First-mile logistics parcel pickup: vehicle routing with packing constraints under disruption

Iván Giménez-Palacios[a], Francisco Parreño[a], Ramón Álvarez-Valdés[b], Célia Paquay[c], Beatriz Brito Oliveira[d], Maria Antónia Carravilla[d], José Fernando Oliveira[d]

[a]*Departamento de Matemáticas, University of Castilla-La Mancha, Albacete, Spain*
[b]*Departament d'Estadística i Investigació Operativa, University of Valencia, Valencia, Spain*
[c]*HEC Liège, Management School of the University of Liège, Liège, Belgium*
[d]*INESC TEC, Faculty of Engineering, University of Porto, Porto, Portugal*

Abstract

First-mile logistics tackles the movement of products from retailers to a warehouse or distribution centre. This first step towards the end customer has been pushed by large e-commerce platforms forming extensive networks of partners and is critical for fast deliveries. First-mile pickup requires efficient methods different from those developed for last-mile delivery, among other reasons due to the complexity of cargo features and volume – increasing the relevance of advanced packing methods. More importantly, the problem is essentially dynamic and the pickup process, in which the vehicle is initially empty, is much more flexible to react to disruptions arising when the vehicles are en route.

We model the static first-mile pickup problem as a vehicle routing problem for a heterogeneous fleet, with time windows and three-dimensional packing constraints. Moreover, we propose an approach to tackle the dynamic problem, in which the routes can be modified to accommodate disruptions – new customers' demands and modified requests of known customers that are arriving while the initially established routes are being covered. We propose three reactive strategies for addressing the disruptions depending on the number of vehicles available, and study their results on a newly generated benchmark for dynamic problems.

The results allow quantifying the impact of disruptions depending on the strategy used and can help the logistics companies to define their own strategy, considering the characteristics of their customers and products and the available fleet.

Keywords: first-mile logistics, disruption, vehicle routing, packing

---

1. Introduction

First-mile logistics refers to the first stage of the journey that products will make in the supply chain process. In recent years, much attention has been paid to the last-mile delivery, the final transfer of the products to customers. Customers like to shop online and receive their products quickly, and companies and researchers have devoted enormous efforts to developing ever faster and more efficient delivery procedures, taking into account the characteristics and regulations imposed on residential areas and inner cities. However, it is becoming increasingly clear that, by comparison, first-mile pickup, the movement of goods from retailers to a warehouse or distribution centre, remains inefficient.

The first-mile echelon of retailing supply chains may operate under different modes. First-mile refers to packing and shipping the products by individual retailers from a factory or retail warehouse to a distribution or a fulfilment centre. Unlike last-mile, where a logistics provider centrally plans and controls outbound transportation for multiple retailers, first-mile operations are run either by a logistics provider or by the seller. Consequently, it may be more difficult to group and centralize multiple retailers' first-mile operations. In the Amazon FBM (Fulfilment By Merchant) system, for example, retailers are fully responsible for the logistics operation, and therefore first-mile operations are not consolidated. These systems do take a toll on the liveability of cities. A recent survey about the citizens' perceptions of urban freight logistics concluded that as the citizens' awareness increases, their expectations of efficient and effective freight operations also increase (Amaya et al., 2021). A step in this direction is to have a single logistics provider who manages the first-mile logistics of multiple retailers, with retailers having no direct control over transportation decisions. The logistics provider may just be a transport company that serves multiple retailers who wish to send their shipments to order fulfilment facilities.

Another setting where first-mile logistics has the characteristics of the problem under study in this paper is the B2B context of couriers acting as 3PL operators, assuming total control over the logistic flow from first-mile pickup in the origin locations until the last-mile delivery in the destination. These carriers have consolidation centres where packages are brought together and then sorted to be sent to the delivery locations, often involving several transportation modes. Under such a centralized first-mile operation scenario, service levels need special attention, and the management of uncertainties such as disruptions (e.g., transportation delays) must be considered since the planning phase.

In this paper, we provide a tool that allows planning the first-mile collection of parcels by a courier or other type of logistics provider, taking into account the uncertainty caused by disruptions and changes while considering different service level contracts, as request cancellation recourse actions can be parameterized at the customer level.

First-mile logistics does not simply mirror the characteristics and strategies of last-mile delivery. On the one hand, retailers are often more geographically disperse than end-customers and delivery strategies and modes developed in last-mile delivery city logistics cannot be applied. On the other hand, retailers may send simultaneously many boxes of different sizes for different customers, and packing constraints, going far beyond simple capacity constraints, must be taken into account. Even more importantly, the pickup process is subject to different disruptions than those arising in last-mile delivery and can react in a very different way. As vehicles are initially empty and progressively filled up, they are more prepared to accommodate new requests or changes to previous requests. It often happens that the exact number and dimensions of the boxes are only known when the vehicle arrives at the customer.

In its basic static version, the first-mile pickup problem can be viewed as a vehicle routing problem with packing constraints. Goods produced by a set of retailers must be picked up by a fleet of vehicles of different types and transported to a depot. As these retailers are, in fact, the customers of the logistics company they will be referred to as customers throughout the paper. The location of customers, their time windows, and the distances between them define a routing problem. The dimensions and weights of the boxes to be picked up and the characteristics of the vehicles define a three-dimensional packing problem, including capacity, stability, and axle weight constraints.

However, this basic version falls very short of the complexity of the situations that logistics companies face in their daily operations. The continuous flow of information and changing roads and even vehicle conditions produce disruptions, changes in the initial data of the problem being solved. According to Pillac et al. (2013) (Table 1), the problem can fall into one of four categories: static and deterministic, when the decisions are made a priori and cannot be changed, and the parameters and data are known with certainty; static and stochastic, when decisions initially made do not change but some data are unknown or uncertain; dynamic and deterministic, when decisions can be changed due to new revealed information, which is known with certainty; and dynamic and stochastic, when decisions can be changed with some new data that may be uncertain. In this paper, we consider the dynamic and deterministic case. The company can take advantage of the new information arriving when the vehicles are already traveling along the initially established routes and modify them, considering the new information as clear and certain. As in Psaraftis et al. (2016) and Eglese and Zambirinis (2018), we assume that there is a fast and reliable communication system between the dispatch office and the drivers so that timely information about the actual status of the vehicles is available, and the drivers can be quickly informed regarding the revised plans. As this is a pickup process, changes and new customer requests can be accommodated if vehicles have empty space left, if the routes can be rearranged to obtain the space required by the new requests, or if additional vehicles are used.

Table 1: Taxonomy of vehicle routing problems by information evolution and quality(adapted from Pillac et al. (2013))

		Information is known with certainty?	
		Yes	No
Decisions can be modified in response to new information revealed after time 0?	No	Static and deterministic	Static and stochastic
	Yes	Dynamic and deterministic	Dynamic and stochastic

There are several types of disruptions. Concerning customers, the main disruption is that new requests arise and it has to be decided on the fly whether they can be met within the current schedule or not. In what follows, we will distinguish the *known customers* whose requests have been received before the realisation of the routing process, and the *new customers* whose requests are revealed over time. Other disruptions may correspond to known customers, who can modify their requests, increasing, decreasing, or changing the number and dimensions of the boxes to be picked up, or change their time windows, due to last minute production problems, or even change their pick-up point. In that case, we will talk about *modified requests* of known customers. The service provided by the logistics company can also be disrupted. The most common situation is that a vehicle breaks down, before or even during the route, and the customers who have not yet been served have to be rearranged on other existing routes. Disruptions can also be due to drivers calling in sick or problems with the depot. A third type of disruption, external to customers and service providers, is due to changes in the road network, when a road segment is blocked, or, more often, the speed at which a segment is traversed is reduced due to congestion or other causes. In this work, we consider the two main important disruptions concerning customers: new customers to be included and, in the case of known customers, modified requests with respect to the number of boxes to be picked up.

A final aspect of the problem is the strategy adopted by the logistics company to deal with disruptions. The strategies to manage disruption risks and improve resiliency are often classified in proactive (mitigation or preventive) and reactive (contingency or responsive) strategies. The former can be handled with a robust approach that compute solutions able to accommodate as much as possible unforeseen disruptions without the need of new computations, the latter aims at providing optimized recovering techniques, that minimizes the changes regarding the initial solutions when disruptions arise. For example, within a supply chain, a mitigation strategy could be to select some facilities to be fortified against disruptions in a planning phase, whereas a contingency strategy would be the re-allocation of customers to facilities after the disruption occurs (Fattahi et al., 2017). In practice, many companies already have a software providing an initial solution, but they seldom have a tool to manage real-time disruptions. In fact, Ivanov et al. (2016) state that a common shortcoming of works tackling disruptions in the supply chain is that they overlook the implementation dynamics. Overall, it is usual to consider disruptions as static events without considering their recovery policies. For this reason, we intend to focus on the reactive approach in this research. Especially focusing on the transportation context, it is critical to focus on the reactive processes to improve the execution and (dynamic) implementation. Mitigating the impact of disruptions is often not dependent on complex and time-consuming methods but can be tackled by simple and efficient responses to uncertain disruptions, such as Kanban-based heuristics (Cui et al., 2016).

When building recovery algorithms, a conservative approach is to stick to the initial plan and only accommodate disruptions if they do not change the order of service to existing customers. At the opposite extreme, a versatile strategy consists of reoptimizing, that is, solving the problem again every time a new piece of information is revealed, obviously considering fixed the part of the route that has already been carried out. In our work, we favour a balanced approach in which the benefits of accepting new or modified requests and the costs of modifying the initial routes are considered. Alternatives concerning the number of vehicles are studied, as it is the common practice of logistics operators who consider the disruptions and balance the cost of changes to the solution and the benefits of accommodating new or modified requests. Balanced strategies can be more conservative, if the number of vehicles cannot be changed, or more flexible, if the possibility of adding a new vehicle is open, and its cost is taken into account.

Tackling first-mile logistics brings new challenges to the literature, compared to last-mile delivery, namely regarding the impact of packing issues and the ability and strategies to accommodate disruptions. In this context, the main contributions of our work are:

- An algorithm for the capacitated VRP with time windows and packing constraints, combining BRKGA with local search, able to efficiently solve the static version of the problem, producing competitive results compared to existing procedures for closely related problems.
- A reactive procedure, based on local search, to deal with disruptions arising in real time, starting from the solution of the static problem and considering the costs and benefits of accommodating new or modified requests in existing routes or even adding new routes.
- Three strategies for addressing the dynamic problem, depending on the number of vehicles available, ranging from the case in which no new vehicles can be added to the extreme case in which an unlimited number of vehicles are available. These strategies are extensively tested and their effects are discussed, to help logistic companies to define their strategies, according to the specificity of the their customers, the type of products being picked, and the characteristics of their vehicle fleet.
- A benchmark of dynamic problems, with a thorough description of the way it has been generated, which can be used to test other procedures for solving dynamic problems.

The paper is structured as follows: Section 2 reviews the related work. The static pickup problem is described in Section 3 and solved in Section 4. In Section 5, the dynamic problem as well as the procedures developed to cope with disruptions are explained. Section 6 contains an extensive computational study, including the description of the instances used and the new instances generated, the results obtained for the static problem, the comparison and discussion of the three strategies developed for the dynamic problem, and the managerial insights drawn from the study. Section 7 presents the conclusions and future work.

2. Related work

Over the very last years, the concept of first-mile logistics has been studied from various perspectives. In passenger transportation, the first-mile ridesharing problem aims at improving the connectivity between passenger's home and the closest public transportation hub bus terminal, metro station, ...), and thus reduce the number of people using their private for commuting (Kåresdotter et al. (2022); Zhang et al. (2020)). Compared to standard VRP, here the passenger satisfaction is key to make the solution applicable. Ning et al. (2021) studied the first-mile ridesharing with uncertain requests and Chen et al. (2020) proposed solutions using autonomous vehicles to transport passengers from their home to metro stations.

The first mile problem is also studied in the context of circular economy supply chains, that needs to organize the collection of used materials from individual consumers. In reverse logistics, the last mile problem is indeed viewed as a first mile problem which relates to managing the flow of reusable and recyclable materials from their users. For instance, Jäämaa and Kaipia (2022) introduce the first mile problem in the context of end-of-life textile collection, study the current end-of-life textile collection system in place, and propose a prediction model for the textile volumes to be collected to improve the collection system.

The first mile problem is also studied from the collaboration perspective. In Wang and Huang (2021), the authors suggest a business model for the first-mile collection of the parcel shipping business. They investigate the possibility of having a common service provider who can help several courier logistics companies to perform collection activities. They use a game theoretical approach to determine the market conditions that would make this collaboration profitable for the courier logistics companies.

Although there are very few papers explicitly addressing the first-mile collection or pickup problem it is very close to the well-known pickup-and-delivery problem. The most comprehensive and relevant survey on the general pickup-and-delivery problem was published in two parts, Parragh et al. (2008a) and Parragh et al. (2008b). Under the designation of general pickup-and-delivery problem, in the first part of the survey, the authors discuss the problem in which goods are transported from the depot to linehaul customers and from backhaul customers to the depot, i.e., what is usually called vehicle routing problem with backhauls, and four subtypes of this problem. The sub-problem closer to the first-mile pickup is the vehicle routing problem with clustered backhauls, in which all linehauls are served before backhauls. In the second part of the survey, pickup and delivery problems are approached, i.e., problems where goods are transported between pickup and delivery locations. This second category of problems encompasses problems as the pickup and delivery vehicle routing problem, where pickup and delivery points are unpaired; the pickup and delivery problem, where pickup and delivery points are paired; and the dial-a-ride problem, which deals with passenger transportation between paired pickup and delivery points while taking into account user inconvenience. For all problems, formulations are given, and solution approaches are discussed. From the same year, it is worthwhile of mentioning the review on one-to-one pickup and delivery problems Cordeau et al. (2008). This topic is still a current challenge for the scientific community. A more recent survey on the dial-a-ride problem can be found in Ho et al. (2018) and a recent application with electric vehicles was proposed in Masmoudi et al. (2018). Vehicle routing problems with simultaneous pickup and delivery are surveyed in Koç et al. (2020), while a very recent survey on dynamic pick-up-and-delivery problems can be found in Wang and Zhao (2021). The integrated pick-up and delivery problem has also been tackled recently in an online setting and incorporating learning effects (Zhang et al., 2019).

The VRP is another strongly related problem widely studied, including when uncertainty is considered. Most of the recent survey papers on the VRP look at the electric vehicle routing problem (Kucukoglu et al., 2021), the vehicle routing problem with side-kick unmanned aerial vehicles (Li et al., 2021), including under a two-echelon perspective, and green vehicle routing (Moghdani et al., 2021). However, much closer to the problem under study are the stochastic VRP and the dynamic VRP. For a general overview over the dynamic VRP, the reader may consult Berbeglia et al. (2010), Pillac et al. (2013), Toth and Vigo (2014), or Eglese and Zambirinis (2018). Nevertheless, the most recent survey on the dynamic VRP (DVRP) is Ojeda Rios et al. (2021). In this survey a taxonomy for the DVRP is proposed and both the dynamic and stochastic and the dynamic and deterministic problems (building on Pillac et al. (2013) classification and subsequent survey and taxonomy of Psaraftis et al. (2016)) are reviewed. According to Ojeda Rios et al. (2021), the main disruption sources in dynamic VRPs are: customer requests, travel time, service time, vehicle availability and customer demand, which may be treated either under a stochastic or deterministic framework. According to these authors, around half of the papers dealing of the dynamic VRP consider that the source of dynamism is at customer requests or at customer demand.

The goal of disruption management is to cope with real-time and unpredictable events while minimizing the deviations w.r.t. initial plans (Clausen et al., 2001). This topic was first studied to help airlines to face schedule issues due to unpredictable weather conditions, but has been rapidly extended to other fields such as production scheduling, supply chain management and transportation planning (Yu and Qi, 2004). More recently, innovative demand-responsive transport systems are leading to the development of approaches that aim to incorporate the uncertainty of disruptions, such as deviations to the planned route to accept late requests (Bruni et al., 2014). A literature review of disruption management in VRP can be found in Eglese and Zambirinis (2018). Wang and Cao (2008) propose a recovery model based on local search operators to handle disruptions for a VRPTW with backhauls. Hu and Sun (2012) present a knowledge-based modeling approach, where the knowledge of experienced schedulers is combined to operations research algorithms, to deal with the modification of a customer request. Wang et al. (2012) consider the VRPTW for identical cargo delivery. They deal with several types of disruptions and assess their solutions by computing the deviations related to customers (service during soft time windows), drivers (change in the route), and logistic provider (change in cost based on number of vehicles and total distance). They use the nested partitions method to solve their recovery model. Spliet et al. (2014) study the vehicle rescheduling problem in retail industry: they assume to have a master schedule for the vehicles on a long-term basis (6 months) and that the demand of the customers is revealed later in time. They aim at minimizing the deviation cost based on the observation that changes are cheaper if performed on a later stage of the routes. They propose a mixed integer programming formulation for moderate-size instances and a two-phase heuristic that removes the last locations of routes and reschedules them.

In this work, we deal with disruption management in the context of a VRP with packing constraints. More precisely, we consider here unpredictable events related to the arrival of new customers during the day and a change related to the number of boxes to be picked up at a customer's location. This type of unpredictable events has been studied in the context of disruption management. Previous works have defined both major disruptions (such as natural hazards) and minor disruptions (such as machine failures) as critical situations to tackle in this context (Parajuli et al., 2021). Transportation delays have also been listed as potential causes of disruption in supply chains (Zhen et al., 2016). The term disruption has also been used to designate events that cause sudden changes in supply levels from specific suppliers (Cui et al., 2016). Therefore, we use the term disruption in the context of first-mile logistics to designate unpredictable events related to the arrival of requests and changes in the demand level to which companies need proper tools to react and adapt. To the best of our knowledge, no previous work has proposed a method to tackle disruption with respect to the packing problem alone. The closest problem is the online packing and dynamic packing problem where the items to be packed arrived online with no previous information available (Coffman et al., 1983; Seiden, 2001). Moreover, the papers studying those two problems remain rather theoretical (approximation algorithms, d-dimensional bin packing). Moreover, even if VRP with packing constraints received more attention during the previous years (Moura and Oliveira, 2009; Ceschia et al., 2013; Pollaris et al., 2015), there is no mention of disruption management in the literature yet.

3. The static pick-up problem

In this problem, there is a set of customers to be served, each one with a location represented as a node in a network. The depot is also a node, with given opening hours. The time to move between two nodes of the network is known and given. Customers have a demand, i.e. a set of boxes, with known dimensions and weight, to be picked up. The boxes must be picked up within a time window associated with each customer, and each demand has an assigned service or picking time, which is the time needed for the driver to pick up and load all the boxes into the vehicle.

The logistics company has a fleet with different types of vehicles, with a given number of vehicles of each type. Each vehicle type has a load volume defined by its three-dimensional dimensions, a weight capacity, and a daily cost. The driver of each vehicle has a defined shift during which he/she must make customer visits.

The objective is to minimize both the number of vehicles used and the total distance to be travelled on all routes. We consider that saving on the number of vehicles used is preferable to reducing the distance travelled, so we will choose the solution with the least number of vehicles, and, in case of a tie, the one with the least distance travelled.

This problem is known as the capacitated vehicle routing problem with pickups, time windows, and packing constraints. A solution to this problem is called a plan and consists of a collection of routes, each with an associated schedule and packing. A route is defined as a sequence of nodes to be visited by a vehicle. The schedule associated with a route is the time information for each node, i.e., the vehicle arrival time at the node, the start time of the load at the node, and the vehicle departure time from the node, depot included.

Associated with each node of the route, the packing of the loaded boxes must be provided. For this purpose, the loading space inside the vehicle is described as a parallelepiped. The packing solution is described by the positions of the front bottom left corner and of the upper right rear corner (to account for different box orientations) of each box inside the vehicle loading space.

The feasibility of a solution depends on the feasibility of the routes, and associated schedule and packing.

Route feasibility. A feasible route must satisfy:

- All boxes must be collected.
- Customers are visited exactly once.
- Each route starts and ends at the depot.

Schedule feasibility. The feasibility of every route depends on the schedule and on the packing. A schedule is feasible if the following constraints are met:

- Time windows for each vehicle: the driver of each vehicle has a time window regarding departure and arrival to the depot.
- Time windows for each customer: at each node, the load start time must be in the time window of the customer, and the process must finish also within this time window.
- The loading starts after arrival at node, and ends at the time the last box is loaded.
- Time consistency: the arrival time at a node is defined as the time the previous node of the route is left plus the time required to traverse the network arc connecting the two nodes.

Packing feasibility. A packing is feasible if it satisfies the following conditions:

- The set of boxes of a customer is assigned to exactly one vehicle.
- Each box lies within the boundaries of the vehicle.
- Boxes cannot overlap each other.
- The total weight of the boxes inside a vehicle cannot exceed the maximum capacity of the vehicle.
- Orthogonality: every box must be loaded with its edges parallel to the vehicle boundaries.
- Vertical stability: boxes must be completely supported by other boxes or the vehicle floor.
- Rotation: each box has six possible rotations, but in some cases only some of them are allowed.

The most common packing policy is to pack the boxes of one customer without moving the boxes of those previously visited, satisfying the sequence-loading constraints (Pollaris et al., 2015) or LIFO constraints (Iori and Martello, 2010). This will be our first packing policy. Nevertheless, in our study we include two more policies, to see if more complex packing procedures can produce significant benefits in the number of trucks needed or in the total distance traveled. In the second policy, the prepacked reachable boxes are allowed to be moved. A box is considered reachable if it can be picked up without moving any other boxes. In the third policy, a new packing is determined at each node, starting from scratch and considering all boxes of the current customer and all other customers previously included in the vehicle. Although this is unlikely to occur in practice, it may be a good point of comparison for the other two policies. A formal description of the static pick-up problem can be found in Appendix A.

4. Solving the static problem

To solve the static problem described in Section 3, we have developed a two-phase algorithm. In the first phase, we resort to a genetic algorithm, based on the Multi-Parent Biased Random-Key Genetic Algorithm with Implicit Path-Relinking (MP-BRKGA-IPR) proposed in Andrade et al. (2021), which uses as decoder a packing algorithm adapted from Alvarez et al. (2015). This first phase focuses on minimizing the number of vehicles required. In a second phase, a local search algorithm improves the solution of the first phase in terms of the total distance traveled.

The MP-BRKGA-IPR is selected as the solution method framework since it is a state-of-the-art development of a genetic algorithm, which has been recurrently used in the literature with good results in several application areas, including the ones tackled in this problem: routing and packing. Genetic algorithms have been developed for routing problems in different contexts, such as containership routing with deliveries and pick-ups (Karlaftis et al., 2009), routing problems under uncertainty (Allahviranloo et al., 2014), or container transportation in intelligent logistics (Fan et al., 2020). More specifically, biased random-key genetic algorithms have been successfully applied to routing problems (Ruiz et al., 2019; Pinto et al., 2020). In previous works, the good performance of biased random-key genetic algorithms has also been demonstrated for packing problems in different settings, such as container loading (Ramos et al., 2018) and closely related berth allocation problems (Correcher and Alvarez-Valdes, 2017). Additionally, Alvarez-Valdes et al. (2013) study the value of integrating path-relinking strategies in other metaheuristic frameworks, as Andrade et al. (2021) propose for genetic algorithms. The successful development of such algorithms for the contexts most related to this problem, as well as the potential demonstrated by the state-of-the-art development of the MP-BRKGA-IPR, make this a suitable and relevant approach for the first-mile logistics problem here tackled, which combines routing and packing problems.

4.1. First phase: the genetic algorithm

The BRKGA is a genetic algorithm that consists of evolving a population of solutions (chromosomes) represented as n-dimensional vectors of values between 0 and 1. These values are called random keys. Bean (1994) proposed this representation, which requires a fitness function and a decoder, to transform the chromosomes into solutions to the problem. In the original BRKGA, there is an initial population $\mathcal{P}$ which includes an elite population $\mathcal{P}_e$ composed of solutions with the best fitness. At each iteration, the elite set is copied to the next generation, the crossover operator produces new solutions by combining elite and non-elite individuals and the remainder of the new generation is filled with new random solutions. In the Multi-Parent crossover, the combination to obtain new individuals is done among several individuals, π_t, of which π_e belong to the elite. Path-Relinking is a search intensification strategy presented in Glover and Laguna (1997), in which the path that links two good solutions is explored. Combined with the Multi-Parent BRKGA (BRKGA-MP), the Implicit Path-Relinking is called after a fixed number of iterations occurs, trying to introduce new and better solutions to the population and speed up the convergence.

In our algorithm, the chromosome is divided into two sections of length C, where C is the number of customers to be served. The first section represents the sequence in which the customers will be loaded into vehicles, and the second the order in which the types of vehicles will be used. In the second section, C works as an upper bound on the number of vehicles required to serve all customers. In the worst case each vehicle may have only one customer assigned. The fitness function is the original objective function of the problem (see equation (A.1)).

3	5	1	2	4	1	2	2	1	1

Figure 1: Representation of a chromosome, with five customers and two types of vehicles

An example chromosome is shown in Figure 1, for an instance with five customers and two types of vehicles. To obtain a solution, the first vehicle, of type 1, is taken and the customers are packed into it in order, first customer 3, then customer 5 and so on. When a customer does not fit completely in the vehicle, the vehicle is closed. For example, if customer 1 does not fit in the first vehicle, it will only include customers 3 and 5, a second vehicle, now of type 2, is taken, and customers are packed into it starting from customer 1.

To decode the chromosomes, the types of vehicles are taken in the order given in the second part of the chromosome, and the customers are packed into them in the order given by the first part of the chromosome. While the routing sub-problem is implicitly solved when knowing the customer sequence, to load the customers' boxes into a vehicle, a packing algorithm is needed to work as a decoder of the genetic algorithm, i.e., to transform the solution representation into an actual packing solution. As there are three different packing policies described in Section 3, three decoders were built: the "no movement", "some movements", and "all movements" decoders.

In all decoders, to pack the boxes of the current customer, we use a constructive algorithm based on maximal spaces, adapted from Alvarez et al. (2015). Given a vehicle and a set of boxes, boxes are placed one by one in a free space of the vehicle. The distinctive aspect of this approach is that a list of non-disjoint parallelepiped spaces represents the overall free space. Each space is limited by a box or by the vehicle (wall, floor or ceiling) and never by another space. Therefore, spaces are as big as possible (maximal) but may, and will, overlap each other. On the one hand, this complicates the update of the free space list, as placing one box will impact several spaces, but on the other hand, it provides very efficient usage of the container's overall space. Therefore, two main decisions are involved in the constructive algorithm: which space to select to be the next one to fill and which box will be packed in that space. Several rules may be used, and a detailed description of the constructive algorithm can be found in Appendix B.

The difference among the three decoders is on the set of boxes that, in each customer, is considered for the packing process. While in the "no movement" only the boxes of the current customer are at stake, in the "some movements" decoder, some boxes from previous customers (the reachable boxes) are removed from the solution and considered for the packing process together with the boxes of the current customer, and, finally, in the “all movements” decoder, the vehicle is virtually emptied, and all boxes of previous customers and the boxes of the current customer are at stake for the packing process. Therefore, in the two last decoders, the already packed boxes may stay in the same place or be moved to a different location in the vehicle if a better solution is generated in this way.

Improving the decoders

It can be expected that the more box movements are allowed, the better the solutions should be. However, as no exact method is used, there may be some cases where better results are obtained with less flexible methods. In an attempt to improve the results, we add two more procedures:

- Combine different methods: in the last two packing policies, where movements of boxes are allowed (reachable boxes or all boxes), whenever the boxes of a set of customers cannot fit in one vehicle, the method that does not allow movements is tried.

- Randomized constructive algorithm: the constructive algorithm is randomized. Every time a maximal space is to be chosen, it is chosen randomly between the one with the largest z-coordinate and the one with the smallest x-coordinate. When a set of customers cannot fit together in the vehicle, the randomized constructive algorithm is run up to 10 times.

The decoder is going to be used a large number of times, so it has to be as efficient and fast as possible. Each time a set of customers is checked for feasibility, a quick check of the volume and time windows is done first, and then it is checked to see if it is a set that has already been studied. Each time a set of customers is checked, the information about whether or not they fit is saved to avoid unnecessary decoder runs.

4.2. Second phase: Local search

The first phase algorithm is designed to fit as many customers as possible in each vehicle, thus minimising the number of vehicles. However, at no point is the total distance travelled explicitly taken into account, so sometimes the solution can be improved in terms of distance. Therefore, in a second phase a local search is introduced to improve the solutions. This local search is based on two different moves:

- Insertion: an attempt is made to insert each customer in all positions of all other routes. If there are some positions where it could be inserted, maintaining the route feasible and decreasing the objective function, the move that produces the largest decrease in the objective function is performed.

- Swapping: each pair of customers is taken, their routes are swapped and all the positions of each customer on their new route are studied. If the two customers are on the same route, the routes are not exchanged, but all the positions for both customers on this route are explored.

The full local search consists of first trying all possible insertions and then all possible swaps. Each time there is an improvement in the objective function, the solution is updated and the process starts from the beginning, ending when no move produces an improvement in the solution.

We use local search in two different moments of the BRKGA: first, once the initial population is generated, we apply local search to some of its chromosomes trying to start with better solutions. Then, we also apply local search to some of the solutions of the final population, depending on the available running time, trying to improve the total distance travelled, and, sometimes, even to reduce the number of vehicles.

5. The dynamic pick-up problem: dealing with disruptions

In first-mile problems, different types of changes can arise in the original data during the course of the routes. When these changes occur, if we allow the initial solution to change, the problem is called a Dynamic Vehicle Routing Problem (Toth and Vigo, 2014). We will call each of the data modifications *a disruption*. Disruptions can be related to parameters or features that were assumed to be known with certainty and are revealed to be different or related to information that is revealed as time unfolds, for instance, new customer requests (as opposed to known requests from known customers).

In the problem at hand, the data can be divided into three categories:

1. Customer related data:
    - New requests: during the route, new customer pickup requests may arrive;
    - Set of boxes to be picked up: the number, dimensions, and weights of the boxes may change. They may increase if the customer receives new orders, or decrease due to last minute production problems;
    - Location: an error may occur when reporting a customer's location. The location may also change if the customer has multiple production facilities;
    - Pickup time window: a change to a customer's time-windows may occur, such as a delayed opening or an early closing;
    - Service duration: it may depend on the number of boxes to be picked up. If this number changes, so will the service time.
2. Data related to the transport provider
    - Vehicle type: similarly to customer boxes, the initial information on the dimensions of a vehicle, the maximum weight it can hold, or the cost associated with its use may change;
    - Driver shift: modifications as in customer time-windows disruptions.
3. Network related data
    - Duration matrix: the time it takes to go from customer i to customer j is a value that usually depends on external conditions such as traffic, weather conditions, possible accidents, etc.

Table 2 summarizes the real-world applicability of the different parameters defining the disruptions, together with their likelihood.

Table 2: Parameters which may suffer disruptions and their likelihood in real-world problems

Parameters	**Likelihood**			
	more likely	likely	less likely	not likely
Customers				
Immediate requests	×			
Number of boxes	×			
Dimensions or weight of boxes		×		
Location				×
Time window	×			
Service duration	depends on boxes			
Service provider				
Vehicles features			×	
Driver shift			×	
Network				
Duration matrix	×			

In this paper we focus on disruptions related to new customer requests revealed throughout the day and to modified requests from known customers. Regarding the latter type of disruption, we consider below that a customer may ask to change the number of boxes of a type initially present in its original request.

As explained in Eglese and Zambirinis (2018), the objective to be optimized can differ from the static problem when dealing with disruptions. In our case, we attempt to accommodate the larger number of disruptions, without drastically changing the solution. Each disruption d arises at time t_d, and is treated independently. The vehicles are already on their routes and therefore changes can only be made to the unrouted legs. In other words, customers that are already served and that the vehicle is going to serve cannot be modified (no rerouting). To avoid major changes in the solution, we will hierarchically apply local operators defined in Section 4.2 , from the lowest impact to the highest. More precisely, we will manage disruptions in the following way:

- **New request**: when a new customer arises, an attempt is made to insert it into each already existing route, using the insertion move. If this is possible on several routes, the one that produces the minimum increase in the total distance is kept. If the insertion is not possible, a new route is created to serve this customer, and the full local search process is applied to the new solution, taking into account only those parts of the routes that have not yet been served and can therefore be modified. The goal is to try to keep the same number of routes, or at least to reduce the total distance travelled. The pseudocode for this procedure can be found in Algorithm 1.

  As an example, Figure 2a shows an original solution with eight customers and three routes. In Figure 2b, a new customer arises when the vehicles are already on their routes. Figure 2c shows the sections of the routes that can be modified in dashed line. In Figure 2d the new customer is inserted into one of the existing routes.

- **Modified request**: when a known customer modifies its demand, it is first checked if the new demand can be met by keeping the customer in the route and position in which it was originally placed. If this is not possible, the customer is removed from the route it was on and treated as a new customer, with the new demand, using Algorithm 1.

  Figure 3a shows the same original solution, in Figure 3b an existing customer increases its demand (marked with a red dot) when the trucks are already serving the customers. The new demand cannot be picked up on the existing route, so it is removed from the route (Figure 3c). Finally, by applying Algorithm 1 the customer is inserted into another existing route (Figure 3d).

Algorithm 1 New request disruption

```
i = new customer
R = set of routes, before the disruption
maxV = maximum number of available vehicles
insert = false
for r ∈ R do
  if (Customer i can be inserted in route r) then
    insert = true
    dist(r) = increase in total distance of route r, putting customer i in the best position in r
  end if
end for
if (insert == true) then
  Insert customer i in the route r with minimum dist(r)
else
  Create new route r to serve i
  Insert r in R
  R' = LocalSearch(R)
  if (|R'| ≤ maxV) then
    R = R'
  else
    The new request is dropped and R is back to the state before the disruption
  end if
end if
```

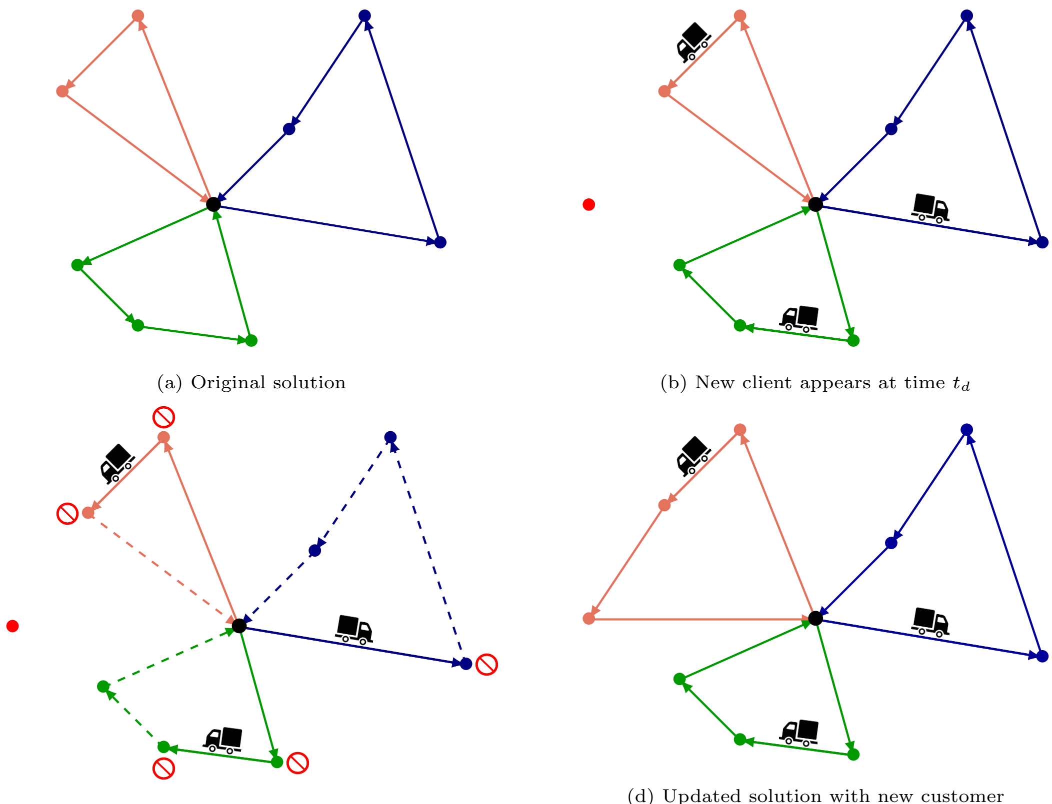
Using these procedures we have designed three strategies to generate post-disruption plans in terms of the maximum number of trucks that can be used once the routes are started:

Strategy 1: as many trucks as we need to cover all the new and modified requests.

Strategy 2: only the number of trucks initially planned.

Strategy 3: the number of trucks required by the solution of the original instance.

It is clear that in the first strategy all the disruptions will be accepted, in the worst case with a new truck for each request. In the second and third strategies, however, some of the disruptions may be rejected because it may be not possible to serve them without exceeding the allowed number of trucks. Note that



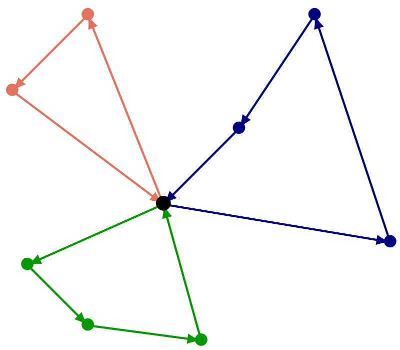
(a) Original solution

(b) New client appears at time t_d

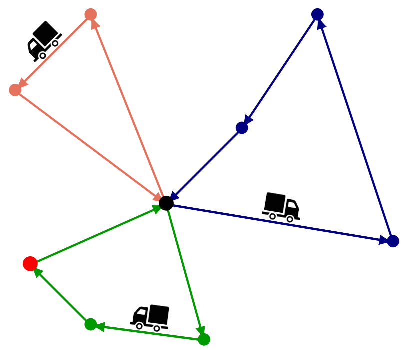
(c) customers served or to be served

(d) Updated solution with new customer

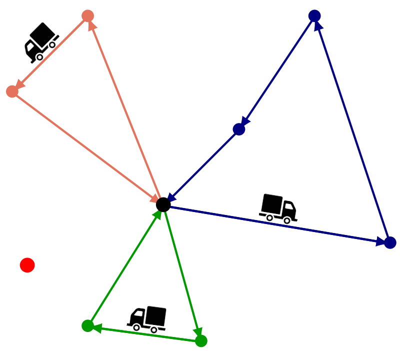
Figure 2: Insertion of new customer



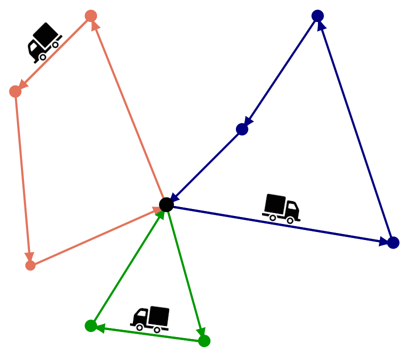
(a) Original solution



(b) A customer increases its demand at time t_d



(c) New demand cannot be picked up



(d) The customer is inserted in other route

Figure 3: Increase of demand of a customer

the maximum number of trucks allowed in the third strategy could cover all requests if they were known in advance. Nevertheless, the fact that part of the information is arriving after the routes are started may cause the number of trucks needed to increase, and some of the disruptions may be rejected. In Section 6.3, the three strategies will be compared in order to give some insights into the quality of the service that would be provided according to the chosen strategy.

6. Computational study

In this section, we present the results of the computational study. Our algorithm was implemented in C++ with 1 CPU, 1 Thread, 2.40 gigahertz, and 4 gigabytes of RAM.

Although the main objective of our study is to solve the dynamic version of the problem, we test first the MP-BRKGA-IPR and the Local Search for the static pickup problem to assess whether they are adequate for the solution process, and then we solve the dynamic problem with the three strategies. In this section, first the test instances are presented, and then the results of the two versions of the problem.

All the instances are available at Giménez-Palacios (2021) and the source code on `https://github.com/ivangipa/FirstMile`.

6.1. Test instances

First, we describe the instances for the static problem and then how they were adapted to include disruptions.

6.1.1. Instances for the static problem

We use the set of instances generated by Moura and Oliveira (2009), here denoted as MO1. These instances are based on the R1 and R2 instances of Solomon (1987) and BR2 of Bischoff and Ratcliff (1995), with only one vehicle type. They combine the information of the time windows and the position of the nodes of the R1 and R2 instances and the information of the BR2 boxes, obtaining two different groups, with 23 instances in each group:

- Group I: Each instance of this group has 1050 boxes, of 5 different types, distributed among 25 customers. Each customer demands between 30 and 80 boxes, 42 on average.
- Group II: The total number of boxes in this group is 1550, of 5 different types and 25 customers. Each customer demands between 50 to 100 boxes, 62 on average.

In addition, to increase the variability of the box types, we generated new instances, the MO2 set. They were created by keeping the time-windows and the customer coordinates, but introducing the box types from BR7 from Bischoff and Ratcliff (1995), BR8, and BR14 from Davies and Bischoff (1999). The total number of boxes and the average are the same, but with increasing box heterogeneity. 46 instances from each BR class were generated, so MO1 and MO2 total 184 instances.

We also use the 13 real-world instances in Ceschia et al. (2013), except for the SD-CSS3 instance because it includes customers that require more than one truck (we do not consider split pickups in this work). These instances have a high variability in terms of number of customers (from 11 to 129), number of box types (from 9 to 97) and total number of boxes (from 254 to 8060), and in some of them 2 vehicle types. For simplicity, we will refer to these as "Ceschia instances" from here onwards. Their characteristics appear in Table 3. The real problem tackled by Ceschia et al. (2013) shares most of the features of our problem, but differs in some characteristics such as including load bearing strength of the boxes, limiting the reachability of the boxes, and the way in which vertical stability is considered. Additionally, neither the depot nor the customers have time windows. Their objective is minimizing the total length of the routes, irrespective of the number of trucks. They also consider a simplified version, denoted as 3L-CVRP, that is more similar to our problem, adding fragility of boxes, relaxing the full support condition to partial support, and not considering weight distribution. Although a direct comparison is not possible, the results obtained for the 3L-CVRP problem can be taken as a close reference for the solution of our problem. In particular, an interesting feature of these instances is that the packing aspect has a strong influence in the solutions. Actually, if packing conditions are relaxed and substituted by one-dimensional weight and volume capacity constraints, the solutions they obtain for the Capacitated Vehicle Routing Problem require less than half the number of trucks, with a 40% reduction in the total distance.

Table 3: Features of Ceschia instances

Instance	**Customers**	**Box types**	**Boxes**
SD-CSS1	11	36	254
SD-CSS2	25	15	350
SD-CSS4	37	13	312
SD-CSS5	41	47	7035
SD-CSS6	43	97	8060
SD-CSS7	45	14	284
SD-CSS8	48	70	3275
SD-CSS9	56	45	1725
SD-CSS10	60	29	1840
SD-CSS11	92	34	3790
SD-CSS12	129	10	745
SD-CSS13	129	63	2880

6.1.2. Instances for the dynamic problem

The three sets of instances used in the static problem (MO1, MO2, and Ceschia) are taken as a basis for generating instances for the dynamic problem. Consider an instance for the static problem with C customers. The percentage of disruption, pod, determines the number of new customers, non, and the number of customers that will modify their demand, nom, so $non = nom = \lceil pod \times C \rceil$. Then, the new instance is divided into two parts: the part initially known and the disruptions. The part initially known consists of $C - non$ known customers. Among these, the information of $C - non - nom$ customers will not change while the demand of nom of them will experience disruption and become a modified request, as shown in Figure 4.

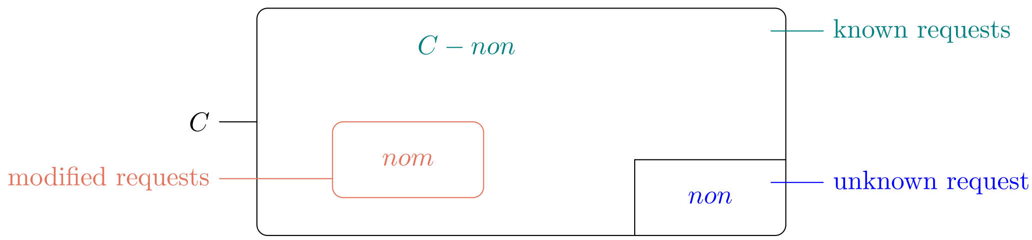


Figure 4: Representation of request types and number of disruptions

The procedure to create the disruptions (new requests and modified requests from known customers) follows these steps:

- Times at which the disruptions will be revealed

  We randomly select the times for the disruptions in $(0, b_0/2)$, where b_0 is the closing time of the depot, using an uniform distribution, to create the set TD of disruption times. All these times are different, as we assume that it is unlikely that two disruptions appear at the exact same time, and we deal with one disruption at a time.

- New customers

  The non new customers are selected from the set of customers of the original instance. We define TD_{used} as the set of disruption times already used. Initially this set is empty. We randomly select one value t from $TD \setminus TD_{used}$, and we select randomly one customer i with a start of its time window greater than $t+t_{0,i}$, where $t_{0,i}$ is the time it takes to go from the depot to customer i. This ensures that ultimately the new customer could be served by sending a new truck. If the number of trucks is fixed, this should give at least some flexibility to serve the new customer by modifying the existing routes. Once a value from TD is selected, it goes to TD_{used} so that it cannot appear in another disruption. The procedure is repeated until non customers have been selected.

  Each time a new customer i is selected, its information is taken from the description of the original instance used. When all these new customers are added to the initial static problem, we will have the complete set of customers of the original instance.

- Modified requests

  In this step, nom customers whose demand will be modified are selected. A value t is randomly selected in the set $TD \setminus TD_{used}$ and a customer i, not selected in the previous step, such that the starting time of its time window is greater than t. For this customer, part of the original demand will be the initially known demand and the remaining part will be added as a disruption. For each box type j with demand q_j in the original instance, a value r_j is taken at random from $(0, q_j)$. The initial demand of this box type will be r_j and the remaining $q_j - r_j$ boxes will appear in the disruption. Thus, when this disruption appears, the total customer demand will be the demand in the original instance.

We have generated 196 instances for each percentage of disruption $pod = \{0.05, 0.10, 0.15, 0.20, 0.25\}$. The way in which the new instances have been generated allows obtaining the value of the total information. That is, it allows quantifying the difference between having the complete information beforehand and having only a part of the information and receiving the remaining information during the run, in the form of disruptions.

6.2. Results for the static problem

In this part of the study, we consider the original instances of Moura and Oliveira (MO1) instances, as well as the ones we have generated based on them (MO2), and the real Ceschia instances. We will analyse how the MP-BRKGA-IPR solves the static problem, i.e., the one in which all the information about the customers to be served is initially known and no modifications occur during the routes. In addition, we will study the three packing policies defined in Section 3.

All runs on the MO1 and MO2 instances have been done with a time limit of 300 seconds, 240 seconds for the initial local search and the MP-BRKGA-IPR, and 60 seconds for the final local search. In the case of the Ceschia instances, as they are derived from a real case, the number of customers, box types, and total boxes are much higher than in the other instances, and the method takes longer to obtain meaningful results. For this reason, we set a maximum time of 8000 seconds for the initial local search and the MP-BRKGA-IPR and 2000 seconds for the final local search, for a total time of 10 000 seconds, as in Ceschia et al. (2013). For the MP-BRKGA-IPR, we have three independent populations of size 100, $|\mathcal{P}| = 100$. 30% of the population will be the elite, so $|\mathcal{P}_e| = 30$. In each generation, we use three parents, $\pi_t = 3$, to generate new individuals, of which two ($\pi_e = 2$) belong to the elite set. Regarding the objective function, a weight of 1 is associated to the total distance travel, and a weight of 1000 is associated to the vehicle use in all instances.

In MO1 and MO2 instances, there is only one type of vehicle, so the associated cost is set to 1. In the case of the Ceschia instances, there are some instances with two different vehicle types. For these cases, the lower capacity vehicle has a cost of 1, whereas the higher capacity vehicle has a cost of $\frac{V_2}{V_1}$, with V_v being the volume capacity of the v type vehicle. The idea is to penalize proportionally the vehicle type with the larger capacity.

Table 4: Results of MP-BRKGA-IPR with Local Search of MO1 instances

				Decoder 1: No moves			**Decoder 2: Some moves**			**Decoder 3: All moves**		
Class	C	BT	NB	*Dist*	*Vehic*	*Iter*	*Dist*	*Vehic*	*Iter*	*Dist*	*Vehic*	*Iter*
I	25	5	1050	32297	131	6584	31995	131	5956	31953	131	8775
II	25	5	1550	38885	149	6464	38646	148	4880	38614	148	8635
			Total	**71182**	**280**	**6524**	**70641**	**279**	**5418**	**70567**	**279**	**8705**

Table 4 shows the results for MO1 instances. The first four columns indicate the instance class, number of customers (C), box types (BT), and total number of boxes (NB). The results for each type of decoder are then shown: Decoder 1, when no movement of the boxes already packed is allowed when loading a new customer's cargo; Decoder 2, when the reachable boxes can be moved; and Decoder 3, when all the movements are allowed at each stop of the route. For each instance group and each decoder, the table shows *Dist*, the total distance of the best solutions, *Vehic*, the total number of vehicles of the best solutions, and *Iter*, the average number of iterations per instance of the MP-BRKGA-IPR.

Comparing the performance of the decoders, the results obtained with Decoder 3 are better overall, as expected, in terms of distance and number of vehicles. However, the results of Decoder 2 are very close, with the same number of vehicles and only 0.1% increase in distance. Decoder 1, on the other hand, needs 0.3% more trucks and increases the distance by 0.7% over Decoder 2. These small differences do not seem to justify the increased complexity in the packing procedures. Therefore, we will keep the Decoder 1 to solve the dynamic version of the problem in Section 6.3.

Table 5 shows the results for the MO2 instances. The information is structured as in Table 4, yet there are three groups of instances, depending on the BR instance class used in their generation. These classes represent instances with increasing box heterogeneity but the same total number of boxes. It can be observed that the distance and the number of vehicles increase with the heterogeneity of the boxes. The comparison of the Decoders is similar to that in Table 4, although the differences are slightly larger, with an increase of 1.1% in the number of trucks and 0.2% in the total distance of Decoder 1 over Decoder 2.

Table 5: Results of MP-BRKGA-IPR with Local Search of MO2 instances

				Decoder 1: No moves			**Decoder 2: Some moves**			**Decoder 3: All moves**		
BR	C	BT	NB	*Dist*	*Vehic*	*Iter*	*Dist*	*Vehic*	*Iter*	*Dist*	*Vehic*	*Iter*
7	25	20	1360	62295	238	5230	62334	234	4240	62019	230	7062
8	25	30	1360	63844	240	4716	62897	238	3662	61951	232	6425
14	25	90	1360	65475	249	3243	66001	247	2277	64080	234	4224
			Total	**191614**	**727**	**4396**	**191232**	**719**	**3393**	**188050**	**696**	**5904**

Finally, Table 6 shows the results for the Ceschia instances. The information is similar to that in previous tables, but presented for each individual instance. The last three columns of table contain the results reported by Ceschia et al. (2013) for three versions of the problem. The 3L-CVRP is a simplified routing and packing problem, very similar to the problem solved when using Decoder 1, although they consider fragility of boxes and their objective function only considers minimizing the total distance. The CLP is a pure packing problem. As no routing is involved, there are no LIFO constraints and therefore it is very similar to the problem solved using Decoder 3. The VRP is a pure vehicle routing version in which the packing constraints are transformed in one-dimensional capacity constraints, so they provide a reference for the impact of the packing constraints. Although a direct comparison is not possible, the results indicate that the algorithms developed for solving the static problem produce competitive solutions, requiring very few vehicles. The comparison with the VRP results shows the strong effect of the packing constraints being considered in this problem.

Table 6: Results of MP-BRKGA-IPR with Local Search of Ceschia instances

	Decoder 1		**Decoder 2**		**Decoder 3**		3L-CVRP	CLP	VRP
Instance	*Dist*	*Vehic*	*Dist*	*Vehic*	*Dist*	*Vehic*	*Vehic*	*Vehic*	*Vehic*
SD-CSS1	4773	4	4773	4	4921	4	5	4	3
SD-CSS2	9152	11	9152	11	9360	10	13	13	8
SD-CSS4	10675	11	10443	10	10328	11	12	11	7
SD-CSS5	6100	2	14749	2	5869	2	12	2	2
SD-CSS6	13412	14	13104	14	13327	14	32	19	11
SD-CSS7	9452	9	9009	9	8739	8	10	10	7
SD-CSS8	16595	20	17408	19	17094	19	36	21	15
SD-CSS9	14729	16	14145	16	15177	16	23	17	12
SD-CSS10	9209	7	9198	7	9187	7	18	7	5
SD-CSS11	12008	9	11440	8	11342	8	13	9	6
SD-CSS12	35090	41	33572	38	33499	37	48	37	28
SD-CSS13	20733	19	19145	18	17494	17	31	19	13
Total	**161928**	**163**	**166138**	**156**	**156337**	**153**	**253**	**169**	**117**

6.3. Results for the dynamic problem

In this second part of the study, we consider 5 different values for the percentage of disruption *pod* (0.05, 0.1, 0.15, 0.2, 0.25), for each of the 196 instances solved in the static case. As described above, each instance is divided into two parts: one with the initial information, that is solved as a static problem; and the other with the disruptions, which are handled one by one as they arise. If the information from these two parts is combined, the result is the original instance, which could be seen as the case with $pod = 0$ or the case with total information at the beginning of the planning horizon. Figure 5 represents the type of dynamic problem solved and resulting plans.

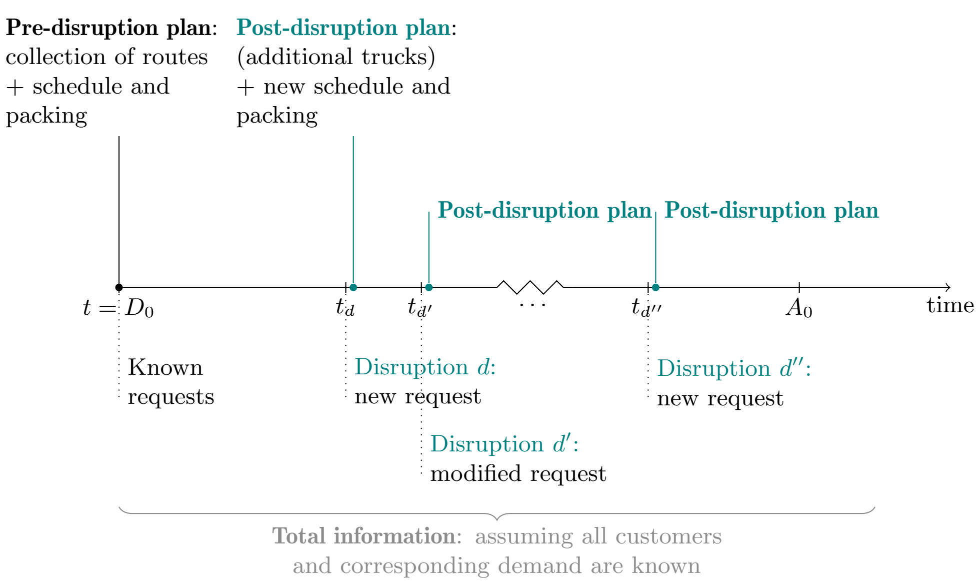


Figure 5: First-mile logistics dynamic problem: representation of the decisions made at the start (before the departure from the depot at D_0), the decisions made after each disruption (until the arrival at the depot at A_0), and of the total information case.

Table 7 shows the results of the dynamic problem using the first strategy, in which we can add as many

Table 7: Solving the dynamic problems adding as many new trucks as needed for the MO1 and MO2 instances

	Static initial problem			**Dynamic problem**			**Total information**	
pod	*Dist*	*Vehic*	*%Vol*	*Dist*	*Vehic*	*%Vol*	*Dist*	*Vehic*
0.05	236052	925	61.2%	318757	1128	53.6%	262795	1005
0.10	223615	882	60.1%	332768	1147	50.7%	262795	1005
0.15	218465	819	59.3%	358559	1317	43.8%	262795	1005
0.20	207298	780	59.4%	381957	1432	40.0%	262795	1005
0.25	187321	706	54.7%	418358	1582	32.8%	262795	1005
Total	**1072751**	**4112**	**58.9%**	**1810399**	**6606**	**44.2%**	**1313975**	**5025**

trucks as needed to cover all the disruptions. The table shows the results for MO1 and MO2 instances, grouped by *pod*, so that each row contains the aggregated results of 184 instances. The table is divided into three parts. The *Static initial problem* part contains the total distance (*Dist*), the total number of vehicles (*Vehic*), and the average percentage vehicle volume occupancy (*%Vol*) for the best solutions of the initial problem. The *Dynamic problem* part contains the information corresponding to the solution of the dynamic problem at the end of the time horizon, and the *Total information* part the solutions obtained if all the information is known from the beginning. Larger values of *pod* mean that larger parts of the information are not previously known, so the initial instances to be solved are smaller, as reflected in the comparison of the second and fifth and the third and sixth columns respectively. Comparing the number of vehicles (*Vehic*) in the initial and dynamic problems, we can observe the effect of increased dynamicity on the number of extra vehicles required to cover all disruptions. A $pod = 0.05$ for both new and modified requests, requires on average 22% more vehicles, and the percentage increases to 30% for $pod = 0.10$, 62% for $pod = 0.15$, 84% for $pod = 0.20$, or 124% for $pod = 0.25$. The comparison of *%Vol* indicates that adding new vehicles to cover the disruptions decreases the average vehicle occupancy, since just one new request can trigger the need for a new vehicle. Comparing *Vehic* in the dynamic and total information columns allows us to assess the value of the perfect information, i.e., how much better the solution would have been if all the dynamically arising information had been known in advance. In this case, having the total information would have reduced the number of trucks by 12%, 14%, 31%, 42%, or 57%, depending on the *pod*.

Furthermore, similar to the analysis performed in Table 5 about MO2 instances, when aggregating the instances based on the cargo heterogeneity (i.e., the BR instance class), the results suggest that cargo heterogeneity positively influences the ability to respond to disruptions, especially for lower percentages of disruption. Figure 6 shows the number of instances in each BR class where no additional trucks besides those initially planned were required to accommodate all disruption requests (total information). It can be observed that for lower *pod* values, it is more frequent for instances with higher box heterogeneity (i.e., higher BR class) to have more robust initial plans that can accommodate disruption without additional vehicles. Consequently, this might indicate that first-mile logistic providers who frequently deal with more homogeneous cargo are more susceptible to disruptions and could benefit the most from efficient methods to tackle them.

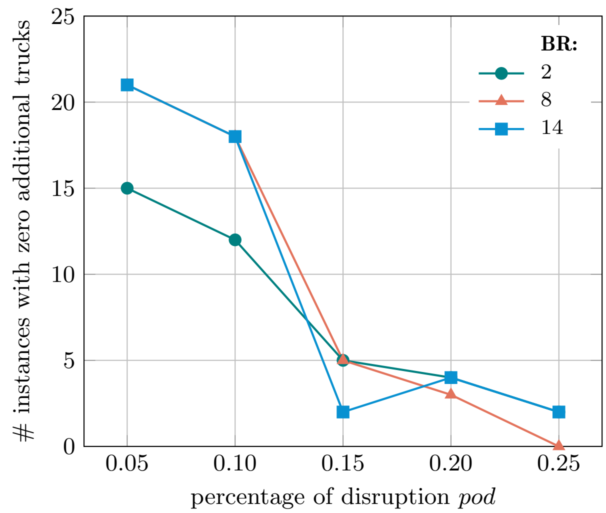


Figure 6: Number of instances (among the 46) in each BR class where no additional trucks were required besides those initially planned to accommodate all disruption requests (total information).

Table 8: Solving the dynamic problems without new trucks for the MO1 and MO2 instances

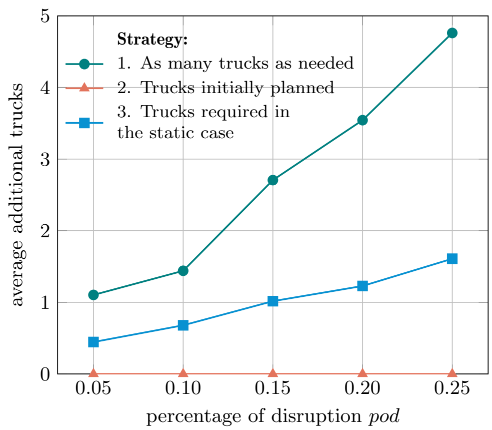
	Static initial problem			**Dynamic problem**			
pod	*Dist*	*Vehic*	*%Vol*	*Dist*	*%Vol*	*%NC*	*%MC*
0.05	235867	927	61.1%	275949	63.7%	41%	36%
0.10	224227	882	60.2%	255063	63.7%	49%	32%
0.15	218417	819	59.3%	248040	63.1%	58%	48%
0.20	207967	776	59.7%	243381	63.9%	62%	53%
0.25	187250	711	54.2%	231261	60.1%	66%	58%
Total	**1073728**	**4115**	**58.9%**	**1253694**	**62.9%**	**55%**	**45%**

Table 8 shows the results of the dynamic problem using the second strategy, in which no more vehicles can be added, so disruptions must be included in the existing routes or rejected. The table is similar to the previous one, although the *Total information* part would be the same as in Table 7 and is therefore not repeated. The *Dynamic problem* part does not show the number of vehicles, because in this strategy it is constant, but adds two new columns. *%NC* and *%MC* are the percentages of new and modified requests that are rejected. It can be seen that large percentages of the disruptions cannot be covered in this strategy, ranging from 41% of new customers for $pod = 0.05$ to 66% for $pod = 0.25$. Covering modified requests is somewhat easier, with rejection percentages ranging from 36% to 58%, as they sometimes do not have a large impact on the original demands. If a first-mile logistics provider does not have access to efficient solution methods to tackle disruptions, such as the one proposed in this paper, its alternatives range from rejecting all disruptive requests, which significantly hinders service quality and customer relations, to employing an additional truck for each disruptive request, which is costly and difficult to sustain in the long-term. A more balanced, albeit naive, approach would be to accept only the requests that can obviously be accommodated in the initial plan. This second strategy, where the vehicles are limited to those of the initial plan, can serve as a proxy for such a balanced approach. Since efficient solution methods are applied to generate post-disruption plans in this strategy, we can assume that the naive approach cannot reduce the percentage of rejected requests presented in Table 8. Thus, the results suggest that the level of rejected requests following a naive approach would be significantly high, demonstrating the value of applying methods tailored to tackle disruptions, as will be seen with the third and final strategy.

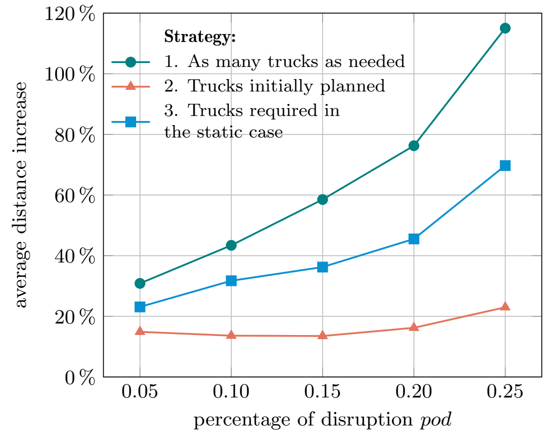
Table 9: Solving the dynamic problems with limited number of new trucks for the MO1 and MO2 instances

	Static initial problem			**Dynamic problem**			
pod	*Dist*	*Vehic*	*%Vol*	*Dist*	*%Vol*	*%NC*	*%MC*
0.05	236826	923	61.5%	298972	59.8%	26%	14%
0.10	224117	880	60.3%	302877	58.2%	19%	9%
0.15	218227	818	59.4%	303951	56.8%	36%	11%
0.20	207654	779	59.5%	309151	56.3%	41%	9%
0.25	187089	709	54.4%	323725	50.9%	41%	10%
Total	**1073913**	**4109**	**59.0%**	**1538676**	**56.4%**	**33%**	**11%**

Table 9 shows the results of the dynamic problem using the third strategy, which can be seen as a mixed strategy in which the number of vehicles can be increased only up to a given value (in this case, the number of vehicles of the original problem). Therefore, it is possible that some disruptions cannot be covered. The table has the same structure as the previous one. The number of total vehicles used always matches the number of vehicles available, shown in the last column in Table 7. The possibility of increasing the number of vehicles reduces the percentages of new and modified requests rejected. In fact, modified requests are nearly always accepted with this strategy, since they are easier to accommodate as they may not require a change in the route. Figure 7 compares the average costs of accepting disruptive requests (measured by the number of additional trucks required and, at a second level, by the additional distance traveled) for the three strategies



(a) Number of additional trucks



(b) Distance traveled

Figure 7: Average increase of the cost of accepting disruptions for the three strategies proposed, measured in terms of additional trucks and distance traveled

proposed. Allowing a limited number of additional trucks appears to avoid a steep increase in the costs of accepting requests in the face of higher disruption levels, while providing significant responsiveness to them, as discussed.

Table 10: Results of dynamic problems with unlimited number of new trucks for the Ceschia instances

	Static initial problem			**Dynamic problem**			**Total information**	
pod	*Dist*	*Vehic*	*%Vol*	*Dist*	*Vehic*	*%Vol*	*Dist*	*Vehic*
0.05	153457	153	60.1%	183278	185	52.0%	161927	163
0.10	144842	143	60.5%	203596	206	46.2%	161927	163
0.15	138470	136	59.6%	221695	229	42.6%	161927	163
0.20	129906	128	59.2%	229855	231	42.5%	161927	163
0.25	124616	114	60.8%	235708	241	39.4%	161927	163
Total	**691291**	**674**	**60.0%**	**1074132**	**1092**	**44.5%**	**809635**	**815**

Tables 10, 11, and 12 contain the results for the Ceschia instances. Although these instances are quite different from MO1 and MO2, the same trends can be observed. Concerning the first strategy, the results in Table 10 indicate that the percentages of additional vehicles are 21% for $pod = 0.05$, 44% for $pod = 0.10$, 68% for $pod = 0.15$, 80% for $pod = 0.20$, or 111% for $pod = 0.25$, very similar to those obtained from Table 7. The percentages of rejected requests in the second strategy and third strategies, Tables 11 and 12, are similar to those in Tables 8 and 9, although slightly larger, possibly due to the larger size of the instances.

As discussed before in Table 3, these instances are highly variable in terms of number of customers, box types and total number of boxes, allowing to study the impact of these instance characteristics in the results. Table 13 shows the percent increase in number of trucks required to accept all disruption requests compared to the trucks resulting from total information, which is the main source of cost to tackle disruptions, for the different values of *pod*. It is observed that it varies significantly across instances. The results suggest that the main driver of this cost increase is the total number of boxes (NB), as opposed to the number of customers

Table 11: Results of dynamic problems without new trucks for the Ceschia instances

	Static initial problem			**Dynamic problem**			
pod	*Dist*	*Vehic*	*%Vol*	*Dist*	*%Vol*	*%NC*	*%MC*
0.05	153457	153	60.1%	163803	61.2%	62.2%	46.5%
0.10	144842	143	60.5%	163938	62.1%	62.8%	64.0%
0.15	138470	136	59.6%	170152	61.7%	73.9%	61.6%
0.20	129906	128	59.2%	157607	61.6%	76.4%	50.8%
0.25	124616	114	60.8%	158187	63.5%	86.0%	55.5%
Total	**691291**	**674**	**60.0%**	**813687**	**62.0%**	**72.3%**	**55.7%**

Table 12: Results of dynamic problems with limited number of new trucks for the Ceschia instances

	Static initial problem			**Dynamic problem**			
pod	*Dist*	*Vehic*	*%Vol*	*Dist*	*%Vol*	*%NC*	*%MC*
0.05	153457	153	60.1%	174546	59.8%	48.3%	20.6%
0.10	144842	143	60.5%	187112	58.5%	43.7%	19.6%
0.15	138470	136	59.6%	203115	58.1%	41.8%	13.5%
0.20	129906	128	59.2%	198135	57.3%	37.1%	13.4%
0.25	124616	114	60.8%	204958	55.9%	35.8%	13.1%
Total	**691291**	**674**	**60.0%**	**967866**	**57.9%**	**41.3%**	**16.0%**

(C) and box types (BT), which seem to have little effect. One would expect that the number of customers would not impact this measure, since it is relative to the initial number of trucks. However, it is interesting to verify that cargo heterogeneity in these instances seems to have little impact on the magnitude of the results unlike the total number of boxes. These results suggest that first-mile logistic providers handling a large number of boxes, even if for fewer customers, could benefit the most from efficient methods of responding to disruptions.

6.4. Managerial insights

Overall, the results from this work allow to derive and summarize the following key managerial insights:

- Disruptions are decisive in first-mile logistics. If all new information were available before the planning phase, the number of trucks required could be decreased in an amount that ranges from 12% to 57% depending on the probability of disruption, which shows the high impact of disruptions in this context.
- Factors that increase susceptibility to disruptions and need for reactive tools:
  - *Homogeneous cargo.* Cargo heterogeneity makes it easier to accommodate and respond to disruptions. First-mile logistic providers who frequently deal with more homogeneous cargo are more susceptible to disruptions and could benefit the most from efficient methods to tackle them. This is a counter-intuitive insight as, for the deterministic problem, problems with homogeneous cargo are easier to deal with.
  - *Large number of boxes.* First-mile logistic providers handling a large number of boxes, even if for fewer customers, could benefit the most from efficient methods of responding to disruptions, as

Table 13: Percentages of increase in number of trucks required to accept all disruption compared to the trucks resulting from total information for the different values of *pod* (percentage of disruption)

				pod				
Instance	C	BT	NB	0.05	0.10	0.15	0.20	0.25
SD-CSS1	11	36	254	25%	25%	25%	25%	25%
SD-CSS2	25	15	350	9%	9%	9%	9%	45%
SD-CSS4	37	13	312	9%	18%	36%	36%	36%
SD-CSS5	41	47	7035	150%	50%	200%	300%	150%
SD-CSS6	43	97	8060	7%	21%	29%	21%	50%
SD-CSS7	45	14	284	11%	33%	56%	33%	44%
SD-CSS8	48	70	3275	0%	10%	5%	10%	10%
SD-CSS9	56	45	1725	6%	25%	38%	44%	38%
SD-CSS10	60	29	1840	29%	57%	71%	43%	71%
SD-CSS11	92	34	3790	44%	78%	122%	122%	89%
SD-CSS12	129	10	745	7%	12%	24%	27%	29%
SD-CSS13	129	63	2880	21%	53%	74%	84%	111%

the number of trucks required to accept all disruptions seems to be more sensitive to the total number of boxes to collect than to the number of customers.

- A flexible fleet capacity is key to accommodate and efficiently react to disruption. If a first-mile logistic provider is working at the limit of its fleet capacity, it can only respond to disruptions that can be easily accommodated in the initial plan, and there are large percentages of the disruptions that cannot be covered (41% to 66% of new customers rejected, and 36% to 58% of modified requests). Having a strategy to quickly access more fleet capacity (such as leasing or renting pre-accorded plans) is key. Even if the number of vehicles can be increased only up to a given value, this strategy appears to avoid a steep increase in the costs of accepting requests in the face of higher disruption levels while providing a significant responsiveness level to them.

- Efficient and balanced methods tailored to cope with disruption bring significant value to the first-mile logistics problem. The results of the proposed approach quantify the advantages of these methods, especially considering the number of trucks required for good service quality. Immediate on-the-fly responses, such as allowing only clearly feasible requests, lead to substantial levels of rejected requests. Also, allowing a limited number of additional trucks and generating efficient post-disruption plans with our proposed method avoids a sharp increase in the costs of accepting requests in the face of higher disruption levels while providing significant positive responsiveness to requests.

- Overall, regarding strategies to recover from disruptions, a balanced approach as described above, where the company allows for a limited increase in the number of vehicles and a fast and efficient re-planning of the routes using the solution method proposed in this work, seems to provide a good balance between good customer service levels and lower logistic costs.

7. Conclusions

In this work, in the context of first-mile parcel pickup, a vehicle routing problem with packing constraints under disruption is addressed. In particular, we consider unpredictable events related to the arrival of new customers and modifications in the number of boxes to be picked up at a customer's location. To tackle this problem we develop a BRKGA plus a local search algorithm for the static version of the vehicle routing problem with packing constraints, in which solutions are encoded in a two-segment chromosome, the first part containing the sequence by which the customers are loaded into the vehicles, and the second part representing the order by which the vehicles are used, as we are considering an heterogeneous fleet. The actual packing of the cargo is obtained during the decoding phase of the algorithm, where an order-based packing algorithm is proposed. Actually, three different decoders are presented, considering different logistics constraints regarding the movement of previously loaded cargo, in order to accommodate the upcoming cargo of each customer. Dealing with the dynamic counterpart of this problem requires disruption management strategies and algorithms. Starting from the solution generated for the static problem, disruption management algorithms are proposed for new requests (new customers to insert in the routes) and modified requests (changes in the demand of existing customers).

Extensive computational experiments were run, aiming to:

- Ascertain the quality of the BRKGA-based algorithm developed for the static problem, including the impact of alternative decoders;
- Assess the impact of disruptions in this problem, under different strategies to generate post-disruption plans, different percentages of disruption and different cargo and customer characteristics.

The computational results also led to insights relevant for first-mile logistics. As discussed above, we distinguished three strategies to generate post-disruption plans regarding the maximum number of trucks that can be used once the routes were initiated. The alternatives were to use as many trucks as needed to cover all the new and modified requests, use only the number of trucks initially planned, or use the number of trucks required by the solution of the original (static) instance. The results demonstrate the relevant role of disruptions in first-mile logistics and allow identifying cargo size and heterogeneity as determinant factors in obtaining good post-disruption plans. Additionally, fleet flexibility is shown to play a relevant role in responding to this kind of disruptions. Overall, this work shows that efficient and balanced methods that are tailored to cope with disruption, such as the ones proposed, are critical in first-mile logistics.

In future work, this approach can be brought even closer to real-world needs by considering other sources of disruption, especially those related to time. Time uncertainty can arise in several aspects of the problem, from customer loading time to route duration, both depending on the time of the day, but also on unexpected events. Time windows can be a major source of disruption. If a customer does not have the parcels ready to be loaded, the courier may wait for them, delaying the route. Congestion can also disrupt the initial plan if travel duration increases dramatically. Initial plans can be adapted to account for variations of deterministic information (Xiao and Konak, 2016), and then a disruption recovery algorithm should be developed to handle traffic jams due to accidents or other unpredictable issues (Chen et al., 2006). Another improvement of this work relies in the recourse actions undertaken to fix the disrupted solution, for instance by considering a split pickup for customers with new boxes to be collected.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Science, Innovation, and Universities, project RTI2018-094940-B-100, partially financed by FEDER funds, by the Junta de Comunidades de Castilla-La Mancha, project SBPLY/17/180501/000282, partially financed by FEDER funds, and by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, I.P., within project POCI-01-0145-FEDER-029609. The authors would like to thank the two anonymous reviewers for their valuable comments, which helped to improve this work.

References

Allahviranloo, M., J. Y. Chow, and W. W. Recker (2014). Selective vehicle routing problems under uncertainty without recourse. *Transportation Research Part E: Logistics and Transportation Review 62*, 68–88.

Alvarez, D., R. Alvarez-Valdes, and F. Parreño (2015). A GRASP algorithm for the container loading problem with multi-drop constraints. *Pesquisa Operacional 35*, 1–24.

Alvarez-Valdes, R., F. Parreño, and J. M. Tamarit (2013). A GRASP/Path Relinking algorithm for two- and three-dimensional multiple bin-size bin packing problems. *Computers and Operations Research 40*(12), 3081–3090.

Amaya, J., M. Delgado-Lindeman, J. Arellana, and J. Allen (2021). Urban freight logistics: What do citizens perceive? *Transportation Research Part E: Logistics and Transportation Review 152*(May), 102390.

Andrade, C. E., R. F. Toso, J. F. Gonçalves, and M. G. Resende (2021). The multi-parent biased random-key genetic algorithm with implicit path-relinking and its real-world applications. *European Journal of Operational Research 289*(1), 17–30.

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing 6*(2), 154–160.

Berbeglia, G., J.-F. Cordeau, and G. Laporte (2010). Dynamic pickup and delivery problems. *European Journal of Operational Research 202*(1), 8–15.

Bischoff, E. and M. Ratcliff (1995). Issues in the development of approaches to container loading. *Omega 23*(4), 377–390.

Bruni, M. E., F. Guerriero, and P. Beraldi (2014). Designing robust routes for demand-responsive transport systems. *Transportation Research Part E: Logistics and Transportation Review 70*(1), 1–16.

Ceschia, S., A. Schaerf, and T. Stützle (2013). Local search techniques for a routing-packing problem. *Computers & Industrial Engineering 66*(4), 1138–1149.

Chen, H.-K., C.-F. Hsueh, and M.-S. Chang (2006). The real-time time-dependent vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review 42*(5), 383–408.

Chen, S., H. Wang, and Q. Meng (2020). Solving the first-mile ridesharing problem using autonomous vehicles. *Computer-Aided Civil and Infrastructure Engineering 35*(1), 45–60.

Clausen, J., J. Larsen, A. Larsen, and J. Hansen (2001). Disruption management-operations research between planning and execution. *OR/MS Today 28*(5), 40–43.

Coffman, Jr, E. G., M. R. Garey, and D. S. Johnson (1983). Dynamic bin packing. *SIAM Journal on Computing 12*(2), 227–258.

Cordeau, J.-F., G. Laporte, and S. Ropke (2008). *Recent Models and Algorithms for One-to-One Pickup and Delivery Problems*, pp. 327–357. Boston, MA: Springer US.

Correcher, J. F. and R. Alvarez-Valdes (2017). A biased random-Key genetic algorithm for the time-invariant berth allocation and quay crane assignment problem. *Expert Systems with Applications 89*, 112–128.

Cui, J., M. Zhao, X. Li, M. Parsafard, and S. An (2016). Reliable design of an integrated supply chain with expedited shipments under disruption risks. *Transportation Research Part E: Logistics and Transportation Review 95*, 143–163.

Davies, A. and E. E. Bischoff (1999). Weight distribution considerations in container loading. *European Journal of Operational Research 114*(3), 509–527.

Eglese, R. and S. Zambirinis (2018). Disruption management in vehicle routing and scheduling for road freight transport: a review. *TOP 26*(1), 1–17.

Fan, T., Q. Pan, F. Pan, W. Zhou, and J. Chen (2020). Intelligent logistics integration of internal and external transportation with separation mode. *Transportation Research Part E: Logistics and Transportation Review 133*, 101806.

Fattahi, M., K. Govindan, and E. Keyvanshokooh (2017). Responsive and resilient supply chain network design under operational and disruption risks with delivery lead-time sensitive customers. *Transportation Research Part E: Logistics and Transportation Review 101*, 176–200.

Giménez-Palacios, I. (2021). First mile. *Mendeley Data, V1*. DOI: 10.17632/fffg27yx9m.1.

Glover, F. and M. Laguna (1997). *Tabu Search Background*, pp. 1–24. Boston, MA: Springer US.

Ho, S., W. Szeto, Y.-H. Kuo, J. Leung, M. Petering, and T. Tou (2018). A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological 111*, 395–421.

Hu, X. and L. Sun (2012). Knowledge-based modeling for disruption management in urban distribution. *Expert Systems with Applications 39*(1), 906–916.

Iori, M. and S. Martello (2010). Routing problems with loading constraints. *TOP 18*(1), 4–27.

Ivanov, D., A. Pavlov, A. Dolgui, D. Pavlov, and B. Sokolov (2016). Disruption-driven supply chain (re)-planning and performance impact assessment with consideration of pro-active and recovery policies. *Transportation Research Part E: Logistics and Transportation Review 90*, 7–24.

Jäämaa, L. and R. Kaipia (2022). The first mile problem in the circular economy supply chains–collecting recyclable textiles from consumers. *Waste Management 141*, 173–182.

Kåresdotter, E., J. Page, U. Mörtberg, H. Näsström, and Z. Kalantari (2022). First mile/last mile problems in smart and sustainable cities: A Case Study in Stockholm County. *Journal of Urban Technology*, 1–23.

Karlaftis, M. G., K. Kepaptsoglou, and E. Sambracos (2009). Containership routing with time deadlines and simultaneous deliveries and pick-ups. *Transportation Research Part E: Logistics and Transportation Review 45*(1), 210–221.

Koç, C., G. Laporte, and I. Tükenmez (2020). A review of vehicle routing with simultaneous pickup and delivery. *Computers and Operations Research 122*.

Kucukoglu, I., R. Dewil, and D. Cattrysse (2021). The electric vehicle routing problem and its variations: A literature review. *Computers & Industrial Engineering 161*, 107650.

Li, H., J. Chen, F. Wang, and M. Bai (2021). Ground-vehicle and unmanned-aerial-vehicle routing problems from two-echelon scheme perspective: A review. *European Journal of Operational Research 294*(3), 1078–1095.

Masmoudi, M. A., M. Hosny, E. Demir, K. N. Genikomsakis, and N. Cheikhrouhou (2018). The dial-a-ride problem with electric vehicles and battery swapping stations. *Transportation Research Part E: Logistics and Transportation Review 118*, 392–420.

Moghdani, R., K. Salimifard, E. Demir, and A. Benyettou (2021). The green vehicle routing problem: A systematic literature review. *Journal of Cleaner Production 279*, 123691.

Moura, A. and J. F. Oliveira (2009). An integrated approach to the vehicle routing and container loading problems. *OR Spectrum 31*, 775–800.

Ning, F., G. Jiang, S.-K. Lam, C. Ou, P. He, and Y. Sun (2021). Passenger-centric vehicle routing for first-mile transportation considering request uncertainty. *Information Sciences 570*, 241–261.

Ojeda Rios, B. H., E. C. Xavier, F. K. Miyazawa, P. Amorim, E. Curcio, and M. J. Santos (2021). Recent dynamic vehicle routing problems: A survey. *Computers & Industrial Engineering 160*, 107604.

Parajuli, A., O. Kuzgunkaya, and N. Vidyarthi (2021). The impact of congestion on protection decisions in supply networks under disruptions. *Transportation Research Part E: Logistics and Transportation Review 145*, 102166.

Parragh, S., K. Doerner, and R. Hartl (2008a). A survey on pickup and delivery problems. *Journal fur Betriebswirtschaft 58*(1), 21–51.

Parragh, S., K. Doerner, and R. Hartl (2008b). A survey on pickup and delivery problems: Part ii: Transportation between pickup and delivery locations. *Journal fur Betriebswirtschaft 58*(2), 81–117.

Pillac, V., M. Gendreau, C. Guéret, and A. Medaglia (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research 225*(1), 1–11.

Pinto, B. Q., C. C. Ribeiro, I. Rosseti, and T. F. Noronha (2020). A biased random-key genetic algorithm for routing and wavelength assignment under a sliding scheduled traffic model. *Journal of Global Optimization 77*(4), 949–973.

Pollaris, H., K. Braekers, A. Caris, G. K. Janssens, and S. Limbourg (2015). Vehicle routing problems with loading constraints: state-of-the-art and future directions. *OR Spectrum 37*(2), 297–330.

Psaraftis, H. N., M. Wen, and C. A. Kontovas (2016). Dynamic vehicle routing problems: Three decades and counting. *Networks 67*(1), 3–31.

Ramos, A. G., E. Silva, and J. F. Oliveira (2018). A new load balance methodology for container loading problem in road transportation. *European Journal of Operational Research 266*(3), 1140–1152.

Ruiz, E., V. Soto-Mendoza, A. E. Ruiz Barbosa, and R. Reyes (2019). Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm. *Computers and Industrial Engineering 133*, 207–219.

Seiden, S. S. (2001). *On the Online Bin Packing Problem*, pp. 237–248. Berlin, Heidelberg: Springer Berlin Heidelberg.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research 35*(2), 254–265.

Spliet, R., A. F. Gabor, and R. Dekker (2014). The vehicle rescheduling problem. *Computers & Operations Research 43*, 129–136.

Toth, P. and D. Vigo (2014). *Vehicle routing: problems, methods, and applications.* SIAM.

Wang, X. and H. Cao (2008). A dynamic vehicle routing problem with backhaul and time window. In *2008 IEEE International Conference on Service Operations and Logistics, and Informatics*, Volume 1, pp. 1256–1261.

Wang, X. and G. Q. Huang (2021). When and how to share first-mile parcel collection service. *European Journal of Operational Research 288*(1), 153–169.

Wang, X., J. Ruan, and Y. Shi (2012). A recovery model for combinational disruptions in logistics delivery: Considering the real-world participators. *International Journal of Production Economics 140*(1), 508–520.

Wang, X. and Z. Zhao (2021). Survey of the dynamic pickup and delivery problems. *Xitong Gongcheng Lilun yu Shijian/System Engineering Theory and Practice 41*(2), 319–331.

Xiao, Y. and A. Konak (2016). The heterogeneous green vehicle routing and scheduling problem with time-varying traffic congestion. *Transportation Research Part E: Logistics and Transportation Review 88*, 146–166.

Yu, G. and X. Qi (2004). *Disruption management: framework, models and applications.* World Scientific.

Zhang, J., F. Liu, J. Tang, and Y. Li (2019). The online integrated order picking and delivery considering Pickers' learning effects for an O2O community supermarket. *Transportation Research Part E: Logistics and Transportation Review 123*, 180–199.

Zhang, Z., A. Tafreshian, and N. Masoud (2020). Modular transit: Using autonomy and modularity to improve performance in public transportation. *Transportation Research Part E: Logistics and Transportation Review 141*, 102033.

Zhen, X., Y. Li, G. G. Cai, and D. Shi (2016). Transportation disruption risk management: Business interruption insurance and backup transportation. *Transportation Research Part E: Logistics and Transportation Review 90*, 51–68.

Appendix A. Formal description of the static pick-up problem

In this problem, there is a set of customers to be served, $\mathcal{C} = \{1, \ldots, C\}$, each one with a location represented as a node n_c in a network. The depot is located at node n_0, with opening hours $[a_0, b_0]$. The matrix T holds the time t_{ij} it takes to move from node i to j, $i, j \in \mathcal{C} \cup \{0\}$. Customers have a demand, i.e. a set of boxes $\mathcal{B}_c = \{1, \ldots, B_c\}$, $c \in \mathcal{C}$ to be picked up. The boxes $b \in \mathcal{B}_c$ are cuboids of which we know their dimensions and weight. They must be picked up within a time window associated with each customer, $[a_c, b_c]$, and each demand has an assigned service or picking time, s_c, which is the time needed for the driver to pick up and load all the boxes into the vehicle.

The logistics company has a fleet with different types of vehicles, $\mathcal{V} = \{1, \ldots, V\}$ and each type $v \in \mathcal{V}$ has k_v vehicles, being $\sum_{v \in \mathcal{V}} k_v = K$. Each vehicle type has a load volume defined by its dimensions $L_v \times W_v \times H_v$, a weight capacity M_v, and a daily cost c_v. The driver of each vehicle k has a defined shift $[\alpha_k, \beta_k]$ during which he/she must make customer visits.

The objective is to minimize both the number of vehicles used and the total distance to be travelled on all routes. We consider that saving on the number of vehicles used is preferable to reducing the distance travelled, so we will choose the solution with the least number of vehicles, and, in case of a tie, the one with the least distance travelled. To achieve this goal, the function f that defines the objective is given by the expression

$$f = d + \kappa \sum_{v \in \mathcal{V}} c_v s_v, \tag{A.1}$$

where d is the total distance, s_v the number of v type vehicles used in the solution, c_v the cost of the v type vehicle, and κ a sufficiently large constant. In our computational experiment, κ is set 1000 in all instances. In MO1 and MO2 instances, there is only one type of vehicle, so the associated cost is set to $c_1 = 1$. In the case of the Ceschia instances, for the instances with two different vehicle types, the lower capacity vehicle has a cost of $c_1 = 1$, whereas the higher capacity vehicle has a cost of $c_2 = \frac{V_2}{V_1}$.

This problem is known as the Capacitated Vehicle Routing Problem with pickups, time windows, and packing constraints. A solution to this problem is called a plan and consists of a collection of routes, each with an associated schedule and packing. A route k is defined as a sequence of nodes $\left(n_0, n_1^k, \ldots, n_i^k, \ldots, n_{I_k}^k, n_0\right)$ to be visited by vehicle k.

The schedule associated with this route k is the time information for each node n_i^k. This time information is composed of three time instants:

- Arrival time A_i^k at node n_i^k.
- Start time of the load at node n_i^k, $H_i^k = \max\{A_i^k, a_i\}$.
- Departure time D_i^k from node n_i^k, $D_i^k = H_i^k + s_i$.

When $i = 0$, we have D_0^k, the departure time at the start of the route, and A_0^k, the arrival time to the depot at the end of the route. All these variables depend on the route k, but when no confusion can arise, we will often omit the index k for the sake of simplicity. A schedule for a given route is shown in Figure A.8.

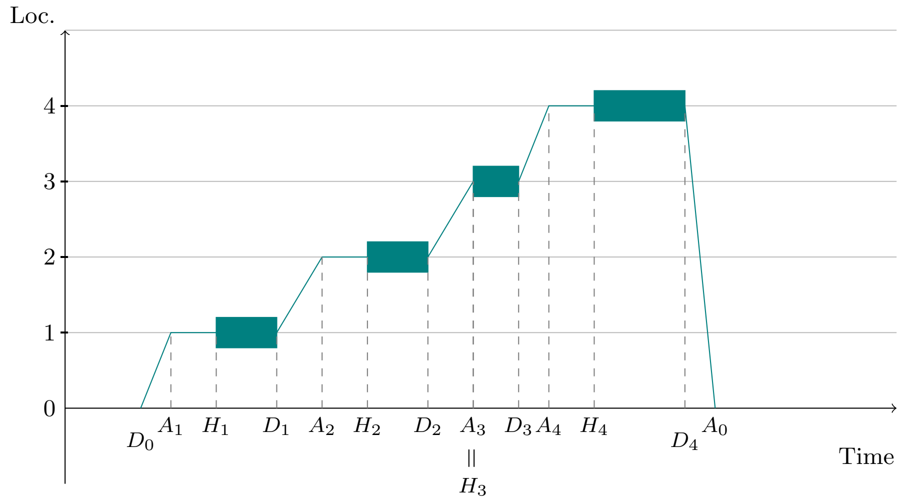


Figure A.8: Schedule of a route

Associated with each node of the route, the packing of the loaded boxes must be provided. For this purpose, the loading space inside the vehicle is described as a parallelepiped. Without loss of generality, the axes of the coordinate system are assumed to be placed so that the length L_v (resp. width W_v, height H_v) of the vehicle type v lies on the x-axis (resp. y-axis, z-axis) $\forall v \in \mathcal{V}$. The origin of this coordinate system is at the bottom left front corner of the vehicle's cargo space. A representation is given in Figure A.9. The position of a box b is described by the position of the front bottom left corner (x_b, y_b, z_b) and the position of the upper right rear corner (x'_b, y'_b, z'_b) of the box inside the vehicle.

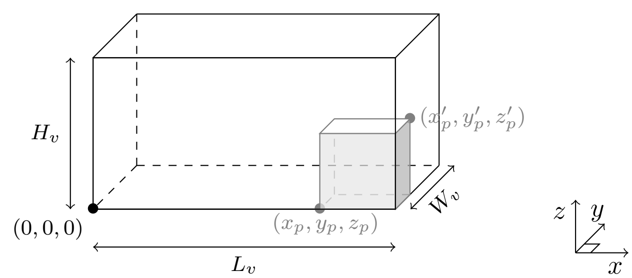


Figure A.9: Representation of the loading space inside a vehicle and description of the position of a box

The feasibility of a solution depends on the feasibility of the routes, and associated schedule and packing.

Route feasibility. A feasible route must satisfy:

(RF1) All boxes must be collected;

(RF2) Customers are visited exactly once;

(RF3) Each route starts and ends at the depot n_0.

Schedule feasibility. The feasibility of every route depends on the schedule and on the packing. A schedule is feasible if the following constraints are met:

(SF1) Time windows for each vehicle: the driver of each vehicle k has a time window $[\alpha_k, \beta_k]$ which implies that the departure time from the depot D_0^k cannot be earlier than time α_k and the arrival time A_0^k cannot exceed time β_k ($D_0^k \geq \alpha_k$ and $A_0^k \leq \beta_k$).

(SF2) Time windows for each customer: at each node n_i^k, the load start time must be in the time window of customer i, and the process must finish also within this time window ($H_i^k \geq a_i$ and ($H_i^k + s_i \leq b_i$).

(SF3) The load starts after arrival at node n_i^k, $A_i^k \leq H_i^k$, and ends at the time the load completes, $D_i^k = H_i^k + s_i$.

(SF4) Time consistency: the arrival time at node n_{i+1}^k is defined as the time the previous node of the route is left plus the time required to traverse the arc $(i, i+1)$: $D_i^k + t_{i,i+1} = A_{i+1}^k$.

Packing feasibility. A packing is feasible if it satisfies the following conditions:

(PF1) The set of boxes of a customer is assigned to exactly one vehicle.

(PF2) Each box lies within the boundaries of the vehicle.

(PF3) Boxes cannot overlap each other.

(PF4) The total weight of the boxes inside a vehicle cannot exceed the maximum capacity of the vehicle

(PF5) Orthogonality: every box must be loaded with its edges parallel to the vehicle boundaries.

(PF6) Vertical stability: boxes must be completely supported by other boxes or the vehicle floor.

(PF7) Rotation: each box has six possible rotations, but in some cases only some of them are allowed.

Appendix B. Genetic algorithm decoders description

Decoder 1: No movement

In this decoder, the boxes previously packed cannot be moved. To pack the boxes of the current customer, we use a constructive algorithm based on maximal spaces, adapted from Alvarez et al. (2015). For a given vehicle type v and a given customer c, the algorithm works as follows:

1. Initialization:

   Two sets are defined:

   $\mathcal{S}$ = the list of empty maximal spaces created when packing the boxes of previous customers into the vehicle. If the vehicle is empty, $\mathcal{S}$ is just the empty cargo space of the vehicle.

   $\mathcal{B}_c$ = the set of boxes to be packed, corresponding to customer c.

2. Choosing the maximal space from $\mathcal{S}$:

   From the list $\mathcal{S}$ the maximal space with the largest coordinate z is chosen. The reason behind this strategy is to stack boxes in piles.

3. Choosing the boxes to pack:

   Once a maximal space $S \in \mathcal{S}$ has been chosen, the remaining boxes of the current customer, $\mathcal{B}_c$, fitting into S, are considered for packing. If there are several boxes of the same type, the possibility of packing a group of boxes forming a column or layer is also taken into account and considered as another single box. The box that best fits in the maximal space is selected. The distance from each side of the box to each side of the maximal space is calculated and these distances are put in a vector in non-decreasing order. The box is chosen using the lexicographic order. If there is a box filling up the space, it is selected. If no such box exists, the box matching two dimensions of the space is selected. If it does not exist, the box matching one dimension, and if it is does not exist, the box whose minimal difference in dimension with respect to the dimension of the space is minimal is chosen.

   The selected box is packed in the corner of the maximal space with the shortest distance to the origin of coordinates.

   The set $\mathcal{B}_c$ is updated by removing the box or boxes just loaded. If $\mathcal{B}_c = \emptyset$ and $\mathcal{S} = \emptyset$, the procedure ends and the decoder will take the next vehicle type and the next customer in the sequence.

4. Updating the $\mathcal{S}$ list:

   Unless the box or group of boxes fits exactly in the space S, packing it produces new empty maximal spaces that will replace S in the list $\mathcal{S}$. Moreover, since the maximal spaces are not disjoint, the box or group of boxes that are packed can intersect with other maximal spaces that will have to be reduced. Once the new spaces have been added and some of the existing spaces have been modified, the list is checked and inclusions are removed. The maximal spaces that cannot accommodate any of the remaining boxes, $\mathcal{B}_c$, are also removed from $\mathcal{S}$. If $\mathcal{B}_c \neq \emptyset$ and $\mathcal{S} \neq \emptyset$, there are still some boxes to pack and some empty spaces, so the procedure goes back to Step 2. However, if $\mathcal{B}_c \neq \emptyset$ but $\mathcal{S} = \emptyset$, the boxes of customer c do not fit into the vehicle; therefore, the state of the vehicle before starting to load the boxes from customer c is retrieved and the algorithm goes back to the decoder with the same customer c and the next vehicle in the sequence. Finally, if $\mathcal{B}_c = \emptyset$ and $\mathcal{S} \neq \emptyset$, the algorithm goes to Step 5 to update the list $\mathcal{S}$ for the next customer in the sequence.

5. Updating the $\mathcal{S}$ list for a new customer:

   The maximal spaces that are not completely visible from the back of the vehicle need to be removed from the list, because they cannot be accessed without moving some of the already packed boxes. A space is said to be visible if it can be completely seen from the back of the vehicle. If $\mathcal{S} \neq \emptyset$, the procedure returns to the decoder with the same vehicle and the next customer in the sequence. If $\mathcal{S} = \emptyset$, the next vehicle in the sequence is taken.

Decoder 2: Some movements

In this decoder, the movement of some boxes is allowed: the driver loads the boxes of the current customer and can also move the reachable boxes, that is, the boxes already loaded that are accessible without moving any other box.

The difference between this decoder and decoder 1 is that in Step 5 we remove the reachable boxes, forming set R, and update the maximal spaces accordingly, and in Step 1 the set of boxes to be loaded will be $\mathcal{B}_c \cup R$.

Decoder 3: All movements

In this decoder, the movement of all boxes is allowed; so all loaded boxes can be moved.

The difference between this decoder and Decoder 1 is that in Step 5 all the boxes loaded from previous customers are removed and the list $\mathcal{S}$ will just be the empty space cargo, and in Step 1 the set of boxes to be loaded will include all boxes from previous customers plus the boxes from customer c.