Integrating autonomous delivery service into a passenger transportation system

Abood Mourad, Jakob Puchinger, Tom van Woensel

ARTICLE TEMPLATE

Integrating autonomous delivery service into a passenger transportation system

Abood Mourad[a,b], Jakob Puchinger[a,b] and Tom Van Woensel[c]

[a]Laboratoire Genie Industriel, CentraleSupélec, Université Paris-Saclay, Gif-sur-Yvette, France;
[b]Institut de Recherche Technologique SystemX, Palaiseau, France;
[c]School of Industrial Engineering, Eindhoven University of Technology, Eindhoven, 5600 MB, The Netherlands

ARTICLE HISTORY
Compiled May 22, 2019

Abstract
The increasing demand for goods, especially in urban areas, together with the technological advances are creating both opportunities and challenges for planning urban freight systems. One of these promising opportunities is to use the underused assets in people-based systems to transport goods. In this paper, we consider an integrated system in which a set of freight requests needs to be delivered using a fleet of grounded, and autonomous, pickup and delivery (PD) robots where a public transportation service (referred to as scheduled line (SL)) can be used as part of PD robot's journey. Passengers and PD robots (carrying freight) share the available capacity on SLs where passengers are prioritized, and their transport demand is stochastic. Thus, the number of available places for PD robots is only revealed upon shuttle arrival to the corresponding SL station. We first formulate this problem as a Pickup and Delivery Problem with Time Windows and Scheduled Lines (PDPTW-SL). We then introduce a sample average approximation (SAA) method along with an Adaptive Large Neighborhood Search (ALNS) algorithm for solving the stochastic optimization problem. Finally, we present an extensive computational study, analyze its results, and give some directions for future research.

KEYWORDS
Pickup and delivery; Scheduled lines; Stochastic demand; ALNS; Sample average.

1. Introduction

The demand for freight transportation basically results from the need of transporting goods from producers to consumers who are geographically apart. In general, this transportation process consists of picking up products at their producer locations (*pre-haul*), transporting them (*long-haul*), and delivering them to final consumers (*end-haul*) at the right time and place and at low costs (Steadieseifi et al. (2014)). The increasing demand for goods in urban areas, together with the emerging information and technological advances are creating both opportunities and challenges for planning urban freight systems (Savelsbergh & Van Woensel (2016)). One of these promising opportunities is to use the low-utilized people transport systems (e.g. off-peak hours

CONTACT A. Mourad. Email: abood.mourad@centralesupelec.fr

of urban rail, buses or private-car trips) to also transport goods. A successful integration of these transportation streams can enhance the service quality of their existing transportation systems as well as their system-wide gains. For example, spare capacity in public transport systems can be used for retail store replenishment (Trentini et al. (2015)) or a taxi can deliver freight when transporting a passenger or during idle time (Li et al. (2014)).

In such combined systems, we have a set of passengers and parcels that need to be transported simultaneously from their origins to their destinations. This combination can lead to minimizing vehicle-miles traveled, traffic congestion and pollution levels in urban areas. It can also yield some travel cost reductions for passengers. However, such a system must ensure that the transportation of goods does not disturb passenger trips. In other words, a passenger would accept only small deviations and short extra times when transporting some parcels in the same trip (i.e. trip times that exceed passenger usual route times significantly might not be acceptable).

In this paper, we consider an integrated system in which a set of freight requests needs to be transported from their origins to their destinations. We use a fleet of grounded and autonomous pickup and delivery (PD) robots where a public transportation service (e.g. a set of shuttles, referred to as *scheduled line (SL)*) can be used as part of PD robot's journey [1]. Most research considers that passengers and goods are transported separately. However, we consider that passengers and PD robots (carrying goods) share the same capacity. This implies that a freight request can be served in one of two ways: (1) a direct delivery (where only a PD robot is used) or (2) transferred through SLs (where both PD robots and SLs are used). Therefore, a parcel might be picked up at its origin location by a PD robot, transported through the scheduled line with passengers, and delivered to its final destination by the PD robot. In order to guarantee an acceptable service quality for passengers, they are assumed to have a higher priority to use SL service. In other words, PD robots are only able to use SLs when there are free places available (i.e. not used by any passengers).

A similar system is considered by Ghilas et al. (2016c) where a scheduled line service is used along with a fleet of heterogeneous vehicles to serve a set of freight requests. In their system, the exact quantities demanded by each customer are only learned upon vehicles' arrival at the corresponding pickup locations. Unlike their problem settings, we consider that freight quantities are known in advance. In addition, we consider that passengers demand for transportation is only learned upon the shuttles' arrival to each SL station. Since passengers and PD robots share the same capacity on SLs, the number of available places for PD robots at each SL departure is thus stochastic. Depending on the actual passengers transport demand, there are two possible violation outcomes: *(i)* the PD robot is not able to take the next SL departure due to the high passenger demand at the corresponding station, and *(ii)* the PD robot needs to get off the SL at an intermediate station, where passengers demand is high, in order to give its place to a passenger. When these route failures occur, a number of recourse actions are needed in order to recover feasibility. Applying these recourse actions might lead to extra handling and transportation costs compared to their original routes.

The key contributions of this paper are as follows. First, we model the proposed pickup and delivery problem as a two-stage stochastic problem. The first stage consists of defining routes for PD robots carrying freight requests. These routes are evaluated over a set of scenarios and their associated recourse costs are calculated in the sec-

[1] This integrated system was inspired from Toyota new e-Palette concept in which small delivery robots travel with passengers in autonomous shuttles moving around in a city.

ond stage. The overall objective is to minimize the overall transportation costs (i.e. the sum of the first-stage routing costs and the second-stage recourse costs). Second, we propose a sample average approximation (SAA) method along with an Adaptive Large Neighborhood Search (ALNS) algorithm for solving the stochastic optimization problem. Finally, we provide a computational study to quantify the impact of passengers demand realization on such combined systems. This is achieved by comparing the solutions obtained when deterministic and stochastic versions of the problem are solved. While the potential benefits of integrating parcel deliveries to SL service were extensively studied in Ghilas et al. (2016a), in this paper we aim at studying the impacts of stochastic passenger demands on this system with different SL frequencies and capacities.

This paper is organized as follows. In section 2, we provide an overview of related literature. In section 3, we describe the problem, provide a mathematical formulation for it, and introduce an algorithm to evaluate its solutions and calculate their recourse costs. The proposed solution method is detailed in section 4. In section 5, we present the computational study and analyze its results. Finally, in section 6, the key findings are summarized and directions for future research are suggested.

2. Background

An increasing amount of research is being directed recently towards studying and developing new transportation systems that integrate passenger and freight flows. These systems can be classified into *single-tiered* and *two-tiered* systems. In single-tiered systems, a set of vehicles transport passengers and goods to their destinations while taking into account some considerations (e.g. request time windows, vehicle capacity, etc.). On the other hand, passenger and freight flows are combined in two-tiered systems thanks to the contribution of a first-tier (e.g. a public mass-transport line), and a second-tier (e.g. a fleet of vehicles) that performs the last-mile deliveries to customers (see Mourad et al. (2019) for a recent review).

Regarding single-tiered systems, Li et al. (2014) introduced the *Share-a-Ride Problem* (SARP) in which passenger and freight requests are transported using a fleet of taxis driving around in a city. As passenger requests are given a higher priority, some parcels are delivered during taxi trips in case this delivery does not affect the passengers significantly. For solving the SARP, a MILP formulation, that extends the classical Dial-a-Ride problem, along with an Adaptive Large Neighborhood Search (ALNS) method were proposed (see also Li et al. (2016a)). Their results demonstrated the benefits of such combination in terms of transportation costs and traveled distances. These benefits were observed by comparing results to those where passenger and freight requests are served separately. In another study, Arslan et al. (2016) presented an event-based rolling horizon framework that dynamically assigns parcel deliveries to self-employed drivers who are willing to earn some extra money by making deliveries on their way to home or work. In addition, the authors proposed a heuristic recursive algorithm for solving the routing subproblem. Their results demonstrated that this integrated delivery can potentially reduce last-mile delivery costs as well as the system-wide vehicle miles. Archetti et al. (2016) considered a similar single-tiered model where a set of occasional drivers is used to supplement the service provided by delivery vehicles and dedicated drivers. Occasional drivers are those willing to make a single delivery using their own vehicle. The authors modeled this problem as a *Vehicle Routing Problem (VRP) with occasional drivers* and proposed a heuristic approach

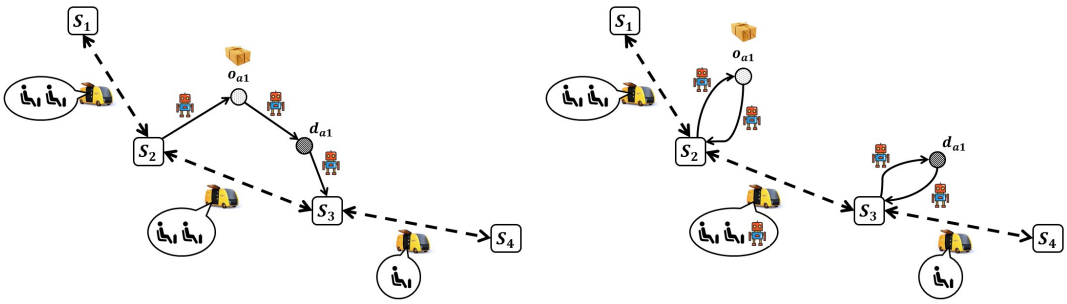
that uses variable neighborhood and tabu search strategies for solving it. Their results showed that introducing more occasional drivers to the system can decrease the total transportation cost and the number of dedicated drivers required. Moreover, Wang et al. (2016) presented a single-tiered model where last-mile deliveries are performed by a large-pool, a *crowd*, of citizen workers. The proposed model was formulated as a network min-cost flow problem and solved using an iterative pruning technique. Furthermore, Dayarian & Savelsbergh (2017) suggested that potential customers can express their interest to participate in making deliveries on their way home. The authors proposed a tabu search heuristic method for generating vehicle routes.

On the other hand, some recent studies have focused on studying two-tiered systems. Fatnassi et al. (2015) introduced an integrated system where passengers and goods are transported to intermediate points using a first-tier (train, bus or truck line), and then delivered using a fleet of electric vehicles (second-tier). The authors proposed a forward periodic-optimization approach which showed that the proposed system can achieve a potential gain in terms of service time and energy consumption. Another study, by Masson et al. (2017), considered a combined system that uses the available capacity in a passenger bus line to transport parcels to specific bus stations where a fleet of low-emission freighters delivers them to final customers. The paper formulated the system as a *Vehicle Routing Problem with transfers* and proposed an ALNS-based heuristic to solve it (see also Trentini et al. (2015)). Similarly, Ghilas et al. (2016a) introduced a two-tiered system where parcels are delivered by a fleet of vehicles such that a part of the delivery process is carried out on a scheduled line of public transport. The paper modeled this integrated system as *Pickup and Delivery Problem with Time Windows and Scheduled Lines* (PDPTW-SL) and introduced an ALNS-based algorithm for solving it. Their results showed that an average cost savings of 10% can be achieved thanks to the use of the scheduled line compared to a pure-freight delivery system. Moreover, Kafle et al. (2017) suggested that parcels can be transported to intermediate points using a set of carrier trucks, and then delivered by a set of potential cyclists and pedestrians who are living in the same neighborhood. The authors proposed a tabu search algorithm for solving the asoociated optimization problem. Their results demonstrated that the use of potential cyclists and pedestrians can reduce the operational costs by 9.25% compared to a truck-based delivery system.

As for studying uncertainty in such combined systems, Li et al. (2016b) extended the SARP, introduced earlier, by considering two stochastic variants. The first variant considered the travel times to be stochastic while the second considered stochastic delivery locations. For solving both variants, a two-stage stochastic programming model with recourse is used with the ALNS heuristic and a scenario generator. Through an extensive experimental study on both stochastic models, the paper concluded that the stochastic travel times have a more noticeable effect on the SARP than the stochastic delivery locations. In addition, Ghilas et al. (2016c) extended their two-tiered model by considering stochastic demand quantities of freight requests which are only revealed upon the vehicle's arrival to their pickup locations. A scenario-based sample average approximation approach was introduced in order to consider this uncertainty. After reviewing the related literature, we provide a detailed description of the considered problem along with the method used to solve it in the following sections.

3. Problem Description

Consider a set of shuttles that operate on a scheduled line (SL) service in both directions. This service consists of a set of physical transfer nodes (i.e. stations) $\mathcal{S}$, where passengers take shuttles as part of their trip to their final destinations, and a set of physical scheduled lines $\mathcal{E}$ linking different transfer nodes. Between every pair of transfer nodes $i, j \in \mathcal{S}$, there are two scheduled lines with opposite directions $(i, j), (j, i) \in \mathcal{E}$. Shuttles move through the scheduled line in fixed routes. Every shuttle moving through scheduled line (i, j) has a capacity Q_{ij}, indicating the number of available places, and a schedule $\mathcal{K}^{ij}$, indicating its departure times at origin transfer node i (denoted by p^w_{ij}, e.g. the second departure from s_1 to s_2 is $p^1_{s_1,s_2} = 60$ time units). Moreover, shipping one unit of package on scheduled line (i, j) is associated with a cost η_{ij} per unit. In addition, a fleet of autonomous, pickup and delivery (PD) robots are located at transfer nodes. Each PD robot $v \in \mathcal{V}$ is assigned to a depot (i.e. transfer node) $o_r \in \mathcal{S}$ and has a capacity Q_v and a maximum service distance δ_v indicating the maximum distance it can go from a transfer node to a request pickup or destination location. Each PD robot is associated with a routing cost per time unit θ_v.



(a) Direct delivery - PD robot (b) Indirect delivery - PD robot and SL

Figure 1.: Request service modes: direct & indirect delivery

In addition, a set of freight requests need to be transported to their final destinations using the fleet of PD robots. Each request is associated with an origin $r \in \mathcal{P}$ and a destination $r + n \in \mathcal{D}$ (where $n = |\mathcal{P}|$ is the number of requests), indicating where it should be picked up and to where it should be delivered. In addition, request r is associated with two time windows, a pickup time window $[e_r, l_r]$ and a delivery time window $[e_{r+n}, l_{r+n}]$, and a demand quantity d_r. Pickup and delivery time windows indicate when the request should be picked up by a PD robot and when it should be delivered to its final destination. Depending on the availability of vacant places in SLs, PD robots carrying freights may travel with passengers between different transfer nodes. A freight, carried by a PD robot, can thus be transported by a shuttle between two transfer nodes as part of its journey.

Indeed, allowing passengers and PD robots to travel simultaneously aims at using the spare capacity in shuttles especially that loading (and unloading) these robots into shuttles at transfer points come with relatively short service times. As a result, delivering a request to its final destination can be done in either direct or indirect way (see Figure 1). In a **direct delivery**, a request is picked up by a PD robot at its origin and delivered directly to its final destination without the use of the scheduled

line (Figure 1a; request a_1 is picked up at its origin o_{a_1} by a PD robot coming from transfer node s_2, and delivered to its final destination d_{a_1} before the PD robot returns to transfer node s_3).

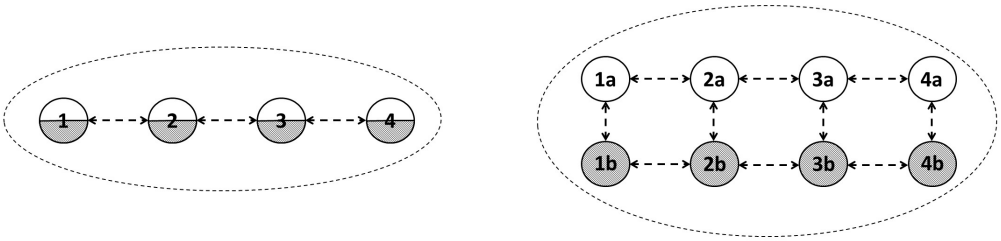
It is important to mention that a direct delivery is only feasible if the distance between the transfer node and request origin/destination, and between request origin and destination locations is less than the maximum distance the robot can travel. In Figure 1b, if the distance between o_{a_1} and d_{a_1} is greater than the robot maximum service distance, a direct delivery cannot be performed and the SL service must be used. On the other hand, in an **indirect delivery**, a request may be collected by one PD robot, transferred through the scheduled line and delivered afterwards to its final destination by the same PD robot (Figure 1b, request a_1 is picked up at its origin o_{a_1} by a PD robot, brought to transfer node S_2, transported through the scheduled line from s_2 to s_3 and finally delivered to its final destination d_{a_1} by the PD robot).

Since passengers and PD robots are using SLs simultaneously in indirect deliveries, we assume that a passenger or a PD robot needs one place in a shuttle while passengers have higher priority to be transported. We also assume that PD robots cannot take over more than a fixed number of places in each shuttle (e.g. if the shuttle capacity is 10 places, PD robots can take over at most 3 places). We assume that each PD robot can serve only one freight request at a time. In other words, a PD robot can only pickup one request from its origin to a transfer point and deliver it from a transfer point to its final destination during one single trip. This assumption can be relaxed so as to consider more realistic settings in which a PD robot can perform multiple pickups and deliveries during a single trip.

Furthermore, the following set of assumptions is used throughout the paper:

- SLs are assumed to be homogeneous in terms of frequency and capacity.
- We assume that all the shuttles operating on SLs have the same capacity. Each shuttle is thus assumed to have a maximum number of places to transport both passengers and PD robots.
- We assume that a PD robot might return to a different station than the one it departed from (as it is the case in Figure 1a) after delivering its request (i.e. relocation operations are not considered).
- As PD robots are likely to be electric ones, a PD robot is assumed to be fully charged at each time it departs from a transfer node for picking up or delivering a request and that this charge is enough to perform its trip (recharging operations are not considered).
- It is also assumed that each PD robot has a storage compartment (where parcels are stored during the robot trip) and those compartments are assumed to be homogeneous.
- Regarding freight demands, it is assumed that the exact quantity and delivery time windows of each request are known beforehand.
- In addition, we assume that each demand unit corresponds to a package of a standardized small size so that it can fit in robot storage compartments (content, nature and weight of the package are disregarded).
- Finally, we assume that travel and service times are known beforehand and remain unchanged during the planning horizon.

Similar to Ghilas et al. (2016b), each scheduled line is replicated in n copies. Figure 2 illustrates an example in which we have four transfer nodes {1, 2, 3, 4}, three physical scheduled lines (i.e., arcs (1,2), (2,1), (2,3), (3,2), (3,4) and (4,3)) and two requests { a, b }. Each replication is assigned to one request, and only that specific request



(a) Physical scheduled line (b) Virtual scheduled line

Figure 2.: Scheduled line with four replicated transfer nodes

can travel on the assigned scheduled line (Figure 2b). As such, the set of all replicated scheduled lines is denoted by $\mathcal{F}$ (i.e., { (1a,2a), (1b,2b), (2a,3a), (2b,3b), (3a,4a), (3b,4b), (2a,1a), (2b,1b), (3a,2a), (3b,2b), (4a,3a), (4b,3b) } in Figure 2b). Furthermore, the set of replicated SLs associate with request r is given as $\mathcal{F}^r$ (e.g., in Figure 2, $\mathcal{F}^a = \{(1a, 2a), (2a, 1a), ..., (3a, 4a), (4a, 3a)\}$). In addition, the set of replicated SLs related to the replicated transfer node t is given as $\mathcal{F}^t$ (e.g. $\mathcal{F}^{1a} = \{(1a, 2a), (2a, 1a)\}$). Finally, $\mathcal{F}^{ij}$ includes all replicated SLs associated with a physical SL $(i, j) \in \mathcal{E}$ (e.g. $\mathcal{F}^{1,2} = \{(1a, 2a), (1b, 2b)\}$ and $\mathcal{F}^{2,1} = \{(2a, 1a), (2b, 1b)\}$).

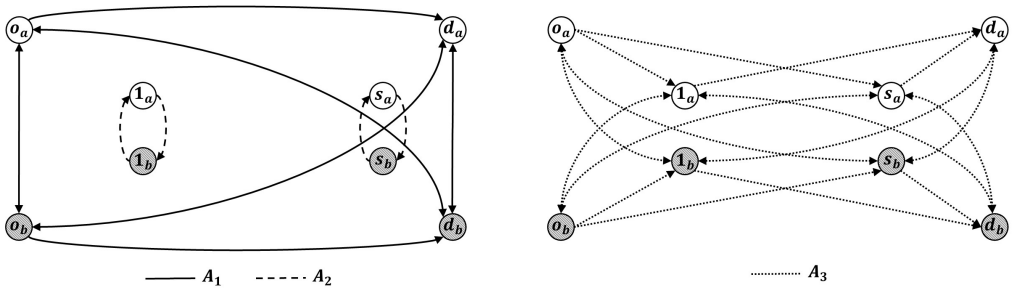


Figure 3.: An example network with s replicated nodes and two requests

Furthermore, each transfer node in $\mathcal{S}$ (i.e. nodes 1, 2, 3 and 4 in Figure 2a) is copied n times. Hence, we denote the set of all replicated transfer nodes by $\mathcal{T}$ (i.e. $\mathcal{T} = \{1a, 1b, 2a, 2b, 3a, 3b, 4a, 4b\}$ in Figure 2b). In addition, we use ψ^t , $\forall t \in \mathcal{T}$ as the physical transfer node represented by the replicated transfer node t (e.g. $\psi^{1a} = \psi^{1b} = 1$). Thus, set $\mathcal{T}^t$ is $\{i \in \mathcal{T} |\ \psi^i = \psi^t \text{ and } i \neq t\}$, $\forall t \in \mathcal{T}$ (e.g. $\mathcal{T}^{2a} = \{2b\}$ in Figure 2b).

The proposed pickup and delivery problem is defined on a digraph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N} = \mathcal{P} \bigcup \mathcal{D} \bigcup \mathcal{T}$, represents the set of graph nodes (i.e. request origins, destinations and replicated transfer nodes), and $\mathcal{A} \equiv \mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_3$ represents the set of feasible arcs connecting different graph nodes, where:

- $\mathcal{A}_1 = ((\mathcal{P} \cup \mathcal{D}) \times (\mathcal{P} \cup \mathcal{D})) \ \setminus \{(r + n, r) : r \in \mathcal{P}\}$
- $\mathcal{A}_2 = \{(i, j) : i, j \in \mathcal{T}, (\psi^i, \psi^j) \notin \mathcal{E}\}$
- $\mathcal{A}_3 = ((\mathcal{P} \cup \mathcal{D}) \times \mathcal{T}) \ \setminus (\{(j, r) : r \in \mathcal{P}, j \in \mathcal{T}^r\} \cup \{(r + n, j) : r \in \mathcal{P}, j \in \mathcal{T}^r\})$

As can be seen in Figure 3, subset $\mathcal{A}_1$ represents arcs linking request origin and

destination nodes, subset $\mathcal{A}_2$ represents arcs linking replicated transfer nodes, and subset $\mathcal{A}_3$ links request origin and destination nodes to transfer nodes.

For modeling the problem, we introduce two binary variables; x_{ij}^v equals to 1 if arc (i,j) is used by PD robot v and 0 otherwise, $\forall (i,j) \in \mathcal{A}, v \in \mathcal{V}$, and q_{ij}^{vw} equals to 1 if replicated scheduled line (i,j) is used by PD robot v that departs from node i at time p_{ij}^w and 0 otherwise, $v \in \mathcal{V}, (i,j) \in \mathcal{F}^i j, w \in \mathcal{K}^{ij}$. In addition, we introduce two timing variables; β_i indicates the departure time of a PD robot from node i, and γ_i^v which indicates the departure time of a PD robot $v \in \mathcal{V}$ from replicated transfer node i (notations and variables used in this paper are summarized in Table 1). We present a two-stage stochastic model in the following sections.

	Notations:
$\mathcal{S}$	Set of physical transfer nodes.
$\mathcal{T}$	Set of replicated (virtual) transfer nodes.
$\mathcal{E}$	Set of physical scheduled lines.
$\mathcal{F}$	Set of replicated (virtual) scheduled lines.
$\mathcal{P}$	Set of requests (represented by their origin location nodes).
$\mathcal{D}$	Set of request destination nodes.
$\mathcal{V}$	Set of PD robots.
$\mathcal{K}^{ij}$	Set of indices for the departure times from origin node i of scheduled line $(i,j) \in \mathcal{E}$.
η_{ij}	Cost of shipping one unit of package on scheduled line $(i,j) \in \mathcal{E}$.
θ_v	Routing cost per time unit of PD robot $v \in \mathcal{V}$
Q_{ij}	Capacity of scheduled line $(i,j) \in \mathcal{E}$.
Q_v	Capacity of PD robot $v \in \mathcal{V}$.
o_v	Origin location of PD robot $v \in \mathcal{V}$.
t_{ij}	Travel time from node i to node j.
s_i	Service time at node i.
	Decision variables:
$x_{ij}^v =$	$\begin{cases} 1 & \text{if arc } (i,j) \text{ is used by robot } v \\ 0 & \text{otherwise} \end{cases}$
$q_{ij}^{vw} =$	$\begin{cases} 1 & \text{if replicated SL } (i,j) \text{ is used by robot } v \text{ that departs from node } i \text{ at time } p_{ij}^w \\ 0 & \text{otherwise} \end{cases}$
	Timing decisions:
β_i	Departure time of a robot from node i.
γ_i^v	departure time of robot $v \in \mathcal{V}$ from transfer node i.

Table 1.: Notations and Variables

The first-stage model

$$Min \sum_{(i,j) \in \mathcal{A}} \sum_{v \in \mathcal{V}} \theta_v t_{ij} x_{ij}^v \; + \; E\left[Q\left(\delta \, , \, \xi \, , \eta \right) \right] \tag{1}$$

subject to

Routing and flow constraints

$$\sum_{i\in\mathcal{N}}\sum_{v\in\mathcal{V}} x_{ij}^{v} = 1 \qquad \forall j \in \mathcal{P}\cup\mathcal{D} \tag{2}$$

$$\sum_{i\in\mathcal{N}} x_{o_v i}^{v} \leq 1 \qquad \forall v \in \mathcal{V} \tag{3}$$

$$\sum_{i\in\mathcal{N}}\sum_{v\in\mathcal{V}} x_{it}^{v} \leq 1 \qquad \forall t \in \mathcal{T} \tag{4}$$

$$\sum_{j\in\mathcal{N}} x_{ij}^{v} - \sum_{j\in\mathcal{N}} x_{ji}^{v} = 0 \qquad \forall i \in \mathcal{N}, \forall v \in \mathcal{V} \tag{5}$$

$$\sum_{t\in\mathcal{T}} x_{it}^{v} - \sum_{t\in\mathcal{T}} x_{tj}^{v} = 0 \qquad \forall v \in \mathcal{V}, \forall (i,j) \in \mathcal{P}\times\mathcal{D} \tag{6}$$

$$t_{ij} x_{ij}^{v} \leq \delta_v \qquad \forall i,j \in \mathcal{N}, \forall v \in \mathcal{V} \tag{7}$$

Capacity constraints

$$\sum_{i\in\mathcal{T}}\sum_{v\in\mathcal{V}} d_j x_{ij}^{v} \leq Q_v \qquad \forall j \in \mathcal{P} \tag{8}$$

Scheduling constraints

$$\sum_{v\in\mathcal{V}} x_{ij}^{v} = 1 \implies \beta_j \geq \beta_i + t_{ij} + s_j \qquad \forall i,j \in \mathcal{N} \tag{9}$$

$$\beta_{r+n} \geq \beta_r + t_{r,r+n} + s_{r+n} \qquad \forall r \in \mathcal{P} \tag{10}$$

$$e_i \leq \beta_i - s_i \leq l_i \qquad \forall i \in \mathcal{P}\cup\mathcal{D} \tag{11}$$

Synchronization constraints

$$\sum_{w\in\mathcal{K}^{\psi^i,\psi^j}} q_{ij}^{vw} = x_{ij}^{v} \qquad \forall v \in V, (i,j) \in \mathcal{F}^v \tag{12}$$

$$q_{ij}^{vw} = 1 \text{ and } x_{ij}^{v} = 1 \implies \gamma_i^v = p_{ij}^w \qquad \forall v \in \mathcal{V}, (i,j) \in \mathcal{F}^v, w \in \mathcal{K}^{\psi^i,\psi^j} \tag{13}$$

Decision variable domains

$$x_{ij}^{v} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, v \in \mathcal{V} \tag{14}$$

$$q_{ij}^{vw} \in \{0,1\} \qquad \forall v \in \mathcal{V}, \forall (i,j) \in \mathcal{F}^v, w \in \mathcal{K}^{\psi^i,\psi^j} \tag{15}$$

$$\beta_i \in \mathcal{R}^+ \qquad \forall i \in \mathcal{N} \tag{16}$$

$$\gamma_i^v \in \mathcal{R}^+ \qquad \forall v \in \mathcal{V}, i \in \mathcal{T} \tag{17}$$

The objective function (1) minimizes the total costs of operating PD robots and the recourse costs incurred by SL capacity violations. In the recourse function, δ is the given routing vector, ξ is the set of scenarios, and η is the cost vector for using the scheduled lines per unit shipped. In this problem, we have four sets of constraints: routing, capacity, scheduling, and synchronization constraints. As for routing and flow constraints, constraints (2) state that all request pickup and delivery nodes (origins and destinations) are visited exactly once by a PD robot. Constraints (3) ensure that each PD robot must leave its depot at most once. Constraints (4) ensure that each replicated transfer node is visited at most once. Flow conservation for PD robots is considered in constraints (5). Constraints (6) ensure that the same PD robot that picked up the request at its origin, will proceed to deliver it to its final destination (i.e. this set of constraints couple the pickup and delivery trips of PD robots). Constraints (7) ensure that the maximum travel distance that PD robots can perform is respected. Since requests demand is known beforehand, constraints (8) ensure that the capacity of PD robots is respected at each time they pickup a request. For the scheduling constraints, constraints (9) ensure that if arc (i,j) is used by PD robot v, the departure time of v from node j should be greater than or equal to the sum of v departure time from node i, the travel time from i to j, and the service time at node j. Precedence relations for each request (i.e. request origins should be visited before their destinations) are considered in constraints (10). Constraints (11) enforce time window restrictions on request pickup and delivery. In order to synchronize PD robot trips and the scheduled line, constraints (12) and (13) ensure that the departure time of a PD robot at a transfer node is equal to the SL departure time at that transfer node (i.e. their departures are synchronized).

Note that constraints (9) and (13) are formulated as implications, and thus, need to be linearized. Using standard linearization techniques, we express them by one or two linear inequalities as follows:

$$\beta_j \geq \beta_i + t_{ij} + s_j - M_{ij}\ (1 - \sum_{v \in \mathcal{V}} x_{ij}^v) \qquad \forall i,j \in \mathcal{N} \tag{18}$$

$$\gamma_i^v \leq p_{ij}^w + M_i\ (2 - q_{ij}^{vw} - x_{ij}^v) \qquad \forall v \in \mathcal{V}, (i,j) \in \mathcal{F}^v, w \in \mathcal{K}^{\psi^i,\psi^j} \tag{19}$$

$$\gamma_i^v \geq p_{ij}^w - M_i\ (2 - q_{ij}^{vw} - x_{ij}^v) \qquad \forall v \in \mathcal{V}, (i,j) \in \mathcal{F}^v, w \in \mathcal{K}^{\psi^i,\psi^j} \tag{20}$$

The second-stage decisions

Due to the uncertainty, the SL capacity might be violated each time a shuttle arrives at a transfer node. This is because passenger demands are unknown by the time of the planning and are assumed to follow a known probability distribution. In other words, the SL service might not be sufficient for the actual passengers demand and PD robots (21). Given the routing solution vector $\boldsymbol{\delta}$, indicating PD robot routes and schedules from the first-stage, the aim of the second-stage is to evaluate this solution over a set of scenarios and calculate the expected recourse cost ($E\ [\ Q\ (\ \boldsymbol{\delta}\ ,\ \xi\ ,\eta\ )\ ]$). At this stage, a scenario indicates the realized passengers demand at each departure from a transfer node, and thus, the number of available places for transporting PD robots.

$$\sum_{r \in \mathcal{P}'} \sum_{(a,b) \in \mathcal{F}^{ij}} q_{ab}^{rw}\ > Q_{ij}^w \qquad \forall (i,j) \in \mathcal{E},\ w \in \mathcal{K}^{ij} \tag{21}$$

Since the number of available places at each shuttle is only revealed upon the shuttle's arrival time at a transfer node, capacity violations might occur at the corresponding transfer node (denoted as failure point). Depending on passenger demand realizations, these capacity violations might occur in two different situations:

- ***Situation#1***: After picking up a request and bringing it to a transfer node, a PD robot may not be able to take the next SL departure at that transfer node due to the high passenger demand (passengers are prioritized over PD robots).
- ***Situation#2***: After taking a shuttle to travel between two transfer nodes as part of its trip, a PD robot may need to get off the SL at an intermediate transfer node due to high passengers demand. In this case, the PD robot needs to give its place to one of the passengers who are willing to take the SL at that transfer node.

In both situations, the same capacity violation is obtained: not enough capacity for transporting PD robots with passengers through the SL service. A set of corrective (or recourse) actions needs to be applied in order to recover feasibility, which might lead to additional costs. We consider the following recourse actions to deal with both situations leading to capacity violation outcome. These are:

- ***Action#1***: If the PD robot cannot take the current departure at the failure point due to high passengers demand, it is transported using the subsequent service of the scheduled line. In other words, the PD robot waits for the next shuttle arriving to the failure point. This recourse action comes with no extra costs as long as waiting the next departure does not violate request delivery time window.
- ***Action#2***: If waiting the next shuttle departure leads to violating the capacity of the subsequent SL service or request delivery time window. If the distance between failure point and request destination is less than the maximum service distance that the PD robot can handle, the PD robot delivers the request to its final destination by itself. This recourse action implies some additional costs since a PD robot might have to perform a longer trip than planned.
- ***Action#3***: If none of the first two recourse actions can be applied, the request

is served by an outsourced service (a dedicated vehicle). This service transports the request from failure point to its final destination. The extra cost implied by using this outsourced service depends on the distance that the outsourced vehicle has to travel.

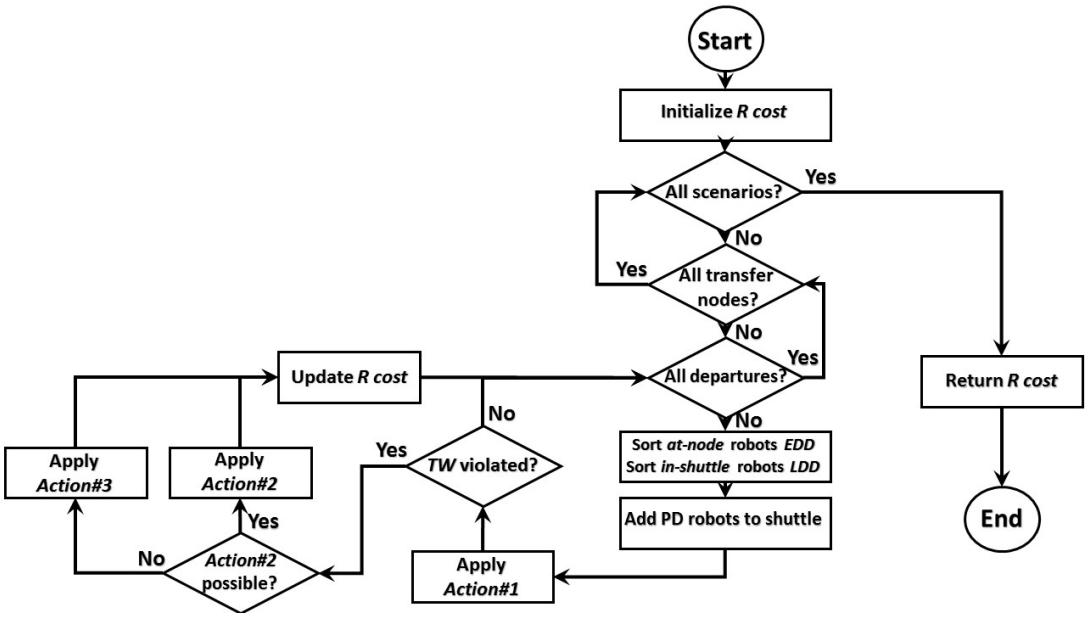


Figure 4.: Calculate-Recourse-Cost Algorithm

Another important issue is to rank or schedule PD robots that are waiting to take the scheduled line at one transfer node, according to some criteria. The model needs to decide which PD robots have the priority to be transported in case the realized number of available places in a shuttle is insufficient (*situation#1*). For this purpose, we sort PD robots at each transfer node according to the earliest delivery date of the requests they carry. The PD robot carrying request with the earliest delivery date is thus the first to be transported when a shuttle with available places arrives to the corresponding transfer node. A similar issue appears when some PD robots need to get off a shuttle at an intermediate transfer node to give space to more passengers (*situation#2*). Therefore, the model also needs to determine the order in which PD robots are asked to get off a shuttle at a certain intermediate transfer node. In this latter case, PD robots already in shuttle are sorted according to their latest delivery date (i.e. PD robot carrying request with the latest delivery date has to get off the shuttle first). It is important to mention that we do not consider the case where a PD robot is asked to get off to allow another one (with an earlier delivery date) to take it's place. PD robots are thus asked to get off only to give place to passengers.

To summarize, the recourse function checks if there are capacity violations at each SL departure. In case SL capacity is violated at a given departure, PD robots that are waiting at the corresponding transfer node, referred to as *at-node* PD robots, are sorted according to the earliest delivery date of their carried requests. In addition, PD robots that are already *in-shuttle* are sorted according to the latest delivery date of their requests. Then, *at-node* and *in-shuttle* PD robots that cannot take the shuttle at the current departure are assigned to the next departure (*action#1*) since it does not imply extra costs. If waiting for the next departure leads to violating the request's

time windows, the recourse function checks if some PD robots can deliver their requests from the corresponding transfer node to their final destination (*action#2*).

Finally, the still remaining requests (i.e. that could not be served using neither *action#1* nor *action#2*) are outsourced using dedicated delivery vehicles (*action#3*). As a result, depending on actual passenger demand, *(i)* some *at-node* PD robots might be able to take SL next departure while others might have to wait, *(ii)* all *at-node* PD robots might not be able to take the next shuttle while no *in-shuttle* PD robots are asked to drop off, or *(iii)* all *in-shuttle* PD robots may have to drop off from the shuttle and join the waiting PD robots at the corresponding transfer node. The algorithm for calculating the recourse cost of a given routing solution is outlined in Figure 4 (see also Appendix A for the detailed recourse function).

4. Solution Approach

In this section, we present our solution approach. This consists of a scenario-based Sample Average Approximation (SAA) framework (Section 4.1), and an ALNS-based heuristic, to solve the corresponding SAA problems (Section 4.2).

4.1. The Sample Average Approximation method

The Sample Average Approximation (SAA) method is an iterative approach for solving stochastic optimization problems. It aims at approximating the expected objective function of the stochastic problem using a sample average estimate derived from a random sample (Verweij et al. (2003)). While the set of possible scenarios might be very large, the SAA iteratively solves the problem using smaller and more tractable sets of scenarios (referred to as SAA problems), and obtains candidate solutions along with their respective optimality gaps.

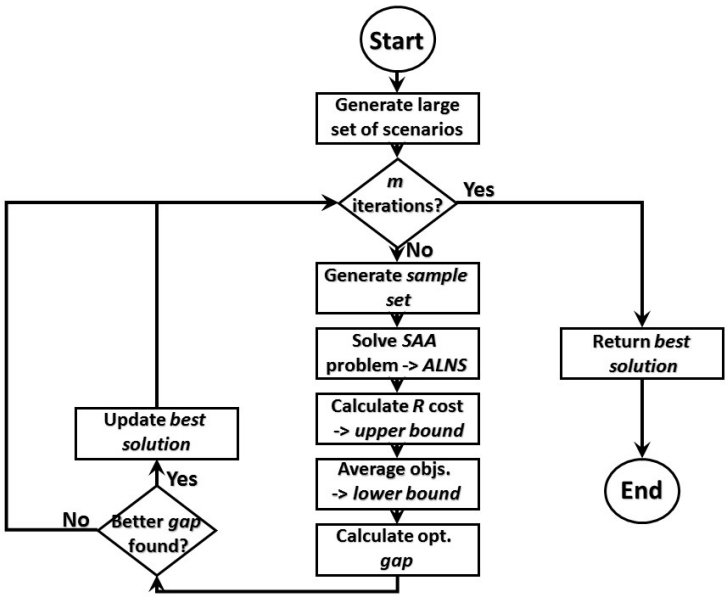


Figure 5.: Sample Average Approximation (SAA) Algorithm

The method starts by generating a large set of scenarios Ω and iterates until the value of the optimal solution is approximated by solving the stochastic problem with smaller sample sets. At each iteration l, a sample set of scenarios $\omega_l : |\omega_l| << |\Omega|$ is generated from the larger set Ω and the corresponding SAA problem is solved using the ALNS heuristic. The obtained solution x_l with objective value $f^l_{\omega_l}$ is then evaluated using the recourse function (Algorithm 2) in order to determine an upper bound $f_\Omega(x_l)$ for the generated set of scenarios Ω:

$$f_\Omega(x_l) = CalculateRecourseCost(x_l, \Omega) \tag{22}$$

Afterwards, a statistical lower bound, denoted by $f'_{\omega_l}(x_l)$, for the optimal solution value of sample ω_l is calculated by averaging the objective function values obtained in previous iterations:

$$f'_{\omega_l}(x_l) = 1/l \sum_{i=1}^{l} f^i_{\omega_l}, \tag{23}$$

where $f^i_{\omega_l}$ is the objective function value obtained at iteration i. To the best of our knowledge, this is the most commonly used approach in the literature for approximating a statistical lower bound in SAA-based methods (see also Ghilas et al. (2016c); Verweij et al. (2003)). Once both bounds are obtained ((22),(23)), the SAA gap is calculated as follows:

$$\epsilon(\omega_l, \Omega) = f'_\Omega(x_l) - f'_{\omega_l}(x_l) \tag{24}$$

The process continues until the best gap $\epsilon(\omega_l, \Omega)$ is found and the corresponding best solution is returned (see Figure 5, and Appendix B for the detailed algorithm).

4.2. ALNS heuristic

An ALNS heuristic algorithm is used to generate routing solutions of minimum total cost. The heuristic is used in combination with the recourse function (Algorithm 2) in order to compute the recourse cost of a generated solution. The main idea of the ALNS is to iteratively apply a set of removal and insertion operators on an initial solution until the best solution is found (Algorithm 1).

The algorithm starts by generating an initial solution indicating initial PD robot routes (section 4.2). The algorithm then applies a removal operator to remove one PD route from the initial solution. The removed PD route is then reconstructed and reinserted to the solution using an insertion operator and a new solution is obtained. The operators are dynamically selected according to their past and current performances through a roulette-wheel mechanism. In other words, each operator, removal or insertion, is associated with a *score* that is increased at each time this operator leads to a better solution, and a probability that indicates how likely this operator is to be selected in the next iteration. This means that operators with better scores have a higher probability to be used by the algorithm. In order to build their scores, operators are selected randomly in the first 100 iterations. The roulette-wheel mechanism is then used based on the calculated operator scores. Once applying these operators yields an improvement, the new solution is stored, and the best solution is updated.

Algorithm 1 The ALNS Framework

```
1: procedure ALNS-Heuristic(Set of removal operators $\mathcal{O}_R$, set of insertion operators $\mathcal{O}_I$)
2:     generate an initial solution: current solution
3:     initialize best solution ← current solution
4:     for a number of iterations do
5:         select a removal operator $r^* \in \mathcal{O}_R$ with probability $P_{r^*}$
6:         apply operator $r^*$ to current solution to obtain a partially destroyed solution
7:         select an insertion operator $i^* \in \mathcal{O}_I$ with probability $P_{i^*}$
8:         apply operator $i^*$ to repair the partially destroyed solution and get new solution
9:         if new solution is better than current solution then
10:            current solution ← new solution
11:            if current solution is better than best solution then
12:                best solution ← current solution
13:        update operator probabilities
14:    return best solution
```

The algorithm continues until either a maximum number of iterations or a certain number of iterations with no improvement is reached.

Generating initial solutions

Since we use simplified problem settings, in which only one pickup/delivery per PD robot trip is allowed, we start with a simple heuristic to generate initial feasible solutions. This heuristic is composed of two main steps:

(1) We start by selecting requests that can be delivered directly by a PD robot depending on the distance between their origin and destination locations (*direct delivery*). These direct PD robot routes are then added to the initial solution.
(2) For the other requests (*indirect delivery*), indirect PD robots are constructed by randomly assigning them to one of the feasible pickup/drop-off transfer nodes while respecting their time restrictions.

To this end, the feasibility of the returned solution, in terms of request time windows and SL departure times, is assured. This initial feasible solution can then be improved by the ALNS operators as it does not lead to min-cost PD robot routes. We describe the removal and insertion operators used by the ALNS algorithm in the following subsections.

Removal operators

- *Random removal (**R1**)*: This operator removes a randomly selected robot route (request) from the solution which helps in diversifying the search for a better solution.
- *Limited random removal (**R2**)*: This operator is similar to *R1* but it limits the number of times a robot route (request) is removed in the last 100 iterations. For other requests, which their counts have not reached the specified limit, *R1* is applied.
- *Tabu-based removal (**R3**)*: This operator also keeps a record of robot route re-

moval counts for the last 100 iterations (as *R2*) and removes those with the smallest frequency of removal rate. This operator also helps in diversifying the search.

- *Early-SL-depart removal (**R4**)*: This operator removes the robot route with the shortest waiting time at the pickup transfer nodes from the solution. The request waiting time at a specific transfer node is obtained from the difference between its arrival to that transfer node and its departure from it (i.e. a PD robot might have to wait at a transfer node until the next SL departure).
- *Late-SL-depart removal (**R5**)*: Unlike *R4*, this operator removes the robot route with the longest waiting time at transfer nodes from the solution.

Insertion operators

- *Pickup Transfer-node insertion (**I1**)*: This operator reconstructs a robot route by assigning it to a different pickup transfer node than the one it was assigned to before being removed. This operator helps diversifying the search by leading to different transportation costs (i.e. operational and recourse costs). This potential improvement highly depends on SL and PD robot transshipment costs as well as the maximum service distance of PD robots which can limit the feasibility of this assignment.
- *Drop-off Transfer-node insertion(**I2**)*: This operator reconstructs a robot route by assigning it to a drop-off transfer node that is different than the one it was assigned to before being removed. Similar to *I1*, this operator can lead to different transportation costs.
- *Early-SL-depart insertion (**I3**)*: This operator reconstructs a robot route by assigning it to the same pickup transfer node but with an earlier departure time. Indeed, changing the SL departure to which a PD robot is assigned, leads to different recourse costs as passengers demand varies between different SL departures.
- *Late-SL-depart insertion (**I4**)*: Unlike *I3*, this operator reconstructs a robot route by assigning it to the same pickup transfer node but with a later departure time. This operator is also important for solving the stochastic optimization problem as it leads to different recourse costs.

That said, these operators are used by the heuristic to remove and insert robot routes to a current solution. They provide a reasonable choice for our problem settings where only one request is served during a PD robot trip. The heuristic can thus be extended by considering different operators when PD robots are allowed to perform multiple pickups and deliveries at one trip (see operators at Ghilas et al. (2016b,c)).

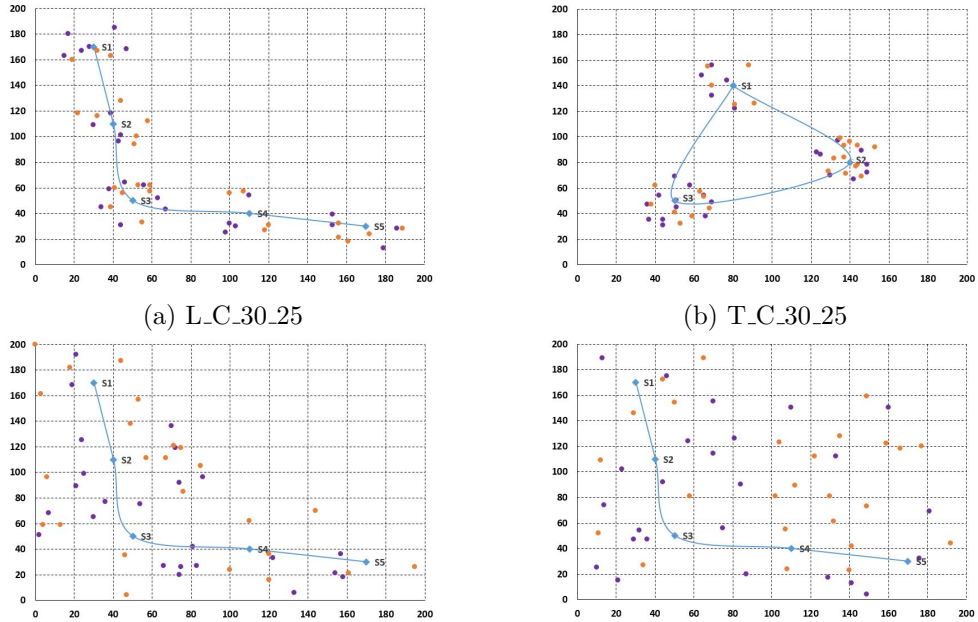
5. Computational study

In this section, an extensive computational study to assess the performance of the proposed solution approach is presented. First, we explain how we generate test instances and we describe the different parameters used (Section 5.1). We then show how we generate the set of scenarios used by the SAA algorithm for solving the stochastic problem (Section 5.2). Afterwards, we analyze the performance of the proposed heuristic approach along with the different operators used, and compare the results obtained from solving the stochastic problem with those of the deterministic one (i.e.

when no uncertainty is considered). Finally, we study the impacts of the considered source of uncertainty on the obtained solutions with different settings (Section 5.3).

5.1. Parameters and instance generation

For testing the proposed solution approach, we generate instances with different network topologies and freight request distributions. Generated instances are named as `P_D_r_n`, where `P` represents the network topology, `D` is the geographical distribution of freight requests, `r` is request nodes range from transfer nodes, and `n` is the number of freight requests. Since the proposed model can adapt different network topologies, we generate instances with line (referred to as "`L`") and triangular (referred to as "`T`") topologies (Figure 6a & 6b). While the number of SLs is different, instances with either topology have the same characteristics. In addition, each instance contains up to 60 freight requests where their origin and destination nodes are distributed over 200 x 200 Euclidean space. We consider three different distributions of freight requests (inspired from Ghilas et al. (2016c)). These are: `C` - freight request origin and destination nodes are clustered within at most 30 time units around transfer nodes (Figure 6a), `RC` - request nodes are randomly clustered within at most 50 time units to one of the available transfer nodes (Figure 6c), and `UR` - freight requests are uniform-randomly distributed over the considered space (Figure 6d). As PD robots are located at transfer nodes, we consider up to three PD robots at each transfer node.



(a) L_C_30_25 (b) T_C_30_25

(c) L_RC_50_25 (d) L_UR_50_25

Figure 6.: Instances with different network topologies and request distributions

We consider a planning horizon of 600 time units where SL departure interval is set to 30 time units (i.e. there is a shuttle departing from each transfer node every 30 time units). We consider that this frequency is enough to cover passengers demand through SL. We generate request pickup and delivery time windows randomly with an average width of 40 time units. A minimum of 100 time units is also assured between the end of pickup time window and the start of delivery time window. Service time at each

location (i.e. pickup, drop-off or transfer node locations) is set to three time units. This service time represents the time needed for a PD robot to pick up or deliver a freight request, or to get in or off a shuttle at a transfer node.

Parameter	Value	Parameter	Value
PD robot cost	0.5	Num. iterations no improvement	50
SL cost	1	The size of the large set of scenarios	10 000
Outsourcing cost	3	The size of the sample set of scenarios	50
PD robot capacity	1	Number of ALNS iterations	10 000
SL capacity	10	Number of SAA iterations	10
Max. num. places for PD robots	3	Score for new best solution	3
Freight request quantity	1	Score for improving the current solution	1

Table 2.: Set of parameters used in the computational study

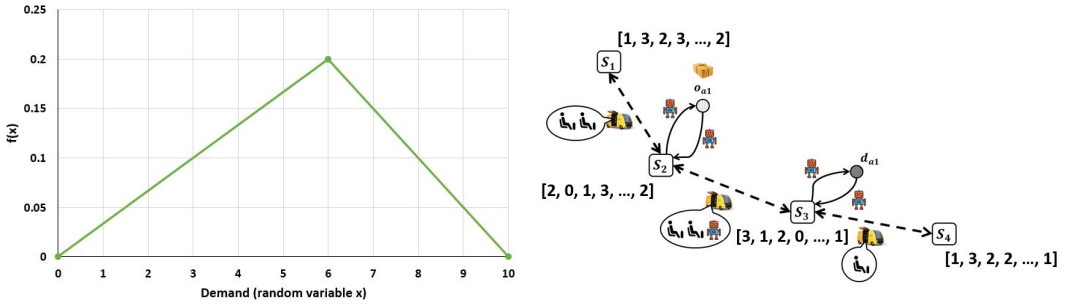
The capacity of each PD robot and the quantity of each freight request are set to 1. This means that each PD robot can serve one freight (i.e. any freight request) at a time. The capacity of shuttles on SL is set to 10 places for both passengers and PD robots where PD robots can take up to 3 places (different limits are analyzed in section 5.3). Regarding transportation costs, we assume the time unit cost for PD robots to be 0.5 unit. This cost includes energy consumption, insurance and transportation expenses induced when PD robots are used. In addition, the time unit cost of using SL service is set to 1 unit. This cost includes loading, unloading, and transportation expenses each time a PD robot uses the SL service. Finally, the recourse cost of using the outsource delivery service is assumed to be 3 units (different SL and robot shipment costs were analyzed in Ghilas et al. (2016a)).

As introduced in section 4.2, we consider two stopping criteria for the heuristic method. These are: the maximum number of ALNS iterations which is set to 10 000 iterations, and the maximum number of consecutive iterations with no improvement which is set to 50 iterations. In addition, the score of an operator is increased by 1 if it leads to improving the current solution, and by 3 if a new best solution is found. For the SAA algorithm, the size of the large set of scenarios (Ω) is set to 10 000 scenarios while the size of the smaller sample (ω) is set to 50 scenarios. Finally, the number of SAA iterations is set to 10 (SAA parameters are fixed based on Ghilas et al. (2016c); Li et al. (2016b) where similar problems and solution methods are considered). The set of parameters used in the computational study along with their values are presented in Table 2.

5.2. Scenario generation

In order to test the proposed SAA algorithm, we need to generate a large set of scenarios which represent the realized passengers demand at each SL departure. The actual passengers demand helps the algorithm to decide whether PD robots can be transported through SLs or some recourse actions need to be applied. For this purpose, passengers demand is assumed to follow a discrete triangular distribution for a given minimum value $a = 0$, mean $b = 6$ and a maximum value $c = 10$ (Figure 7a).

For example in Figure 7b, based on the realized passengers demand, the number of available places for PD robots at the different departures of scheduled line $S_2 \rightarrow S_3$ is respectively $[2, 0, 1, 3, ..., 2]$. Hence, PD robots are not able to take the second shuttle



(a) Passengers demand distribution

(b) An example of SAA scenario

Figure 7.: SAA scenario generation

departure from S_2 to S_3 as there are no available places for them, while there are 3 available places at the fourth departure etc.

5.3. Experiments

The algorithms developed in this paper (i.e. recourse, ALNS and SAA algorithms) are implemented in Java 1.8.0. CPLEX 12.6 solver is used for solving the MIP formulation. Instances are tested on a quad-core i5-5300U machine with 8 GB of RAM. We study the efficiency of the proposed ALNS approach by comparing its results to those obtained with CPLEX solver and analyzing the performance of its operators. We then examine the stochastic solutions obtained by the SAA algorithm, compare them with the deterministic ones, and analyze the impact of different levels of passengers demand, SL frequency and capacity on the obtained solutions.

Analyzing ALNS performance:

The results of solving instances with up to 100 freight requests are presented in Table 3 (results obtained by CPLEX are in **bold**). In this table, *# dir.* indicates the number of direct deliveries, *# ind.* indicates the number of indirect deliveries, and *# usv.* indicates the number of unserved freight requests. In addition, *Cost* column represents the total transportation costs obtained by the ALNS heuristic and CPLEX (respectively) while *Gap (%)* column gives the optimality gap percentage between them. Finally, *CPU* column indicates the execution time needed to run both approaches, and *# iter.* column gives the number of ALNS iterations performed.

Looking at table 3, we observe that the proposed ALNS is always able to find a solution that is identical to the optimal one obtained by solving the MIP in terms of direct and indirect deliveries. In addition, the ALNS is able reach the optimal solutions ($Gap = 0$) for all instances with less than 40 freight requests. For instances with more than 40 requests, the ALNS is still capable of finding solutions that are within 0.6% of the optimal solutions. Moreover, the proposed heuristic returns solutions for instances with 100 requests for which CPLEX is not able to find optimal solutions. This is due to the increasing complexity of the problem (i.e. number of variables) when the number of freight requests gets larger. Since the numbers of direct and indirect deliveries are the same in both solutions, this small gap indicates that there are very few requests that could have been assigned to another pickup or drop-off transfer node so that some costs can be saved. We also observe that total costs are generally lower for

Instance	# dir.	# ind.	# usv.	Cost	Gap (%)	CPU (s)	# iter.
L_C_30_10	0 **(0)**	10 **(10)**	0 **(0)**	1030.1 **(1030.1)**	0.0	0.03 **(4.1)**	19
L_C_30_20	2 **(2)**	18 **(18)**	0 **(0)**	3339.1 **(3339.1)**	0.0	0.31 **(32.3)**	269
L_C_30_30	3 **(3)**	27 **(27)**	0 **(0)**	4156.4 **(4156.4)**	0.0	0.53 **(143.5)**	431
L_C_30_40	5 **(5)**	35 **(35)**	0 **(0)**	4896.9 **(4872.2)**	0.49	0.79 **(537.6)**	728
L_C_30_60	7 **(7)**	53 **(53)**	0 **(0)**	7452.2 **(7443.9)**	0.11	1.47 **(3103.4)**	949
L_C_30_100	12 **(-)**	88 **(-)**	0 **(-)**	12695.1 **(-)**	-	5.01 **(-)**	3157
L_RC_30_10	2 **(2)**	8 **(8)**	0 **(0)**	971.2 **(971.2)**	0.0	0.03 **(3.3)**	20
L_RC_30_20	2 **(2)**	18 **(18)**	0 **(0)**	2910.4 **(2910.4)**	0.0	0.26 **(23.9)**	219
L_RC_30_30	3 **(3)**	25 **(25)**	2 **(2)**	4120.2 **(4120.2)**	0.0	0.41 **(112.9)**	382
L_RC_30_40	5 **(5)**	34 **(34)**	1 **(1)**	5291.3 **(5291.3)**	0.0	0.76 **(351.9)**	656
L_RC_30_60	8 **(8)**	51 **(51)**	1 **(1)**	9006.8 **(8983.8)**	0.25	1.53 **(1802.4)**	976
L_RC_30_100	12 **(-)**	87 **(-)**	1 **(-)**	14747.9 **(-)**	-	5.31 **(-)**	3323
L_UR_30_10	0 **(0)**	8 **(8)**	2 **(2)**	1472.1 **(1472.1)**	0.0	0.03 **(4.5)**	17
L_UR_30_20	1 **(1)**	16 **(16)**	3 **(3)**	3510.0 **(3510.0)**	0.0	0.32 **(26.9)**	228
L_UR_30_30	2 **(2)**	23 **(23)**	5 **(5)**	4707.5 **(4707.5)**	0.0	0.49 **(135.7)**	431
L_UR_30_40	1 **(1)**	34 **(34)**	5 **(5)**	6120.8 **(6099.7)**	0.34	0.69 **(349.3)**	547
L_UR_30_60	4 **(4)**	51 **(51)**	5 **(5)**	9779.1 **(9741.8)**	0.38	1.57 **(2215.9)**	1095
L_UR_30_100	5 **(-)**	85 **(-)**	10 **(-)**	14917.4 **(-)**	-	6.12 **(-)**	4217
T_C_30_10	0 **(0)**	10 **(10)**	0 **(0)**	1087.7 **(1087.7)**	0.0	0.03 **(1.3)**	21
T_C_30_20	5 **(5)**	15 **(15)**	0 **(0)**	1826.9 **(1826.9)**	0.0	0.14 **(7.1)**	239
T_C_30_30	7 **(7)**	23 **(23)**	0 **(0)**	2535.9 **(2535.9)**	0.0	0.47 **(31.1)**	341
T_C_30_40	7 **(7)**	33 **(33)**	0 **(0)**	4087.4 **(4063.5)**	0.0	1.07 **(95.4)**	672
T_C_30_60	12 **(12)**	48 **(48)**	0 **(0)**	6001.2 **(5978.6)**	0.38	1.62 **(444.9)**	1313
T_C_30_100	20 **(-)**	79 **(-)**	1 **(-)**	9473.3 **(-)**	-	4.37 **(-)**	3543
T_RC_30_10	2 **(2)**	7 **(7)**	1 **(1)**	1121.3 **(1121.3)**	0.0	0.03 **(1.2)**	19
T_RC_30_20	1 **(1)**	17 **(17)**	2 **(2)**	2224.4 **(2224.4)**	0.0	0.22 **(5.9)**	227
T_RC_30_30	3 **(3)**	24 **(24)**	3 **(3)**	3666.8 **(3666.8)**	0.0	0.39 **(24.5)**	318
T_RC_30_40	1 **(1)**	36 **(36)**	3 **(3)**	5126.3 **(5126.3)**	0.0	0.96 **(61.7)**	566
T_RC_30_60	6 **(6)**	51 **(51)**	3 **(3)**	6340.7 **(6303.1)**	0.58	1.61 **(371.7)**	1032
T_RC_30_100	11 **(-)**	84 **(-)**	5 **(-)**	10148.9 **(-)**	-	5.91 **(-)**	3782
T_UR_30_10	0 **(0)**	7 **(7)**	3 **(3)**	1164.6 **(1164.6)**	0.0	0.03 **(1.2)**	24
T_UR_30_20	1 **(1)**	16 **(16)**	3 **(3)**	2784.6 **(2784.6)**	0.0	0.19 **(5.8)**	188
T_UR_30_30	2 **(2)**	24 **(24)**	4 **(4)**	3825.7 **(3825.7)**	0.0	0.32 **(20.6)**	240
T_UR_30_40	2 **(2)**	34 **(34)**	4 **(4)**	5673.4 **(5642.6)**	0.54	0.94 **(61.6)**	643
T_UR_30_60	5 **(5)**	44 **(44)**	11 **(11)**	7861.9 **(7829.2)**	0.41	1.38 **(317.6)**	927
T_UR_30_100	8 **(-)**	78 **(-)**	14 **(-)**	13351.1 **(-)**	-	4.34 **(-)**	2976
Average ALNS	4.6	36.1	2.5	5539.6	0.116	1.38	937.6

Table 3.: Analyzing ALNS performance

instances with clustered request distribution (L_C & T_C). This can be explained by the likelihood of performing direct deliveries which is higher in clustered instances, while requests are more scattered in randomly distributed instances (Table 3, "# dir."). This can also be reflected by the increasing number of unserved requests in randomly distributed instances. In this latter case, some requests cannot be brought to transfer nodes due to PD robot distance limitations. Another observation is that the total costs are generally higher in line networks than in triangular ones. This indicates

that a triangular network might provide a better coverage to the service area while reducing transportation costs.

The base case instances, with 10 freight requests, solve in few seconds with CPLEX. This amount of time increases as the number of freight requests increases. We observe that instances with line network topology need longer time to be solved to optimality than those with triangular topology (an average of 6.3 mins for triangular instances with 60 requests compared to 39.5 min for same instances with line topology). The reason is that the number of transfer nodes, and thus the number of variables and graph edges, is bigger in instances with line topologies. This observation also gives an indication that a triangular network topology might be more effective in terms of computational efforts needed to solve its instances. On the other hand, the proposed heuristic solves the different instances in very short running times (1.38 seconds in average) while maintaining near-optimal solutions. These short running times suggest that our approach is suitable for approximating optimal solutions for the stochastic problem where instances have to be solved over a large set of scenarios in the SAA method.

In tables 4 & 5, we analyze the removal and insertion operators used in the ALNS using some relevant information on their performance. For each operator, we present its usage frequency as a percentage of the total number of iterations, and the total time spent on running it (given in parenthesis).

Instance	R1	R2	R3	R4	R5
L_C_30_60	25.4% (0.02)	23.1% (0.02)	23.6% (0.02)	14.1% (0.01)	13.8% (0.01)
L_RC_30_60	24.3% (0.02)	27.2% (0.02)	21.4% (0.02)	12.6% (0.01)	14.5% (0.01)
L_UR_30_60	22.1% (0.02)	24.5% (0.02)	24.1% (0.02)	16.8% (0.01)	12.5% (0.01)
T_C_30_60	26.5% (0.02)	25.3% (0.02)	22.7% (0.02)	12.1% (0.01)	13.4% (0.01)
T_RC_30_60	26.2% (0.02)	23.1% (0.02)	27.6% (0.02)	12.9% (0.01)	10.2% (0.01)
T_UR_30_60	24.9% (0.02)	23.9% (0.02)	25.2% (0.02)	11.8% (0.01)	14.2% (0.01)
Average	**24.9%**	**24.5%**	**24.1%**	13.4%	13.1%

Table 4.: The performance of removal operators

Considering removal operators (Table 4), we observe that operators R1, R2 and R3 are the most frequently used. This is mainly because these three operators randomly select robot routes (requests) and are used to diversify the search for a better solution.

Instance	I1	I2	I3	I4
L_C_30_60	34.3% (0.04)	37.8% (0.04)	15.4% (0.02)	12.5% (0.01)
L_RC_30_60	32.7% (0.04)	38.4% (0.04)	14.8% (0.01)	14.1% (0.01)
L_UR_30_60	28.9% (0.03)	36.8% (0.04)	18.2% (0.02)	16.1% (0.02)
T_C_30_60	31.5% (0.04)	38.3% (0.04)	16.3% (0.02)	13.9% (0.01)
T_RC_30_60	34.7% (0.04)	40.1% (0.04)	13.6% (0.01)	11.6% (0.01)
T_UR_30_60	33.8% (0.04)	35.2% (0.04)	16.4% (0.02)	14.6% (0.01)
Average	**32.6%**	**37.8%**	15.8%	13.8%

Table 5.: The performance insertion operators

We also observe that I1 and I2 are the most frequently used insertion operators (Table 5). This indicates that operators which assign robot route to an earlier, or

later, SL departure (i.e. I3 and I4) are used less than other operators which assign it to a different pickup, or drop-off, transfer node.

Analyzing SAA performance and stochastic solutions

In order to quantify the impact of stochastic passengers demand, we solve the instances, introduced earlier in Table 3, using the proposed SAA algorithm. Results are presented in Table 6 where the first three columns represent the usage frequency of recourse actions 1, 2, and 3 (respectively) as a percentage of the total number of times recourse actions were used for each instance. The total transportation cost is then given along with the associated operational and recourse costs. The additional cost induced by uncertainty is then calculated by comparing the total cost of the stochastic solution with that of solving the deterministic version of the problem using the heuristic (given in Table 3).

In table 6, we observe that the realization of passengers demand can add an average of 3.3% to the total transportation cost. This increase is due to the recourse actions that are used to correct the interrupted robot routes. Indeed, when passengers demand is revealed, the actual number of places for PD robots at each SL departure might not be sufficient and recourse actions need to be applied adding extra expenses to the total transportation cost. Since the recourse function applies recourse actions one by one to recover feasibility, one can observe that *action#1* is the most frequently used among the other recourse actions (93.6% in average). This is because this recourse action uses the subsequent SL service (i.e. waiting the next SL departure) which does not imply additional transportation costs. We also observe that *action#2* is not frequently used by the algorithm (only 1.1%). This indicates that a direct PD robot delivery, from failure point to request destination, is not feasible in most of the time due to PD robot distance limitations. Most of the added recourse costs are thus induced by *action#3* as it guarantees the feasibility of all interrupted deliveries using the outsourced service. Regarding network topology, we observe that the average added cost for instances with line topology is lower than those with triangular topology (2.5% compared to 3.8%). This is because the number of stations in triangular network is less than that of the line network. As a consequence, the number of PD robots at each station is larger in a triangular network and the likelihood of applying recourse actions (i.e. to recover capacity violations at each SL departure) is thus higher. We analyze in the following the impacts of uncertainty under different settings including passengers demand and SL frequency and capacity.

Analyzing uncertainty with different levels of passengers demand

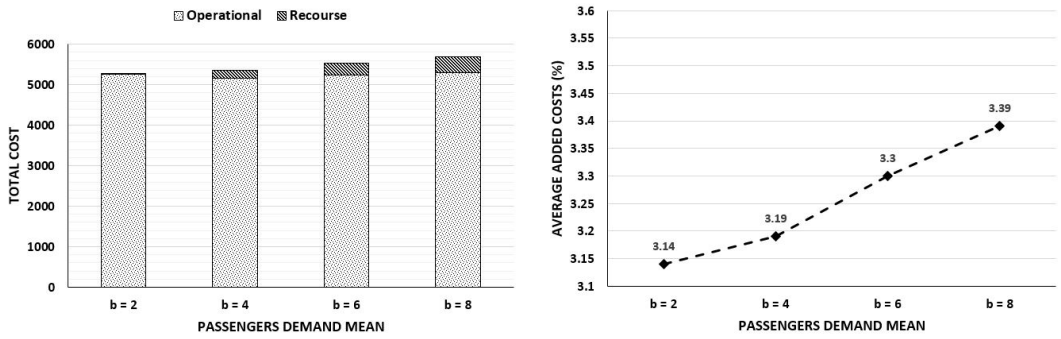
In the original setting, we generate SAA scenarios assuming that passengers demand follows a discrete triangular distribution with a mean value $b = 6$ (Figure 7a). In this section, we analyze the different levels of passengers demand by testing the algorithm with different mean values ($b = 2, 4, 6\&8$ respectively). As the mean value increases, the probability of having a large passengers demand at each SL departure becomes higher. This reflects a real-life case where passengers demand changes over day hours which can limit the integration of PD robot deliveries into the system. The aim of this analysis is thus to investigate the potential impact of these different levels of passengers demand. This is done by performing ten runs of the algorithm for each demand level and taking the average (Figure 8).

Results show that the total transportation costs increases as passengers demand becomes higher (Figure 8a). This increase is mainly induced by the recourse actions

Instance	act1(%)	act2(%)	act3(%)	Cost (oper. , rcs.)	Add(%)	CPU(s)
L_C_30_10	95	1	4	**1066.2** (1019.1 , 47.1)	3.5	15.7
L_C_30_20	81	2	17	**3427.1** (3070.2 , 356.9)	2.6	154.8
L_C_30_30	86	1	13	**4350.3** (3815.3 , 534.9)	4.7	234.5
L_C_30_40	91	2	8	**4983.4** (4382.8 , 600.6)	2.3	395.2
L_C_30_60	89	0.5	1.5	**7498.4** (7382.5 , 115.8)	0.7	635.4
L_C_30_100	94	1	5	**12839.7** (12671.6 , 168.1)	1.1	2505.2
L_RC_30_10	99	1	0	**975.1** (970.7 , 4.5)	0.4	13.8
L_RC_30_20	85	0	15	**3257.1** (2924.8 , 332.2)	5.7	129.8
L_RC_30_30	92	0	8	**4224.6** (4119.4 , 105.2)	2.5	204.5
L_RC_30_40	98	0	2	**5326.1** (5219.6 , 106.5)	0.7	381.2
L_RC_30_60	97	1	2	**9221.7** (8901.9 , 319.8)	2.6	765.1
L_RC_30_100	95	1	4	**15322.6** (14356.7 , 965.9)	3.9	2655.4
L_UR_30_10	99	1	0	**1478.3** (1470.1 , 8.3)	0.4	12.5
L_UR_30_20	93	1	6	**3559.3** (3427.8 , 131.5)	1.4	157.4
L_UR_30_30	94	1	5	**4960.1** (4622.6 , 337.6)	5.3	244.1
L_UR_30_40	98	0	2	**6183.1** (6165.2 , 17.8)	1.4	345.5
L_UR_30_60	98	1	1	**9929.2** (9670.2 , 258.9)	1.9	758.1
L_UR_30_100	96	0	4	**15464.1** (14619.4 , 844.6)	3.7	3060.3
T_C_30_10	97	3	0	**1172.9** (1170.6 , 2.3)	7.8	14.8
T_C_30_20	85	1	14	**2058.4** (1789.6 , 268.4)	8.1	72.4
T_C_30_30	97	0	3	**2582.9** (2483.6 , 99.3)	1.9	233.1
T_C_30_40	94	2	4	**4213.2** (4079.8 , 133.3)	3.1	535.4
T_C_30_60	92	1	7	**6114.8** (5669.5 , 445.2)	2.3	810.1
T_C_30_100	97	1	2	**9589.1** (9424.9 , 164.1)	1.2	2185.7
T_RC_30_10	91	6	3	**1138.7** (1074.3 , 72.4)	1.6	12.9
T_RC_30_20	96	3	1	**2270.6** (2208.6 , 62.4)	2.1	112.4
T_RC_30_30	94	1	5	**3738.8** (3578.6 , 160.2)	1.9	196.2
T_RC_30_40	90	2	8	**5566.4** (5041.7 , 524.7)	8.6	581.3
T_RC_30_60	93	1	6	**6678.8** (6231.9 , 446.9)	5.9	805.7
T_RC_30_100	97	1	2	**10407.3** (10045.2 , 362.1)	2.6	2955.2
T_UR_30_10	95	0	5	**1216.1** (1151.2 , 64.9)	4.4	12.8
T_UR_30_20	97	0	3	**2828.3** (2763.9 , 64.4)	1.6	96.1
T_UR_30_30	91	1	8	**4109.7** (3736.7 , 372.9)	7.4	162.4
T_UR_30_40	97	0.5	2.5	**5704.1** (5626.9 , 77.1)	1.1	570.3
T_UR_30_60	94	1	5	**8159.8** (7732.6 , 427.1)	4.2	691.6
T_UR_30_100	95	1	4	**13893.2** (13207.2 , 686.1)	3.9	2170.8
Average SAA	93.6	1.1	5.3	**5708.6** (5439.6 , 269.2)	3.3	691.3

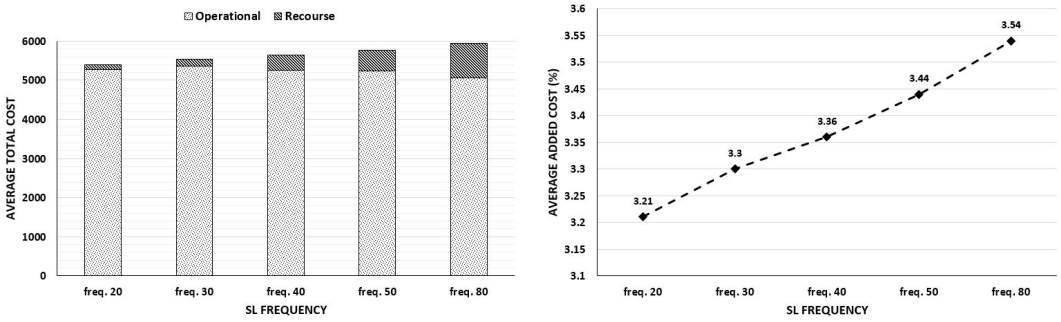
Table 6.: SAA results

that are used more frequently. Relatively, the average added costs slightly increase from 3.14% to 3.39% when passengers demand level goes from 2 to 8 (Figure 8b). These observations are important for two main reasons. First, the increasing transportation costs indicate that allowing PD robots to be transported with passengers through SLs might not always be efficient when passengers demand is high (e.g. morning and evening peak hours). In other words, this combination can prove most efficient during day hours when the probability of having free places in SLs is bigger. Second, the



(a) Average total costs (b) Average added costs

Figure 8.: Passengers demand analysis



(a) Average total costs (b) Average added costs

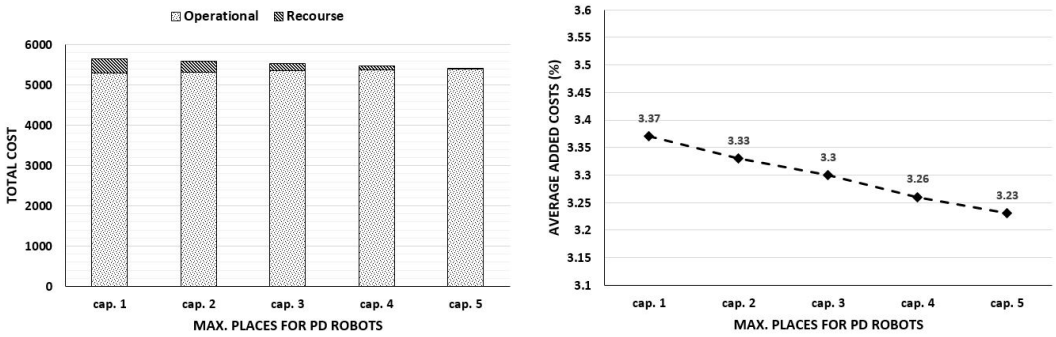
Figure 9.: SL frequency analysis

use of the outsourcing delivery service (*action*#3) will increase in peak hours as the subsequent service might also be fully charged with passengers. This means that road traffic can increase as more vehicles are circulating in the system to deliver freight requests that could not be transported using SL service. However, SL combined service still yields many benefits compared to existing freight delivery services, but these benefits can be maximized in off-peak hours.

Analyzing uncertainty with different SL frequencies

As aforementioned, we consider the SL departure frequency to be 30 time units. We investigate in this section the impact of SL frequency on the total transportation cost. For this purpose, we run the algorithm with SL frequency of 20, 30, 40, 50 and 80 time units and we take the average of ten runs of the algorithm for each SL frequency (Figure 9).

We observe that the total transportation costs increases as SL departures become less frequent. This can be explained by the fact that less frequent SL departures lead to more PD robots waiting at each transfer node which means a higher possibility of having SL capacity violations (Figure 9a). On the other hand, with a more frequent SL service (e.g. 20 time units), the total costs decreases as less recourse actions are needed. This can also be observed by looking at the average added costs with different SL frequencies (Figure 9b). While increasing the SL frequency to 20 time units can reduce the added costs (up to 0.09%) compared to the original ones, decreasing the



(a) Average total costs

(b) Average added costs

Figure 10.: SL capacity analysis

frequency can yield a slightly increasing added costs (up to 0.03%, 0.11% and 0.21% for 40, 50 and 80 SL frequencies respectively). However, increasing SL frequency might also lead to additional costs for SL operators as more shuttles are circulating in the system (e.g. energy, driver wages etc.). Although freight transportation costs can be decreased by making SLs more frequent, it might not be profitable for SL operators especially at passengers off-peak times. On the other hand, reducing SL frequency can also lead to many passengers left unserved at SL stations. To conclude, increasing, or decreasing, SL frequency need to take into account the varying levels of passengers demand.

Analyzing uncertainty with different SL capacities

As introduced earlier, we assumed that PD robots can take up to 3 places in shuttles. In this section, we investigate the effect of changing the maximum number of places allowed for PD robots on the total transportation costs. As such, we take the average of ten runs of the algorithm with up to 5 maximum places (Figure 10).

Looking at the obtained results, we observe that allowing more PD robots at each SL departure has a positive effect in terms of the total transportation costs and the average added costs. This positive effect is justified by a lowering of 0.07% and 0.1% on the added costs when up to 4 or 5 PD robots are allowed at each SL departure. This means that with an extra capacity for PD robots, stochastic solutions become cheaper and less capacity violations can be encountered. However, this might also have a negative effect on the number of PD robots that have to get off at an intermediate transfer node where passengers demand is high leading to many waiting PD robots at that node.

6. Conclusion

In this paper, a transportation service that combines passenger and freight flows has been studied. The associated optimization problem has been formulated as a pickup and delivery problem with time windows, scheduled lines (PDPTW-SL) and stochastic passengers demand. An MIP formulation along with ALNS-based heuristic approach have been introduced. For dealing with uncertainty, a sample average approximation method and a recourse algorithm have been developed. An extensive computational study to evaluate the performance of the proposed approaches and their different

components has been presented.

Results of testing instance with up to 60 freight requests showed that the proposed heuristic approach can return solutions that are within 0.6% of the optimal solutions. The analysis also revealed that an average of 3.3% extra costs can be observed when stochastic passengers demand is realized. These additional costs reflect the effect of uncertainty on the total transportation costs. Analyzing the impact of different SL frequencies and capacities, the results demonstrated the positive effect of increasing the frequency of SL departures and the maximum capacity for PD robots on the system.

Since we build our analysis on a set of assumptions that simplify the problem, there are still a number of challenges facing the deployment of such integrated transportation system. Here we outline some directions for future research: (1) We assumed in this paper that each PD robot can only serve one freight request at a time due to the complexity of the considered problem. A more realistic setting would be to allow multiple request pickup and delivery per PD robot trip. This gives rise to the challenge of coupling, or synchronizing, both pickup and delivery routing problems as the same PD robot performs them. (2) Another interesting direction would be to study the impact of such integrated service on passenger transportation on a daily horizon in which their demand varies during day hours. Finally, (3) Since we consider in this paper one source of uncertainty, which is passengers demand, it is also important to look at other sources of uncertainty like travel times. As PD robots are operating in an urban area, many external factors might affect their travel time and speed. We believe that this study helps in a better understanding of the potential deployment of such integrated systems, and thus, promote more research towards studying this emerging trend in city logistics and transportation.

Acknowledgement

This research work is carried out as part of the ANTHROPOLIS research project at the Technological Research Institute SystemX and is supported by public funding within the scope of the French Program "Investissements d'Avenir".

References

Archetti, C., Savelsbergh, M., & Speranza, G. (2016). The Vehicle Routing Problem with Occasional Drivers. *European Journal of Operational Research*, *254*(2), 472–480. Retrieved from http://dx.doi.org/10.1016/j.ejor.2016.03.049

Arslan, A., Agatz, N., Kroon, L. G., Zuidwijk, R. A. (2016). Crowdsourced Delivery - A Pickup and Delivery Problem with Ad-hoc Drivers. *SSRN Electronic Journal*, 1–29. Retrieved from http://www.ssrn.com/abstract=2726731.

Dayarian, I., & Savelsbergh, M. (2017). Crowdshipping and Same-day Delivery: Employing In-store Customers to Deliver Online Orders. *Optimization Online*, *07*, 42–61.

Fatnassi, E., Chaouachi, J., & Klibi, W. (2015). Planning and operating a shared goods and passengers on-demand rapid transit system for sustainable city-logistics. *Transportation Research Part B: Methodological*, *81*, 440–460. Retrieved from http://dx.doi.org/10.1016/j.trb.2015.07.016

Ghilas, V., Cordeau, J., & Demir, E. (2016). The Pickup and Delivery Problem with Time Windows and Scheduled Lines: Models and algorithms. *Transportation Science*, 1–30. Retrieved from http://dx.doi.org/10.1080/03155986.2016.1166793

Ghilas, V., Demir, E., & Van Woensel, T. (2016). An adaptive large neighborhood search heuristic for the Pickup and Delivery Problem with Time Windows and Scheduled Lines. *Computers and Operations Research*, *72*, 12–30. Retrieved from http://dx.doi.org/10.1016/j.cor.2016.01.018

Ghilas, V., Demir, E., & Van Woensel, T. (2016). A scenario-based planning for the pickup and delivery problem with time windows, scheduled lines and stochastic demands. *Transportation Research Part B: Methodological*, *91*, 34–51. Retrieved from http://dx.doi.org/10.1016/j.trb.2016.04.015

Kafle, N., Zou, B., & Lin J. (2017). Design and modeling of a crowdsource-enabled system for urban parcel relay and delivery. *Transportation Research Part B: Methodological*, *99*, 62–82. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S019126151630265X

Li, B., Krushinsky, D., Reijers, H. A., & Van Woensel, T. (2014). The Share-a-Ride Problem: People and parcels sharing taxis. *European Journal of Operational Research*, *238*(1), 31–40. Retrieved from http://dx.doi.org/10.1016/j.ejor.2014.03.003

Li, B., Krushinsky, D., Reijers, H. A., & Van Woensel, T. (2016). An adaptive large neighborhood search heuristic for the share-a-ride problem. *Computers and Operations Research*, *66*, 170–180. Retrieved from http://dx.doi.org/10.1016/j.cor.2015.08.008

Li, B., Krushinsky, D., Reijers, H. A., & Van Woensel, T. (2016). The Share-a-Ride problem with stochastic travel times and stochastic delivery locations. *Transportation Research Part C: Emerging Technologies*, *67*, 95–108. Retrieved from http://dx.doi.org/10.1016/j.trc.2016.01.014

Masson, R., Trentini, A., Lehuédé, F., Malhéné, N., Péton, O., & Tlahig, H. (2017). Optimization of a city logistics transportation system with mixed passengers and goods. *EURO Journal on Transportation and Logistics*, *6*(1), 81–109. Retrieved from http://link.springer.com/10.1007/s13676-015-0085-5

Mourad, A., Puchinger, J., & Chu, C. (2019). survey of models and algorithms for optimizing shared mobility. *Transportation Research Part B: Methodological*, *123*, 323–346. Retrieved from https://linkinghub.elsevier.com/retrieve/pii/S0191261518304776

Savelsbergh, M., & Van Woensel, T. (2016). 50th Anniversary Invited Article — City Logistics: Challenges and Opportunities. *Transportation Science*, *50*(2), 579–590. Retrieved from http://pubsonline.informs.org/doi/10.1287/trsc.2016.0675

Steadieseifi, M., Dellaert, N. P., Nuijten, W., Van Woensel, T., & Raoufi, R. (2014). Multimodal freight transportation planning: A literature review. *European Journal of Operational Research*, *233*(1), 1–15. Retrieved from http://dx.doi.org/10.1016/j.ejor.2013.06.055

Trentini, A., Masson, R., Lehuédé, F., & Malhéné, N. (2015). Optimization of a city logistics transportation system with mixed passengers and goods. *EURO J Transp Logist*, 1–23. retrieved from http://hal.archives-ouvertes.fr/hal-00859078

Verweij, B., Ahmed, S., Kleywegt, A., Nemhauser, G., & Shapiro, A. (2003). The Sample Average Approximation Method Applied to Stochastic Routing Problems: A Computational Study. *Computational Optimization and Applications*, *24*(1), 289–333. Retrieved from http://dx.doi.org/10.1023/A:1021814225969

Wang, Y., Zhang, D., Liu, Q., Shen, F., & Lee, L. H. (2016). Towards enhancing the last-mile delivery: An effective crowd-tasking model with scalable solutions. *Transportation Research Part E: Logistics and Transportation Review*, *93*, 279–293. Retrieved from http://dx.doi.org/10.1016/j.tre.2016.06.002

Appendix A

Algorithm 2 Algorithm for calculating the average recourse cost of a given routing solution

procedure CALCULATE-RECOURSE-COST(routing solution $\boldsymbol{\delta}$, set of scenarios ξ)
  initialize *recourse cost*: $E\,[\,Q\,(\,\boldsymbol{\delta}\,,\,\xi\,,\eta\,)\,] \leftarrow 0$
  let $c(\boldsymbol{\delta})$ be the routing costs of solution $\boldsymbol{\delta}$
  for each scenario s in ξ **do**
    for each transfer node t in $\mathcal{S}$ **do**
      let $\mathcal{W}_t$ be the set of PD robots waiting at transfer node t: $\mathcal{W}_t \subset \mathcal{P}'$
      for each scheduled departure $p^w_{t,t+1}$ at t **do**
        let $\mathcal{I}^w_t$ be the set of PD robots already in shuttle at $p^w_{t,t+1}$
        rank PD robots in $\mathcal{W}_t$ according to their requests earliest due dates
        rank PD robots in $\mathcal{I}^w_t$ according to their request's latest due dates
        if $|\mathcal{W}_t| > Q^w_{ij}$ **then**
          add PD robots to SL in order until Q^w_{ij} is reached
          for each excessive PD robot **do**
            if direct delivery from t to destination is possible **then**
              update *recourse cost* according to the extra traveled distance by PD robot
            else
              add PD robot to $\mathcal{W}_t$
        let Δ_t be the realized number of passengers waiting for service at t
        if $|\Delta_t| - Q^w_{ij} > 0$ **then**
          for each robot in $\mathcal{I}^w_t$ **do**
            if direct delivery from t to destination is possible **then**
              update *recourse cost* according to the extra traveled distance by PD robot
            else
              add PD robot to $\mathcal{W}_t$
        for each PD robot in $\mathcal{W}_t$ **do**
          if time window is violated **then**
            use outsourced vehicle to make the delivery
            update *recourse cost* according to the traveled distance by outsourced vehicle
            remove PD robot from $\mathcal{W}_t$
  return *recourse cost*

Appendix B

Algorithm 3 Sample Average Approximation (SAA) Algorithm

procedure SAMPLE-AVERAGE-APPROXIMATION(sample size $|\Omega|$, large set of scenarios Ω', number of iterations m)
    generate large set of scenarios Ω
    $m = 0$
    while $m < M$ **do**
        generate sample set ω
        solve corresponding SAA problem using ALNS to get routing solution $\boldsymbol{x^m}$
        Calculate-Recourse-Cost($\boldsymbol{\delta}$, Ω) $\rightarrow$ get *upper bound* for Ω
        calculate *lower bound* for ω by averaging objective values of previous iterations
        calculate SAA gap using *upper bound* and *lower bound*
        if tighter gap is found **then**
            best solution $\leftarrow$ found solution
        $m = m + 1$
    return *best solution*