovation models for companies addressing urban distribution. However, the models considered in Chapter 3 and Chapter 4, as well as other van-based robot routing models cited in Chapter 2, mainly focus on the delivery of goods, and very few studies involve pickup goods operations. To our knowledge, only Karak and Abdelghany, 2019 have studied a one-to-many-to-one problem with van-based robot pickup and delivery. The mothership leaves a depot to deliver commodities to customers, pick up commodities from the customers, and then transport them back to the depot.

We extend the van-based robot pickup and delivery research by integrating multiple pickup and delivery modes in one trip to increase the efficiency of distribution systems, as envisaged by Bergmann et al., 2020.

This chapter presents van-based robot hybrid pickup and deliveries (2E-VRHPD) as a prototypical problem. Larger vans carry small robots along the *1st-level route*. The robots travel along the *2nd-level open route*. Both van and robot can serve customers directly, but some constrained customers can be visited only by robots. The van stops at parking nodes to drop off and/or pick up its robot, replenish its robot and swap its robot's battery if needed. For hybrid pickup and delivery operations, vans and robots can load goods from a depot and deliver them to a customer, or they can pick up goods from a customer (supplier) and deliver them to another customer or to a depot.

Because of its involvement in pickup operations, the allocation and adjustment of goods during the route becomes a critical scientific problem. This holds especially for allocating goods to the van and its robot in the parking nodes before they leave to

do the distribution separately. The difficulties are twofold: (i) Both robots and vans can reach their capacity when they serve customers independently. However, the combined capacity of the robot and van cannot exceed the van's capacity when the robot is on board the van. Cargo load needs to be coordinated. (ii) Whether a couple of pickup-delivery-pair customers are in a van/robot route or not will influence the freight flow of the van/robot. The details of the differences are described in Section 5.2.3.

Our 2E-VRHPD model offers the following advantages over the vehicle routing problem with drones (VRPD) (model of Otto et al., 2018) and the truck-and-trailer routing problem with pickup and delivery (TTRPPD) (model of Battarra et al., 2014). (i) in our 2E-VRHPD model, we extend the VRPD model to cover hybrid pickup and delivery aspects, (ii) the 2E-VRHPD model extends TTRPPD with second-level open routes, and also introduces a new van and robot load trade-off problem, and (iii) small robots are generally safer than other transport modes in cities. For example, when a fast-flying drone stalls, it can cause serious accidents in a city, whereas for a slowly moving robot, accident risks from stalled vehicles are controllable (Yu et al., 2020).

This chapter makes the following contributions: 1. We offer a new two-echelon van-based robot routing problem with hybrid pickup and delivery, which has the potential to improve the efficiency of the distribution system. 2. We introduce five pickup and delivery modes in the van-based robot pickup and delivery problem and the van-robot load trade-off problem. 3. We first offer the 2E-VRHPD model and propose a mathematical formulation. 4. We propose an ALNS algorithm for the newly introduced problem and a capacity feasibility test approach. 5. We introduce a case study and perform a sensitivity analysis on the travel cost rate of a robot, maximum travel distances of a robot, and the impact of van no-go zones on the system. 6. We compare the 2E-VRHPD model with two classical models.

In the following, Section 5.2 describes the problem and formulation, Sections 5.3 deals with the adaptive large neighborhood search approaches, a computational study and a case study are presented in Section 5.4 and Section 5.5, respectively, and Section 5.6 concludes.

5.2 Problem description and formulation

5.2.1 Problem statement

There are customers located in narrow streets, on campuses, or in other communities where the entry of vans is restricted in the real world. We therefore distinguish two kinds of customers. One kind can be visited by either the a van or the a robot: we call these customers van customers. The other kind can only be visited by the a robot: we

call these robot customers. This implies that the a robot can serve all the customers, but the a van can only serve some of the customers.

We consider a two-echelon routing problem with van/robot pickup and delivery. Vans or vans carrying robots move along a *1st-level route*, serve van customers, or drop off/pick up, and replenish or swap their robots' batteries at parking nodes. Robots handle customer services along *2nd-level open routes*: in other words, the robots need not return to the parking node from where they set out.

In our hybrid pickup and delivery system, we define three kinds of pickup and delivery modes: (i) picking up goods from customers and delivering them to a depot, (ii) delivering goods from a depot to customers, and (iii) picking up goods from suppliers (or customers) and delivering them to other customers (or suppliers).

For the pickup-delivery-pair customers, we assume they have pairing (coupling) and precedence constraints. These constraints mean that the pick-up of goods from a customer and then delivery to other customers is one-to-one, and each customer's goods must be picked up before being delivered. We also let the pickup-delivery-pair customers be served by the same van-based robot mothership system.

In our 2E-VRHPD model, there are five pickup and delivery cases for the vehicles to serve the pickup-delivery-pair- customers. Figure 5.1 shows the five cases.

- Case 1: A robot picks up goods from a customer and delivers them to another customer.
- Case 2: A van picks up the goods from a customer and delivers them to another customer.
- Case 3: A van picks up goods from a customer and its robot then delivers them to another customer.
- Case 4: A robot picks up goods from a customer and returns to the van; the van then delivers the goods to another customer.
- Case 5: A robot picks up goods from a customer and returns to the van. The van carries the robot and then drops off the robot letting it deliver the goods to another customer.

These five cases stem mainly from substantial constraint restriction or optimization needs. For example, some pickup-delivery-pair goods can only be picked up and/or delivered by a robot.

We assume there are specific parking nodes for the vehicles, used for the van to rendezvous, replenish and swap the battery for its robot, rather than performing these operations at customer nodes. This is reasonable because some customer nodes cannot

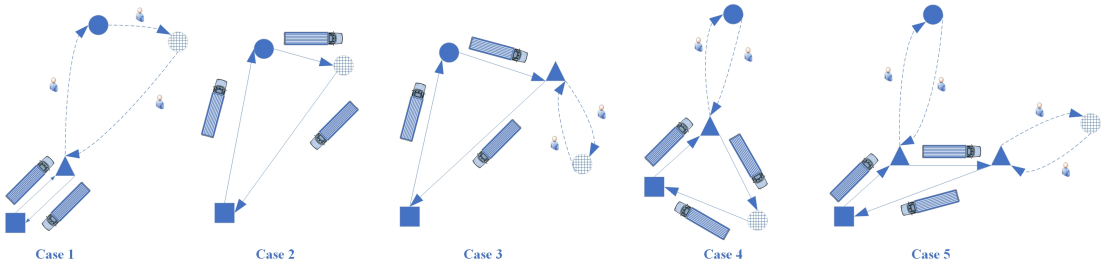

FIGURE 5.1: Pickup and delivery cases

be accessed by the van and some are unsuited to vehicles performing rendezvous operations in the real world. Parking nodes can be visited multiple times by vans/robots. However, we assume that rendezvous and other operations take a fixed amount of time.

The robot has a battery capacity. We note that the battery capacity of the robot determines the maximum distance it can travel by itself. Here we choose battery-swap technologies, and the robot can swap its battery when it meets its corresponding van. We also assume that the van has no maximum travel distance restrictions.

The robot can be replenished with goods from the van at parking nodes, but the van's freight must be loaded at the depot. We do not consider the case where a van replenishes other vans, or where robots replenish other robots. There are capacities for both the van and the robot. We assume that when the robot is on board the van, the van and its robot's total load cannot exceed the van's capacity. We also allow a robot to visit multiple customers during a dispatch instead of serving only one customer, since the robot's capacity is usually greater than a drone's.

Here we assume that each customer node must be visited by exactly one van/robot once. Customer nodes and the depot have their time windows.

We make the following assumptions: (i) each van can carry only one robot, (ii) the robot cannot leave the depot to serve customers directly, (iii) a robot dropped off by a van must be picked up by the same van, (iv) waiting at all locations is costless, and (v) the operating time spent dropping off, picking up, replenishing, and swapping a battery is a fixed value.

We define five types of route here for illustration, drawing on Chapter 4: (i) a van route (*1st-level route*) is a route where a van travels, (ii) a robot route (*2nd-level route*) is a route where a robot travels under its own power, (iii) a van-robot route is a route along which a van carries its robot, (iv) an independent-van route is a route where a van travels independently, and (v) a whole-robot route is a route traveled by a robot.

An example illustrating the problem is given in Figure 5.2. Triangles represent parking nodes, the square represents the depot, and circles correspond to customer nodes (including pickup-customers, delivery-customers, and pickup-delivery-pair-customers).

Green circles correspond to customers that can be served by vans and robots, and yellow circles correspond to customers that can be served only by robots. Solid lines correspond to van routes, dotted lines to robot routes. Circles with P/D/P(1)/D(1) correspond to pickup-customers, delivery-customers, pickup-pair-customer, and delivery-pair-customer, respectively.

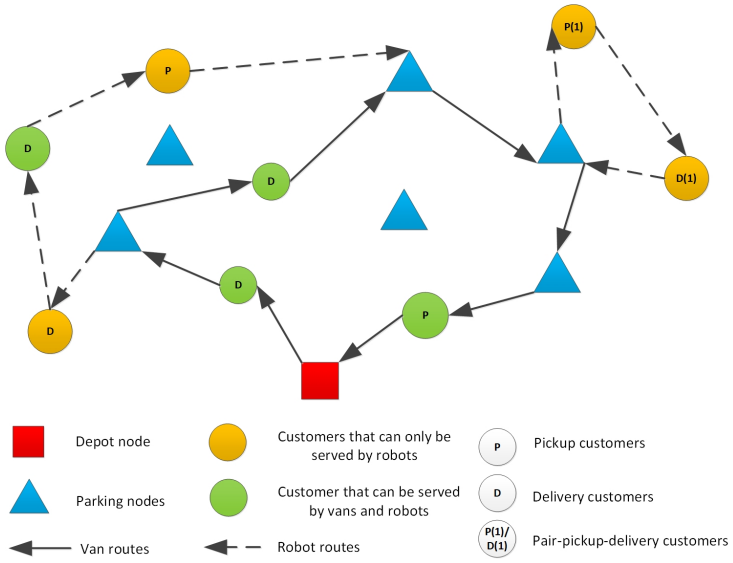

FIGURE 5.2: 2E-VRHPD model

5.2.2 Variable and Parameter Definitions

The problem is defined on a directed graph $G = (V, A)$, where the depot V_0 is represented by two nodes 0 and $0^{'}$. Every van route starts at 0 and ends at $0^{'}$. Let V_r be the set of parking nodes where vans can drop off, pick up, replenish and swap the battery for their robots. A parking node may be visited more than once. We use dummy nodes in the model. We use Num to represent the number of parking nodes in the model. V_{c1} denotes the van customer nodes set and V_c the customer nodes set. We have $V_{c1} \in V_c$, and $V_{c2} = V_c \setminus V_{c1}$ is the robot customer nodes set. The set $V_\alpha^0, V_\alpha^{'}, V_\alpha^{''}$ is the nodes set V_α union the depot $0, 0^{'}, 0 \cup 0^{'}$ respectively, and α can be expressed in a variety of different combinations of customer nodes and parking nodes sets. For the pickup-delivery-pair-customer, $V_{cp} = \{1, ..., l\}$ and $V_{cd} = \{l+1, ..., 2*l\}$. For pickup-customers or delivery-customers, $V_{cpd} = \{2*l+1, ...n\}$. Let $A_1 = \{(i,j) \mid i \in \{0\}; j \in V_{rc1}\} \cup \{(i,j) \mid i,j \in V_{rc1}, i \neq j\} \cup \{(i,j) \mid i \in V_{rc1}; j \in \{0^{'}\}\}$ be the van routes and let $A_2 = \{(i,j) \mid i \in V_r; j \in V_c\} \cup \{(i,j) \mid i,j \in V_c, i \neq j\} \cup \{(i,j) \mid i \in V_c; j \in V_r\}$ be the robot routes. We also let $A_3 = A_1 \cup A_2$ be the complete possible robot routes and $A_4 = A_3 \setminus A_1$ be the routes that the van cannot reach.

For each edge, $d_{i,j}$ is the associated travel distance, $c_1 * d_{i,j}$ ($c_2 * d_{i,j}$) is the associated travel cost and $d_{i,j}/v_1$ ($d_{i,j}/v_2$) is the associated travel time for the van (robot). The freight must be delivered from the depot $\{0\}$ to customer nodes i, or from pickup node V_{cp} to the corresponding delivery node V_{cd}, with the demand d_i and serving time s_i. We let $d_i < 0$ for pickup-customers and let $d_i > 0$ for the delivery-customers, and $d_i = -d_{i-l}$ for the pickup-delivery-pair customers. The time window of the customer nodes $i \in V_c$ is $[a_i, b_i]$, is the time interval where the service at node i is permitted to start. Let $[a_0, b_0] = [a_{0'}, b_{0'}]$, where $a_0/a_{0'}$ represents the earliest possible departure time from the depot $0/0'$ and $b_0/b_{0'}$ is the latest possible arrival time at the depot $0/0'$. These time windows are hard. $F_T = \{1, 2, .., k, .., K\}$ presents the set of vans, where K is the number of vans. $k \in F_T$ represents the robot belonging to the kth van. Let M be an arbitrary large constant number. Let C_1 be the capacity for the van and C_2 the capacity for the robot. Let T be the fixed operational time at depot or parking nodes. Let G be the battery capacity for the robot. Let h be the charge consumption rate of the robot. In addition, we introduce the following decision variables.

Let $x_{i,j,k}$ be equal to 1 if arc (i, j) in A_1 is traveled by the $kth - van$, 0 otherwise.

Let $y_{i,j,k}$ be equal to 1 if arc (i, j) in A_3 is traveled by the $kth - robot$, 0 otherwise.

Let $Q_{i,j,k}$ be equal to 1 if arc (i, j) in A_1 is traveled by the $kth - van$ with its robot on board, 0 otherwise.

Let e_i^k be the remaining battery level of the robot at node i on arrival.

Let W_i^k be the last arrival time for the $kth - van$ (or/and $kth - robot$) at node i.

Let $p_{i,j,k}$ be the freight flow of the robot in arc (i, j) in A_2.

Let $P_{i,j,k}$ be the freight flow of the van in arc (i, j) in A_1. If there is robot on board the van, let $P_{i,j,k}$ be the freight flow of the van and its robot.

Let w_i^k be the binary variable in parking nodes.

Let $Setp_i^k$ be the binary variable for the subroute. If there is a couple of pair-customers in the subroute, then we define $Setp_i^k = 0$, 1 otherwise. Where i is a pickup-pair-customer.

Let $pp_{i,j}^k$ be the specific freight flow of the robot in arc (i, j) in A_2. The specific freight flow is defined as the robot flow carries from the parking nodes.

Let $PP_{i,j}^k$ be the specific freight flow of the van in arc (i, j) in A_1. The specific freight flow is defined as the van flow carries from the parking nodes.

5.2.3 Mixed Linear Integer Programming Model

Objective

$$\min\Big(\sum_{k\in F_T}\sum_{(i,j)\in A_1} c_1 * d_{i,j} * x_{i,j,k} + \sum_{k\in F_T}\sum_{(i,j)\in A_3} c_2 * d_{i,j} * y_{i,j,k} - \sum_{k\in F_T}\sum_{(i,j)\in A_1} c_2 * d_{i,j} * Q_{i,j,k}\Big) \tag{5.1}$$

The objective function (5.1) minimizes the total travel cost. It corresponds to the van routes cost plus the whole-robot routes cost, minus the van-robot routes cost.

Arc constraints

Pure van/robot arc constraints guarantee which route type is allowed and which route type is not allowed. The arc constraints are similar to those in Chapter 4, described in Figure 4.2.

$$\sum_{(i,j)\in A_1} x_{i,j,k} \leq 1, \forall j \in V_{rc1}, k \in F_T \tag{5.2}$$

$$\sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(j,i)\in A_1} x_{j,i,k} = 0, \forall j \in V_{rc1}, k \in F_T \tag{5.3}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} \leq 1, \forall j \in V_{rc}, k \in F_T \tag{5.4}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} - \sum_{(j,i)\in A_3} y_{j,i,k} = 0, \forall j \in V_{rc}, k \in F_T \tag{5.5}$$

$$\sum_{i\in V_{rc1}} x_{i,0',k} = \sum_{j\in V_{rc1}} x_{0,j,k} = \sum_{i\in V_{rc1}} y_{i,0',k} = \sum_{j\in V_{rc1}} y_{0,j,k} \leq 1, \forall k \in F_T \tag{5.6}$$

$$\sum_{k\in F_T} (\sum_{(i,j)\in A_3} y_{i,j,k} + \sum_{(i,j)\in A_3} x_{i,j,k} - \sum_{(i,j)\in A_3} Q_{i,j,k}) = 1, \forall j \in V_c \tag{5.7}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} + \sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(i,j)\in A_1} Q_{i,j,k} \leq 1, \forall i \in V_{c1}^0, k \in F_T \tag{5.8}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} + \sum_{(i,j)\in A_1} x_{i,j,k} - \sum_{(i,j)\in A_1} Q_{i,j,k} \leq 1, \forall j \in V_{c1}^{'}, k \in F_T \tag{5.9}$$

$$\sum_{(i,j)\in A_3} y_{i,j,k} \leq \sum_{(i,j)\in A_1} x_{i,j,k}, \forall i \in V_r, k \in F_T \tag{5.10}$$

$$2 * Q_{i,j,k} \leq x_{i,j,k} + y_{i,j,k} \leq 2 * Q_{i,j,k} + 1, \forall (i,j) \in A_1, k \in F_T \tag{5.11}$$

$$x_{i,j,k}, y_{i,j,k}, Q_{i,j,k} \in \{0,1\}, \forall (i,j) \in A_3, k \in F_T \tag{5.12}$$

$$Q_{i,j,k} = 0, \forall (i,j) \in A_4, k \in F_T \tag{5.13}$$

$$x_{i,j,k} = 0, \forall (i,j) \in A_4, k \in F_T \tag{5.14}$$

$$\sum_{(i,j)\in A_3} x_{i,j,k} + \sum_{(i,j)\in A_3} y_{i,j,k} - \sum_{(i,j)\in A_3} Q_{i,j,k} == \sum_{(j,vd(i))\in A_3} x_{j,vd(i),k} + \sum_{(j,vd(i))\in A_3} y_{j,vd(i),k} - \sum_{(j,vd(i))\in A_3} Q_{j,vd(i),k}, \forall i \in V_P, k \in F_T \tag{5.15}$$

Constraints (5.2)-(5.14) are pure van/robot arc constraints. Constraints (5.2)-(5.6) ensure that every node is visited by a van/robot at most once, and the times of departures are equal to the times of arrivals. Constraints (5.6) force the number of a van leaving the depot to be equal to that of its robot leaving the depot, and equal to the number of van/robot coming back to the depot. Constraints (5.7) ensure that every customer

node is visited by vans or robots exactly once. Constraints (5.8)-(5.9) specify for customer nodes and depot, that a van and its robot cannot visit the same node unless the robot is on board the van. Constraints (5.10) ensure a robot cannot visit a parking node unless its van visited this node, and vice versa. Constraints (5.11) let $Q_{i,j,k}$ be equal to 1 only if $x_{i,j,k}$ and $y_{i,j,k}$ are all equal to 1. Constraints (5.12) are the binary variable constraints. Constraints (5.13)-(5.14) force the arc variable to be equal to 0 where they are not allowed to visit.

Constraints (5.15) are new constraints ensuring that the origin and destination nodes of a request are visited by the same mothership vehicles (van and/or robot).

Time constraints

Van and robot can serve different customers in parallel. However, the case where a van picks up/delivers a pair-customer and its robot delivers/picks up the corresponding pair-customer in the parallel route cannot occur, even if the mothership visits the pickup-pair-customer earlier than the corresponding delivery-pair-customer. Figure 5.3 is a simple example of such a forbidden case. Here we used time constraints to forbid cases where a van and its robot serve a couple of pair-customers in a parallel route.

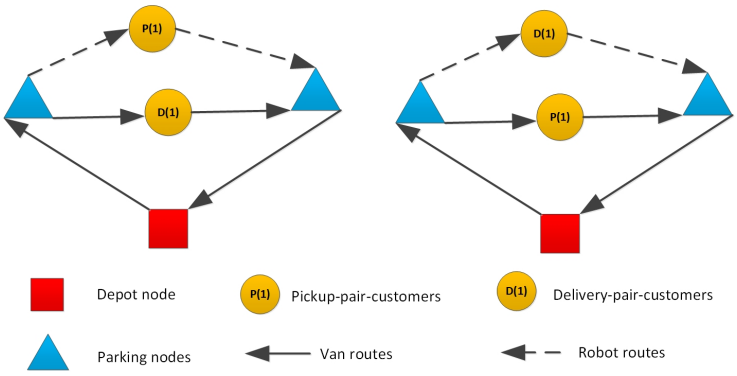

FIGURE 5.3: Forbidden cases for serving a couple of pair-customers in parallel route

$$W_i^k + T + d_{i,j}/v_1 - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_r^0 | (i,j) \in A_1\}, k \in F_T \tag{5.16}$$

$$W_i^k + T + d_{i,j}/v_2 - W_j^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{i \in V_r | (i,j) \in A_2\}, k \in F_T \tag{5.17}$$

$$W_i^k + d_{i,j}/v_1 + s_i - W_j^k \leq M(1 - x_{i,j,k}), \forall\{i \in V_{c1} | (i,j) \in A_1\}, k \in F_T \tag{5.18}$$

$$W_i^k + d_{i,j}/v_2 + s_i - W_j^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall\{i \in V_c | (i,j) \in A_2\}, k \in F_T \tag{5.19}$$

$$W_i^k + T - W_j^k \leq M(1 - y_{i,j,k}), \forall\{(i,j) \in A_3\}, k \in F_T \tag{5.20}$$

$$W_i^k \leq W_{i+l}^k - s_i - d_{i,i+l}/v_1, \forall\{i \in V_p\}, k \in F_T \tag{5.21}$$

$$a_i \leq W_i^k \leq b_i, \forall i \in V_c'', k \in F_T \tag{5.22}$$

$$\sum_{(i,o)\in A_2}(y_{i,o,k} - Q_{i,o,k}) - \sum_{(i+l,o)\in A_1}(x_{i+l,o,k} - Q_{i+l,o,k})) == 0 >> W_i^k <= W_j^k <= W_{i+l}^k, \exists j \in V_r, \forall i \in V_p, k \in F_T \tag{5.23}$$

$$\sum_{(i,o)\in A_1}(x_{i,o,k} - Q_{i,o,k}) - \sum_{(i+l,o)\in A_2}(y_{i+l,o,k} - Q_{i+l,o,k})) == 0 >> W_i^k <= W_j^k <= W_{i+l}^k, \exists j \in V_r, \forall i \in V_p, k \in F_T \tag{5.24}$$

Constraints (5.16-5.22) are the time constraints. Constraints (5.16-5.17) are time flow constraints for the van and robot leaving the parking node/depot. Constraints (5.16) model the time flow of the van route. Constraints (5.17) are the time flow constraints of the independent robot route. Constraint (5.18-5.19) represent time flow constraints for the van and robot leaving the customer node. Constraints (5.18) represent the time flow of the van route. Constraints (5.19) represent the time flow of the robot route. Constraints (5.20) model the time flow of the whole-robot route leaving the parking node. Constraints (5.21) ensure precedence constraints for the visiting of pickup-customers and delivery-customers. Constraints (5.22) enforce time window constraints for all nodes.

Constraint (5.23-5.24) model time and arc constraints to forbid the cases in Figure 5.3. Constraints (5.23) ensure that when a pickup-pair-customer is in a robot route, and its corresponding delivery-pair-customer is in an independent-van route, there exists at least one parking node visited by van after the pickup-pair-customer but before the delivery-pair-customer. Constraints (5.24) ensure that when a delivery-pair-customer is in a robot route, and its corresponding pickup-pair-customer is in an independent-van route, there exists at least one parking node visited by van after the pickup-pair-customer but before the delivery-pair-customer.

Freight and energy constraints

Drexl, 2020 described capacity considerations on subroutes for a one to one pickup and delivery truck and trailer routing problem. Drexl, 2020 pointed out that the following two quantities are relevant: the minimal truck load at decoupling and the subroute load balance at each position. Here we present a freight flow mathematical model for our 2E-VRHPD problem, which can also be used in truck and trailer routing problems.

If there are pair-customers in the subtour (van/robot route between two parking nodes), the freight flow may differ from the case where there are no pair-customers present. Figure 5.4 is a simple example for the freight flow in a robot route, with a pickup-pair-customer and a delivery-pair-customer in the robot subroute. We assume the capacity of the robot is 50. There is no couple of pair-customers in the subroute,

and the robot should load 40 when leaving the parking node in case (i). Before serving delivery-pair-customer 2, the robot should first serve pickup-pair-customer 1, so the freight flow in robot route exceeds the robot's capacity. There is a couple of pair-customers in the subroute in case (ii), and the robot could serve a pickup-pair-customer 1 and then serve a delivery-pair-customer 1, so the freight flow in the robot route will not exceed the robot's capacity of 50.

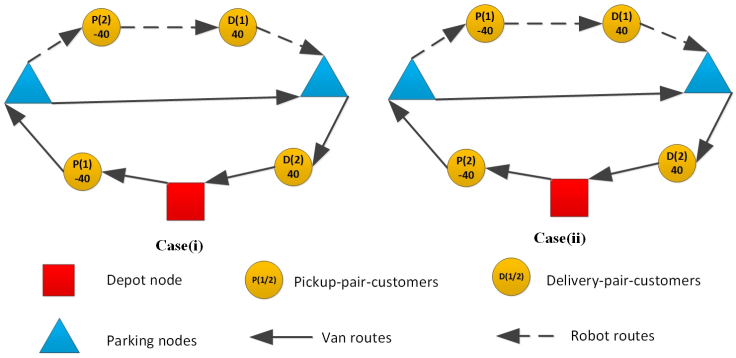

FIGURE 5.4: An example of freight flow in robot route

Freight constraints

$$\sum_{(i,j)\in A_2} p_{i,j,k} - \sum_{(j,i)\in A_2} p_{j,i,k} = d_j * \sum_{(i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}), \forall j \in V_c, k \in F_T \tag{5.25}$$

$$\sum_{(i,j)\in A_1} P_{i,j,k} - \sum_{(j,i)\in A_1} P_{j,i,k} = d_j * \sum_{(i,j)\in A_1} x_{i,j,k}, \forall j \in V_{c1}, k \in F_T \tag{5.26}$$

$$\sum_{(i,j)\in A_1} P_{i,j,k} - \sum_{(j,i)\in A_1} P_{j,i,k} = \sum_{(i,j)\in A_2} p_{i,j,k} - \sum_{(j,i)\in A_2} p_{j,i,k}, \forall j \in V_r, k \in F_T \tag{5.27}$$

$$\sum_{(i,j)\in A_1} P_{i,j,k} + \sum_{(i,j)\in A_1} p_{i,j,k} \leq C_1, \forall j \in V_r, k \in F_T \tag{5.28}$$

$$0 \leq p_{i,j,k} \leq C_2 * (1 - Q_{i,j,k}), \forall (i,j) \in A_1, k \in F_T \tag{5.29}$$

$$0 \leq p_{i,j,k} \leq C_2 * (y_{i,j,k} - Q_{i,j,k}), \forall (i,j) \in A_2, k \in F_T \tag{5.30}$$

$$0 \leq P_{i,j,k} \leq C_1 * (x_{i,j,k}), \forall (i,j) \in A_1, k \in F_T \tag{5.31}$$

$$Setp^k_{i+l} == 0 >> W^k_i \leq W^k_j \leq W^k_{i+l}, \forall i \in V_p, k \in F_T, \exists j \in V_R \tag{5.32}$$

$$Setp^k_{i+l} == 1 >> W^k_i \leq W^k_j \leq W^k_{i+l}, \forall i \in V_p, k \in F_T, \nexists j \in V_R \tag{5.33}$$

$$PP_{i,j,k} == P_{i,j,k}, \forall i \in \{V_r | (i,j) \in A_1\}, k \in F_T \tag{5.34}$$

$$pp_{i,j,k} == p_{i,j,k}, \forall i \in \{V_r | (i,j) \in A_2\}, k \in F_T \tag{5.35}$$

$$0 \leq PP_{i,j,k} \leq C1 * x_{i,j,k}, \forall (i,j) \in A_1, k \in F_T \tag{5.36}$$

$$0 \leq pp_{i,j,k} \leq C_2 * (1 - Q_{i,j,k}), \forall (i,j) \in A_1, k \in F_T \tag{5.37}$$

$$0 \leq pp_{i,j,k} \leq C_2 * (y_{i,j,k} - Q_{i,j,k}), \forall (i,j) \in A_2, k \in F_T \tag{5.38}$$

$$\sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = 0, \forall j \in V_p \cup \{V_{dp} | d_j \leq 0\}, k \in F_T \tag{5.39}$$

$$\sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = d_j * \sum_{(i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}), \forall j \in \{V_{dp} | d_j > 0\}, k \in F_T \tag{5.40}$$

$$Setp_j^k == 1 >> \sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = 0, \forall j \in V_d, k \in F_T \tag{5.41}$$

$$Setp_j^k == 0 >> \sum_{(i,j)\in A_2} pp_{i,j,k} - \sum_{(j,i)\in A_2} pp_{j,i,k} = d_j * \sum_{(i,j)\in A_2} (y_{i,j,k} - Q_{i,j,k}), \forall j \in V_d, k \in F_T \tag{5.42}$$

$$\sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = 0, \forall j \in V_p \cup \{V_{dp} | d_j \leq 0\} \cap V_{c1}, k \in F_T \tag{5.43}$$

$$\sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = d_j * \sum_{(i,j)\in A_1} (x_{i,j,k}), \forall j \in \{V_{dp} | d_j > 0\} \cap V_{c1}, k \in F_T \tag{5.44}$$

$$Setp_j^k == 1 >> \sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = 0, \forall j \in V_d \cap V_{c1}, k \in F_T \tag{5.45}$$

$$Setp_j^k == 0 >> \sum_{(i,j)\in A_1} PP_{i,j,k} - \sum_{(j,i)\in A_1} PP_{j,i,k} = d_j * \sum_{(i,j)\in A_1} (x_{i,j,k}), \forall j \in V_d \cap V_{c1}, k \in F_T \tag{5.46}$$

Energy constraints

$$e_j^k + d_{i,j} * h - e_i^k \leq M(1 - y_{i,j,k} + x_{i,j,k}), \forall \{i \in V_{rc} | (i,j) \in A_2\}, k \in F_T \tag{5.47}$$

$$0 \leq e_i^k \leq G, \forall i \in V_{rc}, k \in F_T \tag{5.48}$$

Constraints (5.25-5.31) model freight flow constraints. Constraints (5.25) represent the freight flow in the *2nd-level route* (robot route). Constraints (5.26) represent the freight flow in the *1st-level route* (van route). Constraints (5.27) ensure the conservation of cargo flow at parking nodes. Constraints (5.28) ensure that at the parking node the load of the van and the robot is less than the capacity of the van. Constraints (5.29-5.31) are the variable constraints.

Constraints (5.32-5.46) are new freight flow constraints for collaborative distribution of vehicles. Constraints (5.32-5.33) are used to mark whether there is a couple of pair-customers in a subroute (independent van/robot route). Constraints (5.32) ensure that if binary variable $Setp_{i+l}^k = 0$ for a delivery-pair-customers in the subroute, then there are no couples of pair-customers in the subroute. In other words, there exist at least one parking nodes visited by van after the pickup-pair-customer but before the corresponding delivery-pair-customer. Constraints (5.33) force that if binary variable $Setp_{i+l}^k = 1$ for a delivery-pair-customers in the subroute, then there is a couple of pair-customers in the subroute. In other words, there is no parking node visited by van after the pickup-pair-customer but before the corresponding delivery-pair-customer.

Constraints (5.34-5.38) represent the continuous variable constraints for the specific freight flow of the robot route. Constraints (5.34-5.35) ensure that when the van/robot

leaves the parking node, the specific freight flow is equal to the freight flow. Constraints (5.39) ensure that when the robot visits a pickup-customer (or pickup-pair-customer), the specific freight flow keeps unchanged. Constraints (5.40) ensure that when the robot visits a delivery-customer, the specific freight flow change is equal to the weight of goods. Constraints (5.41) ensure that when the robot visits a delivery-pair-customer, and its corresponding pickup-pair-customer is also in the subroute, the specific freight flow remains unchanged. Constraints (5.42) ensure that when the robot visits a delivery-pair-customer, and its corresponding pickup-pair-customer is not in this subroute, the change in the specific freight flow is equal to the weight of goods.

Constraints (5.43-5.46) model the continuous variable constraints for the specific freight flow of the van route. Details are not given.

Constraints (5.47-5.48) represent energy constraints for the robot.

Constraint linearization

Constraints (5.23-5.24) and constraints (5.32-5.33) can be linearized by using the binary variable w_i^k in parking nodes as follows. For example, constraints (5.49-5.51) can express the constraints (5.23).

$$Setp_{i+l}^k == 0 >> W_i^k \leq W_j^k + M * w_j^k, \forall i \in V_p, j \in V_r, k \in F_T \tag{5.49}$$

$$Setp_{i+l}^k == 0 >> W_j^k + M * w_j^k \leq W_{i+l}^k, \forall i \in V_p, j \in V_r, k \in F_T \tag{5.50}$$

$$\sum_{i \in V_r} w_i^k \leq Num - 1, \forall k \in F_D \tag{5.51}$$

Constraints (5.24), (5.32), and (5.33) can also be expressed using the same linearization methods.

5.3 Adaptive Large Neighborhood Search

We propose an adaptive large neighborhood search (ALNS) algorithm to solve the 2E-VRHPD problem. The main idea of ALNS is to iteratively apply a set of removal and insertion operators on an initial solution until the best solution is found (Ropke and Pisinger, 2006; Mourad et al., 2020).

Since our problem contains the pickup and delivery, two-echelon, van no-go customer, and VRPD part, we draw on some operations used in (Ropke and Pisinger, 2006; Mühlbauer and Fontaine, 2020; Anderluh et al., 2019; Sacramento et al., 2019). In our problem, we have parking stations for the vehicle, battery, and cargo transshipment, and the parking station is an independent station that can be visited multiple times. The framework of our ALNS is the same as the framework in Chapter 4, which contains simulated annealing (SA) acceptance criterion and weight-adjusting. We also

need to consider the pickup and delivery in the vehicle collaborative distribution scenario during destroy-repair operations. We focus mainly on forbidding the cases in Figure 5.3 and addressing the capacity constraints described in Figure 5.4.

The feasibility of a given route is also important in the ALNS algorithm. We focus mainly on the capacity feasibility in this section, since the time and energy feasibility of a given route is straightforward.

5.3.1 Initial solution

We start with a simple heuristic to generate initial feasible solutions. At the beginning of the heuristic, all customers are in an un-assigned customer list with random order. We then repeat the following two main steps until all the customer nodes are assigned.

1. We randomly choose a customer node in the un-assigned customer list. If the chosen node is a pickup/delivery-customer node and can only be served by robots, we construct a route with van routes and robot routes. If the chosen node is a pickup/delivery-customer node and can both be served by vans and robots, we decide whether to construct a route with van routes and robot routes or to construct a route with only van routes, according to a roulette-wheel mechanism. If the chosen customer is a pickup/delivery-pair-customer, we first find its corresponding pickup/delivery-pair-customer. We then construct a route based on the pickup-pair-customer. Next, we insert the corresponding delivery-pair-customer into the random-feasible position of the constructed route. If the delivery-pair-customer can not be inserted into the constructed route, we try to build a new robot route from the constructed route.

2. The second step is to randomly sort the order of the un-assigned customers list, then choose a customer to insert into the constructed route's random-feasible position until no customer node can be inserted. We note that for the pair-customer, we need to insert customers according to the precedence constraint in order. For example, in the process of a van launching and recycling its robot, the van serving a pickup/delivery pair-customer and its robot serving a corresponding customers is not allowed.

To this end, the feasibility of the returned solution, in terms of request time windows, capacity, energy, reachability synchronization, couple and precedence constraint, is assured. This initial feasible solution can then be improved by the ALNS operators as it does not lead to optimization route. We describe the destroy and repair operators used by the ALNS algorithm in the following subsections.

5.3.2 Destroy operators

The destroy operation destroys a chosen 2E-VREC route by removing nodes or routes. We define six destroy operators:

1. **Random customer removal** operator randomly removes customer nodes from a 2E-VRHPD route. If the removed customer is a pair-customer, we remove its corresponding pair-customer node.

2. **Greedy customer removal** operator removes the customer that can yield the largest cost reduction for a given route. If the removed customer is a pair-customer, we remove its corresponding pair-customer node.

3. **Pair-customer removal** operator removes a couple of pair-customers (pickup-pair-customer and its corresponding delivery-pair-customer) that can yield the largest cost reduction for a given route.

4. **Station-route removal** operator randomly removes a parking station, and the robot routes depart from and arrive at this station.

5. **Random route destruction** operator randomly selects a 2E-VRHPD route in the solution to destroy.

6. **Greedy route destruction** operator selects a 2E-VRHPD route,with a minimum number of customer nodes to destroy.

5.3.3 Repair operators

The route repair operator repairs the existing routes or constructs a new 2E-VRHPD route if needed, using five repair operations. The repair operations do not allow infeasible solutions but allow worse solutions through simulated annealing evaluation and calibration.

1. **Route reconstruction** operator ensures that we always get a viable solution. We adopt the same method as used to get the initial solution in Section 5.3.1.

2. **Random customer insertion** operator randomly chooses a customer to insert into a random-feasible position in a given route until all customers have been tried. If the inserted customer is a pickup/delivery-pair-customer, we randomly insert its corresponding delivery/pickup-pair-customer node after/before this customer.

3. **Greedy customer insertion** operator randomly chooses a customer to insert into a position in the route with the least cost increases. If the inserted customer is a

pickup/delivery-pair-customer, we greedy insert its corresponding delivery/pickup-pair-customer node after/before this customer.

4. **Random station-route insertion** operator randomly chooses a station to insert into a given route and then generates a robot route from this station with one customer node. If the customer is a pickup/delivery-pair-customer, we randomly insert its corresponding delivery/pickup-pair-customer node after/before this customer.

5. **Greedy station-route insertion** operator chooses a station with least total travel cost increases to insert into a given route and then generates a robot route from this station with one customer node. If the customer is a pickup/delivery-pair-customer, we greedy insert its corresponding delivery/pickup-pair-customer node after/before this customer.

5.3.4 Capacity feasibility tests

We draw on the capacity feasibility checking approach of (Drexl, 2020), who studied the one-to-one pickup-and-delivery problem with time windows and trailers. A 2E-VRHPD route usually includes single van-robot routes and parallel routes (contains the independent-van route and robot route). For a single van-robot route, the capacity feasibility test is straightforward, so we focus on testing the capacity feasibility of the parallel routes.

A parallel path is capacity-feasible if and only if the capacity in the parking nodes, together with the independent-van route and robot route, are all feasible. At each parking node for dropping off or picking up the robot, the van and robot's total load should be less than the van's capacity. In a parallel path, capacity considerations on the independent-van route and robot route, the following four amounts usually need to be considered:

(i) How many goods a robot must carry before leaving the parking nodes. This load includes the delivery-customers and the delivery-pair-customers whose corresponding pickup-pair-customer is not in this robot route.

(ii) How many goods a robot can carry at most before leaving the parking nodes. This load should be as large as possible provided the goods do not exceed the robot's capacity on each customer node during a robot route.

(iii) How many goods a van must carry before leaving the parking nodes. This load includes the delivery-customers and the delivery-pair-customers whose corresponding pickup-pair-customer is not in this independent-van route.

(iv) How many goods a van can carry at most before leaving the parking nodes. This load should be as large as possible provided the goods do not exceed the van's capacity on each customer node during an independent-van route.

One easy way to check the capacity feasibility of a parallel route is to ensure the robot capacity in the robot route and let the robot carry as many goods as possible subject to ensuring the van capacity feasibility for the independent-van-route. We then check the capacity feasibility in the parking node used for picking up the robot.

Algorithm 6 is the algorithm to check the capacity of the parallel route. The input is the van and robot's total load (FreightAmount) at the parking node, the parallel routes (RouteRoute and IndependentVanRoute), and the BasicData (Line 1). The BasicData includes the van customer nodes, the demand of customer node ($Load[node]$), the capacity of van (C_1) and robot (C_2), and a large value ($LargeValue$). We note that $Dict[node]$ represents the corresponding pair-customer of the node. We first calculate how many goods a robot must carry before leaving the parking nodes (Line 2 - Line 13). We then test the feasibility of the RobotRoute and calculate the 'Gap' between the number of goods the robot must take and the number of goods the robot can take at most. (Line 14 - Line 29). We further calculate how many goods a van must carry before leaving the parking nodes (Line 30 - Line 41). Next, we calculate how many FreightAmount the van must carry according to the Gap (Line 42 - Line 46), and we test whether the independent-van route's capacity is feasible (Line 47 - Line 54). Finally, we test whether the van and robot's total load in the parking node for picking up exceeds the van's capacity (Line 55 - Line 59).

5.4 Computational study

We implemented extensive computational studies to assess our solution method's performance and evaluated the influence of parking node density on solutions. First, we explained how we generate test instances, and we described the different parameters used in Section 5.4.1. Second, we compared our ALNS results with CPLEX results to assess our algorithm's performance and assessed the influence of parking node density on the solution in Section 5.4.2.

The mathematical programming model was coded in DOCPLEX by calling CPLEX 12.8 to solve the MIP model. The ALNS was coded in python version 3.6.8. Both CPLEX and python were executed on an Intel(R) Core(TM) 2.8GHz processor with 16 GB of memory running under Windows 10. Python was run with single-threading.

Algorithm 6 *TestParallelRouteCapacity*()

```
Input: FreightAmount, RobotRoute, IndenpedentVanRoute, BasicData
for node in RobotRoute do
    RobotFreight = 0
    if node is DeliveryCustomerNode then
        RobotFreight += Load[node]
    end if
    if node is DeliveryPairCustomer and Dict[node] is not in RobotRoute then
        RobotFreight += Load[node]
    end if
end for
if RobotFreight > C2 then
    return False
end if
FreightAmount -= RobotFreight
TempGap = LargeValue
for node in RobotRoute do
    if node is CustomerNode then
        RobotFreight -= Load[node]
        if RobotFreight > C2 then
            return False
        else
            Gap = C2 - RobotFreight
            if Gap < TempGap then
                TempGap = Gap
            end if
        end if
    end if
end for
Gap = TempGap
for node in IndependentVanRoute do
    VanFreight = 0
    if node is DeliveryCustomerNode then
        VanFreight += Load[node]
    end if
    if node is DeliveryPairCustomer and Dict[node] is not in IndependentVanRoute then
        VanFreight += Load[node]
    end if
end for
if FreightAmount < VanFreight then
    return False
end if
if FreightAmount - Gap < VanFreight then
    FreightAmount = VanFreight
else
    FreightAmount -= Gap
end if
for node in IndependentVanRoute do
    if node is CustomerNode then
        FreightAmount = FreightAmount - Load[node]
        if FreightAmount > C1 then
            return False
        end if
    end if
end for
FreightAmount = FreightAmount + RobotFreight + Gap
if FreightAmount > C1 then
    return False
end if
return True, FreightAmount
```

5.4.1 Instance generation and parameters

For testing the proposed solution approach, we used the instances of Dellaert et al., 2018 and Yu et al., 2020 as a basis. The specific instance generation approach is described in those two articles.

We set the speed of the van at 2, and the speed of the robot at 1. The capacity of a robot was 50, and the capacity of a van was 200. When the robot was on board the van, the van and robot's total capacity was 200. The battery capacity of a robot was 100. The robot's energy consumption rate was 1 unit per distance. The depot time window was [0,500]. Vans could access two-thirds of the total customer nodes in each instance. If the calculated number of van customer nodes was fractional, we rounded up. The first two-thirds of customers in the instance were the van customers. We chose $2 * math.ceil(n/8)$ customers as the pair-customers, in which n was the total number of customers. When choosing a couple of pair-customers, we ensured that a van and its robot could feasibly visit a pair of pickup-delivery-pair-customers.

The set of parameters used in the computational study along with their descriptions and values are presented in Table C.1.

5.4.2 ALNS experiments

We first compared the ALNS results with the CPLEX results in small-scale instances to see the proposed ALNS algorithm's overall performance. Next, we analyzed the influence of parking node density on the solution and then obtained a suitable number of parking nodes for the model-comparison below.

Comparison of CPLEX and ALNS

The VRPD model is hard to solve for general-purpose MIP solvers (Wang and Sheu, 2019). In our problem, we replicated the parking nodes for multiple visiting. Dummy nodes increase the number of nodes in the model, making it more difficult to solve and reduce the problem size we can address.

We tested 20 instances with five customer nodes and three parking nodes using CPLEX 12.8, and used the dummy node iterates-growth steps as used by Chapter 4: We iteratively solved our model by CPLEX, gradually increasing the number of dummy nodes. The increase in the number of dummy nodes stops when no improvement in the solution cost is found. However, if the solution is still worse than the ALNS solution, we continue to increase the number of dummy nodes until the CPLEX solution is better than or as good as the ALNS solution. The iterative process will also stop if the procedure runs out of solving time or out of memory.

For each instance, CPLEX 12.8 runs with default settings until it finds an optimal solution, until it reaches the predetermined maximum computation time (10800 s), or until the program runs out of memory. The computational results for all instances are presented in Table 5.1, in which *CplexK* represents the number of times a vehicle can visit a parking node, and UB is the upper bounds of CPLEX solutions. E1 is the CPLEX gap between CPLEX upper bound (baseline) and CPLEX lower bound, and TIME is the CPLEX solving time. BC and AC are the best and average solutions for ALNS in 10 tests. SDC and AT are the standard deviation and solving time of ALNS in 10 tests.

TABLE 5.1: CPLEX and ALNS results comparison

	Cplex K=2			Cplex K=3			Cplex K=4			ALNS			
	UB	E1(%)	TIME(s)	UB	E1(%)	TIME(s)	UB	E1(%)	TIME(s)	BC	AC	SDC	AT(s)
CA1	262.0	0.0	179.6	262.0	0.0	9801.0	253.3	13.1	10800.0	253.3	258.6	4.5	97.1
CA2	365.8	0.0	784.3	365.8	47.8	10800.0	/	/	/	365.8	365.8	0.0	122.9
CA3	333.8	0.0	5933.6	333.8	53.1	10800.0	/	/	/	333.8	353.6	7.0	206.1
CA4	261.2	0.0	23.2	261.2	0.0	3379.2	261.2	43.4	10800.0	261.2	261.2	0.0	94.1
CA5	171.6	0.0	5.1	167.7	0.0	25.5	167.7	0.0	25.4	167.7	169.7	2.0	83.8
CB1	249.6	0.0	93.6	249.6	0.0	8196.2	249.6	21.3	10800.0	249.6	249.6	0.0	107.8
CB2	393.0	0.0	226.1	403.4	59.0	10800.0	/	/	/	**366.9**	366.9	0.0	128.9
CB3	346.2	0.0	6454.5	346.2	49.7	10800.0	/	/	/	346.2	356.7	7.3	133.4
CB4	261.2	0.0	13.3	261.2	0.0	57.0	261.2	0.0	446.7	261.2	261.2	0.0	79.7
CB5	151.0	0.0	6.9	151.0	0.0	136.7	151.0	45.8	10800.0	151.0	151.0	0.0	85.5
CC1	183.8	0.0	11.2	183.8	0.0	39.4	183.8	0.0	852.8	183.8	183.8	0.0	67.9
CC2	289.5	0.0	241.3	289.5	31.3	10800.0	/	/	/	289.5	289.5	0.0	87.0
CC3	297.7	0.0	1012.3	297.7	41.2	10800.0	/	/	/	297.7	309.6	1.1	89.0
CC4	333.0	0.0	839.6	337.9	41.1	10800.0	/	/	/	**277.6**	277.6	0.0	88.1
CC5	140.7	0.0	4.6	138.3	0.0	35.1	138.3	0.0	80.7	138.3	131.2	0.0	57.9
CD1	276.4	0.0	30.2	276.4	0.0	270.4	276.4	0.0	464.6	276.4	276.4	0.0	88.7
CD2	353.4	0.0	32.5	353.4	0.0	2044.2	353.5	22.1	10800.0	353.5	364.6	3.9	111.8
CD3	271.3	0.0	19.5	271.3	0.0	391.6	271.3	0.0	1623.3	271.3	277.7	5.3	98.2
CD4	253.9	0.0	19.7	253.9	0.0	10356.0	253.9	38.8	10800.0	**231.9**	238.5	6.6	94.9
CD5	151.0	0.0	6.6	151.0	0.0	104.4	/	/	/	151.0	151.0	0.0	68.4
AVER	267.3	0.0	796.9	/	/	/	/	/	/	261.4	264.7	1.9	99.6

Table 5.1 shows that our ALNS can always reach the best solutions in very small instances. For case CB2, CC4, CD4, ALNS performed better than the CPLEX solver, which ran out of time in K = 3 or K = 4. We note in ALNS solution of case CB2, CC4, and CD4, K = 3, K = 4, and K = 4 dummy parking nodes were replicated, respectively.

Assessment of the influence of parking node density on the solution

As the number of parking points in the city is likely to affect the delivery efficiency of the 2E-VRHPD system, we analyzed the impact of parking node density (the number of parking nodes) changes on the results.

We chose 20 instances with one depot and 15 customers to conduct the sensitivity analysis experiments. The random distribution scenario was adopted for the parking

node distribution. To quantify the impact of parking node density, we gradually increased the number of parking nodes from 10, 20, up to 50.

Table 5.2 reports the influence of parking node density on the solutions. Row 1 shows the number of parking nodes. Row 2 shows the average best result for the instances. Row 3 reports an average percentage gap between the baseline and experimental group solution.

TABLE 5.2: Influence of parking node density on the solution

Number of parking nodes	10	20	30	40	50
Aver Best Objective	530.8	520.5	507.5	504.5	504.3
Gap between baseline	baseline	1.9%	4.4%	5.0%	5.0%

Table 5.2 shows that the cost function of the system presents a rising trend with the increase in the number of parking nodes. Increasing the number of parking nodes from 10 to 30 had a strong effect on reducing the objective function. Increasing the number of parking nodes from 30 to 50 had little impact on reducing the objective function. Therefore, planning an appropriate number of stops can bring the 2E-VRHPD distribution system's efficiency to a high level. At the same time, there is no need to occupy too many public resources.

5.5 Case study

In this section, we first built realistic scenarios and instances for the case study. We then conducted a sensitivity analysis experiment on the robot's travel cost rate and maximum travel distances and assessed the effect of the restricted area (that are van can not access) on the 2E-VRHPD model. Finally, we compared the 2E-VRHPD model with the other two classical models.

5.5.1 Scenario construction

We examined our distribution concept, its model, and the proposed solution algorithm on realistic scenarios.

Figure 5.5 is an example of a realistic scenario (depot and distribution range) for Xi'an city used in our case study. The distribution range (colored in blue) is the southwest corner of Xi'an (China). The depot we chose is marked in red. It is the Xi'an Intelligent Logistics depot of JD.com.

The van no-go areas, parking nodes, customer nodes, and satellites were set as follows.

Van restricted areas: Van no-go areas were selected within the city walls of Xi 'an and seven universities in the southwest corner of Xi 'an.

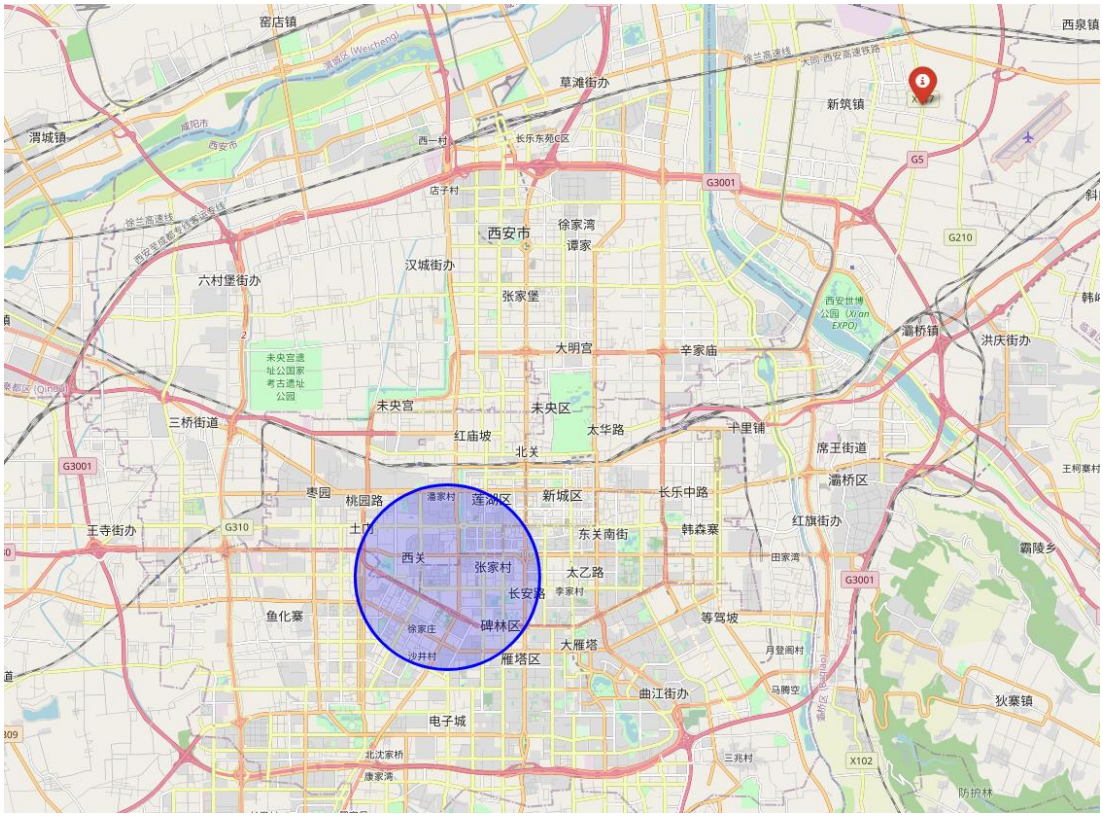

FIGURE 5.5: Example of the distribution range and depot in Xi'an city

Supplier locations: For suppliers, here we only considered four supermarkets, namely Wal-mart, RT-Mart, Yonghui Superstores, and Renrenle Supermarket. The Wal-mart is in the van restricted areas.

Parking node: Some parking nodes were selected near the campuses and city walls, where the van could park. We also randomly selected some parking nodes in the distribution area.

Customer nodes: We randomly generated some customer nodes on the distribution area and ensured that some customers were within van no-go areas.

Satellites: The satellites wer chosen as the JD.com Xi'an GaoXin satellite, YouYi satellite, and ZaoYuan satellite, responsible for serving our selected distribution range. The satellite was only used in model comparisons part.

Figure 5.6 is a detailed example of the scenarios of Xi'an city used in our case study. The van no-go zone is filled in the red area on the map. The blue nodes represent the parking nodes we chose, the black dots represent the suppliers, the gray-green dots represent customer nodes, and the orange dots represent satellites.

The basic parameters used in the case study were set as follows.

According to Yu et al., 2020, simply improving the robot's speed can improve the solution, but improvements remain limited in the complex constraints model. We set

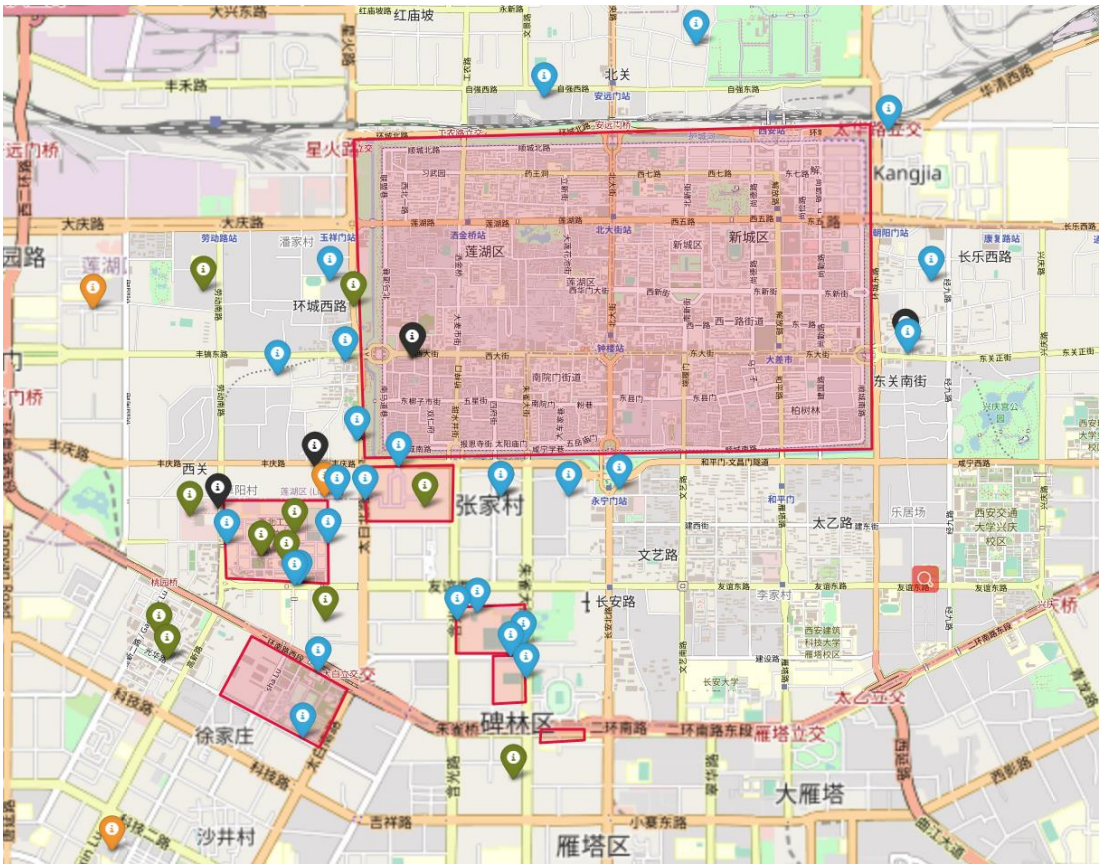

FIGURE 5.6: Example of van no-go zone, parking nodes, suppliers, customer nodes, and satellites in Xi'an city

the robot's speed equal to 5km/h, and the van's speed was 25km/h, suggested by (Yu et al., 2020).

We reset the robot's maximum travel distances equal to 10km. We used artificial customer demands selected in [-20,-10,10,20], and vehicle capacities of 50 for robot and 200 for the van. We used a planning horizon of 8 hours. Accordingly, the depot time windows were set to [0,8 h]. We randomly selected typical values for the customer time windows from [30 min, 8 h], and the service times were 10 min. The associated travel cost rate for a van was 3/km, and the robot's associated travel cost rate was 0.3/km since the robot is smaller than the van and electrically powered.

We built ten instances, and each instance was run ten times in the following 2E-VRHPD model analysis experiments and model comparison experiments. The instances were generated as follows. For ten instances, the locations of the depot, satellites, suppliers, and parking nodes were the same. We set one depot, three satellites, four supplier nodes, and 30 parking nodes for each instance. For customer nodes, we randomly generated 100 customer nodes in the distribution range and ensured that 1/3 of the customer nodes were in the van no-go zone. These 100 customer nodes constituted the customer node pool. We then randomly chose 15 customer nodes from the customer node pool for each instance. We ensured there were customers in the van no-go zone for each instance. We note that some pickup-pair-customers were chosen from suppliers.

5.5.2 2E-VRHPD model analysis experiments

In 2E-VRHPD model analysis experiments, we analyzed the impact of the travel cost rate (TCR) for robots, the maximum travel distance (MTD) of the robot, and the travel cost rate and the maximum travel distance combination (TCR-MTD) of the robot on the mode-computed outputs. We also compared the effects of no-go zones on the solution: we assumed the van could also access the no-go areas in the 2E-VRHPD model, called the 2E-VRHPD-NR model, and compared it with the 2E-VRHPD model, to see how the no-go areas would affect the output of the system.

The experiments were conducted as follows.

Experiment 1: We gradually increased the travel cost rate for the robot from 0.3/km, 0.6/km,.., 3.0/km in 2E-VRHPD and 2E-VRHPD-NR models.

Experiment 2: We gradually increased the maximum travel distance of the robot from 10km, 15km, 20km, ..., to 55km in the 2E-VRHPD and 2E-VRHPD-NR models.

Experiment 3: We gradually increased the travel cost rate for the robot and the maximum travel distance of the robot simultaneously from (0.3/k, 10km), (0.6/km, 15km),...,(3.0/km,55km) in the 2E-VRHPD and 2E-VRHPD-NR models.

Table 5.3 reports the results of sensitivity analysis on different of TCR, MTD, and TCR-MTD combinations of robots. The NR represent the 2E-VRHPD-NR model as shorthand in the table. The e1, e2, and e3 represent the gap between the cost of the 2E-VRHPD model (baseline) and the 2E-VRHPD-NR model in TCR, MTD, and TCR-MTD experiments, respectively.

TABLE 5.3: Impact of TCR, MTD, TCR-MTD and no-go zone constraint

TCR				MTD				TCR-MTD			
TCR	2E-VRHPD	NR	e1	MTD	2E-VRHPD	NR	e2	TCR-MTD	2E-VRHPD	NR	e3
0.3	159.88	155.85	2.52	10	159.88	155.85	2.52	0.3,10	159.88	155.85	2.52
0.6	165.88	160.84	3.03	15	158.70	152.77	3.74	0.6,15	162.43	158.05	2.70
0.9	170.23	164.00	3.66	20	157.89	150.79	4.50	0.9,20	167.76	160.49	4.34
1.2	173.60	167.74	3.37	25	157.91	150.02	5.00	1.2,25	172.30	165.54	3.93
1.5	176.49	169.87	3.76	30	157.87	149.84	5.09	1.5,30	174.32	168.28	3.47
1.8	179.97	171.97	4.44	35	157.87	149.37	5.38	1.8,35	178.88	171.58	4.08
2.1	181.87	173.93	4.36	40	156.76	149.51	4.63	2.1,40	181.67	174.22	4.10
2.4	184.84	175.09	5.28	45	157.66	149.41	5.23	2.4,45	184.36	175.31	4.91
2.7	187.45	175.93	6.15	50	157.28	149.53	4.93	2.7,50	186.36	175.24	5.97
3.0	189.33	176.36	6.85	55	157.29	149.52	4.94	3.0,55	188.46	176.42	6.39

For the sensitivity analysis on TCR (Columns 1-4 in Table 5.3), we have found that the no-go area constraint of the van would probably reduce the efficiency of the 2E-VRHPD distribution system by 2-7%. With the increase in TCR, e1 tends to increase. Besides, with the rise in TCR, TCR's impact on the distribution system is gradually decreasing.

For the sensitivity analysis on MTD (Columns 5-8 in Table 5.3), increasing MTD could increase the system's efficiency, but the increase was limited in the 2E-VRHPD model.

For the sensitivity analysis on TCR-MTD (Column 9-12 in Table 5.3), we found that no-go areas of the van would probably reduce the efficiency of the distribution system by 2-7%.

We knew that the van's no-go restriction had no more than a 7% effect on the system from the above experiments. Increasing the driving distance of robots can improve system efficiency, but the overall improvement is tiny. The unit mileage of a robot has a strong influence on the system cost.

5.5.3 Comparison 2E-VRHPD model with classical models

We compared the 2E-VRHPD model with a parallel van and robot scheduling problem with hybrid pickup and delivery operations (PVRSP-HPD) and model with a two-echelon vehicle routing problem with hybrid pickup and delivery operations (2E-VRP-HPD). We also allowed the van to visit all the customers in the PVRSP-HPD model, used different types of robots, and added the fixed cost of using van, robot, and depot in the 2E-VRP-HPD models to see the influence.

Comparison with PVRSP-HPD model

The PVRSP-HPD model is similar to the parallel drone scheduling TSP (PDSTSP) introduced by (Murray and Chu, 2015), but has hybrid pickup and delivery operations. In the PVRSP-HPD model-setting, the van and robot went alone from the depot to serve customers, and then returned to the depot. Van/robot multiple visits to the depot were not allowed. The other constraints of van, robot, and customers were the same as those in the 2E-VRHPD model.

Through the pre-analysis of instances we found all ten instances were unworkable in the PVRSP-HPD model, because almost no robot customers or pair customers with robot customers could be successfully served by robots (beyond the maximum travel distance of robots).

We tried to gradually increase the robot's maximum travel distance from 10 km to 100 km (far enough); however, instances were also unworkable because robot delivery violated some customer time windows. It is obviously not feasible to unilaterally increase the robot's speed to a value high enough to fit the customer's time window.

In the next experiments, we relaxed the PVRSP-HPD model's constraints to allow vans to access the no-go areas (PVRSP-HPD-NR model). Hence all instances would become feasible.

However, the distribution system efficiency may be affected due to the high cost per unit van mileage. The experimental results show the average best cost for all the instances was 184.47.

We found that even though we increased the travel cost rate for the robot in the 2E-VRHPD model to 1.5/km (half the consumption of vans), the cost was smaller than that for the PVRSP-HPD-NR model (176.49 compared to 184.47). Hence our model is better than the PVRSP-HPD model and to some extent better than the PVRSP-HPD-NR model.

Comparison with 2E-VRP-HPD model

The 2E-VRP-HPD model is similar to the two-echelon vehicle routing problem (2E-VRP) introduced by (Perboli et al., 2011), but has hybrid pickup and delivery operations. In the 2E-VRP-HPD model, vans transport goods from the depot to the satellites and then transport recycled goods from satellites to the depot. Robots travel from satellites to serve customers and then back to satellites. We assumed that there were enough robots in the satellites. The other constraints of van, robot, and customers are the same as those in the 2E-VRHPD model.

Through the pre-analysis of instances, we found that there might be instances that are unworkable in the 2EVRP-HPD model setting. Robots still could not complete the delivery task because some customers or pair-customers lay beyond the robot's maximum travel ranges (10 km).

We then provided different types of robot to see how this would work in the 2E-VRP-HPD model. The types of robot are shown in Table 5.4. Row 1 shows five types of robots. Row 2 represents the maximum travel distances of each kind of robot, and Row 3 gives the travel cost rate of each type of robot.

TABLE 5.4: Types of robot

	Robot L1	Robot L2	Robot L3	Robot L4	Robot L5
Maximum travel distances	10km	15km	20km	25km	30km
Travel cost rate	0.3/km	0.6/km	0.9/km	1.2/km	1.5/km

In our computational study, we used the type of robot that would just do the hybrid pickup and delivery task, and then looked at the system's efficiency. In other words, we assumed there were different types of robots in the satellites, and multiple types of robots could be mixed-used in an experiment.

Experimental results show the average best cost of the 2E-VRP-HPD model with different types of robots was 139.43, better than the average best cost of the 2E-VRHPD model, which cost was 159.88.

We suspected that the travel cost rate might affect the comparison results of the model. We therefore conducted a sensitivity analysis of the robot's different TCR from 0.3, 0.6,..., 3.0 in the 2E-VRP-HPD model like in Section 5.5.2. We found that when TCR was equal to (or larger than) 2.1, the average best cost of the 2E-VRP-HPD model was higher than that of the 2E-VRHPD model: 185.37 for the 2E-VRP-HPD model compared to 181.87 (see Table 5.3) for the 2E-VRHPD model.

Hence, when the objective function is total energy consumption, the model output is closely related to the robot's travel cost rate.

In the above 2E-VRHPD and 2E-VRP-HPD model comparison, we did not consider the fixed cost of robot and van, or the satellites. We therefore added the fixed cost of using van, robot, and satellite into the objective of the two models to see how the fixed cost influenced the model-output.

We set at 10 the cost of using one satellite once, at 10 the cost of using one robot once, and at 20 the cost of using one van once. The experimental results show that the average best cost of the 2E-VRHPD model was equal to 201.88, and the average best cost of 2E-VRP-HPD was equal to 205.43. In this case, the 2E-VRHPD model is more competitive. Hence the price of corporate logistics infrastructure will also strongly affect the output of 2E-VRHPD and 2E-VRP-HPD models.

We therefore advocate using the 2E-VRHPD model for distribution in appropriate scenarios to improve the system's efficiency.

5.6 Conclusion and future research directions

Recent demonstrations by JD.com and Starship Technologies (among others) have shown the potential of robots for small parcel pickup and delivery services. In addition, the service demand of logistics companies has begun to diversify, requiring the combination of multiple modes of pickup and delivery operations in one trip. Current research also shows that combining pickup and delivery operations in one trip can improve the distribution system's overall efficiency.

In this chapter, a van-based robot logistic distribution system that combines hybrid pickup and delivery operations was studied. Larger vans carry small robots along the van route. The robots travel along with the robot open route. The van and robot can serve customers directly, but some constrained customers can be visited only by robots. The van stops at parking nodes to drop off and/or pick up its robot, replenish its robot, and swap its robot's battery if needed. Five detailed pickup and delivery cases in the 2E-VRHPD model were introduced. A van and/or robot can load goods from a depot and deliver them to a customer, or a van and/or robot can pick up goods from a customer and take them to another customer, or a van and/or robot can pick up goods from a customer and take them to a depot.

The associated optimization problem was formulated as a two-echelon heterogeneous vehicle routing problem with hybrid pickup and delivery, time windows, and vehicle collaborations. An MIP formulation including arc, time, freight, and energy parts, along with a simple ALNS-based heuristic approach solving large instances that have been introduced. The capacity feasibility test approaches are proposed for a given 2E-VRHPD route. An extensive computational study to evaluate the performance of the proposed ALNS approaches is presented. How the parking node density influences the output solution was studied. We constructed a case study based on realistic scenarios. We conducted a sensitivity analysis on the robot's travel cost rate and maximum travel distances and compared the van no-go area's effect on the 2E-VRHPD model. The results show that no-go restrictions for a van may significantly impact 2E-VRHPD system efficiency, and that the maximum travel distance of robots has a limited impact on 2E-VRHPD system efficiency if the robot's maximum travel distances exceed 10 km. We also compared the 2E-VRHPD model with two classical models. Results show the 2E-VRHPD model is competitive in appropriate scenarios. We therefore advocate considering the 2E-VRHPD model for distribution in appropriate scenarios to improve the system's efficiency.

Chapter 6

Conclusion and future research

6.1 Key findings and contributions

This dissertation aims at developing optimization models and methods for tour planning in smart urban logistics. Considering the recent booming research and development of the robot distribution system, we focus on van-based robot routing for urban distribution.

We start with three research questions proposed in Chapter 1 to summarize this thesis.

Research question 1: How to overcome the shortages of robot delivery?

We design a van-based robot delivery model (2E-VRRP) to overcome three significant drawbacks of robot delivery: limited travel distance, limited driving speed, and limited freight capacity. The van-based robot delivery system works as follows.

In a 2E-VRRP model, the van carries the robots on the *1st-level route* and drops off and picks them up in the parking nodes, while the robot handles customer services on the *2nd-level route*.

This 2E-VRRP model overcomes the drawbacks of robot delivery by van transports, replenishes, and recharges its robot, realizing the collaborative delivery of vans and robots, and therefore helps improve system efficiency.

Note our 2E-VRRP model contains the time window constraint, capacity constraint, and maximum travel time constraint. In addition, our model allows each van to carry multiple robots, and each robot can visit multiple customers during a trip.

Research question 2: How to make the robot charging more efficiently?

Charging robots takes time. While the time window constraint is vital in the urban distribution system, time is thus becoming a valuable resource. How to make more efficient use of time to recharge is the key of charging problem.

Based on the van-based robot delivery model, we considers van and robot are all electric-driven, and then use reverse charging and en-route charging technologies to achieve more efficient charging.

We further studies a 2E-VREC model: the vans carry small robots along the *1st-level route*, and the robot itself travels along the *2nd-level open route*. The van stops at parking nodes for dropping off and/or picking up its robot, and for recharging and replenishing its robot if needed. The van can charge its robot (if it is onboard the van) during its trip. The van can be recharged at the parking nodes.

The 2E-VREC model makes full use of the time during which vans are carrying robots on the road to charge them, shortening the extra waiting time spent by robots at parking stations for recharging, thereby improving the effectiveness of the distribution system—carried out the transmission of electric power.

Research question 3: How to integrate multiple modes of pickup and delivery operation to make the logistics distribution system more efficient?

Based on Chapter 3 and Chapter 4, using only vans to deliver goods or using only robots to deliver goods has their shortcomings. For example, customers may be located on narrow streets where the entry of vans is restricted. Besides, the increasing diversification of distribution demands in modern cities has brought challenges to logistics companies. Research shows that integrating multiple distribution modes into a distribution system can improve the efficiency of enterprises.

We thus design a van-based robot hybrid pickup and delivery model to integrate various scenarios of pickup and delivery into the model. Various pickup and delivery operations can be completed with the van and robot's coordination, instead of passing through a depot for transshipment, the system's efficiency could be improved. Also, as pair-customers' goods can be transferred between a van and its robot, the flexibility of goods delivery is greatly enhanced.

Next, we introduce the contribution and key finding of this thesis.

Contribution of the thesis

Problem level: We consider a new two-echelon van-based robot urban delivery model with time windows, in which the *2nd-level route is an open route* (Chapter 3 2E-VRRP problem). We then extend the model to the electric level and allow en-route operations by introducing reverse charging and en-route charging technologies into this model (Chapter 4 2E-VREC problem). Furthermore, we extend the van-based robot model to a hybrid pickup and delivery level to adapt to the latest business modes (Chapter 5 2E-VRHPD problem).

Model and theoretical level: We first introduce the fundamental problem and propose the mathematical formulations for 2E-VRRP, 2E-VREC, and 2E-VRHPD problems. For the 2E-VRRP model, a column generation procedure is presented for trying to get a better lower bound for the MIP model. For the 2E-VREC model, we further detailedly introduce time-distance-energy trade-off in the problem. For the 2E-VRHPD model,

we further detailedly introduce five pickup and delivery modes and the van-robot load trade-off in the problem.

Algorithm level: We presented heuristic to solve the 2E-VRRP, 2E-VREC, and 2E-VRHPD problems. For the 2E-VRRP problems, We propose a construction heuristic for the newly introduced problem, which is useful for quickly generating feasible initial solutions and providing a first upper bound. We propose a multi-start hybrid meta-heuristic approach based on iterative local search and backtracking. The backtracking procedure is led to connect the chosen robot routes to the van route accurately. For the 2E-VREC problem, we provide a high-accuracy heuristic for single route feasibility checking of the 2E-VREC problem with time-distance-energy trade-off. We propose an ALNS algorithm for the newly presented problem. For the 2E-VRHPD problem, we propose an ALNS algorithm for the problem and a capacity feasibility tests approach for a single route.

Experiment level: For the 2E-VRRP problem, we analyze how van/robot speed combinations influence the objective value, which can reference real-world implementation of such a service. For the 2E-VRRC problem, we conduct a sensitivity analysis for vehicle charging modes, battery capacities, and charging rate. And for the 2E-VRHPD model, we conduct a sensitivity analysis on the robot's travel cost rate and maximum travel distances and compare the van no-go area's effect on the 2E-VRHPD model. Moreover, we also compared the 2E-VRHPD model with two classical models.

Key finding: (1) After a sensitivity analysis for the van/robot speed combinations, we find increasing the robot's speed rather than the van's speed can reduce the cost in real-world applications. However, simply improving the speed of the robot leads to an only limited reduction in cost. Therefore, we recommend keeping robot speeds rather low because of a more pedestrian-friendly environment in practical implementations. (2) We find that using en-route-charge technologies, with appropriate increases in battery capacity and charging rate, has beneficial effects on cost through a sensitivity analysis for vehicle charging modes, battery capacities, and charging rate. We thus recommend that logistics companies use en-route-charge technology if the new technology's fixed cost is controllable. (3) After comparing the 2E-VRHPD model with two classical models, and results show the 2E-VRHPD model is very competitive in real scenes. We thus recommend that companies consider the 2E-VRHPD model for distribution in appropriate scenarios to improve the system's efficiency.

6.2 Challenges and further research directions

In this thesis, we mainly studied the van-based robot model for urban distribution with time window constraints. We focused on evaluating the potential benefits of van-based

robot systems from the operational point of view. We thus believe that this thesis has opened the door for many exciting research topics.

Problem level. (1) Urban distribution system includes drones, robots, and vans that could be explored. (2) Shared robot delivery systems could be considered. (3) The logistics facility layout for the smart urban distribution system can be studied.

Model level. (1) More concise and efficient mathematical models for the 2E-VREC and 2E-VRHPD problems are worth exploring. (2) Extending the 2E-VREC and 2E-VRHPD models by allowing a van carries multiple robots is an exciting research direction. (3) The other objective of the 2E-VREC model can be studied. For example, if the 2E-VREC model's objective is to minimize travel duration, and there is no time window on customers, the en-route-charge mode may significantly reduce travel duration. Besides, nonlinear objective functions can also be explored.

Theoretical level. (1) A more in-depth study on a new time-distance-energy trade-off problem in Chapter 4, can be conducted. (2) The van-robot load trade-off problem in Chapter 5 can continue to be studied exclusively.

Algorithm level. (1) Future research could design a fast and accurate algorithm for checking the feasibility of a given route in 2E-VRRP, 2E-VREC, and 2E-VRHPD models. (2) The mathematics-heuristic method is also a significant research direction. (3) The branch and pricing method is promising. How to quickly solve the sub-problems of a single path in the 2E-VRHPD model is a critical research field. (4) For particularly large-scale real-time requirement problem, algorithms based on machine learning can be studied.

Appendix A

Appendix for Chapter 3

A.1 Column generation procedure

We design a basic column generation procedure, based on the book of column generation (Desaulniers et al., 2006), to solve the original MIP model. The objective function in Chapter 3 first minimizes the number of vans used, then keep the number of used vans unchanged, to minimize the total travel cost. Since the CPLEX can well solve the primary objective, we only use the column generation to calculate the secondary objective lower bound. In the column generation procedure, we first keep the fixed number of vans (primary objective got from CPLEX) unchanged and then adopt the column generation to get a lower bound of the secondary objective.

Appendix A.1.1 and A.1.2 describe the master problem and pricing sub-problem for the column generation procedure of the original MIP model. A column generation lower bound computational study for small-sized instances is presented in Appendix A.1.3.

A.1.1 Master problem

The master problem is simply stated as a set partitioning problem.

Let R be the set of all feasible tour-trees. We define a tour-tree is feasible if the capacity, time window, travel distance, and synchronization constraints are respected. For each tour-tree $r \in R$, we define c_r as the total cost for tour-tree r. Furthermore, let α_{rz} be a binary coefficient equal to 1 if the customer z is visited by the tour-tree r (0 otherwise). Let y_r denote a binary variable that takes the value 1 if and only if the tour-tree $r \in R$ is included in the solution (0 otherwise). Let K_{fix} be the fixed number of van we used.

Based on these definitions, the van-robot problem formulation can be written as the following set partitioning problem:

Objective

$$\min(\sum_{r\in R} c_r y_r) \tag{A.1}$$

Subject to

$$\sum_{r\in R} \alpha_{rz} y_r = 1, \forall z \in Z \tag{A.2}$$

$$y_r \in \{0,1\}, \forall r \in R \tag{A.3}$$

$$\sum_{r\in R} y_r = K_{fix} \tag{A.4}$$

The objective function (A.1) minimizes the total cost of the used tour-trees. Constraints (A.2) ensure each customer is served by one tour-tree. Constraints (A.3) are the domain constraint for the decision variables. Note that for implementation issues, constraints (A.3) can be replaced by $0 \leq y_r \leq 1$. Constraints (A.4) ensure the number of the vehicle used is fixed.

A.1.2 Pricing sub-problem

We use MIP model to solve the pricing sub-problem.

Let γ_z, $z \in Z$ be the dual variable associated with constraints (A.2). And β be the dual variable of constraint (A.4). The reduced cost $\bar{c}_r = c_r - \sum_{r\in R} \alpha_{rz}\gamma_z - \beta$. Explanation of other symbol definitions and inequalities are the same as the explanation in our van-robot model expect without symbol k. Which means we consider one van, not multi-vans. For the sake of simplicity, we will not explain more.

The pricing sub-problem, aiming to generate columns, with the most negative reduced cost, is formulated as follows:

Objective

$$\min(\sum_{(i,j)\in A_1} c_{i,j} * x_{i,j} + \sum_{(i,j)\in A_2} (c_{i,j} - \gamma_i) y^l_{i,j} - \beta) \tag{A.5}$$

Subject to

$$\sum_{(i,j)\in A_1} x_{i,j} \leq 1, \qquad \forall j \in V_r^{'} \tag{A.6}$$

$$\sum_{i\in V_p} x_{i,0'} = \sum_{j\in V_d} x_{0,j} = 1 \tag{A.7}$$

$$\sum_{(i,j)\in A_1} x_{i,j} - \sum_{(j,i)\in A_1} x_{j,i} = 0, \qquad \forall j \in V_r \tag{A.8}$$

$$\sum_{l\in F_D} \sum_{i\in V_{dc}} y_{i,j}^{l} <= 1, \qquad \forall j \in V_c \tag{A.9}$$

$$\sum_{i\in V_{dc}} y_{i,j}^{l} - \sum_{i\in V_{pc}} y_{j,i}^{l} = 0, \qquad \forall j \in V_c, l \in F_D \tag{A.10}$$

$$\sum_{i\in V_p} Q_{i,0'} = \sum_{j\in V_d} Q_{0,j}, \tag{A.11}$$

$$\sum_{j\in V_d} Q_{0,j} = \sum_{l\in F_D} \sum_{i\in V_c} \sum_{j\in V_d} y_{j,i}^{l}, \tag{A.12}$$

$$\sum_{i\in V_r^0} Q_{i,j} - \sum_{i\in V_r} Q_{j,i} = \sum_{l\in F_D} \sum_{i\in V_c} y_{j,i}^{l}, \qquad \forall j \in V_d \tag{A.13}$$

$$\sum_{i\in V_r} Q_{i,j} - \sum_{i\in V_r^{'}} Q_{j,i} = -\sum_{l\in F_D} \sum_{i\in V_c} y_{i,j}^{l}, \qquad \forall j \in V_p \tag{A.14}$$

$$0 \leq Q_{i,j} \leq x_{i,j} * L, \qquad \forall (i,j) \in A_1 \tag{A.15}$$

$$W_i + t_{i,j} - W_j \leq M(1 - x_{i,j}), \qquad \forall (i,j) \in A_1 \tag{A.16}$$

$$W_i + t_{i,j} - W_j \leq M(1 - y_{i,j}^{l}), \qquad \forall i \in V_d, j \in V_c, l \in F_D \tag{A.17}$$

$$W_i + t_{i,j} + s_i - W_j \leq M(1 - y_{i,j}^{l}), \qquad \forall i \in V_c, j \in V_{pc}, l \in F_D \tag{A.18}$$

$$a_i \leq W_i, \qquad \forall i \in V_c^0 \tag{A.19}$$

$$W_i \leq b_i, \qquad \forall i \in V_c^{'} \tag{A.20}$$

$$\sum_{i\in V_c} d_i \sum_{j\in V_{pc}} y_{i,j}^{l} \leq C, \qquad \forall l \in F_D \tag{A.21}$$

$$\sum_{(i,j)\in A_2} y_{i,j}^{l} t_{i,j} \leq T, \qquad \forall l \in F_D \tag{A.22}$$

$$x_{i,j} \in \{0,1\}, \qquad \forall (i,j) \in A_1 \tag{A.23}$$

$$y_{i,j}^{l} \in \{0,1\}, \qquad \forall (i,j) \in A_2, l \in F_D \tag{A.24}$$

A.1.3 Computational results of column generation lower bound

According to (Desrosiers and Lübbecke, 2005), the lower bound can be calculate as $LB = z + K * c$, where z denotes the optimal objective function value to the restricted master problem. The c is the reduce cost got from the subproblem, and K is an upper bound of the number of van we used. The runtime of the column generation procedure

also limited to 5 hours, and the best of the valid lower bounds, each iteration generated, are chosen as the column generation lower bound. Note that the column generation procedure is code in DOcplex by calling CPLEX 12.8.

The column generation computational results for 15 and 30 customer instances are in Table A.1. In which column 1 indicates the test instances, column 2 and column 3 show the column generation lower bound and the solving time for the instance.

TABLE A.1: Column generation lower bound for 15 and 30 customer instances

	3-15		4-15		5-15		3-30		4-30		5-30	
	CLB	CT(s)	CLB	CT(s)	CLB	CT(s)	CLB	CT(s)	CLB	CT(s)	CLB	CT(s)
CA1	375.6	49.6	411.1	284.6	403.7	522.5	752.7	3375.3	693.7	8332.5	602.2	8484.6
CA2	378.5	80.3	403.2	115.3	418.4	698.5	653.1	13591.0	676.8	6965.7	624.8	18000.0
CA3	387.5	108.5	443.7	113.0	354.4	251.6	680.3	7756.6	639.5	18000.0	602.2	8488.1
CA4	400.9	71.4	397.8	2578.1	343.5	440.8	578.4	2837.0	676.3	18000.0	-1367.4	18000.0
CA5	382.8	52.8	308.7	102.1	386.7	963.2	650.1	9257.5	585.1	9598.6	249.6	18000.0
CB1	391.8	91.2	414.5	505.1	530.0	8060.2	758.2	16025.0	673.1	18000.0	580.1	18000.0
CB2	410.1	45.5	446.9	361.9	406.9	1224.6	626.5	4762.1	642.9	18000.0	588.1	18000.0
CB3	446.9	76.6	438.5	527.3	364.6	311.0	679.9	4886.4	623.1	18000.0	301.2	18000.0
CB4	377.6	59.8	393.6	515.0	334.2	18000.0	609.2	6910.0	654.1	18000.0	-1923.9	18000.0
CB5	408.1	89.6	361.6	127.6	393.7	941.4	621.5	18000.0	601.9	18000.0	315.8	18000.0
CC1	358.8	210.9	415.5	18000.0	332.7	18000.0	389.0	18000.0	503.3	18000.0	316.8	18000.0
CC2	344.8	1198.2	391.5	10235.8	350.7	18000.0	290.0	18000.0	377.4	18000.0	-3355.5	18000.0
CC3	380.9	976.4	419.8	18000.0	337.9	18000.0	450.8	18000.0	-2773.7	18000.0	292.3	18000.0
CC4	373.3	547.3	364.4	18000.0	307.9	18000.0	442.0	18000.0	453.8	18000.0	-3453.5	18000.0
CC5	286.5	5557.1	301.4	7705.9	374.4	18000.0	432.7	18000.0	272.9	18000.0	-983.8	18000.0
CD1	370.8	71.3	405.3	395.8	385.4	792.1	619.7	13819.9	671.2	4944.2	624.5	18000.0
CD2	348.8	95.5	396.9	258.3	384.2	828.9	624.8	12195.6	643.9	9956.6	612.7	18000.0
CD3	382.1	116.8	445.0	403.4	348.3	320.5	645.7	4853.2	606.5	18000.0	588.3	9042.9
CD4	381.6	114.0	355.4	780.9	337.3	585.1	578.4	1612.6	661.7	18000.0	630.1	18000.0
CD5	362.5	60.0	301.4	7790.1	393.2	520.3	641.9	5025.1	630.5	7475.1	441.5	18000.0

Appendix B

Appendix for Chapter 4

B.1 Time warp calculating algorithm

Algorithm 7 is used to calculate the time warp. The input of the algorithm (Line 1) contains the *Path* from station n to station m. *DepartTime* is the time a vehicle leaves the parking station. *Distance* is the distance matrix. *StartTimeWindow*, *EndTimeWindow* and *ServiceTime* are the time window list and service time list for nodes. The algorithm starts with time *DepartTime* (Line 2). For the sequence from node n to node m, we calculate the earliest arrival time at the node m (Line 3-11). Next, we reverse the *Path*, and we then calculate the latest starting time from node n subject to the earliest arrival time at the node m not changing (Line 13-20). The time warp is equal to the latest starting time at node n minus the earliest start time at node n (Line 21).

Algorithm 7 *Time_warp_calculating*

Input: *Path*, *DepartTime*, *Distance*, *StartTimeWindow*, *EndTimeWindow*, *ServiceTime*

$T = DepartTime$
for i in range(len(*Path*)-1) **do**
    T = T + *Distance*[*Path*[i]][*Path*[i+1]]/v
    T = max(*StartTimeWindow*[*Path*[i+1]],T)
    if T $>$ *EndTimeWindow*[*Path*[i+1]] **then**
        return: (False)
    else
        T = T + *ServiceTime*[*Path*[i+1]]
    end if
end for
reverse(*Path*)
for i in range(len(*Path*)-1) **do**
    T = T - *Distance*[*Path*[i]][*Path*[i+1]]/v - *ServiceTime*[*Path*[i+1]]
    T = max(*StartTimeWindow*[*Path*[i+1]],T)
    T = min(T, *EndTimeWindow*[*Path*[i+1]])
    if *StartTimeWindow*[*Path*[i+1]] $>$ T **then**
        return: (False)
    end if
end for
Time Warp = T - *DepartTime*
return: Time Warp

B.2 Formulation of LP model

The LP model is used to check the feasibility of a given 2E-VREC route.

Explanations of symbol definitions are the same as in Subsection 4.2.3 for our 2E-VREC model, expect without symbol k. This means we consider a van, not multi-vans. For the sake of simplicity, no further explanation is given. We add the following new symbols: $R1$: van route; $R2$: robot route; $R3$: van-robot route and independent-van route and robot route; $R4$: independent-van route; $R5$: van-robot route. The LP model is as follows.

Objective

Checking the feasibility of a given route

Subject to

$$W_i + Z_i/g_1 + d_{i,j}/v_1 - W_j \leq 0, \forall\{i \in V_r^0 | (i,j) \in R_1\} \tag{B.1}$$

$$W_i + z_i/g_2 + d_{i,j}/v_1 - W_j \leq 0, \forall\{i \in V_r^0 | (i,j) \in R_1\} \tag{B.2}$$

$$w_i + z_i/g_2 + d_{i,j}/v_1 - W_j \leq 0, \forall\{i \in V_r^0 | (i,j) \in R_1\} \tag{B.3}$$

$$W_i + z_i/g_2 + d_{i,j}/v_2 - w_j \leq 0, \forall\{i \in V_r | (i,j) \in R_2\} \tag{B.4}$$

$$w_i + z_i/g_2 + d_{i,j}/v_2 - w_j \leq 0, \forall\{i \in V_r | (i,j) \in R_2\} \tag{B.5}$$

$$W_i + s_i + d_{i,j}/v_1 - W_j \leq 0, \forall\{i \in V_{c1} | (i,j) \in R_1\} \tag{B.6}$$

$$W_i + z_i/g_2 + d_{i,j}/v_1 - W_j \leq 0, \forall\{i \in V_{c1} | (i,j) \in R_1\} \tag{B.7}$$

$$w_i + s_i + d_{i,j}/v_2 - w_j \leq 0, \forall\{i \in V_c | (i,j) \in R_2\} \tag{B.8}$$

$$W_j = w_j, \forall\{(i,j) \in R_5 | j \in V_{rc1}^{'}\} \tag{B.9}$$

$$a_i \leq W_i, w_i \leq b_i, \forall i \in V_c^{''} \tag{B.10}$$

$$E_j + h_1 * d_{i,j} - Z_i + z_i + z_{i,j} - E_i \leq 0, \forall\{(i,j) \in R_5 | i \in V_r^0\} \tag{B.11}$$

$$E_j + h_1 * d_{i,j} - Z_i + z_i - E_i \leq 0, \forall\{(i,j) \in R_4 | i \in V_r^0\} \tag{B.12}$$

$$E_j + h_1 * d_{i,j} + z_i + z_{i,j} - E_i \leq 0, \forall\{(i,j) \in R_5 | i \in V_{c1}\} \tag{B.13}$$

$$E_j + h_1 * d_{i,j} - E_i \leq 0, \forall\{(i,j) \in R_4 | i \in V_{c1}\} \tag{B.14}$$

$$e_j - z_i - z_{i,j} - e_i \leq 0, \forall\{(i,j) \in R_5 | i \in V_{rc1}^0\} \tag{B.15}$$

$$e_j + h_2 * d_{i,j} - z_i - e_i \leq 0, \forall\{(i,j) \in R_2 | i \in V_r\} \tag{B.16}$$

$$e_j + h_2 * d_{i,j} - e_i \leq 0, \forall\{(i,j) \in R_2 | i \in V_c\} \tag{B.17}$$

$$0 \leq z_i \leq G_2, \forall i \in V_{rc1}^0 \tag{B.18}$$

$$0 \leq Z_i \leq (G_1 + G_2), \forall i \in V_r^0 \tag{B.19}$$

$$0 \leq z_{i,j} \leq G_2, \forall (i,j) \in R_1 \tag{B.20}$$

$$0 \leq E_i + Z_i - z_i \leq G_1, \forall i \in V_r^0 \tag{B.21}$$

$$0 \leq e_i + z_i \leq G_2, \forall i \in V_r^0 \tag{B.22}$$

$$0 \leq E_i \leq G_1, \forall i \in V_{rc1}^{''} \tag{B.23}$$

$$0 \leq e_i \leq G_2, \forall i \in V_{rc}^{''} \tag{B.24}$$

Constraints (B.1-B.10) are the time flow constraints for the given routes. Constraints (B.1-B.3) and (B.6-B.7) enforce the time flow of the van for R_1. Constraints (B.4-B.5) and (B.8) enforce the time flow of the robot for R_2. Constraints (B.9) ensure that when a van carries its robot, their arrival time for each node is equal. Constraints (B.10) are the time window constraints for each node.

Constraints (B.11-B.24) are the energy constraints for the given routes. Constraints (B.11) and (B.13) are the energy flow of the van for the route along which the van carries its robot, and constraints (B.15) are the energy flow of the robot for the routes when the robot is carried in its van. Constraints (B.12) and (B.14) are the energy flow of the van for the independent-van routes. Constraints (B.16) and (B.17) are the energy flow of the robot for the robot routes. Constraints (B.18-B.24) are the energy variable constraints.

B.3 Simulated annealing and weight-adjusted approach

Simulated annealing

The simulated annealing uses a temperature parameter T to control the acceptance probability. The temporary solution x^t is always accepted if x^t has a better objective value than the current best solution x^b ($c(x^t) <= c(x^b)$), and it is accepted with probability $\exp(c(x^b) - c(x^t)/T)$ if this is not the case. Here $T > 0$ is the current temperature. The start temperature is initialized at $T_{st} > 0$ and is decreased gradually by performing an update $T = \alpha * T$ at each iteration, where $\alpha \in [0, 1]$ is the cooling rate of simulated annealing.

Weight-adjusted

For the destroy and repair operations, a score $\psi = max(w_1, w_2, w_3, w_4)$ is used to adjust their respective weights and is computed according the following formula:

$$\psi = \begin{cases} w_1 & \text{if the new solution is a new global best solution,} \\ w_2 & \text{if the new solution is better than the current solution,} \\ w_3 & \text{if the new solution is accepted,} \\ w_4 & \text{if the new solution is rejected.} \end{cases}$$

Normally we have $w_1 \geq w_2 \geq w_3 \geq w_4 \geq 0$.

The components corresponding to the selected destroy-and-repair operations in the ρ^- and ρ^+ vectors are updated using the following equations: $\rho_a^- = \lambda\rho_a^- + (1-\lambda)\psi$, $\rho_b^+ = \lambda\rho_b^+ + (1-\lambda)\psi$, in which a and b are the indices of the destroy-and-repair methods that were used in the last iteration of the algorithm, respectively. The components corresponding to the selected destroy, and repair methods in the ρ^- and ρ^+ vectors are updated using the equations. $\lambda \in [0, 1]$ is the decay parameter that controls how sensitive the weightings are to changes in the performance of the destroy-and-repair methods.

B.4 Parameter notations, descriptions, and values

Table B.1 lists and describes the parameters investigated in our experiment. Column 1 gives the parameter name, and column 2 its corresponding notation and description. Table B.2 lists the parameter notations, the initial value of the parameters, the allowed range for each parameter, and the final values found by the parameter tuning process. Final values are used for the rest of the experiments.

TABLE B.1: Parameters notation and description

Parameter	Notation and description
MaxIteNum	Maximum iteration number of ALNS
MaxNunImp	Maximum iteration number with no improved solution
β	Destruction rate
λ	Decay parameter that controls how sensitive the weights are to changes
T_{st}	Start temperature per customer of SA
α	Cooling rate of SA
w_1	if the new solution is a new global best solution
w_2	if the new solution is better than the current one
w_3	if the new solution is accepted
w_4	if the new solution is rejected

TABLE B.2: Parameters values of ALNS for 2E-VREC

Parameter	Initial value	Search interval	Final value
MaxIteNum	10000	10000-50000 (5000)	20000
MaxNunImp	500	100-2000 (100)	1800
β	0.15	0.05-0.5 (0.05)	0.35
λ	0.9	0.1-1.0 (0.01)	0.95
T_{st}	2000	1000-10000 (1000)	1000
α	0.95	0.90-0.99 (0.01)	0.92
w_1	3	3-30	28
w_2	2	2-27	13
w_3	1	1-12	3
w_4	0	0	0

Appendix C

Appendix for Chapter 5

C.1 ALNS parameters

TABLE C.1: Parameters values of ALNS for 2E-VRHPD

Parameter	Notation and description	Final value
MaxIteNum	Maximum iteration number of ALNS	20000
MaxNunImp	Maximum iteration number with no improved solution	1800
β	Destruction rate	0.35
λ	Decay parameter	0.95
T_{st}	Start temperature per customer of SA	1000
α	Cooling rate of SA	0.92
w_1	new solution is a new global best solution	3
w_2	new solution is better than the current one	2
w_3	new solution is accepted	1
w_4	new solution is rejected	0

Bibliography

Agatz, Niels, Paul Bouman, and Marie Schmidt (2018). "Optimization approaches for the traveling salesman problem with drone". In: *Transportation Science* 52.4, pp. 965–981.

Anderluh, Alexandra et al. (2019). "Multi-objective optimization of a two-echelon vehicle routing problem with vehicle synchronization and 'grey zone' customers arising in urban logistics". In: *European Journal of Operational Research*.

Andrew, J. Hawkins (2019). *Thousands of autonomous delivery robots are about to descend on US college campuses*. https://www.theverge.com/2019/8/20/20812184/starship-delivery-robot-expansion-college-campus. Accessed August 20, 2019.

Avci, Mustafa and Seyda Topaloglu (2016). "A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery". In: *Expert Systems with Applications* 53.Jul. Pp. 160–171.

Bakach, Iurii, Ann Melissa Campbell, and Jan Fabian Ehmke (2020). "A Two-Tier Urban Delivery Network with Robot-based Deliveries". In: *Working Paper Series*.

Bala, Karlo, Dejan Brcanov, and Nebojša Gvozdenović (2017). "Two-echelon location routing synchronized with production schedules and time windows". In: *Central European Journal of Operations Research* 25.3, pp. 525–543.

Baldacci, Roberto et al. (2013). "An exact algorithm for the two-echelon capacitated vehicle routing problem". In: *Operations Research* 61.2, pp. 298–314.

Battarra, Maria, Jean-François Cordeau, and Manuel Iori (2014). "Pickup and Delivery Problems for Goods Transportation". In: *Vehicle Routing: Problems, Methods, and Applications*, pp. 161–191.

Bektaş, Tolga et al. (2019). "The role of operational research in green freight transportation". In: *European Journal of Operational Research* 274.3, pp. 807–823.

Bergmann, Felix M, Stephan M Wagner, and Matthias Winkenbach (2020). "Integrating first-mile pickup and last-mile delivery on shared vehicle routes for efficient urban e-commerce distribution". In: *Transportation Research Part B: Methodological* 131, pp. 26–62.

Bouman, Paul, Niels Agatz, and Marie Schmidt (2018). "Dynamic programming approaches for the traveling salesman problem with drone". In: *Networks* 72.4, pp. 528–542.

Boysen, Nils, Stefan Schwerdfeger, and Felix Weidinger (2018). "Scheduling last-mile deliveries with truck-based autonomous robots". In: *European Journal of Operational Research* 271.3, pp. 1085–1099.

Brandão, José (2018). "Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows". In: *Computers & Industrial Engineering* 120, pp. 146–159.

Breunig, Ulrich et al. (2016). "A large neighbourhood based heuristic for two-echelon routing problems". In: *Computers & Operations Research* 76, pp. 208–225.

Breunig, Ulrich et al. (2019). "The electric two-echelon vehicle routing problem". In: *Computers & Operations Research* 103, pp. 198–210.

Carlsson, John Gunnar and Siyuan Song (2017). "Coordinated logistics with a truck and a drone". In: *Management Science* 64.9, pp. 4052–4069.

Chao, I-Ming (2002). "A tabu search method for the truck and trailer routing problem". In: *Computers & Operations Research* 29.1, pp. 33–51.

Chevallier, Hélène (2017). *TwinswHeel, le livreur du futur*. https://www.franceinter.fr/emissions/c-est-deja-demain/c-est-deja-demain-29-novembre-2017. Accessed November 29, 2017.

Christian, Andrew W and Randolph Cabell (2017). "Initial investigation into the psychoacoustic properties of small unmanned aerial system noise". In: *23rd AIAA/CEAS aeroacoustics conference*, p. 4051.

Coldewey, Devin (2019). *Kiwi's food delivery bots are rolling out to 12 more colleges*. https://techcrunch.com/2019/04/25/kiwisfood-delivery-bots-are-rolling-out-to-12-new-colleges/.. Accessed September 25, 2020.

Crainic, Teodor Gabriel, Antonio Sforza, and Claudio Sterle (2011). *Location-routing models for two-echelon freight distribution system design*. CIRRELT.

Croes, Georges A (1958). "A method for solving traveling-salesman problems". In: *Operations Research* 6.6, pp. 791–812.

Cuda, Rosario, Gianfranco Guastaroba, and Maria Grazia Speranza (2015). "A survey on two-echelon routing problems". In: *Computers & Operations Research* 55, pp. 185–199.

Daimler (2017). *Vans and Robots Paketbote 2.0*. https://www.daimler.com/innovation/specials/future-transportation-vans/paketbote-2-0.html. Accessed Dec 25, 2018.

Dayarian, Iman, Martin W. P. Savelsbergh, and John-Paul Clarke (2020). "Same-Day Delivery with Drone Resupply". In: *Transportation Science* 54, pp. 229–249.

Dellaert, Nico et al. (2018). "Branch-and-Price–Based Algorithms for the Two-Echelon Vehicle Routing Problem with Time Windows". In: *Transportation Science* 53.2, pp. 463–479.

Desaulniers, Guy, Jacques Desrosiers, and Marius M Solomon (2006). *Column generation*. Vol. 5. Springer Science & Business Media.

Desrosiers, Jacques and Marco E Lübbecke (2005). "A primer in column generation". In: *Column generation*. Springer, pp. 1–32.

Drexl, Michael (2020). "On the one-to-one pickup-and-delivery problem with time windows and trailers". In: *Central European Journal of Operations Research*, pp. 1–48.

Duarte, Abraham et al. (2018). "Variable neighborhood descent". In: *Handbook of Heuristics*, pp. 341–367.

Dumas, Yvan, Jacques Desrosiers, and François Soumis (1991). "The pickup and delivery problem with time windows". In: *European Journal of Operational Research* 54.1, pp. 7–22.

European Commission (2014). "Living well, within the limits of our planet. 7th EAP — The New General Union Environment Action Programme to 2020". In:

— (2015). *Urban mobility*. http://ec.europa.eu/transport/themes/urban/urban_mobility/index_en.htm. Access September 10, 2020.

Fred, Lambert (2018). *NIO is courting Tesla owners with mobile charging stations inside electric vans*. https://electrek.co/2018/07/26/nio-courting-tesla-owners-mobile-charging-stations-electric-vans/. Accessed Dec 25, 2018.

Gonzalez-R, Pedro L. et al. (2020). "Truck-drone team logistics: A heuristic approach to multi-drop route planning". In: *Transportation Research Part C-emerging Technologies* 114, pp. 657–680.

Grangier, Philippe et al. (2016). "An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization". In: *European Journal of Operational Research* 254.1, pp. 80–91.

Gschwind, Timo (2019). "Route feasibility testing and forward time slack for the synchronized pickup and delivery problem". In: *OR Spectrum* 41.2, pp. 491–512.

Gschwind, Timo and Michael Drexl (2019). "Adaptive large neighborhood search with a constant-time feasibility test for the dial-a-ride problem". In: *Transportation Science* 53.2, pp. 480–491.

Gu, Liping (2018). *Delivery robots hit the road in Beijing*. http://www.ecns.cn/news/sci-tech/2018-06-20/detail-ifyvmiee7350792.shtml. Accessed June 20, 2018.

Han, Anthony Fu-Wha and Yu-Ching Chu (2016). "A multi-start heuristic approach for the split-delivery vehicle routing problem with minimum delivery amounts". In: *Transportation Research Part E: Logistics and Transportation Review* 88, pp. 11–31.

Hemmelmayr, Vera C, Jean-François Cordeau, and Teodor Gabriel Crainic (2012). "An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics". In: *Computers & operations research* 39.12, pp. 3215–3228.

Hiermann, Gerhard et al. (2016). "The electric fleet size and mix vehicle routing problem with time windows and recharging stations". In: *European Journal of Operational Research* 252.3, pp. 995–1018.

Hiermann, Gerhard et al. (2019). "Routing a mix of conventional, plug-in hybrid, and electric vehicles". In: *European Journal of Operational Research* 272.1, pp. 235–248.

Hunsaker, Brady and Martin Savelsbergh (2002). "Efficient feasibility testing for dial-a-ride problems". In: *Operations Research Letters* 30.3, pp. 169–173.

Jie, Wanchen et al. (2019). "The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology". In: *European Journal of Operational Research* 272.3, pp. 879–904.

Karak, Aline and Khaled Abdelghany (2019). "The hybrid vehicle-drone routing problem for pick-up and delivery services". In: *Transportation Research Part C: Emerging Technologies* 102, pp. 427–449.

Keskin, Merve and Bülent Çatay (2016). "Partial recharge strategies for the electric vehicle routing problem with time windows". In: *Transportation Research Part C: Emerging Technologies* 65, pp. 111–127.

Kitjacharoenchai, Patchara, Byung-Cheol Min, and Seokcheon Lee (2020). "Two echelon vehicle routing problem with drones in last mile delivery". In: *International Journal of Production Economics* 225, p. 107598.

Ko, ar, Gilbert Laporte, and lknur Tükenmez (2020). "A Review of Vehicle Routing with Simultaneous Pickup and Delivery". In: *Computers & operations research*, p. 104987.

Li, Dongmei (2017). *JD.Com Launches Robot Delivery Services In Chinese Universities*. https://www.chinamoneynetwork.com/2017/06/19/jd-com-launches-robot-delivery-services-in-chinese-universities. Accessed June 19, 2017.

Li, Feiyue, Bruce Golden, and Edward Wasil (2007). "The open vehicle routing problem: Algorithms, large-scale test problems, and computational results". In: *Computers & Operations Research* 34.10, pp. 2918–2930.

Li, Hongqi et al. (2016). "The two-echelon time-constrained vehicle routing problem in linehaul-delivery systems". In: *Transportation Research Part B: Methodological* 94, pp. 169–188.

Lithia, Fla (2017). *UPS Tests Residential Delivery Via Drone Launched From atop Package Car*. https://pressroom.ups.com/pressroom/ContentDetailsViewer.page?ConceptType=PressReleases&id=1487687844847-162. Accessed February 21, 2017.

Liu, Tian et al. (2018). "A branch-and-cut algorithm for the two-echelon capacitated vehicle routing problem with grouping constraints". In: *European Journal of Operational Research* 266.2, pp. 487–497.

Luo, Zhihao, Zhong Liu, and Jianmai Shi (2017). "A Two-Echelon Cooperated Routing Problem for a Ground Vehicle and Its Carried Unmanned Aerial Vehicle". In: *Sensors* 17.5, p. 1144.

Martí, Rafael et al. (2019). "Intelligent Multi-Start Methods". In: *Handbook of Metaheuristics*. Springer, pp. 221–243.

Masson, Renaud et al. (2017). "Optimization of a city logistics transportation system with mixed passengers and goods". In: *Euro Journal on Transportation & Logistics* 6.1, pp. 81–109.

Mathew, Neil, Stephen L Smith, and Steven L Waslander (2015). "Multirobot rendezvous planning for recharging in persistent tasks". In: *IEEE Transactions on Robotics* 31.1, pp. 128–142.

Moshref-Javadi, Mohammad, Ahmad Hemmati, and Matthias Winkenbach (2020a). "A truck and drones model for last-mile delivery: A mathematical model and heuristic approach". In: *Applied Mathematical Modelling* 80, pp. 290–318.

Moshref-Javadi, Mohammad, Seokcheon Lee, and Matthias Winkenbach (2020b). "Design and evaluation of a multi-trip delivery model with truck and drones". In: *Transportation Research Part E-logistics and Transportation Review* 136, p. 101887.

Mourad, Abood, Jakob Puchinger, and Tom van Woensel (2020). "Integrating autonomous delivery service into a passenger transportation system". In: *International Journal of Production Research*, pp. 1–24.

Mühlbauer, Ferdinand and Pirmin Fontaine (2020). "A parallelised large neighbourhood search heuristic for the asymmetric two-echelon vehicle routing problem with swap containers for cargo-bicycles". In: *European Journal of Operational Research*.

Murray, Chase C and Amanda G Chu (2015). "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery". In: *Transportation Research Part C: Emerging Technologies* 54, pp. 86–109.

Murray, Chase C and Ritwik Raj (2020). "The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones". In: *Transportation Research Part C: Emerging Technologies* 110, pp. 368–398.

Nagy, Gábor et al. (2013). "The vehicle routing problem with divisible deliveries and pickups". In: *Transportation Science* 49.2, pp. 271–294.

Nguyen, Viet-Phuong, Christian Prins, and Caroline Prodhon (2012). "A multi-start iterated local search with tabu list and path relinking for the two-echelon location-routing problem". In: *Engineering Applications of Artificial Intelligence* 25.1, pp. 56–71.

OECD (2017). "Transition to Shared Mobility". In: *International Transport Forum's Corporate Partnership Board*, pp. 2029–2029.

Otto, Alena et al. (2018). "Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey". In: *Networks* 72.4, pp. 411–458.

Paolo, Toth and Daniele Vigo (2014). *Vehicle routing: problems, methods and applications*. Society for Industrial and Applied Mathematics.

Parragh, Sophie N and Jean-François Cordeau (2017). "Branch-and-price and adaptive large neighborhood search for the truck and trailer routing problem with time windows". In: *Computers & Operations Research* 83, pp. 28–44.

Perboli, Guido, Roberto Tadei, and Daniele Vigo (2011). "The Two-Echelon Capacitated Vehicle Routing Problem: Models and Math-Based Heuristics". In: *Transportation Science* 45.3, pp. 364–380.

Pisinger, David and Stefan Ropke (2019). "Large Neighborhood Search". In: *Handbook of Metaheuristics*. Springer, pp. 99–127.

Poikonen, Stefan and Bruce Golden (2020a). "Multi-visit drone routing problem". In: *Computers & Operations Research* 113, p. 104802.

— (2020b). "The Mothership and Drone Routing Problem". In: *INFORMS Journal on Computing* 32, pp. 249–262.

Poikonen, Stefan, Bruce Golden, and Edward A Wasil (2019). "A branch-and-bound approach to the traveling salesman problem with a drone". In: *INFORMS Journal on Computing* 31.2, pp. 335–346.

Poikonen, Stefan, Xingyin Wang, and Bruce Golden (2017). "The vehicle routing problem with drones: Extended models and connections". In: *Networks* 70.1, pp. 34–43.

Polat, Olcay (2017). "A parallel variable neighborhood search for the vehicle routing problem with divisible deliveries and pickups". In: *Computers & Operations Research* 85, pp. 71–86.

Pugliese, Luigi Di Puglia and Francesca Guerriero (2017). "Last-Mile Deliveries by Using Drones and Classical Vehicles". In: *International Conference on Optimization and Decision Science*. Springer, pp. 557–565.

Qu, Yuan and Jonathan F. Bard (2013). "The heterogeneous pickup and delivery problem with configurable vehicle capacity". In: *Transportation Research Part C-emerging Technologies* 32, pp. 1–20.

Repoussis, Panagiotis P, Christos D Tarantilis, and George Ioannou (2007). "The open vehicle routing problem with time windows". In: *Journal of the Operational Research Society* 58.3, pp. 355–367.

Ropke, Stefan and David Pisinger (2006). "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows". In: *Transportation Science* 40.4, p.455–472.

Rothenbächer, Ann-Kathrin, Michael Drexl, and Stefan Irnich (2018). "Branch-and-price-and-cut for the truck-and-trailer routing problem with time windows". In: *Transportation Science* 52.5, pp. 1174–1190.

Sacramento, David, David Pisinger, and Stefan Ropke (2019). "An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones". In: *Transportation Research Part C: Emerging Technologies* 102, pp. 289–315.

Salama, Mohamed and Sharan Srinivas (2020). "Joint optimization of customer location clustering and drone-based routing for last-mile deliveries". In: *Transportation Research Part C-emerging Technologies* 114, pp. 620–642.

SAP (2019). "E-Commerce Foundation: The Global e-Commerce Report 2019". In:

Savelsbergh and W. P. Martin (1992). "The Vehicle Routing Problem with Time Windows: Minimizing Route Duration". In: *Informs Journal on Computing* 4.2, pp. 146–154.

Schermer, Daniel, Mahdi Moeini, and Oliver Wendt (2019a). "A hybrid VNS/Tabu search algorithm for solving the vehicle routing problem with drones and en route operations". In: *Computers & Operations Research* 109, pp. 134–158.

— (2019b). "A matheuristic for the vehicle routing problem with drones and its variants". In: *Transportation Research Part C: Emerging Technologies* 106, pp. 166–204.

Scherr, Yannick Oskar et al. (2019). "Service network design with mixed autonomous fleets". In: *Transportation Research Part E: Logistics and Transportation Review* 124, pp. 40–55.

Sun, Wei, Yang Yu, and Junwei Wang (2019). "Heterogeneous vehicle pickup and delivery problems: Formulation and exact solution". In: *Transportation Research Part E: Logs and Transportation Review* 125, pp. 181–202.

Tang, Christopher S and Lucas P Veelenturf (2019). "The Strategic Role of Logistics in the Industry 4.0 Era". In: *Transportation Research Part E: Logistics and Transportation Review* 129, pp. 1–11.

USPS (2018). *Summary Report: Public Perception of Delivery Robots in the United States*. https://www.oversight.gov/sites/default/files/oig-reports/RARC-WP-18-005.pdf. Accessed September 25, 2020.

Van Woensel, Tom (2018). *Instances used*. https://tomvanwoensel.com/instances-used/. Accessed June 03, 2018.

Wang, Dan, Hong Zhou, and Ruxin Feng (2019). "A two-echelon vehicle routing problem involving electric vehicles with time windows". In: *Journal of Physics: Conference Series*. Vol. 1324. 1. IOP Publishing, p. 012071.

Wang, Xingyin, Stefan Poikonen, and Bruce Golden (2017). "The vehicle routing problem with drones: several worst-case results". In: *Optimization Letters* 11.4, pp. 679–697.

Wang, Yong et al. (2018). "Two-echelon location-routing optimization with time windows based on customer clustering". In: *Expert Systems with Applications* 104, pp. 244–260.

Wang, Zheng and Jiuh-Biing Sheu (2019). "Vehicle routing problem with drones". In: *Transportation research part B: methodological* 122, pp. 350–364.

Yu, Kevin et al. (2019). "Algorithms and experiments on routing of unmanned aerial vehicles with mobile recharging stations". In: *Journal of Field Robotics* 36.3, pp. 602–616.

Yu, Shaohua, Jakob Puchinger, and Shudong Sun (2020). "Two-echelon urban deliveries using autonomous vehicles". In: *Transportation Research Part E: Logistics and Transportation Review* 124, p. 102018. URL: https://hal.archives-ouvertes.fr/hal-02877730.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Interfaces:
matériaux, systèmes, usages

Titre: Modèles et méthodes d'optimisation pour la planification des tournées dans la logistique urbaine intelligente

Mots clés: Distribution innovante du dernier kilomètre, Services de robots basés sur le vaisseau mère, Routage de véhicules électriques, Recharge en route, Ramassage et livraison hybrides

Résumé: La distribution urbaine désigne les activités de distribution de marchandises desservant les zones urbaines et les banlieues. Les progrès technologiques récents dans le domaine de la distribution sans personnel, ainsi que les nouvelles réglementations limitant l'utilisation des véhicules à moteur à combustion, vont modifier de manière significative la distribution urbaine de marchandises. En outre, le nouveau modèle commercial du commerce électronique apporte également de nouvelles opportunités et de nouveaux défis à la distribution urbaine de marchandises. Cette thèse se concentre sur un nouveau système de distribution visant à fournir de meilleurs services pour la distribution logistique urbaine. Nous allons tout d'abord étudier un système de livraison urbaine robotisé basé sur un fourgon qui peut transporter plusieurs robots, ce qui rend le système de distribution plus flexible et efficace. Ensuite, nous proposons d'intégrer la technologie de recharge en route et de recharge inverse dans le système de livraison urbaine robotisée par camionnette dans les opérations logistiques, afin d'économiser le temps pendant lequel les camionnettes électriques transportent les robots pour les recharger, ce qui augmente l'efficacité des systèmes de distribution. Enfin, nous incorporons les ramassages hybrides et les opérations de livraison dans le système de distribution urbaine robotisée basés sur des fourgons pour nous adapter au nouveau modèle commercial des entreprises de commerce électronique.

Title: Optimization models and methods for tour planning in smart urban logistics

Keywords: Innovative last mile distribution, Mothership based robot services, Electric vehicle routing, En-route charging, Hybrid pickup and delivery

Abstract: Urban distribution refers to the distribution activities of goods serving urban areas and suburbs. Recent technological advances in unmanned distribution field, as well as new regulations limiting the use of combustion engine vehicles, will significantly change urban goods distribution. Besides, the new e-commerce business model also brings new opportunities and challenges to urban goods distribution. This thesis focuses on a novel distribution system to provide better services for urban logistics distribution. We first study a van-based robot urban delivery system and allow a van can carry multiple robot to make the distribution system more flexible. Then we incorporate en-route charging, and reverse charging technology into the van-based robot urban delivery system in logistics operations, to effectively use the time during which electric vans are carrying robots to recharge the robots, thereby increasing distribution systems' efficiency. Finally, we incorporate hybrid pickup and delivery operations into the van-based robot urban distribution system to adapt to the new business model of e-commerce enterprises.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France