future transportation

MDPI

Article

Re-Supplying Autonomous Mobile Parcel Lockers in Last-Mile Distribution

Sajjad Hedayati [1,*], Mostafa Setak [1], Tom Van Woensel [2] and Emrah Demir [3]

[1] Faculty of Industrial Engineering, K. N. Toosi University of Technology, 19991-43344 Tehran, Iran; setak@kntu.ac.ir
[2] School of Industrial Engineering, Eindhoven University of Technology, 5600MB Eindhoven, The Netherlands; t.v.woensel@tue.nl
[3] Panalpina Centre for Manufacturing and Logistics Research, Cardiff Business School, Cardiff University, Cardiff CF10 3EU, United Kingdom; demire@cardiff.ac.uk
* Correspondence: hedayati.s@email.kntu.ac.ir

Abstract: This paper investigates a practical last-mile delivery scenario where a fleet of trucks replenishes autonomous mobile parcel lockers (AMPLs) in an urban setting. The lockers move along specified paths within restricted zones to reach customers' locations. Ensuring seamless coordination between trucks and AMPLs requires the identification of suitable locations to exchange empty or loaded modular lockers. We first introduce a mixed-integer linear programming (MILP) formulation for the investigated problem. The proposed formulation establishes the basis for optimizing meeting point selection and routing decisions. Additionally, the study introduces a cluster-based simulated annealing (CSA) algorithm tailored for addressing larger-scale instances of the studied problem. The CSA algorithm incorporates the K-means clustering method with specialized operators rooted in an extensive neighborhood search, aiming to improve the effectiveness of solution discovery. We generated a new set of benchmark instances to assess the MILP formulation's efficiency and the proposed metaheuristic algorithm and conducted comprehensive numerical experiments.

Keywords: freight transportation; last-mile logistics; autonomous mobile parcel lockers; metaheuristic algorithm

Citation: Hedayati, S.; Setak, M.; Van Woensel, T.; Demir, E. Re-Supplying Autonomous Mobile Parcel Lockers in Last-Mile Distribution. *Future Transp.* **2024**, *4*, x. https://doi.org/10.3390/xxxxx

Academic Editor(s): Name

Received: 22 July 2024
Revised: 1 October 2024
Accepted: 10 October 2024
Published: date

Copyright: © 2024 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Last-mile delivery operations, a crucial aspect of urban logistics, have gained attention due to their environmental and social impact on humans and the environment. These operations, which involve expensive transportation of goods to their destinations, face challenges and opportunities driven by the booming demand for e-commerce services. The growth of megacities and the projected increase in e-commerce sales underscore the need for efficient last-mile logistics solutions [1].

As the complexity of last-mile delivery within supply networks continues to grow, recent technological advancements offer retailers and logistics service providers (LSPs) the potential to address these challenges with greater efficiency and effectiveness [2]. Some of these innovative solutions catering to demanding customer needs include rapid delivery [3], collaborative shipping [4], mobile delivery routes [5], pavement-side delivery [6], storage lockers [7], autonomous delivery robots [8], and drone delivery [9]. However, the feasibility of these solutions varies across locations due to constraints in urban infrastructure development. In contrast, online consumers' expectations for timely and cost-effective order fulfillment persist. Consequently, effective coordination of the flow of goods, collaborative efforts among stakeholders, and optimal order consolidation remain critical components in navigating the complexities of last-mile logistics [10].

A novel innovation in last-mile delivery is the mobile parcel locker (MPL) technology, which allows customers to pick up parcels from convenient locations [11,12,13]. MPLs can be operated autonomously or with a human driver. They can be classified into two groups: fixed locker boxes mounted on mobile platforms, and mobile platforms with separate locker sections. MPLs can travel between different locations throughout the day, serving customers in nearby areas. Consequently, online consumers can conveniently access them during designated time windows for order retrieval or returns.

In a comprehensive study conducted by [13], various last-mile settings and configurations were examined. Based on their research findings, our current problem lies in the space between two distinct configurations. Specifically, this study lies in the middle of the "depot–mobile parcel locker–self-service" configuration, as investigated by [11], and the "depot–autonomous van–locker–self-service" configuration, as studied by [14]. The central focus of this study is to look at the procedure of replenishing a fleet of autonomous MPLs (AMPLs) by using a fleet of trucks, with a specific emphasis on refining last-mile logistics within the delivery network. The primary objective is to enhance the operational efficiency of a designated subset of AMPLs by optimizing the loading of restocked lockers and unloading returned ones, facilitated by a group of delivery trucks.

However, there is a drawback related to the capacity of this delivery assistant. If AMPLs are not replenished regularly, the lockers cannot fulfill orders throughout the day, as anticipated with other MPL variants. Although this system offers considerable potential for the transportation and operations research community, current research remains limited. Therefore, this paper is dedicated to devising a formulation and a solution strategy for a scenario in which a fleet of trucks efficiently loads replenished lockers and unloads returned ones from several AMPLs at selected meeting points during the day. To address this complex problem, we combine two established variants of the vehicle routing problem (VRP), namely, the VRP with multiple time windows (VRPMTW) and the generalized VRPTW (GVRPTW). On the one hand, leveraging the GVRPTW approach enables us to pick an appropriate physical demand location from alternative options, ensuring service within the designated time window for the AMPLs. Nevertheless, unlike VRPTW and VRPMTW, this issue does not require a visit to all nodes. On the other hand, the VRPMTW framework empowers us to deconstruct a physical demand point into other nodes, termed meeting points, within our graph-based model. This partitioning facilitates the segmentation of lengthy waiting periods at each physical point into more manageable time windows, effectively transforming the problem into a recognizable VRPMTW instance.

With our aim centered on selecting the best meeting point from various alternatives, we face the task of choosing a physical location that aligns with a time window within the broader time slot. Moreover, each time window is linked to a unique parking fee, allowing us to integrate this aspect into our model. It is important to note that this innovative integrated approach remains unexplored in previous research in the context of last-mile applications. The problem addressed in this paper differs from the problem discussed by [11], who focused on optimizing the location of human driver MPLs in the last-mile delivery context. In this paper, the main objective is to efficiently re-supply AMPLs (called R-AMPLP), eliminating the need for them to return to the depot for refilling. This approach significantly benefits e-commerce firms or LSPs, as new orders can be processed and transferred to meeting points without the AMPLs returning to the depot.

The contributions in this paper can be summarized into three key dimensions. Firstly, we introduce a novel re-supplying AMPL problem in the context of last-mile distribution, entailing an optimization strategy for establishing a designated set of meeting points. These aspects aid in the efficient loading and unloading of lockers on AMPLs, which customers have emptied in the initial segment of their routes and need to be restocked for the remaining customers in the latter segment. Second, we present a MILP formulation tailored to this problem. Our formulation incorporates various alternative locations characterized by multiple time windows, time-dependent parking charges, and a fleet of capacitated trucks responsible for conveying same-day orders. These orders are stowed in

loaded lockers, moving from the depot to the pre-identified meeting points. Lastly, we propose a cluster-based simulated annealing (CSA) algorithm featuring a specialized enhancement operator to manage the clustering aspect and randomized destroy and repair operators. Particularly, we incorporate the well-known K-means method to cluster nodes as a preparation phase and then customize the simulated annealing (SA) approach to handle the instances.

The following sections of the paper are structured as follows: Section 2 reviews the related literature that addresses similar problems. Section 3 presents the problem definition and the problem's mathematical formulation. Section 4 presents the CSA algorithm and its features, including the clustering method and neighborhood search operators. Section 5 presents the computational outcomes of applying the MILP model and the CSA algorithm on a range of benchmark instances. Finally, Section 6 concludes the paper and provides possible avenues for future research.

2. Literature Review

This section reviews the work related to the problem at hand. Theoretically, the investigated problem falls under the VRPMTW, in which a physical demand location can be defined with multiple time windows during the planning horizon. On the other hand, since it aims to select an optimal meeting point amongst a set of alternative meeting points located at different locations and has different time windows, the GVRPTW concept is also considered. In addition, although it is related to the dynamic VRP, by considering fixed order acceptance and fixed routes, including meeting points, we can also adapt our problem to a static setting.

Regarding last-mile delivery, many studies proposed various innovations for performing delivery operations. We refer to [13] and [2] for more details. In this setting, we only focused on re-supplying MPL. Therefore, our literature review section is divided into three fields. First, we review the recent generalized VRPTW (GVRPTW) studies. Second, the works related to the VRPMTWs are explored. Finally, recent research studies have been reviewed in the context of MPLs.

2.1. The Generalized Vehicle Routing Problem with Time Windows

One of the VRP variants is the VRPTW, where customers must be visited within a specific period [15]. Another type of VRP, known as the generalized VRP (GVRP), considers a setting in which customers are grouped into some unconnected clusters, and the goal is to visit exactly one node in each cluster [16-17]. Theoretically, when we combine the VRPTW and the GVRP, we obtain an extended version of the GVRP, known as the GVRPTW. To our knowledge, the GVRPTW was first studied by [18]. The problem is formulated and solved in their paper by an incremental Tabu search algorithm. After that, [5] presented the VRP with roaming delivery locations (VRPRDL), known as the trunk delivery problem. In this problem, customers can select their preferred locations with time windows for receiving orders. The difference between the VRPRDL and GVRPTW is that time windows in the VRPRDL cannot overlap with each other because the location of each customer may change during the day. For this problem, they introduced an arc-based formulation. Afterward, [19] formulated the VRPRDL as a set-covering problem and promoted a branch-and-cut-and-price algorithm as a solution approach. On the other hand, the main difference between the VRPRDL and our problem is that each location is available as a meeting point within different time windows.

Recently, [20] proposed the VRP with delivery options (VRPDO), which, similar to the VRPRDL, gives customers alternatives to pick up their orders from different locations. They presented an MIP and a large neighborhood search (LNS) algorithm for the problem. The authors considered an extension of the VRPTW with multiple delivery options. Our problem, however, concerns a setting where a fleet of vehicles must re-supply a fleet of mobile parcel lockers in a location (or re-supply point). Therefore, we focused on re-

supply operations rather than customers' pickup alternatives. [21] presented a VRPDO involving optimizing shipments to various locations considering customer preferences and time windows. They introduced a new algorithm to minimize carrier costs while accommodating shared capacity constraints. This approach, tested in scenarios with up to 100 delivery options, includes new solutions to routing problems with roaming delivery locations. [22] introduced a new VRP variant that addresses the delivery of parcels to a customer's vehicle at various locations during the day. It replaces the traditional distance matrix with a probability distribution matrix for variable travel times. The research applied a Monte Carlo method combined with an enhanced greedy randomized adaptive search procedure to find a near-optimal solution, which was validated through various experiments.

2.2. The VRP with Multiple Time Windows

One of the first studies on the VRPMTW was presented by [23]. The authors considered a periodic VRP concept in which demand points are associated with different time windows during the planning horizon. They presented an MIP model and an ant colony algorithm to minimize the total duration. In another study, [24,25] proposed a heuristic algorithm known as the hybrid variable neighborhood with the Tabu search algorithm for the VRPMTW. [26] studied a multi-objective version of the VRPMTW based on customer preference with a case study. The authors applied a local search algorithm to solve it. Recently, [27] presented an exact polynomial–time algorithm based on neighborhood evaluation. The main difference between our work and these works is that, in our setting, each demand point has multiple potential locations and time windows, with time-dependent traveling time and parking fees.

2.3. Mobile Parcel Lockers

Regarding MPLs, the paper presented by [11] was the first research to introduce an optimization problem in the MPL context. The objective of their paper was to find optimal locations for MPLs. They presented three alternative mixed-integer programs and a heuristic algorithm for their problem. The authors did not consider any assumption of responding to new orders and re-supply operations in a same-day context. See Tables 1 and 2 for a comparison of the contributions of this paper with related work in the literature.

Table 1. The main differences between GVRPTW, VRPMTW, and MPLs.

Reference	Setting	GVRPTW	VRPMTW	MPLs
[11]	Alternative locations	*		*
[24,25]	Alternative time windows		*	
[11,20]	Sharing delivery locations			*
This paper	Alternative locations and alternative time windows	*	*	*

Table 2. A list of contributions and solution methodologies from selected publications.

Reference	Model	Solution	Roaming/ GVRPTW	Multiple Time Windows	Parking Slot Fee	AMPL	Sharing Locations
[5]	IP [a]	GRASP [d]-VNS [e]	*				
[18]	-	TS [f]	*				
[21]	MIP [b]	GRASP	*				
[20]	MIP	LNS [g]	*				*
[[21]	SP [c]	BPC [h]	*				*
This paper	MIP	CSA [i]	*	*	*	*	

[a] Integer programming, [b] mixed-integer programming, [c] set-partitioning, [d] greedy randomized adaptive search procedure, [e] variable neighborhood search, [f] Tabu search, [g] large neighborhood search, [h] branch-and-cut-and-price, [i] cluster-based simulated annealing.

3. Problem Description

We now formally introduce the R-AMPLP and present the corresponding mathematical formulation of the problem. Assume an LSP serving its customers in distinct geographical areas or zones with a fleet of AMPLs. In particular, the LSP employs a delivery system with several AMPLs that cover all delivery needs in each zone during the day via already optimized paths. Zones may have restrictions, and not all types of trucks or vans, such as typical delivery vehicles, can be used. In addition, it is assumed that demand is deterministic, leading daily delivery orders through fixed paths with a long planning horizon. As demand information and zones are known and constant, several AMPLs can be assigned to each zone with pre-defined paths, where they travel and stop at known locations associated with time windows.

However, in this setting, there are two main issues for which the LSP should decide. The first problem is the capacity restriction of AMPLs to satisfy all customers' orders during a planning horizon. This implies that all AMPLs are depleted approaching noon and require replenishment to serve the remaining customers or demand points that have not been addressed on their routes. The second issue concerns situations where there are a few parking slots to exchange lockers between trucks and AMPLs located around each zone. Additionally, each parking slot has a different reservation price during the day. It is crucial to highlight that when lockers are fully emptied from autonomous mobile platforms, the synchronization of each box's capacity is disrupted, particularly when previous customers have not received their orders. Therefore, it is assumed that all customers served before noon have collected their orders, leaving all boxes empty.

Furthermore, the LSP operates a distribution center (DC) from which trucks are dispatched to replenish AMPLs with new lockers and retrieve returned lockers. Each truck can transport multiple loaded parcel lockers replenished at the DC. At a designated meeting point, a truck unloads a specified number of parcel lockers onto an equal number of mobile platforms, known as AMPLs. The AMPLs then move from the drop-off point into a zone and follow predetermined routes to serve customers. Customers can visit AMPLs in zones to receive orders from designated lockers. Since demands are already known in the system and AMPLs move in pre-defined routes, the required time in the operation course is clear for replenishment. Therefore, once all delivery orders up to the meeting time have been fulfilled, the AMPL returns to a truck at a specific parking slot to deliver its parcel locker and retrieve a new one. Consequently, exchange operations can occur at a parking space or meeting point.

For further information on the application of this problem, we refer to Figure 1. The biggest advantage of loaded lockers is the autonomy of being separated from the carrying platforms. Figure 1 shows a scenario where three autonomous platforms, each with an empty locker, endeavor to swap their lockers at a meeting point. This is especially important in places that restrict the entry of delivery vans or trucks. AMPLs can offer a viable

solution by collaborating with a truck fleet to serve a diverse range of last-mile customers in such areas (e.g., city centers).

Additional assumptions and system details are outlined as follows: Figure 1 illustrates how the predetermined route of each AMPL is divided into two segments around midday. Indeed, once all customers have retrieved their orders from the boxes, the AMPL becomes empty. Utilizing its navigation, it can then move to a designated parking slot within specified time windows. Each zone has a predetermined number of parking spaces outside (i.e., parking 1–4) that can be reserved at different costs during different day periods. The arrival time of a truck at a meeting point is considered to access a specific parking slot, meaning a time window is associated with each parking slot. AMPLs are available during the designated time windows for delivering and loading parcel lockers at the selected meeting point by automatically changing their paths, as shown in Figure 1. The maximum number of AMPLs visited at each meeting point corresponds to the number of parcel lockers carried by truck. For example, in a zone with a dedicated AMPL, a truck may deliver and load one parcel locker at the selected meeting point. While a truck can visit multiple meeting points from different zones during its trip, the capacity constraint must be adhered to. The battery capacity of the AMPLs is assumed to be sufficient to support daily operations. Trucks are required to reach the selected meeting points (e.g., parking slots within specified time windows) on time. The time needed to load and unload parcel lockers is also incorporated into the travel time.

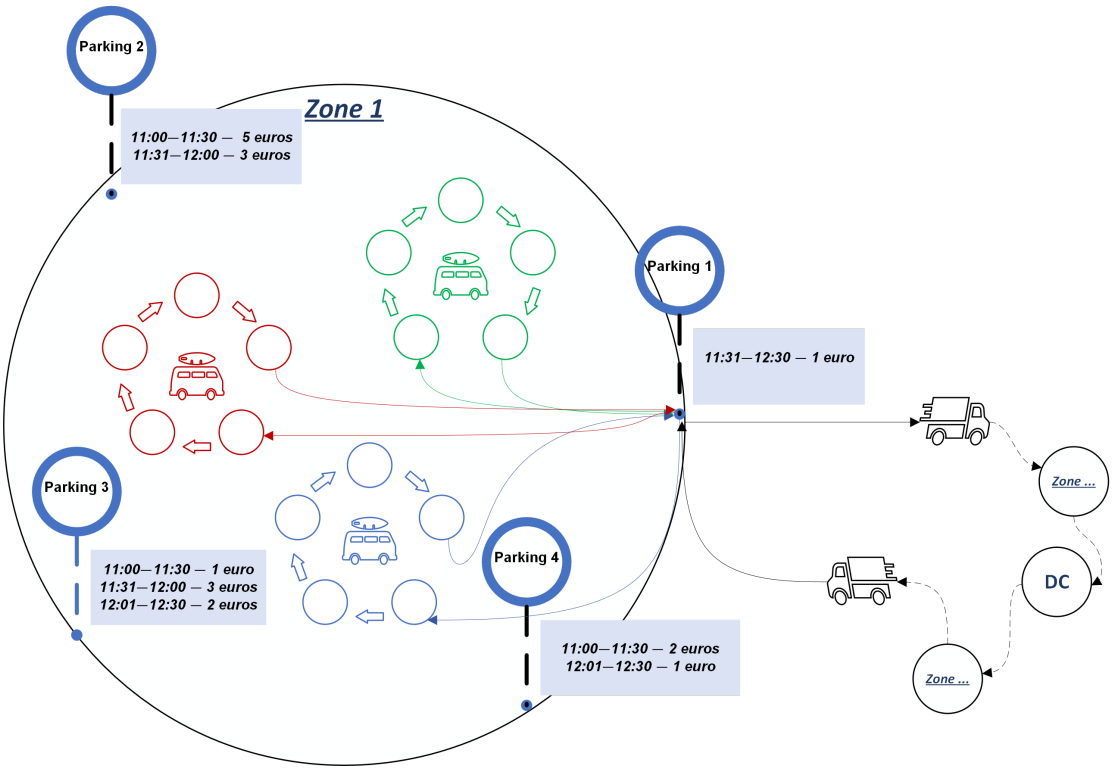


Figure 1. Meeting of trucks and AMPLs at meeting points with different leaving and returning points for AMPLs.

3.1. Mixed-Integer Linear Programming Model

This section presents a mixed-integer linear programming model for the problem. We define the R-AMPLP on a directed graph, $G = (N, A)$, in which N is associated with the node set, $N = \{0,1, \dots, n\}$, and A corresponds to the arc set, $A = \{(i, j): i, j \in C, i \neq j\}$. Set $C = N \backslash \{C^c, 0\}$ is associated with potential meeting points, which are grouped into a set of p distinct clusters (i.e., zones), $C = \{C_1, C_2, \dots, C_p\}$. The complement set, $C^c = \{C^c{}_1, C^c{}_2, \dots, C^c{}_p\}$, corresponds to the AMPLs' leaving and returning points. Note that in

this study, zones and clusters are used interchangeably. These two sets have two main properties, which are as follows: for members in C, $C_h \cap C_{h'} = \emptyset, \forall h, h' \in 1, \dots, p$, where $h \neq h'$ and $C_1 \cup C_2 \cup \dots \cup C_p = N \setminus \{0, C^c\}$, and for members in C^c, ${C^c}_h \cap {C^c}_{h'} = \emptyset, \forall h$, $h' \in 1, \dots, p$, where $h \neq h'$ and ${C^c}_1 \cup {C^c}_2 \cup \dots \cup {C^c}_p = N \setminus \{0, C\}$. Each cluster, ${C^c}_h, 1 \leq h \leq p$, corresponds to several leaving and returning points for AMPL(s). Node 0 represents a depot with a set of k homogeneous trucks, $V = \{1, \dots, k\}$. Each truck has a specific capacity, $L_f, 1 \leq f \leq k$, and a certain constant speed, s_f.

More precisely, for handling multiple time windows associated with each location, we consider multiple copies of a certain physical location, each with a specific time window and identical traveling distances. Hence, for a node $i \in C_h$, there is a time window, $[e_i, l_i]$, which is associated with a specific parking cost, pc_i. All the nodes in a cluster, $C_h, 1 \leq h \leq p$, have an equal non-negative demand, $0 < q_i \leq \max\{L_1, \dots, L_k\}, \forall i \in C_h$, which equals the number of AMPLs available at node i. On the other hand, node $b \in {C^c}_h$ is a node at which an AMPL in zone h leaves its path to go to the meeting point and returns to its path at $b' \in {C^c}_h$ to continue serving customers. Note that $b \in {C^c}_h$ and $b \in {C^c}_h$ might be identical, in other words, $b = b'$, or different based on the system configuration. For instance, as Figure 1 shows, three leaving and three returning points are associated with three AMPLs moving in the presented zone. We assume that at each meeting point, AMPLs are available from the beginning of the time windows, so the operation starts as soon as the truck arrives.

The travel time, d_{ij}/s_f, is associated with each arc, $(i, j) \in A$, where d_{ij} represents the traveling distance from i to j. Also, s_f denotes the speed of the trucks, which is constant and the same for $1 \leq f \leq k$, in our case. Concerning AMPLs' paths, in this paper, we only consider the traveling distance that each AMPL must pass to reach the meeting point. In fact, as AMPLs have pre-defined paths and a navigation system that can help them find the meeting points, we consider the traveling distance they must pass in our objective function.

Moreover, the working time length of the day is defined as $[0, T]$. The solution to this problem is to provide k optimal routes, in which (i) a location is only served by exactly one truck within corresponding time windows (i.e., meeting point); (ii) all of the routes of trucks must start and end at the depot within $[e_0, l_0]$, which is equal to $[0, T]$; (iii) the total demand carried by each truck cannot exceed L_f, which is connected to the capacity of the truck f. Also, each truck's total operation time cannot exceed T. Finally, (iv) the total cost, including traveling and parking costs, is minimized. The required symbols for presenting the MILP model are listed in Table 3.

Table 3. Notations used in the model.

Notation	Description
Sets	
$N = \{0, 1 \dots, n\}$	The set of nodes
$N \setminus \{0, C^c\}$	The set of meeting points
$A = \{(i, j) : i, j \in N \setminus \{0, C^c\}, i \neq j\}$	The set of arcs
$C = \{C_1, C_2, \dots, C_p\}$	The set of clusters corresponding to meeting points
$C^c = \{{C^c}_1, {C^c}_2, \dots, {C^c}_p\}$	The complement set of clusters associated with AMPL paths
$V = \{1, \dots, k\}$	The set of vehicles
$\{0\} \in N$	The depot
$b \in {C^c}_h$	Leaving point in cluster h
$b' \in {C^c}_h$	Returning point in cluster h
Parameters	
d_{ij}	The traveling distance (cost) between meeting points i and j
q_i	The demand of node $i, i \in N\{0\}$
s_f	The speed of vehicle f

L_f	The capacity of vehicle f
e_i	The earliest arriving time at a meeting point i
l_i	The latest arriving time at a meeting point j
pc_i	The parking cost at a meeting point i
Decision variables	
$a_i \geq 0$	The arrival time at a meeting point i
$x_{ijf} \in \{0,1\}$	If the traveling path between node i to node j is traveled by a vehicle f, it is equal to 1, and 0 otherwise
$y_i \in \{0,1\}$	If the node i in a cluster is visited by a vehicle f, it is equal to 1, and 0 otherwise

The MILP model of the R-AMPLP is presented as follows:

$$minimize \sum_{i\in N\setminus C^c}\sum_{j\in N\setminus C^c}\sum_{f\in V} d_{ij}x_{ijf} + \sum_{i\in N\setminus\{0,C^c\}} pc_i y_i + \sum_{C_h\in C}(d_{bi} + d_{ib'})y_i \quad (1)$$

subject to:

$$\sum_{i\in C_h} y_i = 1,\ \forall\, C_h \in C, \quad (2)$$

$$\sum_{j\in N\setminus\{C^c\}}\sum_{f\in V} x_{ijf} = y_i, \forall\, i \in N\setminus\{C^c\}, i \neq j, \quad (3)$$

$$\sum_{j\in N\setminus\{C^c\}}\sum_{f\in V} x_{jif} = y_i, \forall\, i \in N\setminus\{C^c\}, i \neq j, \quad (4)$$

$$\sum_{j\in N\setminus\{C^c\}}\sum_{f\in V} x_{ijf} - \sum_{j\in N\setminus\{C^c\}}\sum_{f\in V} x_{jif} = 0, \forall\, i \in N\setminus\{0,C^c\}, i \neq j, \quad (5)$$

$$\sum_{i\in N\setminus\{0,C^c\}, i\neq 0} x_{0if} \leq 1, \forall\, f \in V, \quad (6)$$

$$\sum_{i\in N\setminus\{0,C^c\}, i\neq 0} x_{i0f} \leq 1, \forall\, f \in V, \quad (7)$$

$$\sum_{i\in N\setminus\{C^c\}} q_i \sum_{j\in N\setminus\{C^c\}} x_{ijf} \leq L_f, \forall\, f \in V, \quad (8)$$

$$e_i \leq a_i \leq l_i, \forall\, i \in N\setminus\{C^c\}, \quad (9)$$

$$a_i - a_j + (d_{ij}/s_f) \leq l_i y_i - e_j y_j - (l_i - e_j)x_{ijf}, \forall\, (i,j) \in N\setminus\{C^c\}, \forall\, f \in V, \quad (10)$$

$$a_i + (d_{i0}/s_f)x_{i0f} \leq l_0, \forall\, i \in N\setminus\{C^c\}, \forall\, f \in V, \quad (11)$$

$$x_{ijf} \in \{0,1\}, \forall (i,j) \in A, \forall\, f \in V, \quad (12)$$

$$y_i \in \{0,1\}, \forall\, i \in N\setminus\{C^c\}, \quad (13)$$

$$a_i \in \mathbb{R}^+, \forall\, i \in N\setminus\{C^c\}. \quad (14)$$

The objective function (1) minimizes the total traveling and related parking costs. We assume that the traveling distance corresponding to each route defines its cost. Hence, we do not add any new parameter to calculate the cost. The first summation shows the total traveling cost associated with trucks' routes. The second summation shows the total parking cost that the system must pay. Finally, the third summation corresponds to the total traveling cost associated with AMPLs' paths from their leaving points to the meeting points and from the meeting points to the returning points. Constraints (2)–(4) ensure that

exactly one node of each cluster is served by a vehicle. Constraint (5) is a flow constraint that guarantees each cluster has the same entry and exit re-supply point. Constraints (6) and (7) ensure that routes must start and end at the depot. Constraint (8) ensures that each route's total demand cannot exceed the assigned vehicle's capacity. Constraints (9)–(11) are time window constraints, in which constraint (10) ensures that the service time associated with each meeting point must be within its time window, and constraint (11) guarantees that arrival and traveling times are consistent.

Moreover, using these constraints, sub-tours are eliminated. Constraint (9) ensures that each tour ends before the end of the working time. Finally, Constraints (12)–(14) present the domains of decision variables. Constraints (2)–(4) and (9)–(11) are based on a study on the generalized traveling salesman problem (GTSPTW) presented by [28], but we adapted them to consider a specific vehicle index in the formulation.

4. The Solution Approach

We initially provide the underlying mathematical principles to introduce our cluster-based simulated annealing (CSA) solution approach for the R-AMPLP. This groundwork enables our solution to manage the objective function and constraints of the problem effectively. For simplicity, we assume from now on that AMPLs' paths cover meeting points with no need to change their route, so the third element in the objective function (1) becomes zero. Regarding the mathematical foundation of the problem, we now present the related equations by which a given objective function, $\mathcal{F}$, must be minimized, subject to a set of constraints. A possible solution is shown by ω, and $\mathcal{F}(\omega)$ represents the value or fitness of the given solution, ω. Assume that a set of meeting points $C = N\backslash C^c = \{0, 1, 2, \dots, m\}$, clusters $C = \{C_1, \dots, C_p\}$, trucks $\mathrm{V} = \{1, \dots, k\}$, and dummy nodes $\mathcal{L} = \{m + 1, \dots, m + k - 1\}$ are given. All possible solutions are shown by set $\mathfrak{V}$. A solution, $\omega \in \mathfrak{V}$, of this optimization problem can be arranged in a row with $S = p + |\mathcal{L}|$ columns, so $|\omega| = S$. Each column is represented by $\mathcal{g}^{\omega}(v) = \varphi, \omega \in \mathfrak{V}, v \in \mathcal{L} \cup C, \varphi \in \{1, \dots, S\}$. As an example, a solution with $m = 20, |C_h| = 4, \forall h \in 1, \dots, p = 5,$ and $k = 2$ is shown in Figure 2. Therefore, in this instance, $|\omega|$ is equal to $S = 5 + 1 = 6$.

Clusters (p = 5)	Meeting points (m = 20)			
1	$v = 1$	v =2	v =**3***	v =4
2	$v = 5$	v =6	v =**7***	v =8
3	$v = 9$	v =**10***	v =11	v =12
4	v =13	v =**14***	v =15	v =16
5	v =17	v =18	v =**19***	v =20

* Selected meeting points from each cluster in a solution.

Truck 1 Truck 2

φ =1	φ =2	φ =3	φ =4	φ =5	φ =6
v =3	v =7	v =10	v =21	v =14	v =19
↓	↓	↓	↓	↓	↓
$\mathcal{g}(3)$ = 1	$\mathcal{g}(7)$ = 2	$\mathcal{g}(10)$ = 3	$\mathcal{g}(21)$ = 4	$\mathcal{g}(14)$ = 5	$\mathcal{g}(19)$ = 6

Figure 2. A solution with $m = 20$, $p = 5$, and $k = 2$.

Let $A = (m + 1) \times (m + 1)$ be a squared matrix, where d_{uv} is the traveling cost between each pair, $u, v \in N\backslash C^c, u \neq v$. To be more precise, assume a zone with three physical parking slots. All three parking slots are available in a single time slot with a one-and-

a-half-hour course between 12: 00 and 14: 00. This timeframe is separated by three distinct shorter time windows (e.g., 12: 30– 13: 00, 13: 01– 13: 30, and 13: 31– 14: 00). It means that this zone must be served only in the mentioned time slot, from 12:30 to 14:00. Hence, in our methodology, we create $3 \times 3 = 9$ virtual nodes or meeting points associated with this zone.

4.1. Cost functions

Here, we present a set of functions to obtain an objective function value.

4.1.1. Total Traveling Cost Formula

Let ψ_f^ω be the total traveling cost corresponding to the distance associated with each truck, $f \in \{1, .., k\}$. To calculate ψ_f^ω, the current solution, ω, must be broken into a set of $\omega_f, \bigcup_{f=1}^{k} \omega_f = \omega \backslash \mathcal{L}$. Since in ω solution, there are dummy nodes to separate each truck's tour, we only need to use $v \in \omega \backslash \mathcal{L}$. Hence, nodes that are between dummy nodes are used to calculate the cost function, ψ_f^ω. In the example above, node 21 is a dummy node. Therefore, all nodes before this node are visited by truck 1, and all nodes after that are served by truck 2. If ω represents a solution, the following Formula (15) shows the total traveling cost, Ψ^ω, associated with the current solution:

$$\forall f: \psi_f^\omega = \sum_{u \in \omega \backslash \mathcal{L}} \sum_{v \in \omega \backslash \mathcal{L}} d_{uv} + \sum_{v \in \omega \backslash \mathcal{L}} d_{0,v} + \sum_{u \in \omega \backslash \mathcal{L}} c_{u,0} \rightarrow \Psi^\omega = \sum_{f \in \mathrm{V}} \psi_f. \quad (15)$$

where $u, v \notin \mathcal{L}$, $d_{0,v}$ represents the distance from the depot or node 0 to the first meeting point, and $d_{u,0}$ shows the distance from the last meeting point to the depot or node 0 corresponding with vehicle f. In the presented example, regarding the truck 1, $d_{0,v} \rightarrow d_{0,3}$ and $d_{u,0} \rightarrow d_{10,0}$ are the distances from the depot to the first meeting point and from the last meeting point to the depot. Acknowledging the constancy of the overall distance that AMPLs must traverse to reach their designated meeting points, we regard this component as a fixed constant within our analysis. Consequently, this distance is excluded from the mathematical formulation underlying our heuristic strategy and the computation of numerical outcomes. This decision streamlines the model by focusing on variable distances directly influenced by routing decisions, thereby enhancing the clarity and effectiveness of our heuristic approach in optimizing traveling costs. As a justification, this element does not affect the total cost calculation, as the distance matrix of leaving points to all possible meeting points and from meeting points to returning points for each cluster is already available.

4.1.2. Capacity Violation and Penalty Cost Formula

Let q_v be the demand associated with meeting point v, and L_f be the total capacity corresponding to the vehicle, $f \in \{1, .., k\}$. If ω represents a solution, the following Formula (16) shows the penalty cost, $\mathcal{M}^\omega$, associated with capacity violation of the current solution:

$$\forall f: \delta_f^\omega = \max\{0, L_f - \sum_{v \in \omega \backslash \mathcal{L}} q_v\} \rightarrow \mathcal{M}^\omega = \sum_{f \in \mathrm{V}} \delta_f^\omega, \quad (16)$$

where $\delta_f^\omega \geq 0$ represents the amount of capacity violation corresponding with vehicle f.

4.1.3. Time Window Violation and Its Penalty Cost Formula

Let e_v, l_v, a_v be the earliest time, latest time, and arrival time at the meeting point $v \in \omega \backslash \mathcal{L}$, respectively. If ω represents a solution, the following Formula (17) shows the penalty cost, Γ^ω, associated with a time window violation of the current solution:

$$\forall v \in \omega \backslash \mathcal{L} \rightarrow \gamma_v^\alpha = \max\{0, e_v - a_v\}$$

$$\forall v \in \omega \backslash \mathcal{L} \rightarrow \gamma_v^\beta = \max\{0, a_v - l_v\}$$

$$\rightarrow \Gamma^\omega = \sum_{v \in \omega \backslash \mathcal{L}} (\gamma_v^\alpha + \gamma_v^\beta), \quad (17)$$

where γ_v^{α} and $\gamma_v^{\beta} \geq 0$ represent the number of time window violations corresponding with the earliest time and latest time associated with meeting point $v \in \omega \backslash \mathcal{L}$, respectively.

4.1.4. Parking Slot Cost Formula

Let $\rho_v, v \in \omega \backslash \mathcal{L}$ be the parking slot fee associated with meeting point v. If ω represents a solution, the following Formula (18) shows the penalty cost, P^{ω}, associated with the current solution's total parking fee:

$$\forall\, v \in \omega \backslash \mathcal{L} \rightarrow\ \mathrm{P}^{\omega} = \sum_{v \in \omega \backslash \mathcal{L}} \rho_v, \tag{18}$$

where $\rho_v \geq 0$ represents the parking fee corresponding to parking slot $v \in \omega \backslash \mathcal{L}$.

Objective function:

The total cost corresponding to the solution ω is presented in Equation (19). We base our assumption on the idea that the traveling distance for each route determines its associated cost. Therefore, we do not introduce any additional parameters for cost calculation.

$$\mathcal{F}(\omega) = \Psi^{\omega} + \mathcal{M}^{\omega} + \Gamma^{\omega} + \mathrm{P}^{\omega} \tag{19}$$

Regarding the optimal solution, ω^*, the objective function value, $\mathcal{F}(\omega^*)$, shows the minimum cost of the system. More precisely,

$$\omega^* \ \overleftarrow{\omega \in \mho}\ \min \mathcal{F}(\omega). \tag{20}$$

As mentioned in this section, $\mho$ represents all possible solutions in which $0 \notin \mho$.

4.2. A Cluster-Based Simulated Annealing Algorithm

This section presents the procedure of a cluster-based simulated annealing (CSA) algorithm. According to [29] and [30], using simulated annealing (SA) with a set of neighborhood search operators in the context of the VRP and its variants is successful.

We first provide a general rule for the CSA and then step-by-step define sub-algorithms in three main categories: clustering, initial solution, and improvement algorithms. In Algorithm 1, we define steps to solve the problem by applying the CSA algorithm. This algorithm uses three phases to deliver a solution. The first phase is to obtain data from Algorithm 2, by which nodes are grouped into distinct clusters by applying a K-means algorithm. The second phase creates an initial solution using Algorithm 3 based on the output of Algorithm 2. Finally, by calling Algorithm 4 and its sub-functions (see Appendix A) improvement algorithms, Algorithm 1 tries to find an optimal solution based on a standard simulated annealing step, as shown in Algorithm 1.

Algorithm 1. The cluster-based simulated annealing algorithm (CSA)

```
//Initialization phase creates an initial solution (instance) by calling
Algorithm 3;
ω^best ← ω;
calculate the fitness value (ω);
F(ω^best) ← F(ω);
T ← T0;
Tf ← 0;
//improvement phase
for externaliteration=1 to EImax do
    if T>Tf then
        for internaliteration=1 to IImax do
            ω^new ←createneighborhoodsolutions(ω);
```

$F(\omega^{new}) \leftarrow calculatethefitnessvalue(\omega^{new})$;
if $F(\omega^{new}) <= F(\omega)$ **then**
$\omega \leftarrow \omega^{new}$;
else
$\Delta = T_0 - T_f$;
$\eta = e^{(\Delta/T)}$;
if $random(0,1) <= \eta$ **then**
$\omega \leftarrow \omega^{new;}$
end
end
end
end
if $F(\omega) <= F(\omega^{best})$ **then**
$\omega^{new} \leftarrow \omega$;
end
$T = \varepsilon T$;
end

In Algorithm 2, a given dataset is transformed into a dataset with clusters grouped into distinct clusters based on clustering criteria, such as nodes' coordinates. In this algorithm, we consider the coordinates of nodes as grouping criteria by applying a K-means procedure. This algorithm is based on the silhouette method [31]. The method is a widely used technique in cluster analysis, particularly in the context of unsupervised machine learning and data mining. It serves as a valuable tool for evaluating the quality of clustering results. The primary goal of clustering is to partition a dataset into distinct groups or clusters, each containing similar data points. The silhouette method quantitatively measures how well this goal has been achieved. Several steps are involved in computing the silhouette score for a particular data point, as follows: (i) Calculate the average distance (ϖ) between the data point and all other data points within the same cluster. This value represents the cohesion of the data point with its cluster, indicating how similar it is to the other points in the same group. (ii) Calculate the average distance (κ) between the data point and all data points in the nearest neighboring cluster that the data point is not a part of. This value measures the separation of the data point from other clusters, indicating how dissimilar it is to points in neighboring clusters. (iii) Compute the silhouette score (ς) for the data point using the formula: $\varsigma = (\kappa - \varpi)/\max(\varpi, \kappa)$.

Algorithm 2. Clustering algorithm.

Input: Nodes' coordinates, nodes' labels N

Output: A set of H clusters, $instance$

Step 1: Call data to extract horizontal and vertical coordinates.

Step 2: Analyze the maximum number of distinct clusters by calling the silhouette method.

Step 3: Select $p\ clusters$ based on the silhouette method.

Step 4: Apply the K-means method for clustering.

Step 5: Print zones (clusters) with memberships.

Concerning producing an initial solution, we apply Algorithm 3. This algorithm yields a set of nodes obtained from Algorithm 2 and then creates an initial solution randomly.

Algorithm 3. Initial random solution algorithm.

Input : An instance

Output : An initial solution, ω

function create initial solution (*instance*)

$N \leftarrow N^{instance}$;

$p \leftarrow p^{instance}$;

$k \leftarrow k^{instance}$;

$|\omega| \leftarrow p + k - 1$;

$\omega \leftarrow 0$;

for *g=1* to *p* **do**

    cluster(g)←random_sampling(g,1);

end

ω←[permutation([p+1:p+k-1],[cluster(g) from 1 to p])];

To improve the initial solution, Algorithm 4 is applied. The algorithm employs six internal operators that are invoked randomly to enhance the initial solution. A detailed explanation of these operators can be found in Appendix A. Each operator is linked to a distinct selection function, which identifies a group of nodes and generates new routes.

Algorithm 4. Create neighborhoods algorithm.

Input: An initial solution, ω

Output: A new solution, ω^{new}

function *create neighborhoods (ω)*

r ←random_integer_number[1,6]

switch r **do**

    case 1 **do**

        ω^{new} *←2_opt(ω);*

    end

    case 2 **do**

        ω^{new} *←remove_insert1(ω);*

    end

    case 3 **do**

        ω^{new} *←shake_cluster(ω);*

    end

    case 4 **do**

        ω^{new} *←3_opt(ω);*

    end

    case 5 **do**

        ω^{new} *←remove_insert_2(ω);*

```
        end
        case 6 do
            ω^new ←reverse(ω);
        end
    end
```

5. Computational Experiments

This section demonstrates the computational outcomes achieved by solving various instances, adapted according to the methodology described in this paper. It also offers substantial proof of the effectiveness of the CSA.

5.1. Dataset and Strategy Definition

Our experiments used the Solomon dataset [32]. It is important to note that for obtaining multiple time windows associated with each node, we defined four internal strategies, including *25-nodes-per-set* (*25nps*), *50-nodes-per-set* (*50nps*), *75-nodes-per-set* (*75nps*), and *100-nodes-per-set* (*100nps*). These strategies indicate how many nodes from each dataset with which range were selected. For example, regarding class C1, 25nps means we chose the first 25 nodes. We applied this approach to create a wide range of instances, and each dataset, such as c101 from class C1, was divided into four strategies as instances.

Moreover, we designed three and four scenarios to combine strategies for each class, C1, C2, R1, R2, RC1, and RC2, as explained in the following tables. For instance, scenario c101—c103 means combining internal strategies from c101 to c103. The number of scenarios depended on the dataset class, as depicted in Tables 4, 6, 8, and 10. In fact, there were four scenarios for classes R1 and R2, and for the rest, there were three. Our instances had a specific ID based on both scenarios and strategies mentioned before. For example, an instance with ID "c101—103:25nps" indicates that it called the first 25 nodes from each dataset from c101 to c103, creating 75 meeting points. The reason for this approach was that it created multiple time windows for several nodes. As each dataset, such as C1, had identical coordinates but different time windows associated with each node, c101—103:25nps represents 25 alternative locations, with 3 time windows for each node. The number of clusters was obtained by applying Algorithm 2.

5.2. Performance Metrics and Parameters

The performance metrics included the number of nodes ($|N|$), the number of capacitated arcs ($|C|$), demand (q_i), vehicle capacity (L_f), the number of time windows (NTWs), and the average solutions. Moreover, based on the length of the value of the earliest time associated with each node, we assigned a parking slot fee between 1 and 4. Particularly, we divided the maximum earliest time windows into four intervals. Then, we considered the earliest time windows associated with each node to assign the parking fee. If it was below the first interval, then we assigned 1, and so on. Therefore, the parking slot fee was calculated for the total cost of each instance. Moreover, as we already mentioned, we assumed that the third summation in the objective function was a constant in our calculations, so we could ignore it in the final results.

Regarding the CSA parameters, we set them at $EI_{max} = 150{,}000, II_{max} = 50, T_0 = 100, \varepsilon = 0.9999,$ and $T_f = 0$, empirically. To confirm that the CSA is a reliable algorithm, we solved a set of small and medium-sized instances using the IBM CPLEX solver [33], as shown in Table 16. We selected the first dataset from each scenario corresponding to each class. All instances were run on a 64-bit Windows system with 8 GB RAM and Core i5 CPU. The maximum computational time for CPLEX was set to 3600 s. Here, parking fees were disregarded to emphasize the contrast between RRAMPL and the original VRPTW.

To illustrate the impact of clustering and to assess the effects of various factors on our defined problem, we presented outcomes dually linked to each class. Initially, we

showcased the outcomes of the R-AMPLP, a solution yielded through CSA. Subsequently, we offered results for the fundamental VRPTW problem, maintaining an equal count of clusters and nodes. For instance, in C1, there were nine different subsets, C1–C9, each with 25, 50, and 100 nodes. For this, in CSA, we set the number of clusters to the size of the instance (nodes) and obtained the objective value related to the dataset. The results are shown in Tables 5, 7, 9, and 11. It is important to note that no parking slot fee was associated with VRPTW instances.

5.3. Results of the R-AMPLP and VRPTW

Focusing on the results, we compared the R-AMPLP solution with the VRPTW. This comparison highlighted the CSA algorithm's efficacy in reducing operational costs and demonstrated the impact of clustering on solution quality.

Regarding class C1, as depicted in Table 4, identical strategies employed across distinct scenarios—such as *25nps* in c101—c103 and c101—c106—exhibited comparable cluster counts and average solutions. This observation was evident from the graphical representation, where the time window variability in classes c103 and c104 was noticeably more uneven than the rest. These findings were further substantiated by the results presented in Table 5 for the VRPTW.

Table 4. Numerical results based on Solomon's dataset C1.

Class	Scenario	Strategy	ID	$\|N\|$	$\|C\|$	q_i	L_f	NTWs	Average of Solutions
C1	c101—c103	*25nps*	*c101-c103:25nps*	75	3	20	60	3	83.48
		50nps	*c101-c103:50nps*	150	6	20	60	3	231.58
		75nps	*c101-c103:75nps*	225	9	20	60	3	466.84
		100nps	*c101-c103:100nps*	300	10	20	60	3	478.41
	c101—c106	*25nps*	*c101-c106:25nps*	150	3	20	60	6	82.48
		50nps	*c101-c106:50nps*	300	7	20	60	6	266.94
		75nps	*c101-c106:75nps*	450	10	20	60	6	495.03
		100nps	*c101-c106:100nps*	600	10	20	60	6	470.24
	c101—c109	*25nps*	*c101-c109:25nps*	225	3	20	60	9	71.12
		50nps	*c101-c109:50nps*	450	6	20	60	9	243.29
		75nps	*c101-c109:75nps*	675	8	20	60	9	446.26
		100nps	*c101-c109:100nps*	900	10	20	60	9	488.05
Average	-	-	-	375	8.5	20	60	6	318.64

The Pearson correlation coefficient of 0.985 indicated a strong positive correlation, suggesting that as the number of clusters increased, there was a tendency for the average objective values to increase as well. The statistical significance of this correlation was supported by a *p*-value of approximately 6.35×10^{-9}, indicating that the observed relationship was not due to random chance.

Table 5. Numerical results for confirming the CSA performance compared to optimal solutions on Solomon's dataset C1.

Dataset	Subset	$\|N\|$	$\|C\|$	Optimal (Benchmark)	CSA Solution	Gap %
C1	c101	25	25	191.3	191.81	0.27%
	c101	50	50	362.4	363.24	0.23%
	c101	100	100	827.3	828.93	0.20%
	c102	25	25	190.3	190.73	0.23%
	c102	50	50	361.4	362.17	0.21%
	c102	100	100	827.3	828.93	0.20%

	c103	25	25	190.3	190.73	0.23%
	c103	50	50	361.4	362.17	0.21%
	c103	100	100	826.3	834.75	1.02%
	c104	25	25	186.9	187.45	0.29%
	c104	50	50	358	358.88	0.25%
	c104	100	100	822.9	834.94	1.46%
	c105	25	25	191.3	191.81	0.27%
	c105	50	50	362.4	363.24	0.23%
	c105	100	100	827.3	828.93	0.20%
	c106	25	25	191.3	191.81	0.27%
	c106	50	50	362.4	363.24	0.23%
	c106	100	100	827.3	828.93	0.20%
	c107	25	25	191.3	191.81	0.27%
	c107	50	50	362.4	363.24	0.23%
	c107	100	100	827.3	828.93	0.20%
	c108	25	25	191.3	191.81	0.27%
	c108	50	50	362.4	363.24	0.23%
	c108	100	100	827.3	828.94	0.20%
	c109	25	25	191.3	191.81	0.27%
	c109	50	50	362.4	363.92	0.42%
	c109	100	100	827.3	828.94	0.20%
			Average	459.65	463.45	0.36%

Tables 4 and 5 show that the objective value increased when the number of nodes and clusters increased. The mean overall cost of R-AMPLP within this specific class, denoted as C1, was 318.64. This value contrasted notably with the VRPTW context, where the cost was 459.65. As such, it can be deduced that the R-AMPLP demonstrated enhanced cost efficiency compared to the original VRPTW, mainly when clustering was a viable option within the given configuration. The performance shown by the introduced algorithm for this class revealed an exceedingly slight gap compared to the exact solutions associated with it.

Regarding class C2, Table 6 shows that the average objective values were related to the number of clusters. That is, scenarios with more clusters had greater solution values. In addition, the length of time window intervals significantly impacted the cost values. Although the coordinates corresponding to each node were the same in both class C1 and C2, the length of the time window intervals was tighter in class C1, and solution values regarding class C1 were greater than class C2 on average.

Table 6. Numerical results based on Solomon's dataset C2.

Class	Scenario	Strategy	ID	$\|N\|$	$\|C\|$	q_i	L_f	NTWs	Average of Solutions
C2	c201—c203	*25nps*	*c201-c203:25nps*	75	8	20	60	3	173.50
		50nps	*c201-c203:50nps*	150	9	20	60	3	213.34
		75nps	*c201-c203:75nps*	225	10	20	60	3	373.65
		100nps	*c201-c203:100nps*	300	8	20	60	3	294.02
	c201—c206	*25nps*	*c201-c206:25nps*	150	9	20	60	6	224.05
		50nps	*c201-c206:50nps*	300	5	20	60	6	145.77
		75nps	*c201-c206:75nps*	450	8	20	60	6	221.37
		100nps	*c201-c206:100nps*	600	10	20	60	6	341.01
	c201—c208	*25nps*	*c201-c208:25nps*	200	10	20	60	8	266.68

		50nps	*c201-c208:50nps*	400	7	20	60	8	204.00
		75nps	*c201-c208:75nps*	600	10	20	60	8	306.95
		100nps	*c201-c208:100nps*	800	7	20	60	8	216.77
Average	-	-	-	354.16	8.41	20	60	5.66	248.42

The statistical analysis of Table 6 from Solomon's dataset C2 showed a positive correlation between the number of clusters ($|C|$) and the average objective values (average of solutions), with a correlation coefficient of approximately 0.774. This indicated a moderate to strong correlation. The *p*-value was approximately 0.0031, suggesting this correlation was statistically significant.

Regarding Table 7, as the results revealed, the performance of the CSA compared to the optimal solutions was worse than that of class C1 in Table 5. Consequently, the average gap in class C2 was greater than that in class C1, which means that the CSA worked better with the same solution resources in cases with tighter time windows, such as C1. Also, for small instances, the CSA quality was near optimal.

Table 7. Numerical results for confirming the CSA performance compared to benchmarks on Solomon's dataset C2.

Dataset	Subset	$\|N\|$	$\|C\|$	Optimal (Benchmark)	CSA Solution	Gap %
C2	c201	25	25	214.7	215.54	0.39%
	c201	50	50	360.2	361.79	0.44%
	c201	100	100	589.1	673.16	14.27%
	c202	25	25	214.7	215.54	0.39%
	c202	50	50	360.2	395.81	9.89%
	c202	100	100	589.1	674.29	14.46%
	c203	25	25	214.7	215.54	0.39%
	c203	50	50	359.8	393.98	9.50%
	c203	100	100	588.7	674.95	14.65%
	c204	25	25	213.1	215.54	1.15%
	c204	50	50	350.1	393.37	12.36%
	c204	100	100	588.1	640.59	8.93%
	c205	25	25	214.7	215.54	0.39%
	c205	50	50	359.8	364.75	1.38%
	c205	100	100	586.4	652.41	11.26%
	c206	25	25	214.7	215.54	0.39%
	c206	50	50	359.8	361.41	0.45%
	c206	100	100	586	642.57	9.65%
	c207	25	25	214.5	215.54	0.48%
	c207	50	50	359.6	402.32	11.88%
	c207	100	100	585.5	675.75	15.41%
	c208	25	25	214.5	215.54	0.48%
	c208	50	50	350.5	352.12	0.46%
	c208	100	100	585.5	637.07	8.81%
Average	-	-	-	386.41	417.53	8.05%

Table 8 shows that scenarios with larger clusters had greater solution values regarding dataset R1. The results revealed that the effect of the number of clusters on solution values was considerable.

Table 8. Numerical results based on Solomon's dataset R1.

Class	Scenario	Strategy	ID	$\|N\|$	$\|C\|$	q_i	L_f	NTWs	Average of Solutions
R1	r101-r103	*25nps*	*r101-r103:25nps*	75	10	20	60	3	276.05
		50nps	*r101-r103:50nps*	150	4	20	60	3	88.46
		75nps	*r101-r103:75nps*	225	5	20	60	3	102.31
		100nps	*r101-r103:100nps*	300	4	20	60	3	78.92
	r101-r106	*25nps*	*r101-r106:25nps*	150	9	20	60	6	253.62
		50nps	*r101-r106:50nps*	300	10	20	60	6	264.91
		75nps	*r101-r106:75nps*	450	4	20	60	6	78.84
		100nps	*r101-r106:100nps*	600	4	20	60	6	78.81
	r101-r109	*25nps*	*r101-r109:25nps*	225	9	20	60	9	284.7
		50nps	*r101-r109:50nps*	450	8	20	60	9	240.79
		75nps	*r101-r109:75nps*	675	4	20	60	9	76.25
		100nps	*r101-r109:100nps*	900	4	20	60	9	65.59
	r101-r112	*25nps*	*r101-r112:25nps*	300	8	20	60	12	207.54
		50nps	*r101-r112:50nps*	600	4	20	60	12	77.25
		75nps	*r101-r112:75nps*	900	3	20	60	12	34.26
		100nps	*r101-r12:100nps*	1200	4	20	60	12	68.69
Average	-	-	-	468.75	5.87	20	60	7.5	142.31

The analysis of Table 8, using Solomon's dataset R1, demonstrated a notably strong positive correlation between the number of clusters ($|C|$) and the average objective values, as indicated by a correlation coefficient near 0.989. This reflected a nearly ideal correlation. With a *p*-value of around 5.35×10^{-13}, the statistical significance of this correlation was extremely high, confirming its reliability.

Table 9 asserts that increasing the number of nodes impacted both objective function values and, in most cases, the gap between optimal solutions and the CSA. However, an improvement was achieved in the gap regarding class C1. We can interpret this improvement based on the structure of class R1, in which nodes were scattered randomly in the geographical area. On the other hand, as the numerical results of the R-AMPLP presented, applying the clustering approach could reduce the operation cost of the system and algorithm.

Table 9. Numerical results for confirming the CSA performance compared to benchmarks on Solomon's dataset R1.

Dataset	Subset	$\|N\|$	$\|C\|$	Optimal (Benchmark)	CSA Solution	Gap %
R1	r101	25	25	617.1	627.13	1.63%
	r101	50	50	1044	1065.9	2.10%
	r101	100	100	1637.7	1699.34	3.76%
	r102	25	25	547.1	550.20	0.57%
	r102	50	50	909	923.22	1.56%
	r102	100	100	1466.6	1523.34	3.87%
	r103	25	25	454.6	464.82	2.25%
	r103	50	50	772.9	813.05	5.19%
	r103	100	100	1208.7	1289.84	6.71%
	r104	25	25	416.9	437.08	4.84%
	r104	50	50	625.4	644.07	2.99%
	r104	100	100	971.5	1043.67	7.43%
	r105	25	25	530.5	531.53	0.19%

	r105	50	50	899.3	948.36	5.46%
	r105	100	100	1355.3	1462.89	7.94%
	r106	25	25	465.4	466.48	0.23%
	r106	50	50	793	830.78	4.76%
	r106	100	100	1234.6	1299.89	5.29%
	r107	25	25	424.3	437.67	3.15%
	r107	50	50	711.1	746.12	4.92%
	r107	100	100	1064.6	1127.83	5.94%
	r108	25	25	379.3	398.29	5.01%
	r108	50	50	617.87	630.53	2.05%
	r108	100	100	-	1001.17	-
	r109	25	25	441.3	442.62	0.30%
	r109	50	50	786.8	819.21	4.12%
	r109	100	100	1146.9	1211.66	5.65%
	r110	25	25	444.1	447.47	0.76%
	r110	50	50	697	735.05	5.46%
	r110	100	100	1068	1150.07	7.68%
	r111	25	25	428.8	440.04	2.62%
	r111	50	50	707.2	740.56	4.72%
	r111	100	100	1048.7	1101.81	5.06%
	r112	25	25	393	413.44	5.20%
	r112	50	50	630.2	646.85	2.64%
	r112	100	100	-	999.19	-
Average	-	-	-	792.31 *	831.71 *	4.97%

* These results are restricted to the available optimal solutions in the benchmark.

Regarding class R2, Table 10 reveals that scenarios with larger clusters had greater solution values. The results also showed that the number of clusters' effect on solution values was more dominant than the time windows' interval length. To be more precise, when considering a given strategy across various scenarios, it became evident that the impact of the time window duration was relatively less significant than the influence exerted by the number of nodes and clusters.

Table 10. Numerical results based on Solomon's dataset R2.

Class	Scenario	Strategy	ID	$\|N\|$	$\|C\|$	q_i	L_f	NTWs	Average of Solutions
R2	r201—r203	*25nps*	*r201-r203:25nps*	75	10	20	60	3	271.9
		50nps	*r201-r203:50nps*	150	4	20	60	3	85.06
		75nps	*r201-r203:75nps*	225	5	20	60	3	100.42
		100nps	*r201-r203:100nps*	300	4	20	60	3	84.01
	r201—r206	*25nps*	*r201-r206:25nps*	150	9	20	60	6	287.70
		50nps	*r201-r206:50nps*	300	10	20	60	6	247.64
		75nps	*r201-r206:75nps*	450	4	20	60	6	79.52
		100nps	*r201-r206:100nps*	600	4	20	60	6	77.36
	r201—r209	*25nps*	*r201-r209:25nps*	225	9	20	60	9	285.15
		50nps	*r201-r209:50nps*	450	8	20	60	9	232.81
		75nps	*r201-r209:75nps*	675	4	20	60	9	81.04
		100nps	*r201-r209:100nps*	900	4	20	60	9	70.90
	r201—r211	*25nps*	*r201-r211:25nps*	275	8	20	60	11	225.76
		50nps	*r201-r211:50nps*	550	9	20	60	11	280.85

		75nps	*r201-r211:75nps*	825	3	20	60	11	34.30
		100nps	*r201-r211:100nps*	1100	4	20	60	11	72.23
Average	-	-	-	453.12	6.18	20	60	7.25	157.29

The examination of Table 10, drawn from Solomon's R2 dataset, revealed a powerful positive correlation between the count of clusters ($|C|$) and the average values of the solutions. With a correlation coefficient near 0.981, the relationship approached an almost perfect correlation. The *p*-value, approximately 2.20×10^{-11}, underscored the remarkable statistical significance of this correlation.

As the results in Table 10 show, class R1 had a shorter interval length and a smaller average solution value than class R2. In classes R1 and R2, known as scattered datasets, the average of the solution depended on the average time window interval length. This conclusion is different from that of classes C1 and C2, where dataset C1, with a shorter interval length on average than C2, had a greater average solution value. Moreover, the average solution values in classes R1 and R2 were smaller than those in C1.

Table 11 reveals the same conclusions regarding the correlation between the number of nodes and the value of the objective functions and gaps when we compared the optimal solutions with the CSA results. On the other hand, the average cost of class R2 was less than that of class R1, so having tighter time windows led to higher costs.

Table 11. Numerical results for confirming the CSA performance compared to benchmarks on Solomon's dataset R2.

Dataset	Subset	$\|N\|$	$\|C\|$	Optimal (Benchmark)	CSA Solution	Gap %
R2	r201	25	25	463.3	475.51	2.64%
	r201	50	50	791.9	836.80	5.67%
	r201	100	100	1143.2	1271.95	11.26%
	r202	25	25	410.5	421.28	2.63%
	r202	50	50	698.5	765.80	9.63%
	r202	100	100	-	1265.5	-
	r203	25	25	391.4	399.67	2.11%
	r203	50	50	605.3	618.98	2.26%
	r203	100	100	-	977.39	-
	r204	25	25	355	358.57	1.01%
	r204	50	50	506.4	530.84	4.83%
	r204	100	100	-	824.16	-
	r205	25	25	393	407.00	3.56%
	r205	50	50	690.1	723.90	4.90%
	r205	100	100	-	1074.42	-
	r206	25	25	324.0	325.10	0.34%
	r206	50	50	632.4	692.00	9.42%
	r206	100	100	-	970.02	-
	r207	25	25	361.6	392.32	8.50%
	r207	50	50	-	583.01	-
	r207	100	100	-	914.83	-
	r208	25	25	328.2	331.79	1.09%
	r208	50	50	-	510.88	-
	r208	100	100	-	778.72	-
	r209	25	25	370.7	399.21	7.69%
	r209	50	50	600.6	619.53	3.15%
	r209	100	100	-	954.34	-

	r210	25	25	404.6	431.10	6.55%
	r210	50	50	645.6	679.01	5.18%
	r210	100	100	-	959.19	-
	r211	25	25	350.9	351.9	0.28%
	r211	50	50	535.5	581.60	8.61%
	r211	100	100	-	817.02	-
Average	-	-	-	526.33 *	553.04 *	5.07% *

* These results are restricted to the available optimal solutions in the benchmark.

Regarding dataset RC1, as with previous datasets, scenarios with larger clusters had greater solution values than others, see Table 12. These results showed that the number of clusters defined the dominant element on solution values.

Table 12. Numerical results based on Solomon's dataset RC1.

Class	Scenario	Strategy	ID	$\|N\|$	$\|C\|$	q_i	L_f	NTWs	Average of Solutions
RC1	rc101—rc103	*25nps*	*rc101-rc103:25nps*	75	4	20	60	3	187.33
		50nps	*rc101-rc103:50nps*	150	5	20	60	3	235.24
		75nps	*rc101-rc103:75nps*	225	8	20	60	3	278.85
		100nps	*rc101-rc103:100nps*	300	10	20	60	3	350.11
	rc101—rc106	*25nps*	*rc101-rc106:25nps*	150	3	20	60	6	124.08
		50nps	*rc101-rc106:50nps*	300	4	20	60	6	201.48
		75nps	*rc101-rc106:75nps*	450	9	20	60	6	288.80
		100nps	*rc101-rc106:100nps*	600	10	20	60	6	316.13
	rc101—rc108	*25nps*	*rc101-rc108:25nps*	200	7	20	60	8	235.24
		50nps	*rc101-rc108:50nps*	400	5	20	60	8	255.85
		75nps	*rc101-rc108:75nps*	600	9	20	60	8	347.09
		100nps	*rc101-rc108:100nps*	800	9	20	60	8	284.67
Average	-	-	-	354.16	6.91	20	60	5.66	258.74

The analysis of Table 12 from Solomon's dataset RC1 revealed a strong positive correlation between the number of clusters ($|C|$) and the average objective values (average of solutions), with a correlation coefficient of approximately 0.916. This indicates a substantial correlation. The p-value was around 2.84×10^{-5}, suggesting that this correlation was statistically significant.

Based on Tables 12 and 13, the mean objective values within this class lay intermediate to those of classes R1, R2, and C1. This positioning arose because the configuration linked to this particular class, RC1, no longer adhered strictly to pure clustering or random scattering.

Table 13. Numerical results for confirming the CSA performance compared to benchmarks on Solomon's dataset RC1.

Dataset	Subset	$\|N\|$	$\|C\|$	Optimal (Benchmark)	CSA Solution	Gap %
RC1	rc101	25	25	461.1	464.57	0.75%
	rc101	50	50	944	968.80	2.63%
	rc101	100	100	1619.8	1721.18	6.26%
	rc102	25	25	351.8	352.74	0.27%
	rc102	50	50	822.5	890.26	8.24%
	rc102	100	100	1457.4	1539.48	5.63%

	rc103	25	25	332.8	333.91	0.33%
	rc103	50	50	710.9	753.52	6.00%
	rc103	100	100	1258	1391.77	10.63%
	rc104	25	25	306.6	307.14	0.18%
	rc104	50	50	545.8	546.51	0.13%
	rc104	100	100	-	1191.07	-
	rc105	25	25	411.3	434.29	5.59%
	rc105	50	50	855.3	894.08	4.53%
	rc105	100	100	1513.7	1649.14	8.95%
	rc106	25	25	345.5	347.26	0.51%
	rc106	50	50	723.2	814.49	12.62%
	rc106	100	100	-	1475.35	-
	rc107	25	25	298.3	298.95	0.22%
	rc107	50	50	642.7	671.77	4.52%
	rc107	100	100	1207.8	1278.37	5.84%
	rc108	25	25	294.5	295.74	0.42%
	rc108	50	50	598.1	599.17	0.18%
	rc108	100	100	1114.2	1217.62	9.28%
Average	-	-	-	764.33 *	807.76 *	5.68% *

* These results are restricted to the available optimal solutions in the benchmark.

Table 14 shows that scenarios with more clusters had greater solution values for class RC2. In addition, several clusters dominated the solution compared to the effect of the time window interval length on solution values.

Table 14. Numerical results based on Solomon's dataset RC2.

Class	Scenario	strategy	ID	$\|N\|$	$\|C\|$	q_i	L_f	NTWs	Average of Solutions
RC2	rc201—rc203	*25nps*	*rc201-rc203:25nps*	75	4	20	60	3	187.33
		50nps	*rc201-rc203:50nps*	150	5	20	60	3	238.58
		75nps	*rc201-rc203:75nps*	225	8	20	60	3	297.67
		100nps	*rc201-rc203:100nps*	300	10	20	60	3	357.66
	rc201—rc206	*25nps*	*rc201-rc206:25nps*	150	3	20	60	6	143.4
		50nps	*rc201-rc206:50nps*	300	4	20	60	6	209.03
		75nps	*rc201-rc206:75nps*	450	9	20	60	6	313.10
		100nps	*rc201-rc206:100nps*	600	10	20	60	6	347.58
	rc201—rc208	*25nps*	*rc201-rc208:25nps*	200	7	20	60	8	240.23
		50nps	*rc201-rc208:50nps*	400	5	20	60	8	250.89
		75nps	*rc201-rc208:75nps*	600	9	20	60	8	359.58
		100nps	*rc201-rc208:100nps*	800	9	20	60	8	369.96
Average	-	-	-	354.16	6.91	20	60	5.66	276.25

The examination of Table 14, based on Solomon's RC2 dataset, showed a significantly strong positive correlation between the number of clusters ($|C|$) and the average objective values, evidenced by a correlation coefficient of about 0.951. This demonstrates a substantial and meaningful correlation. The p-value, approximately 2.05×10^{-6}, emphasized the statistical importance of this correlation.

In class RC1, as Table 15 shows, the average gap was less than that in class C2 but greater than that in classes C1, R1, and R2. In fact, as in class RC1 and RC2 the dispersing method was no longer random and clustered, the results were also in the middle of those

approaches. Regarding the general insights from Table 15, the numbers of objective function values and gaps increased when the number of nodes increased in most cases.

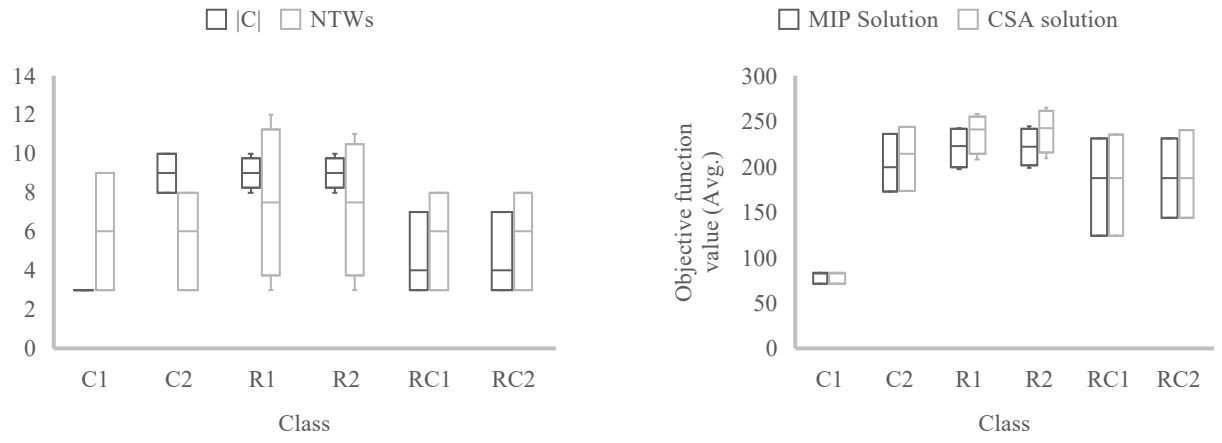
Table 15. Numerical results for confirming the CSA performance compared to benchmarks on Solomon's dataset RC2.

Class	Subset	$\|N\|$	$\|C\|$	Optimal (Benchmark)	CSA Solution	Gap %
RC2	rc201	25	25	360.2	361.24	0.29% *
	rc201	50	50	684.8	686.31	0.22%
	rc201	100	100	1261.8	1395.70	10.61%
	rc202	25	25	338	338.82	0.24%
	rc202	50	50	613.6	665.53	8.46%
	rc202	100	100	1092.3	1233.72	12.95%
	rc203	25	25	326.9	328.44	0.47%
	rc203	50	50	555.3	625.60	12.66%
	rc203	100	100	-	1082.26	-
	rc204	25	25	299.7	315.95	5.42%
	rc204	50	50	444.2	462.98	4.23%
	rc204	100	100	-	847.32	-
	rc205	25	25	338	338.93	0.28%
	rc205	50	50	630.2	631.98	0.28%
	rc205	100	100	1154	1241.2	7.56%
	rc206	25	25	324	325.10	0.34%
	rc206	50	50	610	611.75	0.29%
	rc206	100	100	-	1190.45	-
	rc207	25	25	298.3	298.95	0.22%
	rc207	50	50	558.6	561.42	0.50%
	rc207	100	100	-	1081.70	-
	rc208	25	25	269.1	269.56	0.17%
	rc208	50	50	-	491.46	-
	rc208	100	100	-	810.35	-
Average	-	-	-	564.38 *	594.06 *	5.26% *

* These results are restricted to the available optimal solutions in the benchmark.

In Table 15, objective values increased while we increased the number of nodes. On the other hand, regarding classes R1 and R2, the average gap and cost of the RC2 were greater than those in both classes R1 and R2. Here, again, we can recognize that tighter time windows produced higher costs. The results from RC2 with wider time windows showed that the average cost was less than the average cost in RC1.

As a comprehensive class comparison, Figure 3 shows that R1 and R2 had smaller objective values than RC1, RC2, C1, and C2. In fact, after applying the clustering method, pure scattered datasets had the smallest objective values. In addition, the average number of clusters in classes R1 and R2 was smaller than that in C1, C2, RC1, and RC2. Also, the average number of clusters in classes RC1 and RC2 was smaller than that in C1 and C2.



(**a**) (**b**)

Figure 3. Visual representation of R-AMPLP outcomes. Part (**a**) illustrates the distribution of clusters for each class, while part (**b**) displays the distribution of objective values (OFV) across different classes.

5.4. Comparison with CPLEX Solutions

We compared the performance of the CSA algorithm with CPLEX on a set of small and medium-sized datasets that were solved before, as shown in Table 16.

By comparing CSA solutions with those obtained from the CPLEX solver, this section evidences the reliability and effectiveness of our proposed algorithm.

Table 16. Numerical results needed to confirm the CSA performance, solved by CPLEX.

Class	Scenario	Strategy	ID	$\|N\|$	$\|C\|$	q_i	L_f	NTWs	MIP Solution	CSA Solution	Gap %
C1	*c101-c103*	*25nps*	*c101-c103:25nps*	75	3	20	60	3	83.48	83.48	0.00%
C1	*c101-c106*	*25nps*	*c101-c106:25nps*	150	3	20	60	6	82.48	82.48	0.00%
C1	*c101-c109*	*25nps*	*c101-c109:25nps*	225	3	20	60	9	71.12	71.12	0.00%
C2	*c201-c203*	*25nps*	*c201-c203:25nps*	75	8	20	60	3	172.81	173.5	0.40%
C2	*c201-c206*	*25nps*	*c201-c206:25nps*	150	9	20	60	6	199.38	214.39	7.53%
C2	*c201-c208*	*25nps*	*c201-c208:25nps*	200	10	20	60	8	236.03	243.67	3.24%
R1	*r101-r103*	*25nps*	*r101-r103:25nps*	75	10	20	60	3	242.32	246.33	1.65%
R1	*r101-r106*	*25nps*	*r101-r106:25nps*	150	9	20	60	6	207.00	234.7	13.38%
R1	*r101-r109*	*25nps*	*r101-r109:25nps*	225	9	20	60	9	238.70	257.61	7.92%
R1	*r101-r112*	*25nps*	*r101-r112:25nps*	300	8	20	60	12	197.00	207.54	5.35%
R2	*r201-r203*	*25nps*	*r201-r203:25nps*	75	10	20	60	3	244.32	264.52	8.27%
R2	*r201-r206*	*25nps*	*r201-r206:25nps*	150	9	20	60	6	209.32	233.73	11.66%
R2	*r201-r209*	*25nps*	*r201-r209:25nps*	225	9	20	60	9	234.34	250.73	6.99%
R2	*r201-r211*	*25nps*	*r201-r211:25nps*	275	8	20	60	11	198.63	209	5.22%
RC1	*rc101-rc103*	*25nps*	*rc101-rc103:25nps*	75	4	20	60	3	187.33	187.33	0.00%
RC1	*rc101-rc106*	*25nps*	*rc101-rc106:25nps*	150	3	20	60	6	124.08	124.08	0.00%
RC1	*rc101-rc108*	*25nps*	*rc101-rc108:25nps*	200	7	20	60	8	231.23	235.24	1.73%
RC2	*rc201-rc203*	*25nps*	*rc201-rc203:25nps*	75	4	20	60	3	187.33	187.33	0.00%
RC2	*rc201-rc206*	*25nps*	*rc201-rc206:25nps*	150	3	20	60	6	143.39	143.39	0.00%
RC2	*rc201-rc208*	*25nps*	*rc201-rc208:25nps*	200	7	20	60	8	231.23	240.23	3.89%
Average									186.07	194.52	4.54%

The results showed that the outputs from CPLEX could confirm the output of CSA. Moreover, the important factor in the gap between CSA and CPLEX was the number of clusters and nodes in each instance (ID). The findings indicated that the algorithm's performance was notably superior for class C1 compared to the remaining classes. Figure 4 depicts a graphical presentation of the CSA algorithm's performance compared with CPLEX on small and medium-sized datasets.

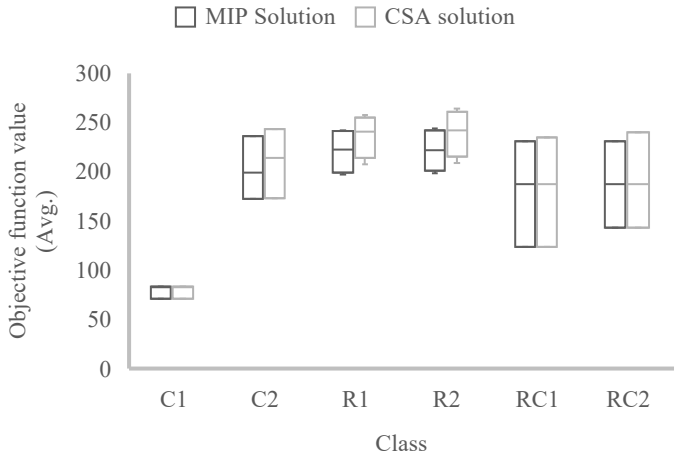


Figure 4. Graphical depiction of the CSA algorithm's performance compared with CPLEX on small and medium-sized datasets.

5.5. Summary of Findings

Here, we have summarized insights from the presented numerical results. Firstly, by applying the clustering approach we used in the R-AMPLP instances, the traveling cost decreased regarding the original cases in the VRPTW setting of the Solomon dataset (Solomon, 1987), in which the clustering approaches were not considered. Secondly, having wider time windows caused smaller costs than cases of tighter time windows, either in the pure VRPTW or R-AMPLP setting. An important point relates to class C2, where the outcomes suggested that combining clustering with extended time windows resulted in lower costs. Figure 5 depicts the numerical results based on Solomon's datasets (VRPTW) compared to the R-AMPLP.

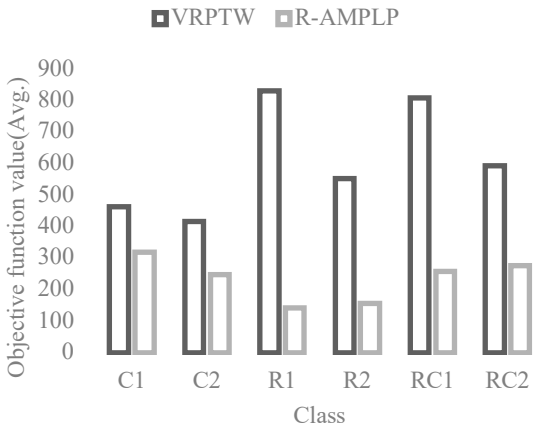


Figure 5. Graphical depiction of numerical results based on Solomon's datasets (VRPTW) compared to the R-AMPLP.

It is critical to note that these results revealed that even when considering the parking slot fee, the operation cost related to the R-AMPLP was significantly less than the VRPTW setting.

5.6. Managarial Insights

In this study, we divided the last-mile delivery operation into two subsystems to coordinate the handling of delivery items using different technologies and facilities. However, the scope of this paper was limited to situations where access to final customers for order delivery may not be possible due to restrictions. In these cases, AMPL units were employed to fulfill orders in certain clusters. Additionally, due to the real-world limitations posed by the capacity of AMPLs, a fleet of trucks was utilized to replenish these units with new orders. This replenishment can either serve remaining customers along pre-defined AMPL routes or fulfill same-day delivery requests. Moreover, serving AMPLs at designated meeting points, whether included in their paths or not, can significantly reduce travel costs, as this approach eliminates the need for AMPLs to return to distribution centers. Consequently, trucks also avoid the need to cover all customer nodes, which can be costly in last-mile delivery due to increased demands for drivers, trucks, and operational costs, as well as negative environmental impacts.

Unlike the GVRPTW, this study addressed not only issues related to alternative physical points but also managed multiple time windows for each physical node. This added complexity increased the challenges associated with GVRPTW. The situation became even more complicated when each time window at each physical node incurred specific or time-dependent parking costs during the operational delivery period.

The proposed algorithm overcame the complexities of the presented problem by incorporating the K-means clustering method. This approach first classified physical nodes into distinct clusters and then created copies of each node to apply the multiple time windows strategy, making the model more realistic and closer to real-world scenarios in last-mile delivery. Given that the number of meeting points is typically extremely large when handling multiple time windows for a set of physical nodes, having an effective algorithm to manage these issues is crucial. The results revealed that the algorithm could effectively address the complexity, even for those VRPTW instances, such as a basic case of R-AMPLP, where no exact solution existed. In such cases, the algorithm could provide a feasible solution within a reasonable timeframe. Furthermore, this algorithm was not sensitive to the structure of meeting points concerning their associated time windows. For instance, a physical node in a zone may have three distinct time windows, each with a parking fee, or three time windows arranged serially within a long time slot, or even overlapping with each other.

6. Conclusions

In summary, we presented a significant advancement in the efficiency of last-mile distribution by developing a mixed-integer linear programming (MILP) model tailored for re-supplying autonomous mobile parcel lockers (AMPLs) through a coordinated effort with a fleet of trucks. Although AMPLs move in pre-defined paths, we considered the traveling cost from leaving points in their paths to selected meeting points and from the meeting points to returning points in their paths. Moreover, we introduced a novel approach utilizing cluster-based simulated annealing (CSA) to tackle the complexities of large-scale re-supply scenarios, with the model's robustness validated against smaller instances for a comprehensive understanding of its effectiveness. The CSA heuristic efficiently managed up to 1200 potential meeting points, showcasing the model's scalability and applicability for practical deployment. This system significantly aided in the strategic planning of AMPL re-supply points, ensuring that the AMPLs could maintain optimal service routes while integrating re-supply needs effectively. By employing advanced

navigation systems, the AMPLs were systematically directed to designated re-supply points, minimizing the total re-supplying cost and optimizing the network's overall efficiency.

Our findings illuminated the cost benefits of a clustering strategy coupled with assigning multiple time windows to alternative meeting locations, which collectively lowered the operational expenses compared to traditional methods that necessitate individual visits to each AMPL individually. This streamlined approach negated the requirement for AMPLs to return to distribution centers for re-supply, enhancing operational efficiency and extending the reach of service routes without significant deviations. In summarizing the comprehensive class comparison and performance evaluation presented, it was evident that the application of clustering methods significantly impacted the efficiency and objective values in routing problems, as demonstrated by the comparison across different classes (R1, R2, RC1, RC2, C1, and C2). Particularly, the groups R1 and R2 demonstrated lower objective values and smaller average cluster counts compared to their counterparts, underscoring the efficiency of scattered meeting points in yielding more efficient and cost-effective solutions relative to clustered or semi-clustered strategies. Moreover, comparing the CSA algorithm with CPLEX across various dataset sizes confirmed the CSA algorithm's robust performance, with minimal discrepancies observed between the two approaches. This alignment further emphasized the reliability of clustering strategies in optimizing routing problems. Notably, the findings underscored the significant advantages of applying clustering, particularly in the context of the VRPTW settings from Solomon's dataset, where the introduction of broader time windows coupled with clustering approaches yielded lower operational costs. This suggests a promising avenue for enhancing the efficiency of routing problems by strategically integrating clustering methods and adjusting time window parameters, ultimately leading to significant cost reductions.

The research suggested promising research avenues for the future. These include integrating considerations of electric consumption, addressing parameter uncertainties and synchronization challenges between AMPLs and trucks, and investigating comprehensive location-allocation and routing problems within specific geographic areas. Notably, the study identified the dynamic and intricate case of replenishing mobile vendor machines in urban settings as a viable domain for further investigation.

Author Contributions: Conceptualization, Mostafa Setak, Sajjad Hedayati, and Tom Van Woensel; methodology, Sajjad Hedayati, Tom Van Woensel, and Emrah Demir; software, Sajjad Hedayati; validation, Sajjad Hedayati, Tom Van Woensel, and Emrah Demir; formal analysis, Sajjad Hedayati; resources, Mostafa Setak, Sajjad Hedayati, and Tom Van Woensel; data curation, Sajjad Hedayati; writing— Mostafa Setak, Sajjad Hedayati, X.X.; writing—review and editing, Tom Van Woensel and Emrah Demir; visualization, Sajjad Hedayati; supervision, Mostafa Setak and Tom Van Woensel; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data availability statement: The datasets generated and/or analyzed during the current study are not publicly available but are available from the corresponding author upon reasonable request.

Acknowledgments: We would like to express our sincere gratitude to the anonymous reviewers for their valuable feedback and insightful comments. Their suggestions have significantly contributed to improving the quality of this work. We appreciate their time and effort in carefully reviewing the manuscript.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A

In Algorithms A1–A6, we present improvement functions that were randomly selected at each internal iteration to create an improvement in the current solution. Figures

A1–A6 graphically show operations that were performed by these algorithms, corresponding to the example provided in Section 4.

In Algorithm A1, the 2_opt operator is depicted. This operator selects two meeting points and then exchanges their positions within the current route.

Algorithm A1. 2_opt algorithm.

Input: An initial solution, ω

Output: A new solution, ω^{new}

function 2_opt (ω)

$S \leftarrow |\omega|$;

$g(v) \leftarrow 0$;

$g(u) \leftarrow 0$;

$[list(1),list(2)] \leftarrow random\ sampling((1,S),2)$;

$g(v) \leftarrow list(1)$;

$g(u) \leftarrow list(2)$;

update$(\omega) \leftarrow \omega([g(u)\ g(v)])$;

$\omega^{new} \leftarrow \omega$;

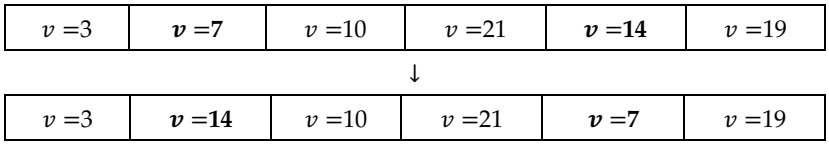


Figure A1. Graphical depiction of Algorithm A1 corresponding to the example in Section 4.

Algorithm A2 operates by selecting two meeting points within the current solution. It then assesses their order and removes the meeting point with the lesser order from its route, placing it instead after the second selected meeting point. This process effectively destroys and constructs new routes.

Algorithm A2. remove_insert_1 algorithm.

Input: An initial solution, ω

Output: A new solution, ω^{new}

function *remove_insert_1 (ω)*

$S \leftarrow |\omega|$;

$g(v) \leftarrow 0$;

$g(u) \leftarrow 0$;

$[list(1),list(2)] \leftarrow random\ sampling((1,S),2)$;

$g(v) \leftarrow list(1)$;

$g(u) \leftarrow list(2)$;

if g(v)<g(u) **then**

    update$(\omega) \leftarrow \omega([[1:g(v)-1][g(v)+1:g(u)][g(v)][g(u)+1:|\omega|])]$;

else

    update$(\omega) \leftarrow \omega([[1:g(u)][g(v)][g(u)+1:g(v)-1][g(v)+1:|\omega|])]$;

end

$\omega^{new} \leftarrow \omega$;

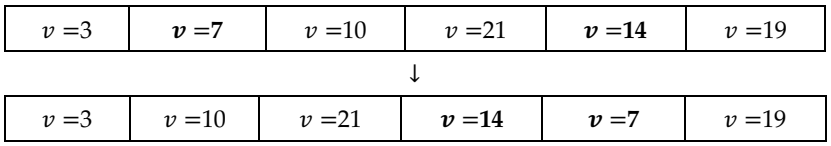


Figure A2. Graphical depiction of Algorithm A2 corresponding to the example in Section 4.

Algorithm A3 functions by shaking up the current solution, achieved by selecting a new meeting point from each cluster. This is a fundamental operator for managing alternative locations that have multiple time windows.

Algorithm A3. shake_cluster algorithm.

Input: An initial solution, ω

Output: A new solution, ω^{new}

function *shake-cluster* (ω)

$S \leftarrow |\omega|$;

$L \leftarrow Linstance$;

$C \leftarrow Cinstance$;

for $g(v)$ *do*

  if $v \setminus \in L$ **then**

  update $(\omega) \leftarrow \omega([random\_sampling(g(g(v)).1)})$;

  end

end

$\omega^{new} \leftarrow \omega$;

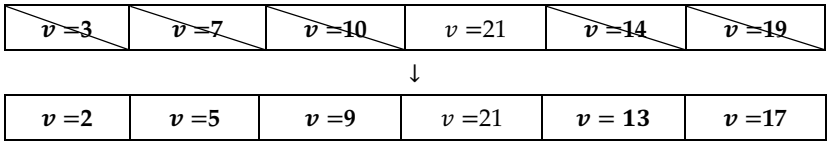


Figure A3. Graphical depiction of Algorithm A3 corresponding to the example in Section 4.

Algorithm A4 selects three meeting points within the current solution and proceeds to exchange their positions with each other.

Algorithm A4. 3_opt algorithm.

Input: An initial solution, ω

Output: A new solution, ω^{new}

function *3_opt* (ω)

$S \leftarrow |\omega|$;

$g(v) \leftarrow 0$;

$g(u) \leftarrow 0$;

$g(w) \leftarrow 0$;

$[list(1),list(2),list(3)] \leftarrow random\ sampling((1,S),3)$;

g(v) ← *list(1);*
g(u) ← *list(2);*
g(w) ← *list(3);*
update*(ω)* ← *ω([[g(u)] [g(w)] [g(v)]]);*
ω^{new} ← *ω*;

v =3	***v*** **=7**	*v* =10	***v*** **=21**	*v* =14	***v*** **=19**

↓

v =3	***v*** **=21**	*v* =10	***v*** **=19**	*v* =14	***v*** **=7**

Figure A4. Graphical depiction of Algorithm A5 corresponding to the example in Section 4.

Algorithm A5 involves selecting a single meeting point in the current solution and then transferring it from its original route to a different route.

Algorithm A5. remove_insert_2 algorithm.

Input: An initial solution, *ω*
Output: A new solution, *ω*^{new}
function *remove_insert_2 (ω)*
S ← |*ω*|;
g(v) ← *0;*
g(u) ← *0;*
[list(v)] ← *random sampling((1,S),1);*
[list(u)] ← *find dummy(ω,1);*
g(v) ← *list(v);*
g(u) ←*list(u);*
if *g(v)* < *g(u)* **then**
    update*(ω)*←*ω([[1:g(v)-1][g(v)+1:g(u)[g(v)][g(u)+1:|ω|]]);*
else
    update*(ω)*←*ω([[g(v)][1:g(u)[g(u)+1:g(v)+1][g(v)+1:|ω|]]);*
end
ω^{new} ← *ω*;

v =3	*v* =7	*v* =10	*v* =21	***v*** **=14**	*v* =19

↓

v **=14**	*v* =3	*v* =7	*v* =10	*v* =21	*v* =19

Figure A5. Graphical depiction of Algorithm A4 corresponding to the example in Section 4.

Algorithm A6 involves selecting two elements within the current solution and disrupting the routes. This is carried out by reversing the order of meeting points located between these two selected points.

Algorithm A6. reverse algorithm.

Input : An initial solution, ω

Output: A new solution, ω^{new}

function *reverse* (ω)

$S \leftarrow |\omega|$;

$g(v) \leftarrow 0$;

$g(u) \leftarrow 0$;

$g(w) \leftarrow 0$;

$g(y) \leftarrow 0$;

$[list(1), list(2)] \leftarrow random\ sampling((1,S),2)$;

$g(v) \leftarrow list(1)$;

$g(u) \leftarrow list(2)$;

$g(w) \leftarrow find\ min(g(v),g(u))$;

$g(y) \leftarrow find\ max(g(v),g(u))$;

update$(\omega) \leftarrow \omega(reverse([[g(y)] : [g(w)]]))$;

$\omega^{new} \leftarrow \omega$;

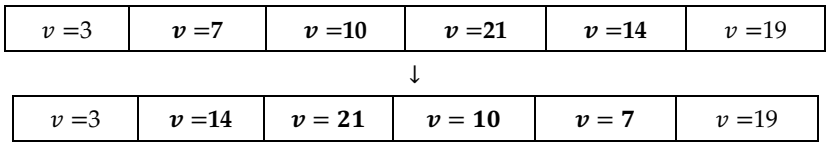


Figure A6. Graphical depiction of Algorithm A6 corresponding to the example in Section 4.

References

1. Siegfried, P.; Zhang, J.J. Developing a Sustainable Concept for Urban Last-Mile Delivery. *Open J. Bus. Manag.* **2020**, *9*, 268–287.
2. Demir, E.; Syntetos, A.; Van Woensel, T. Last mile logistics: Research trends and needs. *IMA J. Manag. Math.* **2022**, *33*, 549–561.
3. Gu, Q.; Fan, T.; Pan, F.; Zhang, C. A vehicle-UAV operation scheme for instant delivery. *Comput. Ind. Eng.* **2020**, *149*, 106809.
4. Alnaggar, A.; Gzara, F.; Bookbinder, J.H. Crowdsourced delivery: A review of platforms and academic literature. *Omega* **2021**, *98*, 102139.
5. Reyes, D.; Savelsbergh, M.; Toriello, A. Vehicle routing with roaming delivery locations. *Transp. Res. Part C Emerg. Technol.* **2017**, *80*, 71–91.
6. Figliozzi, M.A. Carbon emissions reductions in last mile and grocery deliveries utilizing air and ground autonomous vehicles. *Transp. Res. Part D Transp. Environ.* **2020**, *85*, 102443.
7. Lin, Y.H.; Wang, Y.; He, D.; Lee, L.H. Last-mile delivery: Optimal locker location under multinomial logit choice model. *Transp. Res. Part E Logist. Transp. Rev.* **2020**, *142*, 102059.
8. Chen, C.; Demir, E.; Huang, Y.; Qiu, R. The adoption of self-driving delivery robots in last mile logistics. *Transp. Res. Part E: Logist. Transp. Rev.* **2021**, *146*, 102214.
9. Rojas Viloria, D.; Solano-Charris, E.L.; Muñoz-Villamizar, A.; Montoya-Torres, J.R. Unmanned aerial vehicles/drones in vehicle routing problems: A literature review. *Int. Trans. Oper. Res.* **2021**, *28*, 1626–1657.
10. Savelsbergh, M.; Van Woensel, T. 50th anniversary invited article—City logistics: Challenges and opportunities. *Transp. Sci.* **2016**, *50*, 579–590.
11. Schwerdfeger, S.; Boysen, N. Optimizing the changing locations of mobile parcel lockers in last-mile distribution. *Eur. J. Oper. Res.* **2020**, *285*, 1077–1094.
12. Schwerdfeger, S.; Boysen, N. Who moves the locker? A benchmark study of alternative mobile parcel locker concepts. *Transp. Res. Part C Emerg. Technol.* **2022**, *142*, 103780.
13. Boysen, N.; Fedtke, S.; Schwerdfeger, S. Last-mile delivery concepts: A survey from an operational research perspective. *OR Spectr.* **2021**, *43*, 1–58.
14. Ulmer, M.W.; Streng, S. Same-day delivery with pickup stations and autonomous vehicles. *Comput. Oper. Res.* **2019**, *108*, 1–19.

15. Desrochers, M.; Lenstra, J.K.; Savelsbergh, M.W.; Soumis, F. Vehicle Routing with Time Windows: Optimization and Approximation; **1987**, *(No. OS-R8715). CWI. Department of Operations Research and System Theory [BS] .* Vancouver
16. Ghiani, G.; Improta, G. An efficient transformation of the generalized vehicle routing problem. *Eur. J. Oper. Res.* **2000**, *122*, 11–17.
17. Hedayati, S.; Setak, M.; Demir, E.; Van Woensel, T. A new approach to the joint order batching and picker routing problem with alternative locations. *IMA J. Manag. Math.* **2024**, *35*, 241–265.
18. Moccia, L.; Cordeau, J.-F.; Laporte, G. An incremental tabu search heuristic for the generalized vehicle routing problem with time windows. *J. Oper. Res. Soc.* **2012**, *63*, 232–244.
19. Ozbaygin, G.; Karasan, O.E.; Savelsbergh, M.; Yaman, H. A branch-and-price algorithm for the vehicle routing problem with roaming delivery locations. *Transp. Res. Part B Methodol.* **2017**, *100*, 115–137.
20. Dumez, D.; Lehuédé, F.; Péton, O. A large neighborhood search approach to the vehicle routing problem with delivery options. *Transp. Res. Part B Methodol.* **2021**, *144*, 103–132.
21. Tilk, C.; Olkis, K.; Irnich, S. The last-mile vehicle routing problem with delivery options. *OR Spectr.* **2021**, *43*, 877–904.
22. Lombard, A.; Tamayo-Giraldo, S.; Fontane, F. Vehicle routing problem with roaming delivery locations and stochastic travel times (VRPRDL-S). *Transp. Res. Procedia* **2018**, *30*, 167–177.
23. Favaretto, D.; Moretti, E.; Pellegrini, P. Ant colony system for a VRP with multiple time windows and multiple visits. *J. Interdiscip. Math.* **2007**, *10*, 263–284.
24. Belhaiza, S.; Hansen, P.; Laporte, G. A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. *Comput. Oper. Res.* **2014**, *52*, 269–281.
25. Belhaiza, S.; M'Hallah, R.; Brahim, G.B. A new hybrid genetic variable neighborhood search heuristic for the vehicle routing problem with multiple time windows. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC), San Sebastián, Spain, 5–8 June2017; IEEE: Piscataway, NJ, USA, 2017.
26. Beheshti, A.K.; Hejazi, S.R.; Alinaghian, M. The vehicle routing problem with multiple prioritized time windows: A case study. *Comput. Ind. Eng.* **2015**, *90*, 402–413.
27. Hoogeboom, M.; Dullaert, W.; Lai, D.; Vigo, D. Efficient neighborhood evaluations for the vehicle routing problem with multiple time windows. *Transp. Sci.* **2020**, *54*, 400–416.
28. Yuan, Y.; Cattaruzza, D.; Ogier, M.; Rousselot, C.; Semet, F. Mixed integer programming formulations for the generalized traveling salesman problem with time windows. *4OR* **2021**, *19*, 571–592.
29. Wei, L.; Zhang, Z.; Zhang, D.; Leung, S.C. A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints. *Eur. J. Oper. Res.* **2018**, *265*, 843–859.
30. Wang, C.; Mu, D.; Zhao, F.; Sutherland, J.W. A parallel simulated annealing method for the vehicle routing problem with simultaneous pickup–delivery and time windows. *Comput. Ind. Eng.* **2015**, *83*, 111–122.
31. Rousseeuw, P.J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* **1987**, *20*, 53–65.
32. Solomon, M.M. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* **1987**, *35*, 254–265.
33. Ilog. IBM ILOG CPLEX Optimization Studio [Online]. 2023. Available online: https://www.ibm.com/products/ilog-cplex-optimization-studio (accessed on 22 July 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.