CIRRELT-2025-23

A Double ALNS Metaheuristic for the Multi-commodity Location-Network Design Problem with Heterogeneous Vehicles

Francesco Contu
Teodor Gabriel Crainic
Massimo Di Francesco Enrico Gorgone

August 2025

A Double ALNS Metaheuristic for the Multi-commodity Location-Network Design Problem with Heterogeneous

Francesco Contu[3], Teodor Gabriel Crainic[1,2]*, Massimo Di Francesco Enrico Gorgone[3]

1. Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)
2. School of Management, Université du Québec à Montréal
3. Department of Mathematics and Computer Science University of Cagliari, Italy

Abstract. This work presents a decision support system for planning a consolidation-based urban distribution system, in which inbound freight packed in containers arrives at an inter-modal terminal. As the terminal lacks storage and transdock capabilities, containers must be transferred to satellites, where their contents are unloaded and redistributed into smaller vehicles for final delivery. The problem is approached from the perspective of an urban mobility manager, whose goal is to select satellite facilities and vehicle types, define vehicle routes and the flow of goods to final customers while optimizing transportation resource utilization. The system is modeled using a Mixed-Integer Lin-ear Programming (MILP) formulation. To address instances of realistic size, a solution approach based on two integrated Adaptive Large Neighborhood Search (ALNS) meta-heuristics is proposed. In each iteration of the first ALNS, the selection of satellites and the assignment of containers to them are updated. These configurations are then evaluated by the second ALNS, which determines the associated second-tier routing and flow decisions. Extensive computational experiments are conducted to assess the algorithm's performance and to analyze the practical benefits of the proposed approach, using the city of Cagliari (Italy) as a real-world case study.

Keywords: Location, Network Design, multi-commodity flows, Adaptive Large Neighborhood.

Acknowledgments: This study was carried out within the MOST – Sustainable Mobility National Research Center and received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.4 – D.D. 1033 17/06/2022, CN00000023). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them. This publication was produced by Francesco Contu while attending the PhD program in Mathematics and Computer Science at the University of Cagliari, Cycle XXXVIII, with the support of a scholarship co-financed by the Ministerial Decree no 352 of 9th April 2022 based on the NRPP – funded by the European Union – Next Generation EU- Mission 4 "Education and Research", Component 2 "From Research to Business", Investment 3.3. and by the company Spiva. While working on the project, the second author held the UQAM Chair on Intelligent Logistics and Transportation Systems Planning, and was Adjunct Professor, Department of Computer Science and Operations Research, Universite´ de Montréal. We gratefully acknowledge the financial support provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Discovery Grant program, as well as that of Fonds de recherche du Québec through their infrastructure grants.

---

* Corresponding author: teodorgabriel.crainic@cirrelt.net

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2025

1 Introduction

The freight transport sector plays a vital role in supporting economic growth, but it also poses significant challenges to urban environments, such as traffic congestion, air pollution, and noise. In light of evolving political, economic, and societal pressures, transportation and logistics systems are increasingly expected to become more sustainable, without sacrificing efficiency. In response to these demands, innovative business models — collectively known as City Logistics — have emerged. City Logistics seeks to reduce the negative externalities of urban freight transport while fostering social and economic development. These systems are based on freight consolidation strategies and require coordinated planning across strategic, tactical, and operational levels.

In this work, we address the tactical planning of a Two-Tier City Logistics (2T-CL) system. First introduced in [3], in these systems inbound freight is moved to City Distribution Centers (CDCs), which are typically located on the outskirts of urban areas. From these CDCs, freight is transported by large vehicles to *satellites* — small transdock facilities positioned within the inner part of the city. Final deliveries to customers are then carried out by city freighters, which are smaller, environmentally friendly, and cost-effective. The resulting two-tier structure presents complex optimization challenges.

This paper examines a 2T-CL system incorporating an *intermodal facility* (e.g., a port) that serves as the entry point for inbound containers, each carrying commodities (or pallets) destined for multiple customers. Unlike traditional systems, where City Distribution Centers (CDCs) are located on the city's outskirts, the intermodal facility is situated within the urban area. Due to space constraints and the need to swiftly clear the docks for incoming shipments, the deconsolidation of containers is not permitted at the facility. Instead, containers are transported to *satellites* — transdock-type facilities where goods are unpacked, sorted, and dispatched to final destinations using smaller, environmentally sustainable vehicles. To the best of our knowledge, the tactical planning aspect of such a system has not yet been investigated in the literature.

Our objective is to address existing knowledge gaps by introducing a novel problem setting that captures the complexities inherent in both demand and supply dynamics. Specifically, we define two distinct service types, each corresponding to one tier of the logistics system: the first type involves the allocation of resources (vehicles, capacities, infrastructure) for transporting containers from the intermodal terminal to satellites; the second type concerns the movement of pallets from satellites to final customers using suitable delivery means. The main aspects investigated include: (1) the selection of satellites to which inbound containers are assigned; (2) the specification of specialized vehicles for each tier — large vehicles for container transport in the first tier, and smaller, more sustainable vehicles for pallet delivery in the second tier; (3) the design of second-tier delivery routes and the associated flow of commodities; and (4) the incorporation of system constraints, such as a heterogeneous vehicle fleet and satellite capacity limitations, expressed in terms of pallet, vehicle, and container handling capabilities.

The main contributions of this paper are as follows: (1) the definition of a novel problem setting for 2T-CL systems, in which transportation services must be jointly selected in both tiers; (2) the formulation of a Location-Network Design model to support the tactical planning of the proposed 2T-CL system; (3) the development of a solution approach based on two Adaptive Large Neighborhood Search (ALNS) algorithms tailored to the problem structure; and (4) the demonstration of the method's effectiveness through extensive computational experiments and managerial insights derived from a real-world case study.

The remainder of the paper is structured as follows. In **Section 2** the problem setting is described. An overview of the literature on two-tiered systems is provided in **Section 3**. In **Section 4** we propose a Mixed-Integer Linear Programming (MILP) formulation for the problem at hand. In **Section 5** the ALNS-based algorithm is presented. The results of our experiment are shown in **Section 6**. Finally, in **Section 7** we summarize the main findings, discuss the implications of our results, and indicate potential directions for future research.

2 Problem setting

The section begins by introducing the 2T-CL system under study, detailing its core components and operational mechanisms. Emphasis is placed on the novel features of the proposed approach in comparison with existing literature. The section concludes with an overview of the tactical planning challenges associated with the system.

2.1 The system

This study addresses the tactical planning problem within a 2T-CL system, schematically illustrated in **Figure 1**. The 2T-CL network comprises two types of facilities—the *intermodal terminal* and the *satellites*—as well as customer zones, an external zone, and a set of connections representing different transportation services. Satellites are typically located on the outskirts of the city, whereas customer zones are situated within the urban core, where access for large vehicles is often restricted due to traffic regulations. Connections linking the intermodal terminal to the satellites define the first tier, while those connecting satellites to customer zones constitute the second tier.

We focus on inbound demand, where customer zones represent demand destinations and the external zone serves as the point of origin. Freight must be transported from this external origin to customers located within the city. A single unit of demand, referred to as a pallet, is assumed to be the standard unit handled in the 2T-CL system. Demand is thus quantified in terms of the number of pallets to be transported from origin to destination. To optimize long-haul transportation toward the intermodal facility, pallets destined for different customer zones are consolidated into containers. Upon

arrival at the intermodal terminal, containers must be transferred to a satellite facility for unloading. Notably, the demand for a specific destination may be distributed across multiple containers. However, due to the rear-loading nature of containers and the operational difficulty of rearranging pallets, each container is assumed to be unpacked at a single satellite. In the second tier, the demand for a given customer may be split among multiple delivery vehicles.

Pallet demand is fulfilled through two types of transportation services, one for each tier of the system. First-tier services are responsible for transporting containers from the intermodal terminal (e.g., port) to satellite facilities, providing the necessary vehicles, capacities, and related resources. Second-tier services handle the distribution of pallets from satellites to customer zones within the city.

In daily operations, containers are transferred from the intermodal facility to satellites using first-tier urban vehicles, also referred to as Container-Compatible Vehicles (CCVs). Satellites function as transfer points between first-tier and second-tier operations, typically operating under trans-dock principles, with minimal or no storage capacity. First-tier and second-tier vehicles may be simultaneously present at a satellite, potentially competing for limited docking or parking space, as well as for cargo handling resources. Second-tier deliveries to customer zones are carried out by city freighters or Pallet-Compatible Vehicles (PCVs), which are better suited to urban environments due to their smaller size and maneuverability. We assume heterogeneous vehicle fleets in both tiers with respect to cost structures. While PCVs may vary in pallet-carrying capacity, all CCVs are assumed to transport only one container at a time.

Finally, all containers are considered available at the intermodal facility at the start of the workday. PCV routes are assumed to be open, i.e., vehicles do not necessarily return to their starting points.

2.2 Tactical planning

We assume that the City Logistics (CL) system is planned and managed by a single decision-maker, although vehicles and satellite facilities may be owned and operated by multiple public and private stakeholders engaged in a collaborative framework involving cost and capacity sharing. Within this context, the tactical plan defines the selection of active satellites, their operational roles, and the set of vehicles required to support the selected services in both tiers.

This tactical plan is developed for a short-term period, referred to as the schedule length (e.g., one day), and is intended to be applied repeatedly over a longer planning horizon (e.g., six months). It is assumed that the core elements of the plan — such as the selected satellites and vehicle configurations — remain fixed throughout the planning horizon. Operational adjustments to accommodate fluctuations in daily demand are handled by modifying routing decisions in real time, which falls outside the scope of this

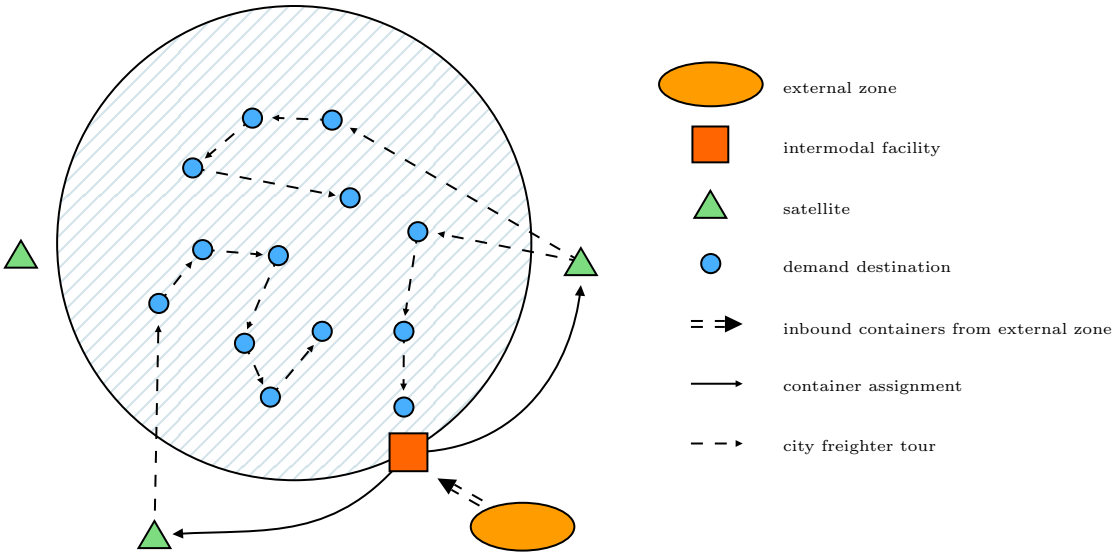

Figure 1: Two-tier city logistics system.

study.

The proposed Location-Network Design formulation addresses the key tactical planning decisions for the 2T-CL system, including: (1) selection of satellite facilities capable of handling all inbound containers from the intermodal terminal; (2) assignment of each container to a specific satellite; (3) selection of Container-Compatible Vehicles (CCVs) to transport containers from the terminal to satellites; (4) selection of Pallet-Compatible Vehicles (PCVs) to fulfill last-mile deliveries; (5) assignment of each selected PCV to a satellite; (6) routing of each PCV from the assigned satellite to the customer zones; and (7) determination of the number of pallets delivered by each PCV to each customer.

The objective is to identify the most cost-effective configuration that satisfies demand requirements using the available sets of satellites and vehicles. The generalized cost function incorporates all relevant operational costs associated with the selected transportation and facility services.

3 Literature review

This section begins with a brief overview of the concept of 2T-CL systems. We then shift our focus to network design problems within this context. Finally, we highlight the main

research gaps identified in the existing literature.

3.1 Two-tier City Logistics

The concept of Two-Tier City Logistics (2T-CL) systems was first introduced by [3], who investigated the optimal location of satellite facilities within urban freight distribution networks. City logistics systems generally comprise one or more layers of facilities designed to consolidate freight flows through the coordination of transportation, transshipment, and storage activities. These facilities are typically categorized as either City Distribution Centers (CDCs) or satellites. CDCs primarily receive inbound freight for storage, sorting, and consolidation, thereby enabling coordinated and efficient deliveries within urban areas. Satellites, in turn, operate as intermediate facilities within a multi-tier structure, linking first-tier CDCs with lower-tier facilities and end customers. The study in [3] demonstrated that implementing a 2T-CL system can significantly reduce the distance traveled by large trucks within the city, allowing final deliveries to be performed by smaller, more sustainable vehicles.

A general modeling framework for tactical planning in 2T-CL systems was later proposed by [4], who addressed the temporal characteristics of demand and the need for coordinated vehicle operations across both tiers. Their study considered a simplified setting with a single mode of road-based transport per tier and focused exclusively on inbound demand, with each demand instance pre-assigned to a City Distribution Center (CDC).

Subsequent research explored related problems such as the Two-Echelon Vehicle Routing Problem (2E-VRP) and Location Routing Problems [2], even if limited research was carried out to capture the multi-commodity feature of these problems ([1], [7]).

As for Scheduled Service Network Design (SSND) problems (e.g., [6]), the work by [8], incorporated demand uncertainty into tactical planning through a two-stage stochastic programming approach, and analyzed various strategies for adapting plans in response to real-time demand fluctuations. Building on this, [9] refined the SSND framework to better reflect the specific requirements of 2T-CL systems.

[11] contributed to the literature by introducing more realistic assumptions, such as the simultaneous consideration of inbound and outbound flows and multiple transportation modes. Their approach relied on an SSND model solved via Benders decomposition. In parallel, [10] investigated the integration of public transport services into two-tier logistics systems to support freight movement from peripheral areas to satellite facilities.

More recently, [12] examined the conditions under which single-tier or two-tier city logistics structures are preferable, offering managerial insights for urban logistics design.

It is important to note that existing studies assume the presence of City Distribution Centers that allow for initial consolidation of freight flows from external regions. In

contrast, the setting addressed in this work considers an intermodal facility that lacks consolidation capabilities, thereby requiring containers to be directly transferred to satellites for unpacking and distribution.

3.2 Service Network Design

This study addresses decisions related to the selection of satellites and vehicles, as well as the design of the service network. Accordingly, we review the relevant literature in the domains of facility location and service network design. However, unlike traditional approaches that emphasize resource acquisition — such as selecting vehicle fleets or facility sites — our focus lies on selecting the services provided by available vehicles and satellites. Notably, most existing studies concentrate on strategic or integrated strategic–tactical decision-making frameworks, whereas our work emphasizes tactical planning within a fixed infrastructure context.

In [13], the authors provide a unified framework for problems that integrate facility location and network design decisions.

[14] propose an optimization model that jointly determines the locations of uncapacitated facilities and the design of a transportation network used to move customer shipments from their origins to the nearest open facility. The objective is to minimize the total cost, including facility opening, service operation, and shipment routing. Building upon this work, [15] extend the model to incorporate facility capacity constraints, introducing a combined facility location and capacitated network design problem. However, their formulation does not account for the operational resources required to support the transportation network, such as vehicle availability or service provisioning.

Early studies (e.g., [17], [18]) investigated strategic planning problems that impose a design-balance constraint, requiring that the number of services entering and leaving a terminal at any given time be equal. These models typically assume a single type of resource, with each service supported by one unit of that resource. As a result, the design-balance constraint ensures temporal and spatial equilibrium of resources across the network. However, incorporating such constraints significantly increases the complexity of the problem, as rounding-based solution techniques often lead to infeasible solutions [26]. In response, several tailored solution methods have been developed to effectively handle design balance, including those proposed in [19], [20], and [21].

A problem integrating both strategic and tactical planning was addressed by [25] and [24]. In their approach, tactical decisions involve the selection of transportation services to operate and the routing of freight through the resulting service and terminal network, where services are supported by vehicles assigned to specific satellites. Strategic decisions, on the other hand, encompass the location of facilities and the selection and allocation of vehicles. Although these decision levels are often treated separately, the authors propose solving them jointly, allowing strategic resource acquisition and allocation choices to be

informed by accurate estimates of their impact on operational transportation costs. In contrast, the present work interprets satellite selection as a service selection decision, rather than a strategic facility location problem.

[22] examine a combined facility location and network design problem with multiple types of capacitated links. They propose a mixed-integer programming model to simultaneously optimize facility locations and the associated transportation network, aiming to minimize total transportation and operational costs. The model includes various link types, each characterized by distinct capacities and fixed costs; however, vehicle selection is not considered in their framework.

In a related contribution, [23] propose an integrated tactical planning model combining hub location decisions with the design of a frequency-based service network. Notably, their formulation assumes that no fixed costs are incurred for using, not using, or opening hubs, which contrasts with many practical applications where such costs are significant.

3.3 Gaps in the literature

To the best of our knowledge, this is the first work to address a service network design problem for the tactical planning of a Two-Tier City Logistics (2T-CL) system that integrates an intermodal facility without deconsolidation capabilities—unlike conventional City Distribution Centers (CDCs). Uniquely, our model simultaneously considers the selection of services, vehicles, and satellite facilities, capturing operational constraints and interdependencies that have been overlooked in prior research.

4 Modeling

The physical distribution network is represented by a directed graph $G = (N, A)$, where N denotes the set of nodes and A the set of arcs. The node set N includes the port node p, the set of satellites S and the set of customer locations Γ. The arc set A can be partitioned into 2 sets: A_1, representing the first-tier arcs from the port $p \in N$ to each satellite $s \in S$; and A_2, representing second-tier arcs connecting each satellite $s \in S$ and each customer $\gamma \in \Gamma$ to any other customer within the network.

Let K_1 and K_2 denote the sets of Container-Compatible Vehicles (CCVs) and Pallet-Compatible Vehicles (PCVs), respectively. Let C be the set of containers and Π_c^γ the number of pallets for the customer $\gamma \in \Gamma$ carried by container $c \in C$. The former notation is also illustrated in **Table 1**. Accordingly, the total number of pallets in container $c \in C$ is given by $\Pi_c := \sum_{\gamma \in \Gamma} \Pi_c^\gamma, \forall c \in C$ and the total number of pallets destined for customer $\gamma \in \Gamma$ is given by $\Pi^\gamma := \sum_{c \in C} \Pi_c^\gamma, \forall \gamma \in \Gamma$.

Figure 2 illustrates a simplified configuration involving three containers c_1, c_2 and c_3,

Symbol	Description
p	Seaport
C	Set of containers
S	Set of satellites
Γ	Set of customers
K_1	Set of CCVs
K_2	Set of PCVs
$G = (N, A)$	A directed graph
N	$\{p\} \cup S \cup \Gamma$
A	$A_1 \cup A_2$
A_1	$\{(p, s) : s \in S\}$
A_2	$\{(i, j)\, i \in S \cup \Gamma, j \in \Gamma,\, i \neq j\}$

Table 1: Sets

three satellites s_1, s_2 and s_3, and three customers γ_1, γ_2 and γ_3. The total demand for customer γ_2 is 7 pallets: 3 are packed in container c_1 and 4 in container c_3. If both containers are assigned to the same satellite, the corresponding pallets can be either consolidated onto a single PCV for delivery or split among multiple PCVs. The decision variables and the data in the MILP formulation are defined in **Table 2**.

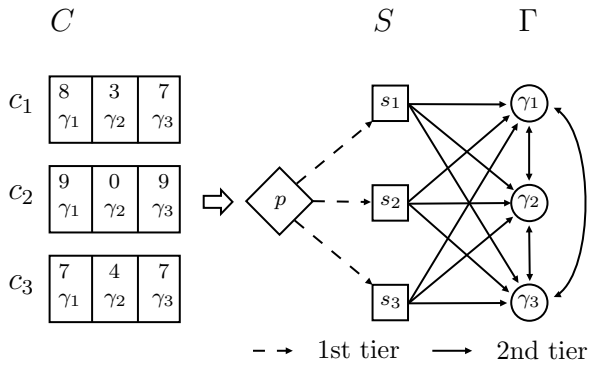

Figure 2: Example of a two tiered system in a maritime city

Symbol	Description
u_s	Maximum number of *containers* that satellite $s \in S$ can handle
v_s	Upper limit on the number of *PCVs* assigned to satellite $s \in S$
π_s^1	Maximum number of *pallets* that satellite $s \in S$ can handle
π_k^2	Maximum number of *pallets* that PCV $k \in K_2$ can carry
f_s	Fixed selection cost of satellite $s \in S$
β_s	Unit cost per pallet served by $s \in S$
λ_{ks}^1	Fixed operation and handling cost of CCV $k \in K_1$ at satellite $s \in S$
λ_{ks}^2	Fixed operation and handling cost of PCV $k \in K_2$ at satellite $s \in S$
$\mu_{k(i,j)}$	Operation cost of PCV $k \in K_2$ traversing arc $(i,j) \in A_2$
$\alpha_{k\gamma(i,j)}$	Unitary transportation cost per pallet for customer $\gamma \in \Gamma$ carried by PCV $k \in K_2$ along arc $(i,j) \in A_2$
y_s	Satellite selection variable: it takes value 1 if satellite $s \in S$ is selected, 0 otherwise
x_{ksc}^1	Container transportation variable: it takes value 1 if container $c \in C$ is moved by CCV $k \in K_1$ from the port to satellite $s \in S$, 0 otherwise
$w_{k(i,j)}^2$	Vehicle routing variable: it takes value 1 if PCV $k \in K_2$ traverses arc $(i,j) \in A_2$, 0 otherwise
$x_{k\gamma(i,j)}^2$	Number of pallets shipped along arc $(i,j) \in A_2$ to customer $\gamma \in \Gamma$ by PCV $k \in K_2$, 0 otherwise

Table 2: Data and decision variables

The problem can be formulated as follows.

$$
\begin{aligned}
z^* = \min & \sum_{s\in S}\left(f_s y_s + \beta_s \sum_{k\in K_1}\sum_{c\in C}\Pi_c x_{ksc}^1\right) \\
& + \sum_{k\in K_1}\sum_{s\in S}\sum_{c\in C}\lambda_{ks}^1 x_{ksc}^1 + \sum_{k\in K_2}\sum_{s\in S}\sum_{j\in\Gamma}\lambda_{ks}^2 w_{k(s,j)} \\
& + \sum_{k\in K_2}\sum_{(i,j)\in A_2}\mu_{k(i,j)} w_{k(i,j)} \\
& + \sum_{k\in K_2}\sum_{\gamma\in\Gamma}\sum_{(i,j)\in A_2}\alpha_{k\gamma(i,j)} x_{k\gamma(i,j)}^2 && (1) \\
& \sum_{k\in K_1}\sum_{s\in S} x_{ksc}^1 = 1 && \forall c \in C \quad (2) \\
& \sum_{s\in S}\sum_{c\in C} x_{ksc}^1 \le 1 && \forall k \in K_1 \quad (3) \\
& \sum_{s\in S}\sum_{j\in\Gamma} w_{k(s,j)} \le 1 && \forall k \in K_2 \quad (4)
\end{aligned}
$$

$$\sum_{k\in K_1}\sum_{c\in C} x^1_{ksc} \le u_s y_s \qquad \forall s \in S \tag{5}$$

$$\sum_{k\in K_2}\sum_{j\in \Gamma} w_{k(s,j)} \le v_s y_s \qquad \forall s \in S \tag{6}$$

$$\sum_{k\in K_1}\sum_{c\in C} \Pi_c x^1_{ksc} \le \pi^1_s y_s \qquad \forall s \in S \tag{7}$$

$$\sum_{\gamma\in\Gamma} x^2_{k\gamma(i,j)} \le \pi^2_k w_{k(i,j)} \qquad \begin{array}{r}\forall k \in K_2,\\ \forall (i,j) \in A_2\end{array} \tag{8}$$

$$\sum_{(j,l)\in A_2} w_{k(j,l)} \le \sum_{(i,j)\in A_2} w_{k(i,j)} \qquad \begin{array}{r}\forall k \in K_2,\\ \forall j \in \Gamma\end{array} \tag{9}$$

$$\sum_{k\in K_2}\sum_{(s,j)\in A_2} x^2_{k\gamma(s,j)} - \sum_{k\in K_1}\sum_{c\in C} \Pi^\gamma_c x^1_{ksc} = 0 \qquad \begin{array}{r}\forall s \in S,\\ \forall \gamma \in \Gamma\end{array} \tag{10}$$

$$\sum_{k\in K_2}\sum_{(i,\gamma)\in A_2} x^2_{k\gamma(i,\gamma)} - \Pi^\gamma = 0 \qquad \forall \gamma \in \Gamma \tag{11}$$

$$\sum_{k\in K_2}\sum_{(\gamma,i)\in A_2} x^2_{k\gamma(\gamma,i)} = 0 \qquad \forall \gamma \in \Gamma \tag{12}$$

$$\sum_{(i,\gamma')\in A_2} x^2_{k\gamma(i,\gamma')} - \sum_{(\gamma',j)\in A_2} x^2_{k\gamma(\gamma',j)} = 0 \qquad \begin{array}{r}\forall k \in K_2,\\ \forall \gamma, \gamma' \in \Gamma, \gamma' \neq \gamma\end{array} \tag{13}$$

$$y_s \in \{0,1\} \qquad \forall s \in S \tag{14}$$

$$x^1_{ksc} \in \{0,1\} \qquad \begin{array}{r}\forall k \in K_1,\ \forall s \in S,\\ \forall c \in C\end{array} \tag{15}$$

$$w_{k(i,j)} \in \{0,1\} \qquad \begin{array}{r}\forall k \in K_2,\\ \forall (i,j) \in A_2\end{array} \tag{16}$$

$$x^2_{k\gamma(i,j)} \in \mathbb{N} \qquad \begin{array}{r}\forall k \in K_2,\ \forall \gamma\ \in \Gamma,\\ \forall (i,j) \in A_2\end{array} \tag{17}$$

In (1) we minimize the total cost, which includes: satellite activation costs; vehicle selection and assignment costs; costs for pallet handling at satellites; and vehicle routing costs in the second tier. Moreover, it accounts for the transportation cost paid by customers for moving pallets across both the first and second tiers. Constraints (2) ensure that each container $c \in C$ is collected from the port and assigned to a satellite $s \in S$

by a CCV $k \in K_1$. Constraints (3) and (4) enforce that each CCV and PCV is assigned to exactly one satellite, respectively. Constraints (5), (6) and (7) impose upper bounds on the number of containers, PCVs and pallets assigned to each satellite $s \in S$, ensuring these do not exceed their respective capacities u_s, v_s and π_s^1, provided the satellite is selected. Constraints (8) link flow and routing variables and ensure that the pallet load carried by each PCV $k \in K_2$ does not exceed its capacity. Constraints (9) enforce the flow conservation for each PCV at every node in the 2^{nd} tier or the end of its route, if the last customer is visited. Constraints (10) guarantee that the pallets requested by each customer $\gamma \in \Gamma$ pass through the selected satellite $s \in S$. Constraints (11) and (12) ensure that each customer $\gamma \in \Gamma$ receives exactly the number of requested pallets and that no pallets for a customer leave this customer after delivery. Constraints (13) represent the flow balance of pallets for customer $\gamma \in \Gamma$ at intermediate customer nodes $\gamma' \in \Gamma$ visited before the final destination. Finally, constraints from (14) to (17) define the domain for the decision variables.

5 Solution method

In this section, we present an algorithm based on Adaptive Large Neighborhood Search (ALNS) to solve the proposed problem. ALNS builds upon the Large Neighborhood Search (LNS) framework originally introduced by Shaw [28], where the solution space is explored through an iterative destroy-and-repair mechanism. At each iteration, a destroy operator partially removes elements from the current solution, and a repair operator reconstructs a complete solution by reinserting the removed components. This approach enables the exploration of large neighborhoods, thereby enhancing the ability to escape local optima [29]. ALNS extends the LNS framework by incorporating an adaptive mechanism that dynamically selects the most promising destroy and repair operators during the search process, based on their historical performance [30].

ALNS has been effectively applied to various extensions of the Vehicle Routing Problem (VRP), including the Two-Echelon VRP (2E-VRP) and the Location-Routing Problem (LRP). For example, Grangier et al. [31] developed ALNS-based approaches for solving the 2E-VRP, Schiffer et al. [32] employed ALNS for the LRP and Hemmelmayr et al. [5] proposed an ALNS heuristic for both 2E-VRP and LRP. A comprehensive review of ALNS algorithms was recently provided by Mara et al. [33].

The key idea of the proposed solution approach is to decompose the overall problem into two interrelated components, referred to as the main problem and the sub-problem. The main problem, denoted as $Prob_1$, focuses on exploring feasible configurations of satellite selection and container assignments. For each candidate configuration generated in $Prob_1$, the associated second-tier decisions are determined by solving a corresponding sub-problem that is consistent with the chosen configuration. The overall procedure proceeds through the following steps:

1. A configuration s_1 of satellites and container assignments is determined.
2. The sub-problem, denoted as $Prob_2(s_1)$, is solved within the framework of $Prob_1$ to determine the distribution of pallets to customers, based on the configuration defined in s_1. Let s_2 represent the corresponding solution to this sub-problem.
3. The solution to the complete problem (s_1, s_2) is built from s_1 and s_2 and compared against the best incumbent solution (s_1^*, s_2^*).

A high-level flowchart of the solution method is presented in **Figure 3**. The method follows a nested loop structure. In each iteration of the outer loop, a new configuration s_1' is generated and an initial feasible solution s_2 is built for the corresponding sub-problem $Prob_2(s_1')$. In the inner loop, s_2 is iteratively modified by generating new candidate solutions s_2', which are accepted or rejected based on their quality and possibly stored as the best solution s_2^b to $Prob_2(s_1')$. When the stopping criterion of the inner loop is satisfied, the combined solution (s_1', s_2^b) is compared with the current solution (s_1, s_2) and the best overall solution (s_1^*, s_2^*), updating them if improvements are observed. The process repeats until the global termination criterion is reached.

The configurations s_1 and the corresponding solutions s_2 are obtained by two ALNS metaheuristics, referred to as $ALNS_1$ and $ALNS_2$, which are described in **Section 5.1** and **Section 5.7**, respectively.

5.1 $ALNS_1$

We present the algorithm $ALNS_1$, which is designed to modify satellite configurations and the assignment of containers to satellites. At each iteration of $ALNS_1$, the current solution s_1 is transformed into a new configuration s'_1, and the associated sub-problem $Prob_2(s'_1)$ is solved to evaluate the resulting solution.

In $ALNS_1$, the destroy-and-repair process operates on s_1 as follows: a number n_C of containers is removed from their assigned satellites by a destroy operator, placed into a temporary pool, and then reassigned to any open satellite by a repair operator. The value of n_C is selected at each iteration within a predefined range $[n_C^{\min}, n_C^{\max}]$, which is an algorithmic hyperparameter. The longer the number of iterations since the last accepted solution, the larger the value of n_C, thus promoting diversification (see **Section 6.2**).

Some destroy operators are designed to explicitly open or close satellites, thereby exploring a wider solution space, while others perform more localized modifications, preserving the current satellite configuration. Destroy-repair operator pairs are selected using a roulette-wheel mechanism based on their past performance. Each pair is associated with a score that reflects its past effectiveness, influencing its selection probability in subsequent iterations (see **Section 5.8** for further details).

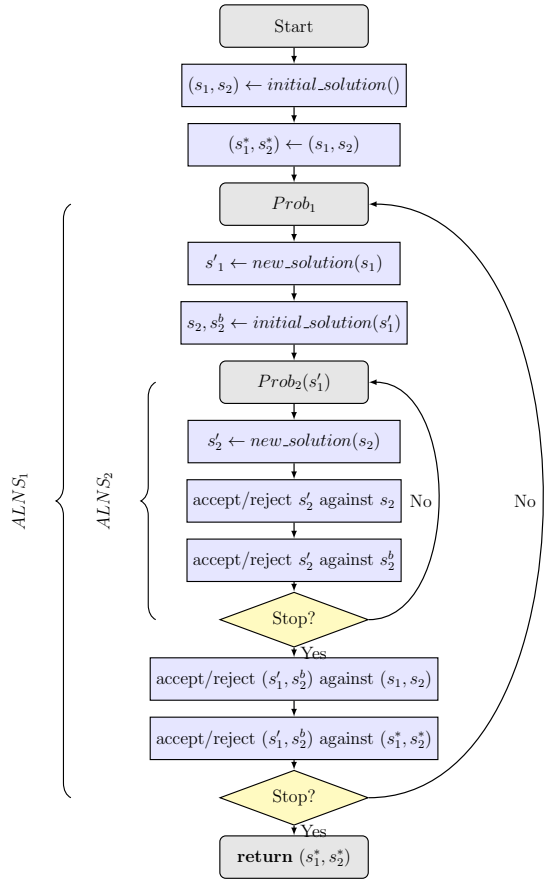

Figure 3: General flowchart of the solution method

The algorithm proceeds as outlined in **Algorithm 1**. An initial solution (s_1, s_2) for the overall problem is generated using a greedy heuristic (see line 1 and refer to Section 5.3 for details). At each iteration, a pair of destroy and repair operators is selected to modify the configuration s_1. Specifically, a destroy operator removes container assignments from satellites, and a repair operator reassigns them to available satellites.

Two categories of destroy operators are employed in $ALNS_1$: small-impact destroy operators, denoted by D^S (see line 9 and Section 5.5), which only remove container assignments without altering the satellite configuration; large-impact destroy operators, denoted by D^L (see line 7 and Section 5.4), which may also open or close satellites, thereby resulting in a relevant change of the overall configuration. These are applied every τ iterations since the last accepted solution, to force diversification when the search stagnates. When D^L is executed, the number of containers removed is set to $n_C = n_C^{\max}$.

Repair operators, denoted by R, are responsible for reassigning the removed containers to the current set of open satellites. The destroy–repair procedure results in a new configuration s'_1, and the corresponding sub-problem $Prob_2(s'_1)$ is solved (line 13, function $solve\_Prob_2(s'_1)$, yielding a solution (s'_1, s'_2).

The acceptance of this new solution is determined with respect to the current one (s_1, s_2). If a large-impact destroy operator was used, the solution is automatically accepted (free pass). Otherwise, acceptance occurs only if the new solution strictly improves upon the current one. This mechanism balances diversification and intensification by allowing broader exploration when necessary, while focusing on refinement under stable configurations.

The iterative process continues until a predefined time limit is reached or a specified number of consecutive non-improving iterations occur, at which point the best-found solution (s_1^*, s_2^*) is returned.

Algorithm 1 : $ALNS_1$

$(s_1, s_2) \leftarrow initial\_solution()$
$(s_1^*, s_2^*) \leftarrow (s_1, s_2)$
$initialize\_scores(\sigma)$
$i \leftarrow 0$
repeat
  if $i == \tau$ **then**
    $N^- \leftarrow select\_destroy\_operator(D^L, \sigma)$
  else
    $N^- \leftarrow select\_destroy\_operator(D^S, \sigma)$
  end if
  $N^+ \leftarrow select\_repair\_operator(R, \sigma)$
  $s'_1 \leftarrow destroy\_repair(s_1, N^-, N^+)$
  $s'_2 \leftarrow solve\_Prob_2(s'_1)$
  if $i == \tau$ **or** $f(s'_1, s'_2) < f(s_1, s_2)$ **then**
    $s_1 \leftarrow s'_1$
    $s_2 \leftarrow s'_2$
    $i \leftarrow 0$
    if $f(s_1, s_2) < f(s_1^*, s_2^*)$ **then**
      $s_1^* \leftarrow s_1$
      $s_2^* \leftarrow s_2$
    end if
  else
    $i \leftarrow i + 1$
  end if
  $update\_scores(\sigma)$
until stopping condition is met
return (s_1^*, s_2^*)

Note that:

- The operators in D^L are applied less frequently (every time τ iterations since the

last accepted solution).

- The operators in D^S are applied at every iteration, except when a large-impact operator from D^L is selected.

In line 13 of $ALNS_1$, the sub-problem $Prob_2(s_1)$ is a multi-commodity multi-depot network design problem. To tackle this sub-problem efficiently, we introduce a dedicated ALNS metaheuristic, referred to as $ALNS_2$, which is detailed in **Section 5.7**.

5.2 Penalized cost function

During the search process, infeasible solutions may also be explored due to potential violations of satellite capacity constraints related to containers, pallets, and PCVs (i.e., constraints 5, 6, and 7 in **Section 4**). To account for such infeasibilities, a penalty is imposed for each violated constraint and added to the objective value of the new incumbent solution. Specifically, for a given solution (s_1, s_2), a weighted penalty function $\bar{f}(s_1, s_2)$ is used to account for these violations:

$$f(s_1, s_2) = z(s_1, s_2) + \omega \sum_{s \in S} \bar{u}(s, s_1) f_s + \epsilon \sum_{s \in S} \bar{\pi}^1(s, s_1) f_s + \psi \sum_{s \in S} \bar{v}(s, s_1) f_s \tag{18}$$

where $z(s_1, s_2)$ is the cost associated with solution (s_1, s_2), while $\bar{u}(s, s_1)$, $\bar{\pi}^1(s, s_1)$, and $\bar{v}(s, s_1)$ quantify the violations of, respectively, the container (CCV), pallet, and PCV capacity constraints at satellite $s \in S$ under configuration s_1. The scalar coefficients ω, ϵ, and ψ are penalty weights (or hyperparameters) used to modulate the impact of each type of violation in the penalized objective function, and must be properly calibrated.

5.3 Initial solution

This solution method requires a feasible initial solution to initialize the optimization process. To this end, we propose a simple constructive heuristic capable of generating a feasible solution for the full problem. The procedure starts with all satellites initially closed. At each iteration, a container is assigned to the satellite that yields the minimum incremental cost, computed as the sum of (i) the cost of serving the pallets within the container and (ii) the cost of opening the satellite, if currently closed. When a satellite is selected and not yet active, it is opened. Once all containers are assigned, the corresponding second-tier routing is determined by sequentially serving customers in a random order using randomly chosen PCVs without violating their capacity, until the entire demand is satisfied.

5.4 Large-impact destroy operators

Large-impact destroy operators modify the satellite configuration by opening or closing one or more satellites and, when applicable, remove n_C containers affected by the resulting configuration change. The removed containers are transferred to a container pool for subsequent reassignment by a repair operator.

- **Satellite closing:** An open satellite is closed and all its containers and routes are removed. This operator can be selected only if at least one satellite is open. Five different selection mechanisms are defined, each resulting in a different operator:
  - **Most penalized (penalized_c)**: The satellite with the highest penalty is chosen.
  - **Random (random_c)**: This operator randomly selects a satellite from the set of currently open facilities and marks it for closure.
  - **Least Used (emptiest_c)**: This operator selects the satellite that handles the fewest number of pallets. The rationale is the assignment of containers to fewer satellites by closing the one with the least number of pallets.
  - **Worst (worst_c)**: The satellite with the highest removal gain is selected. Specifically, the removal gain is computed as the difference between the total cost of the current solution and the cost after removing the satellite, normalized by the number of pallets processed at that satellite. This metric reflects the cost-efficiency of each satellite in serving pallets, and the least efficient satellite — i.e., the one with the highest cost per pallet — is chosen for removal.
  - **Most-selected (most_selected_c)**: The satellite with the highest cumulative number of container assignments across all iterations of the algorithm is selected. The underlying rationale is to discourage over-reliance on frequently chosen satellites by promoting diversification in the solution space.
- **Satellite opening:** A closed satellite is selected for opening, and the n_C containers that have been assigned to it the fewest number of times throughout the search are removed from their current satellites and placed in the container pool. If no closed satellites are available, this operator is deactivated. Although a satellite is opened, the operator is categorized as a destroy operator, as it disrupts the current solution by unassigning containers, which must subsequently be reassigned during the repair phase. Five distinct selection strategies are implemented, each defining a different variant of this operator:
  - **Largest (largest_o)**: The satellite with the highest pallet handling capacity π_s^1 is selected for opening;
  - **Random (random_o)**: A random satellite is selected from the set of currently closed satellites;

  - **Smallest (smallest_o)**: The satellite with lowest pallet handling capacity π_s^1 is selected for opening;
  - **Best (best_o)**: The satellite with the lowest estimated cost is selected. This estimate is calculated as the average per-pallet transportation cost from the port to the customers via the satellite, computed and updated throughout the algorithm's iterations;
  - **Least-selected (least_selected_o)**: The satellite with the lowest cumulative number of container assignments across all iterations of the algorithm is selected. This strategy aims to prioritize underutilized satellites for the potential reassignment of containers.

- **Satellite swap:** An open satellite is closed while simultaneously opening a previously closed satellite. All containers and routes associated with the closed satellite are transferred to the newly opened one. Consequently, this operator inherently performs the repair phase by reassigning containers from the pool during the transfer. Despite incorporating repair actions, due to its substantial impact on the overall solution, it is still classified as a large-impact operator. Five distinct selection mechanisms are defined, each corresponding to a different variant of this operator:

  - **Most penalized (penalized_s)**: The open satellite with the highest penalty is closed, while the closed satellite with the highest pallet capacity is opened;
  - **Random (random_s)**: Two satellites are randomly selected;
  - **Least used (emptiest_s)**: The open satellite managing the fewest pallets and the closed satellite with the smallest pallet capacity are selected; the former is closed, and the latter is opened;
  - **Worst (worst_s)**: The satellite with the highest removal gain is closed, and the satellite with the best cost estimate is opened;
  - **Most-selected (most_selected_s)**: The satellites with the highest and lowest total container assignments across all iterations are selected: the former is closed, and the latter is opened.

5.5 Small-impact destroy operators

Small-impact destroy operators remove (when applicable) n_C containers without altering the satellite configuration. The removed containers are placed into a container pool for subsequent reassignment.

- **Container removal**: This operator directly selects the containers to be removed from their current satellites. Five distinct container selection mechanisms are defined, each giving rise to a different operator:

  - *Most penalized (penalized_r)*: Containers are removed from the satellites with the highest penalty;
  - *Random removal (random_r)*: Containers are randomly selected;
  - *Least-used removal (emptiest_r)*: Containers are removed from satellites that currently serve the fewest number of pallets;
  - *Worst removal (worst_r)*: Containers are removed from the satellites with the highest removal gain;
  - *Most-selected removal (most_selected_r)*: Containers are removed from the satellites with the highest number of assignments.

- **Container removal from satellite**: this operator selects a satellite and removes at most n_C random containers from this satellite. Five different satellite selection mechanisms are defined, each resulting in a different operator:
  - *Most penalized (penalized_rs)*: This operator selects the satellite with the highest penalty;
  - *Random removal (random_rs)*: This operator selects a random satellite;
  - *Least-used removal (emptiest_rs)*: This operator selects the satellite with the lowest number of pallets;
  - *Worst removal (worst_rs)*: The satellite with the highest removal gain is selected;
  - *Most-selected removal (most_selected_rs)*: The satellite with the highest cumulative number of container assignments across all iterations of the algorithm is selected.

5.6 Repair operators

The repair operators reassign containers from the pool to open satellites. Containers are selected in random order, and after each assignment, the solution is immediately updated. Three distinct assignment mechanisms are defined, each corresponding to a different repair operator:

- **Most-used insertion (most_used_i)**: Assignment to the satellite with the highest number of pallets;
- **Best insertion (best_i)**: Assignment to the best estimated satellite;
- **Least-selected insertion (least_selected_i)**: Assignment to the least selected satellite.

For all these operators, "noised" and "forbidden" versions are considered. In the first case, a random multiplicative noise in the range [0.85,1.15] is applied to the evaluation metric used to select the satellite for container assignment. This stochastic perturbation promotes exploration by preventing overly deterministic behavior and encouraging alternative choices. In the second case, the satellite to which a container was previously assigned is excluded from the candidate set during reassignment. This mechanism enforces diversification by discouraging the re-selection of recently used configurations, thereby promoting exploration of new assignment patterns.

5.7 $ALNS_2$

For each solution s'_1 generated through a destroy–repair pair in $ALNS_1$, the associated sub-problem $Prob_2(s'_1)$ must be solved to construct a complete solution (s'_1, s'_2) to the overall problem. Given a fixed configuration s_1 defining the set of active satellites and the assignment of containers to them, $Prob_2(s_1)$ determines the distribution of pallets to customers. Specifically, it returns vehicle routing decisions (i.e., customer sequences for each vehicle) and the associated pallet flows, all consistent with the configuration imposed by s_1.

To solve $Prob_2(s'1)$ efficiently, we propose a dedicated ALNS-based metaheuristic, denoted as $ALNS_2$, and described in **Algorithm 2**. At each iteration, a destroy operator removes a subset of $n\Gamma$ customers from their assigned routes and places them into a customer pool. Subsequently, a repair operator reinserts the customers into the solution. The value of n_Γ is dynamically selected at each iteration from the interval $[n^{\min}\Gamma, n^{\max}\Gamma]$, which is a hyperparameter of the algorithm. This value increases with the number of consecutive non-improving iterations (see **Section 6.2**).

This search is restricted to feasible solutions only. When a customer is removed, the associated pallet flow is also eliminated to maintain consistency. As in $ALNS_1$, destroy–repair operator pairs are selected using a roulette-wheel mechanism, guided by a scoring system based on past performance. Higher scores increase the likelihood of selecting more effective operator pairs in subsequent iterations (see **Section 5.8** for details).

Whenever a new configuration s'_1 is generated, an initial feasible solution s_2 for $Prob_2(s'_1)$ is constructed as follows: customers affected by changes in the satellite configuration or in container-to-satellite assignments are randomly assigned to available PCVs with sufficient remaining capacity. If required, PCVs are redistributed across satellites to ensure that each satellite has adequate transportation capacity. In cases where the construction of a feasible initial solution is not possible due to an insufficient number of PCVs, the configuration s'_1 is deemed infeasible and discarded, and the associated search in $ALNS_2$ is not performed.

During this search process, a destroy operator removes a subset of customers from

their current PCV routes, and a repair operator subsequently reinserts them (see line 7 of **Algorithm 2**). This destroy–repair process generates a new solution s'_2, which replaces the current solution s_2 if its objective value lies within a threshold $\theta\%$ of the best solution found so far, denoted by $s^b{}_2$ (see line 9 of the algorithm).

Algorithm 2 : $ALNS_2$

1: s_2^b, $s_2 \leftarrow initial\_solution\_Prob_2(s'_1)$
2: $initialize\_scores(\sigma)$
3: **if** s_2^b is feasible **then**
4: **repeat**
5: $N_2^- \leftarrow select\_destroy\_operator(D_2, \sigma)$
6: $N_2^+ \leftarrow select\_repair\_operator(R_2, \sigma)$
7: $s'_2 \leftarrow destroy\_repair(s_2, N_2^-, N_2^+)$
8: **if** $f_2(s'_2) < f_2(s_2^b) * (1 + \theta)$ **then**
9: $s_2 \leftarrow s'_2$
10: **if** $f_2(s'_2) < f_2(s_2^b)$ **then**
11: $s_2^b \leftarrow s'_2$
12: **end if**
13: **end if**
14: $update\_scores(\sigma)$
15: **until** stopping condition for $ALNS_2$ is met
16: **end if**
17: **return** s_2^b

The destroy and repair operators of $ALNS_2$ are described in **Section 5.7.1** and **Section 5.7.2**, respectively.

5.7.1 Destroy operators

Destroy operators in $ALNS_2$ remove, when applicable, n_Γ customers from their current routes, along with the associated pallet flows. The destroy operators are formally defined as follows.

- **Random Route removal (route_r)**: a route is randomly selected, and all customers assigned to it are removed from the solution along with their corresponding pallet flows;
- **Partial customer removal**: Customers are selected and removed from one of their assigned routes (remember that split deliveries may allow a customer to appear in multiple routes). Three distinct versions of this operator are implemented, each differing in the criteria used to select both the customers and the specific routes from which they are removed:

  - *Random (random_r)*: randomly selected customers are removed from one of their routes, with the specific route also chosen at random.
  - *Worst (worst_r)*: customers are removed from the route that yields the highest removal gain, defined as the difference in the total solution cost before and after the customer's removal.
  - *Smallest (smallest_r)*: customers are removed from the route delivering the fewest number of pallets.

- **Customer removal**: this operator removes a subset of selected customers from all their routes. Three variants are proposed, differing in the method used to select customers:
  - *Random (random_rs)*: customers are randomly selected.
  - *Worst (worst_rs)*: customers with the highest removal gain are selected.
  - *Most fragmented (most_fragmented_rs)*: customers with the highest number of visits by PCVs are selected.

5.7.2 Repair operators

The repair operators assign PCVs to deliver pallets to customers in the pool, respecting the residual capacity of the selected PCVs. A customer remains in the pool until its entire pallet demand is fulfilled. Customers in the pool are processed in random order. Five distinct methods for selecting PCVs are proposed, each defining a different repair operator:

- Greedy delivery insertion (greedy_i): for each customer, select the PCV minimizing the customer's insertion cost.
- Best fit delivery insertion (best_fit_i): for each customer, select the PCV with the smallest residual capacity to accommodate the residual demand of the customer.
- Best delivery insertion (best_i): for each customer, the PCV minimizing the estimated insertion cost is selected.
- Least selected delivery insertion (least_selected_i): for each customer, select the PCV that has served that customer the fewest number of times.
- Largest delivery insertion (largest_i): for each customer, the largest PCV is selected.

Even in this case, a "noised" version of the operators is considered as well.

5.8 Selection of operators

At each iteration of both ALNS algorithms, operator pairs are selected using a roulette wheel mechanism, where selection probabilities are based on scores assigned to individual *destroy-repair* pairs. All scores are initialized to a common value σ at the start of the algorithm and are updated during its execution, based on the quality of incumbent solutions produced by the destroy-repair process. Higher scores reflect more effective operator pairs, increasing their likelihood of being selected in subsequent iterations.

In $ALNS_1$, two separate roulette wheels are used: one for large-impact pairs and another for small-impact pairs. This separation ensures that pairs from different categories are selected independently. Large-impact destroy operators, which cause significant changes to the solution, are applied every τ consecutive iterations without improvement, while small-impact operators are applied during all other iterations. The score associated with each operator pair i is denoted by ρ_i, and its selection probability ϕ_i is calculated as the ratio of ρ_i to the sum of all scores within the corresponding roulette wheel:

$$\phi_i := \begin{cases} \rho_i / \sum_{j \in N_L^1} \rho_j & \text{if } i \in N_L^1 \\ \rho_i / \sum_{j \in N_S^1} \rho_j & \text{if } i \in N_S^1 \end{cases} \text{ for } ALNS_1 \tag{19}$$

$$\phi_i := \rho_i / \sum_{j \in N^2} \rho_j \text{ if } i \in N^2 \qquad \text{for } ALNS_2 \tag{20}$$

Given the pair of destroy-repair operators i and the incumbent solution s', $\sigma_{s'}$ represents the score obtained by the pair i that generated the incumbent solution s'. This score depends on whether the incumbent solution is accepted or not:

$$\sigma_{s'} := \begin{cases} \sigma^+ & \text{if } s' \text{ is the new best solution} \\ \sigma & \text{if } s' \text{ is an accepted solution} \\ \sigma^- & \text{otherwise} \end{cases} \tag{21}$$

Finally, the score ρ_i is updated as the weighted average between the previous score and the score $\sigma_{s'}$:

$$\rho_i := \rho_i(1 - \zeta) + \sigma_{s'}\zeta \tag{22}$$

The scores of all pairs range in the interval $[\sigma^-, \sigma^+]$; setting σ^- with a positive value ensures that the least performing pairs are not totally excluded and have the chance to improve their score in the following iterations of the algorithm. σ^-, σ, σ^+, ζ are hyper-parameters of the algorithm. Usually, one expects $0 \leq \sigma^- \leq \sigma \leq \sigma^+$ and $0 \leq \zeta \leq 1$.

6 Experimentation

We conduct computational experiments to tune parameter values and identify effective configurations for the ALNS algorithm, which are then benchmarked against a MILP solver. Additionally, we compare the performance of our proposed system with that of a traditional distribution system, where shippers predefine their partners, including container-to-satellite assignments and vehicle selections.

The algorithm was implemented in Microsoft Visual C++ 2022, compiled with the Visual C++ compiler, and executed on a 2.9 GHz AMD Ryzen 7 4800H processor. To solve the proposed MILP model, we employed IBM ILOG CPLEX Concert Technology for C++ (version 22.1.1). **Section 6.1** presents the instance set used in the experiments. The parameter calibration procedure is outlined in **Section 6.2**. In **Section 6.3**, we report a comparison between our algorithm and the CPLEX branch-and-cut solver. Finally, **Section 6.4** evaluates the performance of the 2T-CL system relative to the traditional distribution system.

6.1 Instance description

We generate a set of instances based on a real-world scenario centered around the city of Cagliari, Italy. Customer locations within the city are randomly selected from among commercial sites, while transportation-related facilities — such as satellites and vehicle depots — are chosen from actual logistics hubs located on the outskirts. Real driving distances are computed for each arc of the graph formed by the selected nodes.

Each container is assumed to carry 18 pallets, and customer pallet demands are set to 3, 6, or 9 units. Customers may be served via split deliveries, allowing their pallets to be transported in different containers. Cost parameters are derived from actual logistics contracts. Fixed selection costs for PCVs, CCVs, and satellites are generated within ranges observed in the analyzed contracts. The unit cross-docking cost per pallet is negatively correlated with satellite size, reflecting the expectation that larger satellites offer greater efficiency in unloading operations.

Vehicle operating costs are set within the ranges of [2.00, 2.20] per kilometer for CCVs and [1.80, 2.00] for PCVs. Average per-pallet, per-kilometer transportation costs are estimated from the data of a real logistics provider and are perturbed using a random multiplicative coefficient in the range $[0.85, 1.15]$ to reflect variability across different commodities. Additional random noise within the same range is applied to all generated costs to simulate broader data variability.

Two sets of instances were created: 45 small instances featuring 3 satellites, 3–5 containers, and 6–30 customers; and 30 large instances with 10–16 satellites, 25–75 containers, and 50–300 customers. All instances are available upon request.

6.2 Parameter calibration

Several parameters of the ALNS algorithm require tuning. To identify suitable values, we conducted a calibration experiment on a subset of 15 selected small instances, referred to as the calibration set. For each instance, multiple parameter configurations were tested, with each configuration evaluated over 15 independent runs, each limited to a maximum runtime of one hour. The final configuration was selected based on the best average performance, measured by the gap to the best bound obtained by the CPLEX solver within the same time limit.

Given the high dimensionality of the parameter space, an exhaustive search was deemed computationally impractical. Therefore, the calibration process was structured into sequential steps, each focusing on a subset of parameters while keeping the others fixed. Parameters optimized in one step were held constant in subsequent steps. The initial values used prior to calibration are presented in **Table 3**. As the process progresses, parameters are updated at each step. **Table 4** summarizes the final calibrated values, along with parameter descriptions.

The first step of the calibration focuses on tuning the parameters $[n^{\min}C, n^{\max}C]$ and $[n^{\min}\Gamma, n^{\max}\Gamma]$, which govern the level of destruction in both ALNS algorithms. The destruction level, when used, increases proportionally with the number of consecutive iterations without improvement. These parameters define the minimum and maximum destruction rates that a destroy operator can apply, expressed as percentages of the solution affected. A limited set of candidate value pairs was considered for each parameter: $[10\%, 30\%]$, $[10\%, 50\%]$, $[10\%, 70\%]$, $[20\%, 50\%]$, $[20\%, 70\%]$, and $[30\%, 50\%]$.

All possible combinations of these candidate pairs were tested for $[n^{\min}C, n^{\max}C]$ and $[n^{\min}\Gamma, n^{\max}\Gamma]$. Following this step, the parameters τ, M_1, and M_2 were calibrated. Here, τ represents the number of consecutive iterations without an accepted solution before triggering the application of large-impact destroy operators. The parameters M_1 and M_2 define the maximum number of iterations allowed for $ALNS_1$ and $ALNS_2$, respectively. A constraint was imposed to fix the product $M_1 \cdot M_2 = 500,000$, in order to maintain consistent total computational effort across different configurations.

Several combinations of τ, M_1, and M_2 were evaluated to identify a balanced trade-off between solution quality and computational time:

1. $\tau = 10, M_1 = 500, M_2 = 1000,$
2. $\tau = 20, M_1 = 500, M_2 = 1000,$
3. $\tau = 10, M_1 = 1000, M_2 = 500,$
4. $\tau = 20, M_1 = 1000, M_2 = 500,$
5. $\tau = 50, M_1 = 1000, M_2 = 500,$

6. $\tau = 20, M_1 = 2000, M_2 = 250$

7. $\tau = 50, M_1 = 2000, M_2 = 250$.

Next, the parameter θ, which defines the threshold of acceptability relative to the best solution found by $ALNS_2$, was calibrated. The tested values for θ were 0%, 1%, 2%, and 5%.

Following this, the parameters governing the score assignment to destroy-repair operator pairs — namely ζ, σ^-, σ, and σ^+ — were calibrated. The parameter ζ, which determines the weight assigned to recent performance in score updates, was fixed at 0.5. The following value sets were evaluated for σ^-, σ, and σ^+: $[0, 1, 2]$, $[0, 1, 4]$, and $[0, 1, 1.5]$.

Lastly, the parameters of the penalized objective function — ω, ϵ, and ψ — were assigned the same value within each configuration. The values tested for these parameters were: 1.25, 1.50, 2.00, and 2.25.

Parameter	Description	Value
τ	max iterations satellite configuration	10
n_C	destruction percentage range in $ALNS_1$	$[10\%, 50\%]$
M_1	max iterations without improvement $ALNS_1$	2000
θ	acceptance threshold $ALNS_2$	5%
n_Γ	destruction percentage range in $ALNS_2$	$[10\%, 50\%]$
M_2	max iterations without improvement $ALNS_2$	250
ω	CCV capacity penalty weight	1.50
ϵ	pallet capacity penalty weight	1.50
ψ	PCV capacity penalty weight	1.50
ζ	scores update weight	0.5
σ^-	rejected solution score	0.5
σ	initial score / accepted solution score	1.0
σ^+	best solution score	2.0

Table 3: Parameter setting before the calibration phase

For both ALNS variants, we conducted an additional calibration phase focused on evaluating the effectiveness of the proposed operators, based on the best-performing parameter configuration. This analysis was performed on the same calibration set of instances, with each instance solved three times under a time limit of one hour per run. For each run and for each destroy-repair operator pair, we recorded the average computational time required to perform the operation and the average number of times the pair contributed to finding a new best solution.

Based on these metrics, an overall ranking of operator pairs was established. Each pair was evaluated using two criteria: a *speed score*, reflecting its average computational efficiency, and a *quality score*, representing its average contribution to discovering im-

Parameter	Description	Value
τ	max iterations for satellite configuration	20
n_C	destruction percentage range in $ALNS_1$	$[10\%, 50\%]$
M_1	max iterations without improvement $ALNS_1$	1000
θ	acceptance threshold $ALNS_2$	2%
n_Γ	destruction percentage range in $ALNS_2$	$[10\%, 50\%]$
M_2	max iterations without improvement $ALNS_2$	500
ω	CCV capacity penalty weight	1.25
ϵ	pallet capacity penalty weight	1.25
ψ	PCV capacity penalty weight	1.25
ζ	scores update weight	0.5
σ^-	rejected solution score	0.5
σ	initial score / accepted solution score	1.0
σ^+	best solution score	2.0

Table 4: Parameter setting after the calibration phase

proved solutions. Both scores were averaged across all instances in the calibration set. The top operator pairs in this ranking were selected for inclusion in the final algorithm.

Table 5 presents the top 20 operator pairs for both ALNS variants. The **quality** and **speed** columns report the respective *quality score* and *speed score* for each pair, while the **overall** column indicates their average score.

$ALNS_1$					$ALNS_2$				
destroy	**repair**	**overall**	**speed**	**quality**	**destroy**	**repair**	**overall**	**speed**	**quality**
worst_rs	best_in	0,39	0,62	0,36	route_r	best_fit_i	0,57	0,88	0,40
worst_rs	most_used_in	0,39	0,55	0,43	route_r	smallest_i	0,59	0,88	0,43
random_r	most_used_in	0,39	0,47	0,46	route_r	largest_i	0,60	0,88	0,43
worst_rs	best_if	0,40	0,84	0,26	route_r	best_fit_in	0,61	0,85	0,48
random_rs	best_in	0,40	0,63	0,31	smallest_rs	greedy_in	0,63	0,82	0,51
random_r	most_used_i	0,40	0,70	0,37	smallest_r	greedy_in	0,63	0,74	0,55
worst_rs	least_selected_i	0,40	0,80	0,26	smallest_rs	greedy_i	0,63	0,84	0,51
worst_r	least_selected_if	0,40	0,74	0,29	route_r	smallest_in	0,64	0,85	0,50
penalized_r	best_if	0,41	0,68	0,43	route_r	greedy_in	0,64	0,89	0,50
penalized_r	best_i	0,41	0,66	0,41	smallest_r	greedy_i	0,64	0,76	0,55
penalized_r	most_used_i	0,42	0,70	0,43	route_r	greedy_i	0,64	0,90	0,50
random_s	random_s	0,42	0,76	0,34	route_r	largest_in	0,65	0,85	0,52
worst_r	best_if	0,42	0,74	0,32	random_r	greedy_in	0,66	0,86	0,52
penalized_s	penalized_s	0,42	0,71	0,43	random_rs	greedy_in	0,66	0,82	0,54
worst_c	best_if	0,42	0,68	0,43	route_r	best_in	0,66	0,86	0,53
penalized_rs	best_i	0,45	0,70	0,45	route_r	best_i	0,67	0,88	0,52
random_rs	fullest_i	0,45	0,85	0,40	random_r	greedy_i	0,67	0,88	0,53
most_selected_rs	best_i	0,45	0,66	0,43	random_rs	greedy_i	0,67	0,84	0,55
worst_rs	most_used_if	0,46	0,76	0,38	fragmented_rs	greedy_in	0,74	0,77	0,74
penalized_r	least_selected_if	0,48	0,68	0,51	fragmented_rs	greedy_i	0,76	0,80	0,75

Table 5: Top 20 pairs of operators sorted by overall score

Table 6 presents a comparison, on the calibration set, between the branch-and-cut algorithm implemented in the CPLEX MILP solver and our algorithm, with results

averaged over 15 runs. Three different configurations are considered, labeled as *top*-20, *top*-10, and $full$ in the table. The *top*-20 and *top*-10 configurations include the 20 and 10 highest-ranked destroy-repair operator pairs, respectively, as identified in **Table 5**. The $full$ configuration includes all available operator pairs.

In both the *top*-20 and *top*-10 configurations, the highest-ranked pair containing an opening operator was explicitly added to $ALNS_1$ to ensure that both satellite opening and closing functionalities were available.

In **Table 6**, the column **avg. gap (%)** reports the average optimality gap between the solutions obtained by each algorithm and the best lower bounds identified by the MILP solver. The column **avg. T(s)** reports the average computational time required.

Algorithm	**avg. gap(%)**	**avg. T(s)**
$top-20$	1,54	21,74
$top-10$	1,64	22,55
$full$	1,65	19,24
$MILP$	7,06	2039,15

Table 6: Average results after the selection of operators

For the remainder of the experimental analysis, we adopt the *top*-20 configuration, as it achieved the best performance on the calibration set in terms of average optimality gap with respect to the best lower bound obtained by the MILP solver.

6.3 Computational results

Table 7 presents a comparison between two variants of the *top*-20 ALNS configuration and the MILP solver, focusing on the set of small instances. Results for the ALNS algorithms are averaged over 15 independent runs. In the first variant, denoted as $ALNS$-1000, the parameters max_it_1 and max_it_2 are set to 1000 and 500, respectively. In the second variant, $ALNS$-500, these parameters are set to 500 and 250. As such, $ALNS$-1000 is expected to yield higher-quality solutions, while $ALNS$-500 is designed to reduce computational time.

Table 7 reports the characteristics of each instance, including the number of satellites (S), containers (C), customers (Γ), CCVs (K_1), and PCVs (K_2). The column *gap* represents the optimality gap between the obtained solutions and the best lower bounds determined by the MILP solver. For the ALNS algorithms, the columns **avg gap** and **min gap** report the average and minimum gaps, respectively. The column T indicates the total runtime (in seconds), while T^* denotes the time at which the best solution was found during each run.

Table 8 summarizes these results by reporting average values across all small instances in the test set.

The $ALNS$-1000 variant demonstrates strong performance in terms of both solution quality and computational efficiency. Across the 45 small instances, it achieved optimal solutions in 9 cases, improved upon the MILP solver's results in 16 instances — primarily among the largest — and returned slightly worse solutions in 20 cases. Notably, $ALNS$-1000 consistently required less computational time to reach its best solution compared to the MILP solver. In cases where $ALNS$-1000 produced suboptimal solutions, the deviations from the MILP results were minimal.

The $ALNS$-500 variant, while offering faster runtimes, generally produced solutions slightly worse than those of $ALNS$-1000. However, it still outperformed the MILP solver in some instances, highlighting its potential as a viable, time-efficient alternative.

	$DATA$					$MILP$			$ALNS-1000$				$ALNS-500$			
#	S	C	Γ	K_1	K_2	$gap(\%)$	$T^*(s)$	$T(s)$	$avg\ gap(\%)$	$min\ gap(\%)$	$T^*(s)$	$avg\ T(s)$	$avg\ gap(\%)$	$min\ gap(\%)$	$avg\ T^*(s)$	$avg\ T(s)$
1	3	3	6	4	16	0	0,57	2,85	0,04	0,04	0,36	4,37	0,04	0,04	0,29	1,31
2	3	3	6	5	15	0	0,93	3,41	0	0	0,20	3,71	0,29	0,19	0,52	1,38
3	3	3	6	5	15	0	0,30	4,37	0	0	1,45	5,16	0,79	0,30	0,71	1,53
4	3	3	6	5	11	0	0,41	3,04	0,03	0,03	1,57	5,54	0,27	0,04	0,71	1,64
5	3	3	6	5	14	0	2,94	3	0,12	0,08	3,33	8,44	1,03	0,63	0,89	2,06
6	3	3	9	4	11	0	13,39	54,80	0,10	0,01	3,05	7,74	0,39	0,01	0,71	1,85
7	3	3	9	4	16	0	5,18	10,48	0	0	0,80	6,40	0,41	0,03	0,69	1,78
8	3	3	9	4	18	0	6,63	89,36	0	0	3,71	9,21	1,05	0,63	0,66	2,13
9	3	3	9	5	12	0	1,88	6,31	0	0	0,90	6,37	1,24	0,91	1,16	2,29
10	3	3	9	4	16	0	0,89	2,08	0	0	1,49	7,30	0,82	0,54	1,49	2,63
11	3	3	18	4	13	0,47	1676,35	TL	0,44	0,42	9,00	17,34	1,96	1,70	2,15	4,22
12	3	3	18	4	13	1,53	1338,98	TL	1,50	1,37	5,86	14,97	3,86	3,15	1,99	3,94
13	3	3	18	5	12	5,12	1493,60	TL	6,74	4,56	4,25	11,51	10,48	5,12	0,86	2,57
14	3	3	18	5	14	0,17	3317,87	TL	0,16	0,16	6,63	16,62	1,05	0,27	1,62	3,65
15	3	3	18	4	10	0,41	3297,39	TL	0,32	0,31	6,64	15,18	0,89	0,44	1,32	3,26
16	3	4	8	6	13	0	10,87	30,56	0,16	0,06	2,22	7,62	0,36	0,16	0,84	2,02
17	3	4	8	6	21	0	14,59	17,96	0,21	0,20	2,35	8,27	0,80	0,62	1,38	2,77
18	3	4	8	5	18	0	8,26	8,33	0	0	0,34	6,92	1,00	0,74	0,80	2,20
19	3	4	8	6	18	0	4,91	5,05	0	0	4,77	11,13	1,09	0,44	0,97	2,41
20	3	4	8	5	18	0	98,24	98,83	0	0	2,03	7,91	0,42	0,42	0,46	1,83
21	3	4	12	6	20	0,89	2484,79	TL	0,96	0,94	8,78	20,20	2,20	1,79	1,54	3,49
22	3	4	12	6	17	0	179,69	206,88	0,02	0,02	3,67	10,55	0,90	0,72	1,15	2,84
23	3	4	12	6	18	0,61	18,78	TL	0,53	0,52	6,50	13,41	1,48	1,30	0,98	2,58
24	3	4	12	6	16	0	147,38	153,31	0,10	0,02	2,51	9,58	0,14	0,12	1,90	3,73
25	3	4	12	5	18	0	61,97	151,41	0,10	0,10	3,38	11,19	0,51	0,46	1,38	3,14
26	3	4	24	6	15	0,61	3278,98	TL	0,44	0,42	13,02	27,70	1,61	1,36	2,97	6,01
27	3	4	24	6	16	1,50	3392,55	TL	1,09	1,03	16,24	33,93	2,00	1,73	1,75	4,85
28	3	4	24	5	17	3,33	985,33	TL	1,58	1,49	10,47	25,30	2,22	1,86	2,55	5,68
29	3	4	24	6	18	5,43	3062,82	TL	2,43	2,16	13,99	31,58	4,70	3,79	2,24	5,15
30	3	4	24	6	17	6,83	1267,39	TL	3,17	2,62	9,02	25,60	4,21	3,61	2,33	5,06
31	3	5	10	6	19	0,79	3447,85	TL	1,17	0,84	10,17	19,37	5,00	2,34	1,79	3,74
32	3	5	10	8	21	0	148,32	180,52	0,88	0,06	7,08	17,23	3,34	1,74	1,62	3,72
33	3	5	10	6	21	0	518,37	566,71	0,50	0,22	7,39	18,70	2,70	1,78	1,60	3,82
34	3	5	10	7	23	1,51	1061,64	TL	3,47	3,05	10,38	21,86	5,32	4,50	1,72	4,08
35	3	5	10	7	22	0,65	2151,74	TL	0,76	0,67	11,10	21,90	5,45	3,42	1,40	3,49
36	3	5	15	7	27	0,68	3059,64	TL	0,81	0,73	15,71	36,41	2,51	2,24	2,00	5,27
37	3	5	15	7	23	3,70	304,11	TL	3,50	3,40	18,67	35,94	7,11	5,98	2,21	5,19
38	4	5	15	6	20	2,58	2116,52	TL	10,06	4,49	12,69	26,98	10,69	4,73	1,61	4,29
39	3	5	15	8	23	0,71	639,46	TL	0,72	0,72	10,60	26,29	2,72	1,28	1,56	4,39
40	3	5	15	8	25	2,26	2403,02	TL	6,57	2,59	16,57	34,11	13,89	8,01	1,42	4,18
41	3	5	30	7	21	NF	0,43	TL	2,59	2,25	31,58	65,04	4,55	3,85	3,04	7,70
42	3	5	30	7	23	87,89	3597,48	TL	5,90	5,63	30,88	67,17	10,30	9,04	4,19	9,19
43	3	5	30	8	22	27,42	3589,31	TL	4,00	3,86	38,94	75,90	5,60	5,10	3,90	9,30
44	3	5	30	8	24	12,24	3594,15	TL	7,79	6,63	28,92	63,05	9,55	8,35	3,15	7,65
45	3	5	30	7	20	50,78	3572,05	TL	0,62	0,44	23,94	57,64	1,89	1,42	4,39	9,62

Table 7: Comparison of result between ALNS and MILP solver on small instances. The best gaps are underlined.

Table 9 compares the performance of $ALNS$-500 and $ALNS$-1000 on the set of large instances, with results averaged over 15 runs. Let obj denote the average objective

Algorithm	*avg gap*(%)	*avg* $T(s)$	*avg* $T^*(s)$
$ALNS-1000$	1,55	21,74	9,40
$ALNS-500$	3,08	3,90	1,66
MILP	7,07	2039,15	1253,24

Table 8: Results averaged over the instances of the set

value of the solutions obtained by each algorithm. The columns $\#S$ and $\#K_2$ report the average number of selected satellites and PCVs, respectively. Column T indicates the total runtime in seconds, while column T^* specifies the time at which the best solution was found.

As expected, $ALNS$-500 achieves significantly shorter computational times across all instances, due to the reduced number of iterations. For $ALNS$-1000, the average time to reach the best solution and the total runtime increase by 54.17% and 66.89%, respectively. While $ALNS$-1000 consistently yields better objective values, the average difference between the solutions of the two variants is only 1.85%, indicating that $ALNS$-500 provides comparable solution quality with substantially lower computational effort.

6.4 Management insights

In this section, we compare the proposed city logistics system with a traditional system in which shippers have *a priori* commercial agreements. In such a system, satellite configurations, container-to-satellite assignments, and vehicle selections are predefined. In addition, we evaluate three hybrid configurations in which some of these decisions are fixed, while the remaining ones are optimized by the *urban mobility manager*. Specifically, five configurations—referred to as rigidity levels—are considered in the analysis:

0% City logistics without predefined decisions.

25% City logistics with 25% of predefined defined decisions according to existing contracts (i.e., 75% of decisions are made by the *urban mobility manager*).

50% City logistics with 50% of predefined defined decisions according to existing contracts.

75% City logistics with 75% of predefined decisions according to existing contracts.

100% City logistics with 100% of predefined decisions according to existing contracts.

The experimental analysis is conducted using the $ALNS$-500 algorithm on the set of large instances, with results averaged over 15 runs per instance. The comparison focuses on two primary performance indicators: economic efficiency, measured by the

	$DATA$					$ALNS-500$					$ALNS-1000$				
#	S	C	Γ	K_1	K_2	obj	$\#S$	$\#K_2$	$T^*(s)$	$T(s)$	obj	$\#S$	$\#K_2$	$T^*(s)$	$T(s)$
0	20	25	75	26	62	14636,45	4	56	32,79	52,13	14635,10	4	57	164,21	231,78
1	20	25	75	26	63	14557,00	3	56	14,31	34,29	14393,67	3	56	25,21	78,88
2	20	25	75	26	69	15826,32	4	60	19,23	43,66	15714,16	4	61	79,41	157,39
3	20	25	75	26	63	14667,65	4	57	26,46	46,28	14345,28	3	56	70,21	140,69
4	20	25	75	26	65	14858,48	4	57	10,76	32,60	14696,74	4	58	31,44	116,85
5	20	25	50	26	64	14699,04	4	58	2,69	17,52	13895,81	3	55	85,13	141,99
6	20	25	50	26	61	13537,47	3	53	5,74	19,85	13515,09	3	53	49,03	97,93
7	20	25	50	26	65	14144,53	3	55	21,01	34,37	13282,82	4	55	54,12	100,43
8	20	25	50	26	60	14520,95	4	54	5,72	20,55	14042,66	3	55	72,93	121,91
9	20	25	50	26	62	14017,70	3	55	6,25	20,01	13889,33	3	56	17,90	71,80
10	25	50	150	51	124	30592,56	9	117	159,42	349,91	30290,49	9	117	1102,30	1778,36
11	25	50	150	51	121	31117,25	10	113	134,08	288,27	30055,85	9	113	180,98	874,14
12	25	50	150	51	124	29469,98	7	114	249,07	450,36	28520,75	9	114	389,82	981,61
13	25	50	150	51	123	29275,09	8	113	327,07	508,86	29172,52	9	112	633,32	1294,63
14	25	50	150	51	122	30133,51	10	111	118,74	300,49	29649,34	8	114	91,49	726,54
15	25	50	100	51	130	30572,87	9	120	184,29	347,26	30097,35	8	116	565,51	1176,24
16	25	50	100	51	126	30705,51	8	117	73,66	222,43	30407,99	9	116	317,34	927,79
17	25	50	100	51	124	29775,22	8	116	328,46	494,66	29475,07	8	113	402,33	950,39
18	25	50	100	51	129	29648,72	7	118	485,03	635,08	29355,68	7	117	321,71	873,87
19	25	50	100	51	124	30329,02	8	116	204,52	348,28	29315,44	7	114	1235,45	1791,48
20	30	75	225	76	184	45343,36	14	173	945,22	1508,78	44658,08	13	175	3425,14	3600,24
21	30	75	225	76	183	44325,42	11	174	453,18	995,48	43344,30	10	172	1811,11	3600,23
22	30	75	225	76	189	47768,94	16	176	667,14	1229,68	46304,89	16	175	3355,02	3600,23
23	30	75	225	76	190	46062,80	11	176	873,95	1484,75	44958,24	13	177	2968,00	3600,23
24	30	75	225	76	182	46170,70	13	172	956,51	1458,30	44865,82	12	171	738,17	2897,18
25	30	75	150	77	189	44716,38	12	176	467,13	940,21	43936,90	11	176	724,78	2611,37
26	30	75	150	76	182	45033,99	14	173	88,80	527,60	45015,56	13	172	598,74	2316,76
27	30	75	150	77	181	45534,15	14	169	400,46	848,13	45436,02	13	169	1116,46	2746,97
28	30	75	150	77	190	45325,61	14	176	107,93	511,90	44786,79	13	176	615,16	2242,55
29	30	75	150	77	188	45623,15	14	175	321,53	801,39	45526,70	13	176	313,31	1879,17

Table 9: Comparison between $ALNS-500$ and $ALNS-1000$ on large instances

total distribution cost, and the impact on urban traffic, evaluated by the number of vehicles deployed during the distribution process.

Figure 4 shows the average number of satellites selected for each rigidity level. The results are reported for the instances with 25, 50, and 75 containers.

As expected, the average number of selected satellites increases with the level of rigidity. In instances with 25 containers, the average number of satellites selected increases from 3.42 with predefined selections 0% to 12.25 with predefined decisions 100%, which represents a significant increase of 258%. This shows the greatest relative growth. In instances with 50 containers, the number of satellites increases from 8.75 with 0% rigidity level to 20.33 with 100%, i.e. an increase of 132%. In instances with 75 containers, the number of satellites increases from 13.58 with predefined selections of 0% to 26.75 with 100%, resulting in an increase of 97%. Generally speaking, the largest decrease in the average number of satellites arises when the rigidity is changed from 100% to 75% and from 75% to 50%.

As expected, the average number of selected satellites increases with the level of rigid-

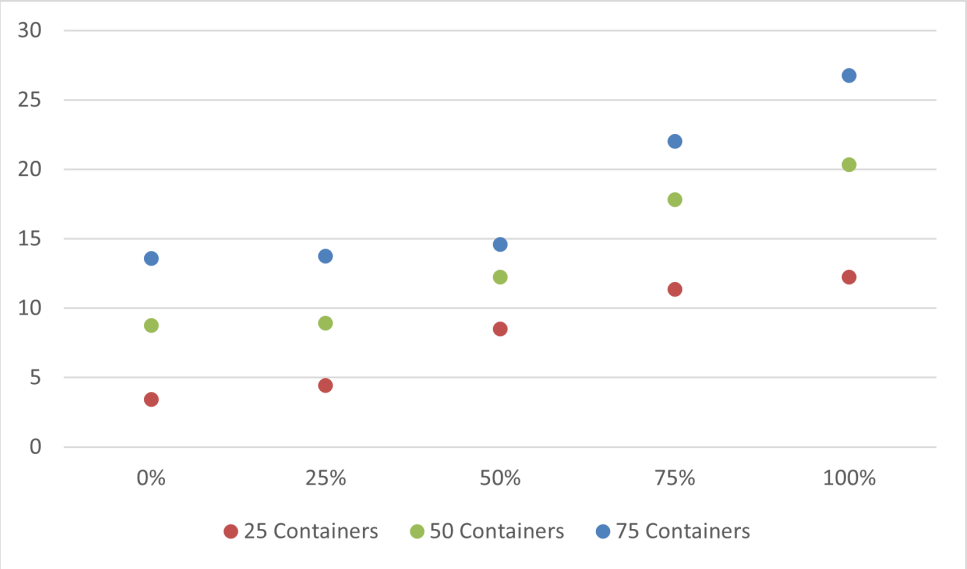

Figure 4: Average number of satellites selected as a function of the rigidity level.

ity in the system configuration. In instances with 25 containers, the average number of selected satellites rises from 3.42 under 0% predefined decisions to 12.25 under 100% predefined decisions — an increase of 258%. For instances with 50 containers, the number of satellites increases from 8.75 to 20.33 across the same range of rigidity levels, corresponding to an increase of 132%. Similarly, in instances with 75 containers, the number of satellites grows from 13.58 to 26.75, reflecting a 97% increase.

Overall, the most substantial reductions in the average number of satellites occur when decreasing the rigidity level from 100% to 75% and from 75% to 50%.

Figure 5 illustrates the average distribution costs across the five system configurations, grouped by the number of containers (25, 50, and 75). The total cost is broken down into three components: PCV costs (including selection costs, routing costs, and pallet flow costs incurred by PCVs), satellite costs (comprising satellite selection costs and total pallet cross-docking costs at the selected satellites), and CCV costs (covering selection costs for CCVs). For each cost category, the percentage increase relative to the base configuration of the two-tiered system is reported, allowing for a detailed comparison of how rigidity levels affect the cost structure.

As expected, cost percentages tend to increase with higher rigidity levels. The CCV selection cost is the least affected by the increase in predefined selections, as the same number of containers must always be transported from the port to the satellites, regardless of the configuration. In contrast, satellite costs are the most significantly impacted, showing an increase of 54% in the fully traditional system (i.e., with 100% of rigidity level).

The cost of PCVs displays a non-linear trend relative to the rigidity level. From 0% to

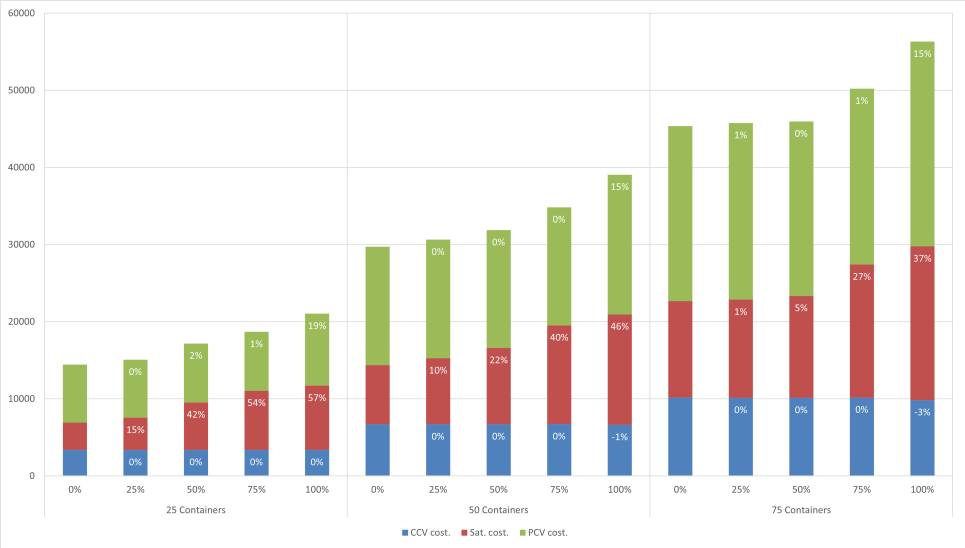

Figure 5: Comparison of system costs by rigidity level and container group. Costs for PCVs, satellites, and CCVs are reported in green, red, and blue, respectively.

	25 Containers				50 Containers				75 Containers			
	25%	50%	75%	100%	25%	50%	75%	100%	25%	50%	75%	100%
CCV	0%	0%	0%	0%	0%	0%	0%	-1%	0%	0%	0%	-3%
Sat.	15%	42%	54%	57%	10%	22%	40%	46%	1%	5%	27%	37%
PCV	0%	2%	1%	19%	0%	0%	0%	15%	1%	0%	1%	15%
Total	4%	19%	29%	46%	3%	7%	17%	32%	1%	1%	11%	24%

Table 10: The percentage of cost increases with respect to the 0% rigidity level.

75% predefined selections, PCV-related costs remain stable or even decrease. However, when the rigidity level exceeds 75%, PCV costs increase substantially. This behavior can be attributed to the greater number of available satellites at lower rigidity levels, which allows PCVs to be more geographically dispersed across the city, thereby reducing their average travel distances. Furthermore, even at higher rigidity levels, no significant issues are observed regarding the saturation of PCVs' capacities.

Overall, the total cost increases by 46%, 32%, and 24% when moving from 0% to 100% rigidity in the instances with 25, 50, and 75 containers, respectively.

Table 10 reports the average percentage cost increase - total and by type - for all levels of rigidity and instance sizes. Generally speaking, the largest cost decrease occurs when switching the rigidity from 100% to 75% and from 75% to 50%.

Finally, **Figure 6** compares the average number of PCVs used for each rigidity configuration. Three trends are reported, one for each group of instances.

The average number of PCVs increases with the level of predefined selections. In

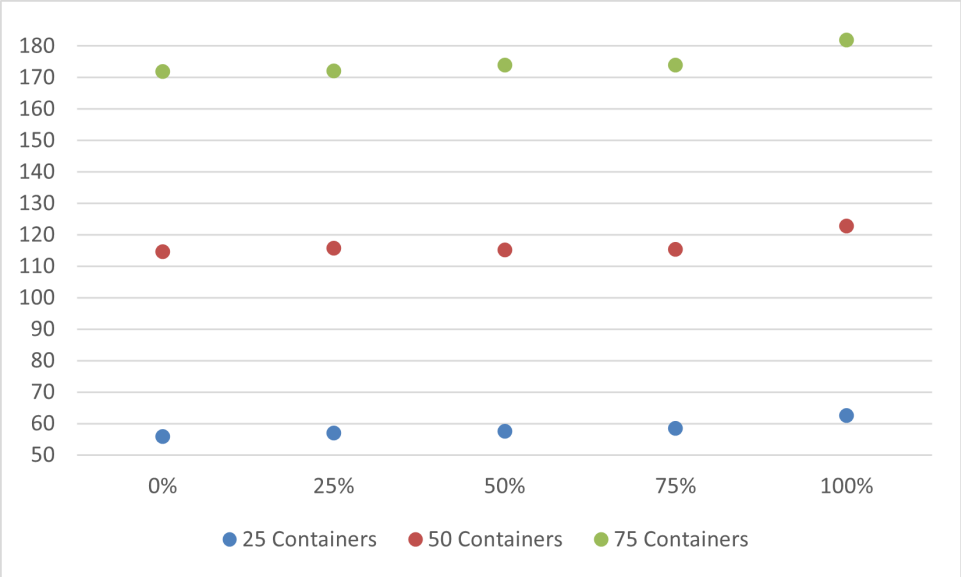

Figure 6: Average number of PCVs selected as a function of the rigidity level.

instances with 25 containers, the number of PCVs rises from 55.83 at 0% rigidity to 62.50 at 100% rigidity, corresponding to an increase of 11.94%. These instances show the highest sensitivity to changes in rigidity. In the 50-container group, the number of PCVs increases from 114.67 to 122.75, marking a 7.04% increase. For instances with 75 containers, the variation is less pronounced, with the number rising from 171.92 to 182.00, which represents a 5.87% increase.

Overall, the most significant reduction in the average number of PCVs occurs when decreasing the rigidity level from 100% to 75%, indicating that even partial flexibility in operational decisions can lead to a meaningful reduction in vehicle usage.

7 Conclusion

In this work, we investigated a two-tiered multi-commodity service network design problem inspired by urban contexts featuring large intermodal terminals without transdock capabilities (e.g., ports). To the best of our knowledge, this problem has not yet been explored in the literature. The main contributions of this paper are as follows:

- The formulation of a Mixed Integer Linear Programming (MILP) model that captures the essential features of the problem. The model integrates facility location and network design decisions, considers multi-commodity flows, and includes the selection of vehicles from a heterogeneous fleet, accounting for varying capacities and cost structures.

- The design and implementation of an effective solution method based on two interacting Adaptive Large Neighborhood Search (ALNS) metaheuristics, tailored to efficiently solve large-scale instances. Specifically, the first ALNS addresses the configuration of satellites and the assignment of containers to satellites, while the second ALNS handles the second-tier decisions given a fixed satellite configuration and container assignment.

A logistic analysis was conducted to assess the impact of progressively incorporating decision-making capabilities into the proposed framework. We evaluated several key aspects of the resulting solutions, including satellite and vehicle selection and cost distribution. Experimental results indicate that the most significant gains occur when transitioning from no decision of the mobility manager to the adoption of approximately 25% of the available decision options. This highlights the strategic value of introducing city logistics initiatives, particularly in the early stages, rather than merely expanding existing systems.

Future research will explore exact solution methods for this problem and investigate extensions to more complex and realistic scenarios in city logistics and multi-tier systems. Notably, time dependency and the synchronization of CCV and PCV arrivals at satellites represent critical enhancements to improve model applicability. Further extensions will include satellite-to-satellite transfers, container repositioning, and increased heterogeneity in container and pallet types, which would enhance both the scope and practical relevance of the problem.

8 Acknowledgements

This study was carried out within the MOST – Sustainable Mobility National Research Center and received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.4 – D.D. 1033 17/06/2022, CN00000023). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them. This publication was produced by Francesco Contu while attending the PhD programme in Mathematics and Computer Science at the University of Cagliari, Cycle XXXVIII, with the support of a scholarship co-financed by the Ministerial Decree no 352 of 9th April 2022 based on the NRPP – funded by the European Union – Next Generation EU- Mission 4 "Education and Research", Component 2 "From Research to Business", Investment 3.3. and by the company Spiva. While working on the project, the second author held the UQAM Chair on Intelligent Logistics and Transportation Systems Planning, and was Adjunct Professor, Department of Computer Science and Operations Research, Université de Montréal. We gratefully acknowledge the financial support provided by the Natural Sciences and Engineering Council of Canada (NSERC), through its Discovery Grant

program, as well as that of Fonds de recherche du Québec through their infrastructure grants.

References

[1] Maurizio Boccia, Teodor Gabriel Crainic, Antonio Sforza, and Claudio Sterle. Multi-commodity location-routing: Flow intercepting formulation and branch-and-cut algorithm. *Computers & Operations Research*, 89:94–112, 2018.

[2] R. Cuda, G. Guastaroba, and M. G. Speranza. A survey on two-echelon routing problems. *Computers & Operations Research*, 55:185–199, 2015.

[3] Teodor Gabriel Crainic, Nicoletta Ricciardi, and Giovanni Storchi. Advanced freight transportation systems for congested urban areas. *Transportation Research Part C: Emerging Technologies*, 12(2):119–137, 2004.

[4] Teodor Gabriel Crainic, Nicoletta Ricciardi, and Giovanni Storchi. Models for evaluating and planning city logistics systems. *Transportation Science*, 43(4):432–454, 2009.

[5] Vera C. Hemmelmayr, Jean-François Cordeau, and Teodor Gabriel Crainic. An adaptive large neighborhood search heuristic for Two-Echelon Vehicle Routing Problems arising in city logistics. *Computers & Operations Research*, 39(12):3215–3228, 2012.

[6] Teodor Gabriel Crainic and Antonino Sgalambro. Service network design models for two-tier city logistics. *Optimization Letters*, 8(4):1375–1387, 2014.

[7] Paolo Gianessi, Laurent Alfandari, Lucas Létocart, and Roberto Wolfler Calvo. The multicommodity-ring location routing problem. *Transportation Science*, 50(2):541–558, 2016.

[8] Teodor Gabriel Crainic, Fausto Errico, Walter Rei, and Nicoletta Ricciardi. Modeling demand uncertainty in two-tier city logistics tactical planning. *Transportation Science*, 50(2):559–578, 2016.

[9] Teodor Gabriel Crainic, Guido Perboli, and Nicoletta Ricciardi. City Logistics. In Michel Gendreau and Bernard Gendron (eds.), *Network Design with Applications to Transportation and Logistics*, pages 507–537. Springer International Publishing, Cham, 2021.

[10] Jeanette Schmidt, Christian Tilk, and Stefan Irnich. Using public transport in a 2-echelon last-mile delivery network. *European Journal of Operational Research*, 317(3):827–840, 2024.

[11] Pirmin Fontaine, Teodor Gabriel Crainic, Ola Jabali, and Walter Rei. Scheduled service network design with resource management for two-tier multimodal city logistics. *European Journal of Operational Research*, 294(2):558–570, 2021.

[12] Pirmin Fontaine, Stefan Minner, and Maximilian Schiffer. Smart and sustainable city logistics: Design, consolidation, and regulation. *European Journal of Operational Research*, 307(3):1071–1084, 2023.

[13] Ivan Contreras and Elena Fernández. General network design: A unified view of combined location and network design problems. *European Journal of Operational Research*, 219(3):680–697, 2012.

[14] Sanjay Melkote and Mark S. Daskin. An integrated model of facility location and transportation network design. *Transportation Research Part A: Policy and Practice*, 35(6):515–538, 2001.

[15] Sanjay Melkote and Mark S. Daskin. Capacitated facility location/network design problems. *European Journal of Operational Research*, 129(3):481–495, 2001.

[16] Matthias Winkenbach, Paul R. Kleindorfer, and Stefan Spinler. Enabling urban logistics services at La Poste through multi-echelon location-routing. *Transportation Science*, 50(2):520–540, 2016.

[17] Daeki Kim, Cynthia Barnhart, Keith Ware, and Gregory Reinhardt. Multimodal express package delivery: A service network design application. *Transportation Science*, 33(4):391–407, 1999.

[18] M. F. Lai and Hong K. Lo. Ferry service network design: optimal fleet size, routing, and scheduling. *Transportation Research Part A: Policy and Practice*, 38(4):305–328, 2004.

[19] Duc Minh Vu, Teodor Gabriel Crainic, and Michel Toulouse. A three-phase matheuristic for capacitated multi-commodity fixed-cost network design with design-balance constraints. *Journal of Heuristics*, 19(5):757–795, 2013.

[20] Teodor Gabriel Crainic, Mike Hewitt, Michel Toulouse, and Duc Minh Vu. Service network design with resource constraints. *Transportation Science*, 50(4):1380–1393, 2014.

[21] Mervat Chouman and Teodor Gabriel Crainic. Cutting-plane matheuristic for service network design with design-balanced requirements. *Transportation Science*, 49(1):99–113, 2015.

[22] Ragheb Rahmaniani and Abdolsalam Ghaderi. A combined facility location and network design problem with multi-type of capacitated links. *Applied Mathematical Modelling*, 37(9):6400–6414, 2013.

[23] Ann-Kathrin Rothenbächer, Michael Drexl, and Stefan Irnich. Branch-and-price-and-cut for a service network design and hub location problem. *European Journal of Operational Research*, 255(3):935–947, 2016.

[24] Teodor Gabriel Crainic, Mike Hewitt, Michel Toulouse, and Duc Minh Vu. Scheduled service network design with resource acquisition and management. *EURO Journal on Transportation and Logistics*, 7(3):277–309, 2018.

[25] Teodor Gabriel Crainic, Michel Toulouse, Mike Hewitt, and Minh Võ. Location and service network design. CIRRELT Report, 2015.

[26] Michael Berliner Pedersen, Teodor Gabriel Crainic, and Oli B. G. Madsen. Models and tabu search metaheuristics for service network design with asset-balance requirements. *Transportation Science*, 43(2):158–177, 2009.

[27] Teodor Gabriel Crainic and Benoit Montreuil. Physical Internet Enabled Hyperconnected City Logistics. *Transportation Research Procedia*, 12:383–398, 2016.

[28] Paul Shaw. Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In Michael Maher and Jean-François Puget (eds.), *Principles and Practice of Constraint Programming — CP98*, pages 417–431. Springer, Berlin Heidelberg, 1998.

[29] David Pisinger and Stefan Ropke. Large Neighborhood Search. In Michel Gendreau and Jean-Yves Potvin (eds.), *Handbook of Metaheuristics*, pages 99–127. Springer International Publishing, Cham, 2019.

[30] Stefan Ropke and David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472, 2006.

[31] Philippe Grangier, Michel Gendreau, Fabien Lehuédé, and Louis-Martin Rousseau. An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. *European Journal of Operational Research*, 254(1):80–91, 2016.

[32] Maximilian Schiffer and Grit Walther. An adaptive large neighborhood search for the location-routing problem with intra-route facilities. *Transportation Science*, 52(2):331–352, 2018.

[33] Setyo Tri Windras Mara, Rachmadi Norcahyo, Panca Jodiawan, Luluk Lusiantoro, and Achmad Pratama Rifai. A survey of adaptive large neighborhood search algorithms and applications. *Computers & Operations Research*, 146:105903, 2022.