On the Hierarchical Directed Capacitated Arc Routing Problem

Minh Hoàng Hà[a], Thu Huong Dang[b], Ba Luat Le[a], Trung Thanh Nguyen[c], André Langevin[d]

[a] *SLSCM and CADA, Faculty of Data Science and Artificial Intelligence, College of Technology, National Economics University, Hanoi, Vietnam*
[b] *Department of Management Science, Lancaster University, Lancaster LA1 4YX, UK*
[c] *DataOptLab, Faculty of Data Science and Artificial Intelligence, College of Technology, National Economics University, Hanoi, Vietnam*
[d] *Department of Mathematics and Industrial Engineering and CIRRELT, École Polytechnique de Montréal, C.P. 6079, Succursale Centre-ville, Montréal, Qué. Canada H3C 3A7*

Abstract

The Hierarchical Directed Capacitated Arc Routing Problems (HDCARP) is a variant of the Capacitated Arc Routing Problems (CARPs), in which the arcs in a graph are partitioned into priority classes. However, unlike traditional CARPs that aim to minimise total time, the HDCARP focuses on minimizing the maximum completion time of each priority class in a hierarchical fashion. Practical applications of the HDCARP include snow plowing, salt spreading, street cleaning, and waste collection. In this study, we explore two variants of the HDCARP. The key difference between these variants lies in the consideration of precedence relations between classes within routes. We propose MILP formulations and matheuristics for both HDCARP variants. The MILP formulations enable us to find optimal solutions for small-scale instances and evaluate the quality of matheuristics. Our matheuristics are based on decomposing the problem into multiple sub-problems, resulting in faster running time for large-scale instances. We conduct extensive computational experiments to assess the performance of these approaches and present our findings.

Keywords: Arc routing problem; Hierarchical objective; Mixed-integer linear programming, Matheuristic.

1. Introduction

The well-known *Chinese Postman Problems* (CPPs) are a special kind of *Arc Routing Problems* (ARPs), in which all roads should be traversed at least once by a single vehicle, assuming unlimited capacity [17, 19, 22]. While the CPPs can be solved in polynomial time [6, 10, 18], most ARPs of interest are NP-hard. The readers are referred to [14] for recent results on ARPs. In the CPP, road segments are assigned equal priorities. However, in practice, to optimise route scheduling or improve service efficiency, roads are usually categorised into classes based on factors such as traffic volume, urgency, and specific applications [9, 25, 30, 31, 33, 34, 35, 38]. For instance, in disaster relief operations, routing problems are crucial for tasks like delivering medical aid, transporting supplies, and distributing food. The affected areas are categorized based on damage intensity, urgency of needs, and road conditions to guide effective emergency responses and resource allocation.

Email address: hoanghm@neu.edu.vn (Minh Hoàng Hà)

The Hierarchical Chinese Postman Problem (HCPP) is an NP-hard variant of the CPP, where edges are grouped into priority classes and a precedence relationship determines their traversal order. The objective is to find the shortest route that starts and ends at a depot, travels each edge at least once, and respects the precedence relationship (see [16]). The *Hierarchical Rural Postman Problems* (HRPPs) are the generalization of the HCPP where only a subset of road segments require a service (see [4, 12, 13]). Typical applications of HCPPs and HRPPs include flame cutting, street cleaning, garbage collection, salt spreading, and snow plowing (e.g., [4, 8, 16, 21, 24, 25, 28]).

While the precedence relationship is commonly encountered in real-life scenarios, there are only a limited number of studies that specifically address ARPs incorporating service hierarchy. Moreover, most research efforts have focused on the HCPPs and HRPPs, assuming unlimited vehicle capacity, which is unrealistic for real-world scenarios. However, constraints on vehicle capacity are typically critical in various applications. For example, in snow removal, routes must be planned to ensure that the total amount of de-icing materials loaded on a vehicle does not exceed its capacity (see [3]).

In this study, we examine the *Hierarchical Directed Capacitated Arc Routing Problem* (HDCARP), which extends the HRPPs by incorporating practical aspects such as one-way road segments, multiple vehicles, and capacity constraints. To be specific, the HDCARP aims to determine a set of routes for a fleet of identical vehicles, subject to the following conditions: (i) each vehicle must start and end its route at the depot; (ii) each required road must be serviced exactly once; (iii) the total demand on each route must not exceed the vehicle capacity, and (iv) the order in which the roads are serviced on each route must respect the specified precedence relation.

The objective of the HDCARP is to minimise the maximum completion time of the first priority class, followed by the second priority class, and so on. The maximum completion time of a class is defined as the minimum time required for all vehicles to complete servicing all roads within that class. This objective is referred to as the *hierarchical objective*, as described in [8, 30]. The hierarchical objective is particularly suitable for practical applications where roads are categorized based on their priority. It ensures that roads with higher priority must take precedence over roads with lower priority. Therefore, this objective aligns with the concept of precedence constraints.

There are two variants of precedence constraints studied in the literature. The predominant focus is on the *linear precedence relations*, which establish a unique lexicographical ordering for all classes within a route [8, 13, 16, 26, 30]. This variant arises when roads belonging to a class can only be serviced after all higher priority roads have been successfully serviced. Another variant of precedence constraints that has been studied in the literature is the *general precedence relations* [16, 30]. In this variant, all roads of high-priority class must be serviced before those of low-priority class. However, medium-priority roads are allowed to be serviced before or after certain high-priority and low-priority roads. There are several studies [30, 39] in the literature that do not impose precedence constraints between any pair of classes. Instead, the ordering of classes is determined solely by the hierarchical objective. This is referred to as *class upgrading* possibility, which is beneficial for improving the service quality of low priority classes and reducing the total completion time. The linear precedence and upgrading possibility have received more attention in the literature, primarily due to their practical applications and relevance [32, 33, 34]. Consequently, we consider these variants in this paper.

This paper has several contributions. First, this research represents the pioneering effort in the study of HDCARP, a new problem that generalizes both the HCPPs and the HRPPs. We

study both linear precedence constraints and the upgrading possibility, resulting in two HDCARP variants. Second, we provide *Mixed Integer Linear Programming* (MILP) formulations for both variants. We point out a special case that is not adequately addressed by existing formulations in the literature on ARPs with a hierarchical objective. Additionally, we propose a new approach for expressing the hierarchical objective. Third, we improve the computational time by developing efficient matheuristics for both variants that decompose the original problem into several sub-problems. Finally, we present some extensive computational results on randomly created instances.

The rest of the paper is structured as follows. Section 2 contains a brief literature review. Section 3 provides a formal description and mathematical formulation for each HDCARP variant. Section 4 presents MILP-based matheuristics, and Section 5 presents the computational results. Finally, some conclusions and future research directions are given in Section 6.

2. Literature review

In this section, we briefly review the relevant literature. We discuss the HCPPs in Subsection 2.1, the HRPPs in Subsection 2.2, and other ARPs with hierarchical services in Subsection 2.3.

2.1. Hierarchical chinese postman problems

To our knowledge, the first paper that formally addressed the HCPP was Dror et al. [16]. The authors showed that the HCPP is generally NP-hard. However, when all subgraphs induced by the classes are connected and the precedence relations are linear, the HCPP can be solved in polynomial time via a matching problem and a series of shortest path problems. Improved versions of this algorithm, suitable for large-scale instances, were given in [21, 25, 37]. More recently, Afanasev et al. [1] proved that under the assumption of connected classes, the HCPP is generally classified as NP-hard. The authors also proposed a 5/3-approximation algorithm in polynomial time for the HCPP under the assumption of linear precedence relations.

Several algorithms have been proposed for the HCPP with linear precedence relations in general cases. Lemieux and Gampagna [26] proposed a simple heuristic based on Euler circuits for the case of two priority classes. Cabral et al. [8] converted the HCPP to the equivalent RPP and applied branch-and-cut procedures to solve it. Çodur and Yılmaz [11] formulated the HCPP as an MILP and proposed two metaheuristics based on a genetic algorithm and a hybrid simulated annealing for large-scale instances. Damodaran et al. [15] used the optimal solution to the CPP as a lower bound for the HCPP. The authors also proposed a heuristic algorithm to enhance this lower bound within a short computational time.

Regarding directed graphs, Alfa and Liu [4] addressed the *Directed HCPPs* (HDCPPs) with general precedence relations. They introduced a new constraint, requiring the service of a higher priority class to be started and completed before that of a lower priority class. They proposed a three-phase heuristic approach to address this problem, which involved connecting classes, balancing non-symmetric nodes, and identifying feasible routes. A dynamic programming algorithm was later presented in [20] for HDCPPs with general precedence relations and class connectivity.

Perrier et al. [30] addressed the problem of *Mixed Multi-vehicle* HCPPs (m-HMCPP) with class upgrading possibilities. The m-HMCPP is the generalisation of the HCPP where both edges and arcs can be present, and multiple vehicles with unlimited capacity are available. They first performed a graph transformation and then formulated the m-HMCPP as a MILP model. The model was adaptable to incorporate additional constraints and different situations.

Furthermore, the authors introduced two constructive heuristics based on decomposing the problem into sub-problems. For more on the HCPPs, see the surveys [33, 34].

2.2. Hierarchical rural postman problems

Quirion-Blais et al. [36] proposed an adaptive large neighborhood search metaheuristic for the multi-vehicle HRPP with general precedence relations. The metaheuristic is suitable for large-scale instances and can be customized to accommodate some operational constraints.

The concept of the Hierarchical Mixed RPP (HMRPP) was introduced by Perrier et al. [29] in 2006 and further studied in 2017 [12, 13, 35]. Colombi et al. [12] proposed a mathematical programming formulation, a matheuristic, and a Tabu Search for the HMRPP with linear precedence relations. The matheuristic solves a series of Mixed Rural Postman Problems for each class. Colombi et al. [13] provided polyhedral results and introduced facet-inducing inequalities based on the formulation in [12]. Quirion-Blais et al. [35] focused on the multi-vehicle version and developed an adaptive large neighborhood metaheuristic.

2.3. Other ARPs with hierarchy services

Recently, Ahabchane et al. [2] introduced the *Hierarchical Mixed Capacitated General Routing Problem* (HMCGRP). The HMCGRP is a generalisation of the HMRPP in which both nodes and edges may require service. Notably, this work is the first and only one in the literature to consider capacity constraint and demand uncertainty. In their approach, road hierarchies are modelled with time-dependent costs. For small-scale instances, a robust formulation with graph transformation was developed, while for large-scale instances, a two-phase metaheuristic based on simulated annealing and ruin-and-recreate strategies was proposed.

Paper	Lexicographic Objective	Precedence Relation	Upgrading Allowed	Multiple Vehicles	Capacity Constraint	Data Type	Graph Type
Dror et al. 1987 [16]	-	X	-	-	-	Deterministic	(Un)Directed
Cabral et al. 2004 [8]	X	X	-	-	-	Deterministic	Undirected
Perrier et al. 2008 [30]	X	X	X	X	-	Deterministic	Directed
Quirion-Blais et al. 2015 [36]	X	-	X	X	-	Deterministic	Mixed
Colombi et al. 2017 [12, 13]	-	X	-	-	-	Deterministic	Mixed
Quirion-Blais et al. 2017 [35]	X	X	-	X	-	Deterministic	Mixed
Ahabchane et al. 2021 [2]	-	-	X	X	X	Uncertain	Mixed
This paper	X	X	X	X	X	Deterministic	Directed

Table 1: Synthesis of studies related to routing problems with service hierarchy.

Table 1 situates our research problem within the context of studies on the service hierarchy in the ARP literature. We extend the research direction of [8, 12, 13, 16] by considering multiple vehicles and capacity constraints. Our main objective is to analyze and compare the results of our proposed algorithms to understand the computational difficulty of solving these variants. This study is the first to report such results for capacitated ARPs with hierarchical service.

3. Mathematical formulations

In this section, we present the problems and mathematical formulations. First, in Subsection 3.1, we provide a formal description of two HDCARP variants, along with the notations and terminology. Second, the graph transformation will be presented in Subsection 3.2. Then, in Subsections 3.3 and 3.4, we present the mathematical formulations for both variants.

3.1. Problem definition

We are given a strongly connected directed graph $G = (V, A)$, where V is the vertex set and A is the set of (directed) arcs. The vertices are denoted by $v_0, v_1, \ldots, v_n$, and vertex v_0 is called the *depot*. This graph represents a road network, in which road junctions and key landmarks are represented by vertices, one-way road segments are represented by arcs, and two-way road segments are represented by a pair of directed arcs, one in each direction. We are also given a set $A_r \subseteq A$ of *required arcs*. We call $A_{nr} = A \setminus A_r$ the set of non-required arcs. The set of required arcs A_r is partitioned into p pairwise disjoint classes $A_r^1, A_r^2, \ldots, A_r^p$ ($A_r = A_r^1 \cup A_r^2 \cup \ldots \cup A_r^p$ and $A_r^h \cap A_r^k = \emptyset$ for $h \neq k$ and $h, k \in \{1, 2, \ldots, p\}$). For convenience, we denote $P = \{1, 2, \ldots, p\}$. Let V_t^k be the set of tail vertices of arcs in A_r^k for each $k \in P$. Let V_t denote $V_t^1 \cup V_t^2 \cup \ldots \cup V_t^p$.

Figure 1 illustrates the concept of classes and notations. Required and non-required arcs are drawn by thick solid and dashed lines, respectively. The class numbers are indicated on the required arcs. Nodes in V_t are represented by hollow circles. One can check that $V_t^1 = \{v_0, v_2, v_3, v_4\}$, $V_t^2 = \{v_1\}$, and $V_t^3 = \{v_3, v_4\}$.

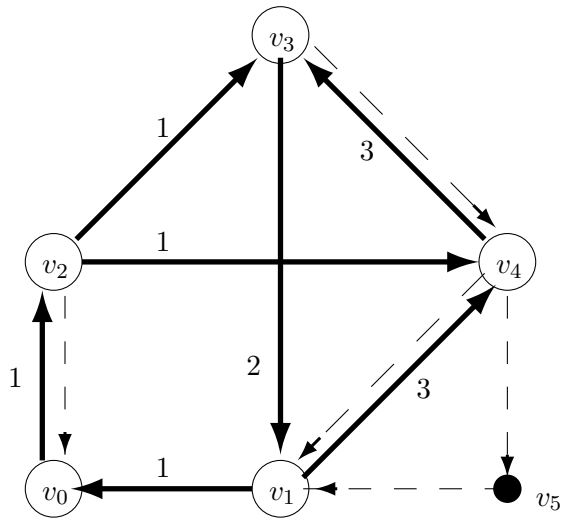


Figure 1: Graph G with priority classes

Each arc $a \in A$ has a positive *traversal time* d_a. Each required arc $a \in A_r$ is associated with a positive *demand* q_a, *servicing time* s_a, and *priority level* p_a. A fleet of identical vehicles M is located at the depot, each with positive capacity Q. Hence, the load of each vehicle must not exceed Q at any time. In the HDCARP, a vehicle must depart from the depot, service some required arcs while respecting the capacity constraint and the linear precedence relations between classes, and must return to the depot. Each required arc must be serviced by exactly one vehicle. Traversing an arc without servicing is called *deadheading*.

In this study, we consider both linear precedence relations and class upgrading possibility, resulting in two variants of the HDCARP, called the HDCARP-P and HDCARP-U, respectively. The only difference between the two variants is whether low priority arcs are allowed to be served before higher priority arcs. Specifically, in the HDCARP-P, arcs in class k can only be served after completing the service of all arcs in class $k-1$. In contrast, in the HDCARP-U, vehicles can serve arcs in class k before completing the service of all arcs in class $k-1$.

We now consider the HDCARP instance shown in Figure 1. Assume that each arc has a unit traversal time, each required arc has a unit servicing time and a unit demand, and two identical vehicles are located at v_0 with a capacity of $Q = 4$ units. Let's examine a feasible solution for the HDCARP-P shown in Figure 2(a). The first route serves three required arcs of class 1, while the second route sequentially serves the remaining required arcs according to the linear precedence relations. The completion time for class 1 is 4 on route 1 and 2 on route 2, resulting in a maximum completion time of 4 for class 1. Similarly, one can check that the maximum completion times for classes 2 and 3 are 4, and 6, respectively. Note that this solution is also feasible for the HDCARP-U. Figure 2(b) illustrates another feasible solution for the HDCARP-U, where the required arc in class 2 is serviced before the one in class 1 in route 1. It can be verified that the maximum completion times of classes 1, 2, and 3 are $4, 3$, and 5, respectively. If upgrades are permitted, the solution shown in Figure 2(b) is considered better than the solution in Figure 2(a) in terms of completing classes at the earliest possible time (and minimising the total time). However, the solution in Figure 2(b) is infeasible for the HDCARP-P.

This suggests that the HDCARP-U can improve the maximum completion time of each class in the HDCARP-P's solution, especially if the majority of time is spent deadheading. This is further confirmed in Section 5. However, the main drawback of the HDCARP-U is the unknown order in which classes are traversed on each route, making it more challenging to solve.

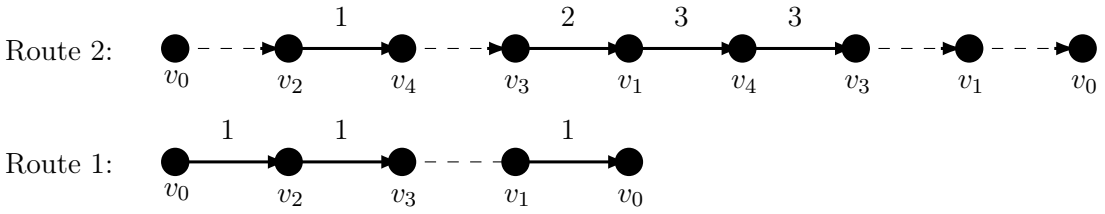
We consider the hierarchical objective for both HDCARP-P and HDCARP-U. A common way to express this objective in the literature is the weighted sum of the maximum completion times of each priority class [8, 13, 35, 36]. In other words, the weighted sum $\sum_{k=1}^{p} \beta_k T_k$ is minimised, where T_k is the maximum completion time of the class k, and β_k are coefficients such that $\beta_1 \gg \beta_2 \gg ... \gg$
$beta_p$. It can be seen that handling the large constants β_k $(k \in P)$ in the objective function can be difficult, as often happens in integer programming. To get around this issue, we use the lexicographic optimization approach to model the hierarchical objective. Simply put, the approach starts with the minimization of the first objective, then among the possible optima, minimises the second objective, and so on, until all p objective functions are minimised. We call sequence $(T_1, \ldots, T_p)$ the *maximum completion time sequence*. To compare any two solutions, we use the lexicographic order to compare their maximum completion time sequences. More precisely, the solution is considered better if the first different element in its maximum completion time sequence is smaller.

When dealing with the HDCARPs, we observe special cases where a low priority class may have a shorter maximum completion time than a higher priority class. These could happen in both non-upgrade and upgrade variants. However, these cases are excluded in any formulation for multiple-vehicle HCPPs or HRPPs in the literature. For example, the ones in [30, 35] defined that $T_1 \leq T_2 \leq \ldots \leq T_p$.

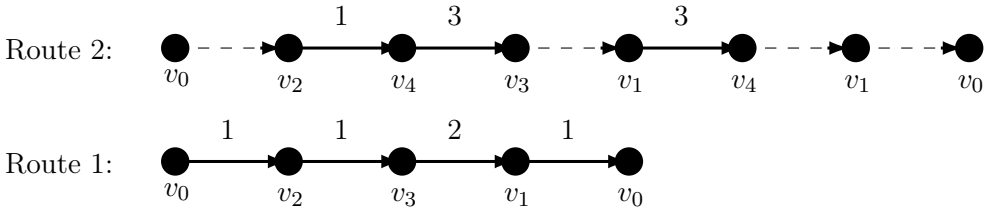
These special cases are best illustrated in Figure 3. It shows a feasible solution to the HDCARP-P instance in Figure 1. One can check that $T_1 = 4$, $T_2 = 3$ and $T_3 = 5$. It is clear that $T_2 < T_1$ even though class 2 has a lower priority than class 1. We would like to highlight that our MILP formulations and matheuristics are capable of capturing and addressing such special cases.

3.2. Graph Transformation

For what follows, it is helpful to define an auxiliary directed multigraph, which we denote by $G' = (V', A')$. Our transformation is inspired by the one in [30] where a "dummy" vertex is introduced to ensure connectivity between classes on each route. However, we propose a slightly



(a) A feasible solution for the HDCARP-P



(b) A feasible solution for the HDCARP-U

Figure 2: Feasible solutions for the HDCARP instance in Figure 1.

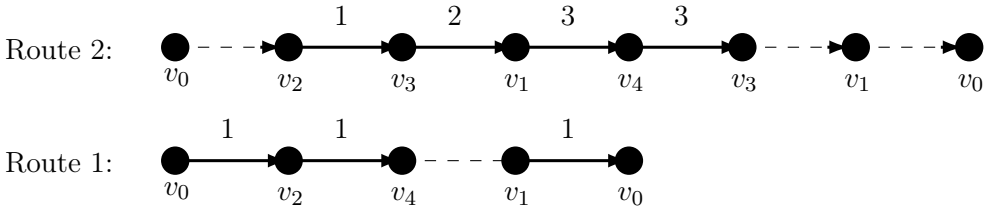


Figure 3: A feasible solution for the HDCARP-P instance in Figure 1.

different way to transform graph G to the sparser multigraph and utilise the dummy node to address the special cases, which will become clear.

The multigraph is constructed as follows: Initially, $G' = (V', A')$ is a duplicate of G. Then, we add a dummy node v'_0 to the node set V' ($V' = \{v'_0\} \cup V$). This dummy node serves as an intermediate node when transitioning between different classes on each route. For instance, when a vehicle completes the service of certain required arcs in class $k \in P$, it is required to visit node v'_0 before starting the service of required arcs in another class $k' \in P$ ($k' \neq k$). Moreover, the node visited immediately after v'_0 must be the same as the node visited right before v'_0. Every vehicle must make a visit to v'_0 both before departing from the depot and after completing the service of the last required arcs on its assigned route.

Next, we construct two sets of dummy arcs A_f and A_t. For each vertex $v \in V_t \cup \{v_0\}$, we add the arcs $\{v, v'_0\}$ and $\{v'_0, v\}$ to A_t and A_f, respectively. These arcs have zero traversal time. Finally, for each arc $a \in A_f \cup A_t$, we add arc a to A'. In other words, $A' = A \cup A_f \cup A_t$, which

means that $|A'|$ won't exceed $|A| + 2|V_t| + 2$.

Figure 4 represents how the above-mentioned transformation works for the graph G in Figure 1, with dummy arcs represented as dotted lines. Furthermore, Figure 5(a) and 5(b) show the routes in G' corresponding to the feasible solution of the HDCARP-P in Figure 2(a) and the feasible solution of the HDCARP-U in Figure 2(b), respectively.

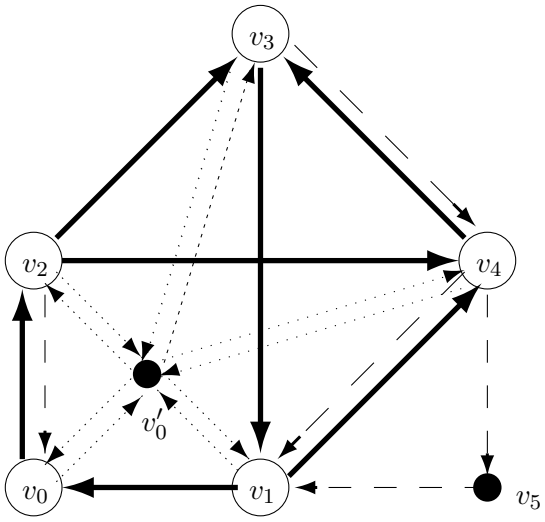


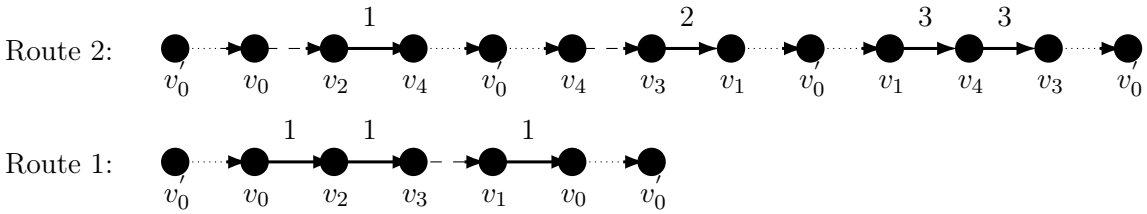
Figure 4: Auxiliary directed multigraph G' for the HDCARP instance in Figure 1

For convenience, we use some additional notations. Given a set $S \subseteq V'$, $A(S)$ denotes the set of arcs with both end-nodes in S, and $\delta(S)$ denotes the set of arcs with exactly one end-node in S. We denote $\delta^+(S)$ and $\delta^-(S)$ as the sets of arcs in G' leaving and entering S, respectively. Let's denote $A_r(S)$ as $A(S) \cap A_r$, and similarly for $\delta_r(S)$, $\delta_r^+(S)$, and $\delta_r^-(S)$. For each class $k \in \{1, 2, \ldots, p\}$, we also let $A_r^k(S)$ denote $A(S) \cap A_r^k$, and similarly for $\delta_r^k(S)$, $\delta_r^{+k}(S)$ and $\delta_r^{-k}(S)$. For simplicity, we sometimes write $\delta(v)$ instead of $\delta(\{v\})$. For improved notations, we use $\delta_{r,k}(S), \delta_{r,k}^+(S)$, and $\delta_{r,k}^-(S)$ instead of $\delta_r^k(S)$, $\delta_r^{+k}(S)$, and $\delta_r^{-k}(S)$, respectively.

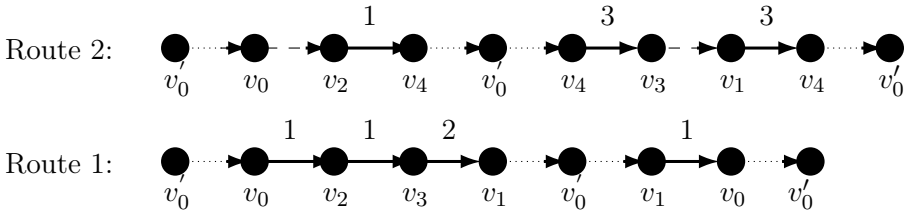
3.3. MILP model for the HDCARP-P

Our MILP model uses the following variables:

- Binary variables x_a^m indicate whether vehicle $m \in M$ services arc $a \in A_r$.
- Integer variables y_{ak}^m count the number of times vehicle $m \in M$ deadheads through arc $a \in A'$ on its chosen route while serving class $k \in P$.
- Non-negative real variables t_k^m represent the service completion time of class $k \in P$ on route $m \in M$ (assigned to vehicle m).
- Binary variables r_k^m indicate whether vehicle $m \in M$ services any required arcs in class $k \in P$.
- A non-negative real variable T_k represent the service completion time of class $k \in P$.



(a) The routes in G' corresponding to the solution in Figure 2(a).



(b) The routes in G' corresponding to the solution in Figure 2(b).

Figure 5: An illustration of routes in G'

The MILP formulation for the HDCARP-P is then as follows:

$$\underset{k=1,\ldots,p}{\text{lex-min}}\ T_k \tag{1}$$

$$\text{st. } T_k \geq t_k^m - \beta(1 - r_k^m) \qquad m \in M; k \in P \tag{2}$$

$$t_k^m = t_{k-1}^m + \sum_{a \in A_r^k} s_a x_a^m + \sum_{a \in A} d_a y_{ak}^m \qquad m \in M; k \in P \tag{3}$$

$$t_0^m = 0 \qquad m \in M \tag{4}$$

$$\sum_{a \in A_r^k} x_a^m \leq |A_r^k| r_k^m \qquad m \in M; k \in P \tag{5}$$

$$y_{\{v_0', v_0\},1}^m = 1 \qquad m \in M \tag{6}$$

$$\sum_{a \in A_f} y_{ak}^m = 1 \qquad m \in M; k \in P \tag{7}$$

$$y_{\{v_0', v\},k}^m = y_{\{v, v_0'\},k-1}^m \qquad m \in M; k \in P \setminus \{1\}; v \in \bigcup_{k'=1}^{k-1} V_t^{k'} \cup \{v_0\} \tag{8}$$

$$\sum_{m \in M} x_a^m = 1 \qquad a \in A_r \tag{9}$$

$$\sum_{k=1}^{p} \sum_{a \in A_r^k} q_a x_a^m \leq Q \qquad m \in M \tag{10}$$

$$\sum_{a \in \delta_{r,k}^+(v)} x_a^m + \sum_{a \in \delta^+(v)} y_{ak}^m = \sum_{a \in \delta_{r,k}^-(v)} x_a^m + \sum_{a \in \delta^-(v)} y_{ak}^m \qquad m \in M; k \in P; v \in V' \tag{11}$$

$$\sum_{a \in \delta_{r,k}^+(S)} x_a^m + \sum_{a \in \delta^+(S)} y_{ak}^m \geq x_b^m \qquad m \in M; k \in P; S \subseteq V; b \in A_r^k(S) \tag{12}$$

$$x_a^m \in \{0, 1\} \qquad m \in M; a \in A_r \tag{13}$$

$$y_{ak}^m \in \mathbb{N} \qquad m \in M; k \in P; a \in A' \qquad (14)$$

$$t_k^m \in \mathbb{R}^+ \qquad m \in M; k \in P \qquad (15)$$

$$r_k^m \in \{0,1\} \qquad m \in M; k \in P \qquad (16)$$

$$T_k \in \mathbb{R}^+ \qquad k \in P \qquad (17)$$

The objective function (1) represents the hierarchical objective, while constraints (2) ensure that the maximum completion time of class k is greater than or equal to the completion time of that class on any route that services at least one required arc in that class. Here, β is a suitably large number. (β can be set as the total time of the route that traverses all the required arcs while respecting the precedence relations. Constructing such a route can be easily achieved in a greedy manner.) Constraints (3) and (4) are time-conservation constraints, ensuring route connectivity. Constraints (5) limit the number of arcs in class k that can be serviced on each route. Constraints (6), (7), and (8) define class transitions on each route via node v_0'. Constraints (9) and (10) guarantee that each required arc is serviced exactly once and that vehicle capacity is not exceeded. Constraints (11) ensure that the number of times a vehicle departs from a node is equal to the number of times it arrives at that node for each class. Constraints (12) state that if route m services arc b, it must traverse any arc cutset that separates b from node v_0'. The remaining constraints define the domains of the variables.

3.4. MILP model for the HDCARP-U

In the HDCARP-U, where class upgrading is allowed, the specific order in which classes are traversed within each route is unknown. To address this, we introduce the concept of *hierarchy level*. The hierarchy level indicates the completion of a class's service within each route. This concept is illustrated using the feasible solution described in Figure 2(b). In route 1, three required arcs $\{v_0, v_2\}$, $\{v_2, v_3\}$, and $\{v_3, v_1\}$ are in hierarchy level 1, marking the service completion of class 2. The required arc $\{v_1, v_0\}$ is in hierarchy level 2, marking the service completion of class 1. Hierarchy level 3 does not have any required arcs. Similarly, in route 2, hierarchy levels 1 and 2 indicate the completion of servicing classes 1 and 3, respectively, while hierarchy level 3 does not contain any required arcs. It's important to note that in each route, the number of hierarchy levels is equal to the number of classes, including the possibility of empty sets. To represent transitions between hierarchy levels, we use the dummy node v_0'. To avoid confusion, we use the notation h to represent the hierarchy level and k to represent the class.

The MILP model for the HDCARP-U, similar to HDCARP-P, makes use of five variable types: x, r, y, t, and T. While the definitions of the last variable type remain unchanged, the first four variable types have been redefined as follows:

- Binary variables x_{ah}^m indicate whether vehicle $m \in M$ services the required arc $a \in A_r$ in hierarchy level $h \in P$.
- Integer variables y_{ah}^m count the number of times vehicle $m \in M$ deadheads through arc $a \in A'$ on its chosen route while serving hierarchy level $h \in P$
- Non-negative real variables t_h^m represent the time at which vehicle $m \in M$ completes servicing hierarchy level $h \in P$.
- Binary variables r_{kh}^m indicate whether hierarchy $h \in P$ services any required arcs in class k on route $m \in M$ (assigned to vehicle m).

The MILP model for the HDCARP-U is described as follows:

$$\underset{k=1,\ldots,p}{\text{lex-min}}\ T_k \tag{18}$$

$$\text{st. } T_k \geq t_h^m - \beta(1 - r_{kh}^m) \qquad m \in M; h, k \in P \tag{19}$$

$$t_h^m = t_{h-1}^m + \sum_{a \in A_r} s_a x_{ah}^m + \sum_{a \in A} d_a y_{ah}^m \qquad m \in M; h \in P \tag{20}$$

$$t_0^m = 0 \qquad m \in M \tag{21}$$

$$\sum_{a \in A_r^k} x_{ah}^m \leq |A_r^k| r_{kh}^m \qquad m \in M; h, k \in P \tag{22}$$

$$y_{\{v_0', v_0\},1}^m = 1 \qquad m \in M \tag{23}$$

$$\sum_{a \in A_f} y_{ah}^m = 1 \qquad m \in M; h \in P \tag{24}$$

$$y_{\{v_0', v\},h}^m = y_{\{v, v_0'\},h-1}^m \qquad m \in M; h \in P \setminus \{1\}; v \in V_t \cup \{v_0\} \tag{25}$$

$$\sum_{m \in M} \sum_{h=1}^{p} x_{ah}^m = 1 \qquad a \in A_r \tag{26}$$

$$\sum_{h=1}^{p} \sum_{a \in A_r} q_a x_{ah}^m \leq Q \qquad m \in M \tag{27}$$

$$\sum_{a \in \delta_r^+(v)} x_{ah}^m + \sum_{a \in \delta^+(v)} y_{ah}^m = \sum_{a \in \delta_r^-(v)} x_{ah}^m + \sum_{a \in \delta^-(v)} y_{ah}^m \qquad m \in M; h \in P; v \in V' \tag{28}$$

$$\sum_{a \in \delta_r^+(S)} x_{ah}^m + \sum_{a \in \delta^+(S)} y_{ah}^m \geq x_{bh}^m \qquad m \in M; h \in P; S \subseteq V; b \in A_r(S) \tag{29}$$

$$x_{ah}^m \in \{0, 1\} \qquad m \in M; h \in P; a \in A_r \tag{30}$$

$$y_{ah}^m \in \mathbb{N} \qquad m \in M; h \in P; a \in A' \tag{31}$$

$$t_h^m \in \mathbb{R}^+ \qquad m \in M; h \in P \tag{32}$$

$$r_{kh}^m \in \{0, 1\} \qquad m \in M; h, k \in P \tag{33}$$

$$T_k \in \mathbb{R}^+ \qquad k \in P \tag{34}$$

Constraints (19) ensure that the maximum completion time of a class is greater than or equal to the completion time of any hierarchy level on any route that includes at least one required arc of that class. Constraints (20), (26-29) are analogous to constraints (3), (9-12) of the HDCARP-P model, respectively, with the exception that required arcs can be serviced at any hierarchy level. Constraints (22) limit the maximum number of required arcs of class k that can be serviced in a single hierarchy level on each route. The remaining constraints and objective function are straightforward, and therefore, we omit them for brevity. To distinguish between the two MILP models, we refer to the MILP model for HDCARP-P as MILP-P and for HDCARP-U as MILP-U.

Since the HDCARP-U is an upgraded version of HDCARP-P, any feasible solution for the HDCARP-P is also feasible for the HDCARP-U. Therefore, the optimal maximum completion time sequence of the HDCARP-U is considered to be lexicographically better to that of the HDCARP-P. However, solving the HDCARP-U model can be more challenging and time-consuming due to the larger solution space.

3.5. Branch-and-cut algorithms

We applied the branch-and-cut algorithms from [23] to solve the MILP formulations. The models are first solved without the connectivity constraints ((12) for the HDCARP-P and (29)

for the HDCARP-U). We proceed by searching for violated connectivity inequalities. Any violated constraints are then included in the current MILP, which is subsequently reoptimised. This process is repeated until all the connectivity constraints are satisfied.

In order to identify violated connectivity inequalities, both heuristic and exact procedures can be used. Though an exact method can be done by solving min-cut problems, it is quite time-consuming. Hence, we first implemented a simple heuristic described in [23]. This heuristic initially computes the connected components and then checks if a connectivity constraint for each component is violated. The exact procedure is only called if no violated connectivity constraints are found by the simple heuristic.

4. MILP-based matheuristics

In this section, we propose matheuristics for both variants based on MILPs. For clarity, we propose two matheuristics for the HDCARP-P variant, namely MB1 and MB2-P, while we propose one matheuristic for the HDCARP-U variant named MB2-U, as we will see below. These matheuristics are specially tailored to provide good solutions within a reasonable time limit. The details are outlined in Algorithm 1. The idea is to decompose the original problems into p sub-problems, each focused on constructing partial routes for a subset of A_r. The kth sub-problem ($k \in P$) keeps track of the flow of time t_k^m and the current load l_k^m for each vehicle $m \in M$. These values are subsequently used in the $(k+1)$th sub-problem. Each sub-problem consists of two objectives prioritised as follows: (1) Minimise the maximum duration of partial routes, and (2) Minimise the total duration of all partial routes. The second objective is introduced with the aim of potentially decreasing the value of the primary objective for the following sub-problem.

This approach offers the advantage of reducing the problem size so that it can be tackled more easily and quickly. The sub-problems and their MILP formulation are described in the following subsections.

Algorithm 1: MILP-based matheuristic

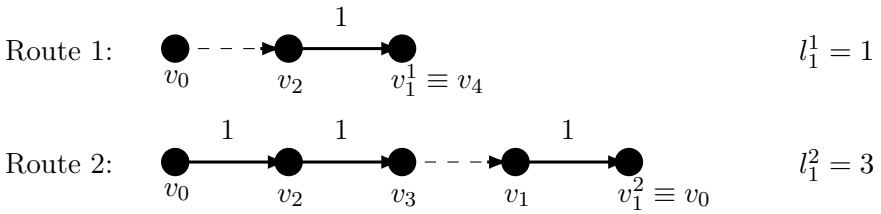
```
Set t_0^m = l_0^m = 0 for all m ∈ M;
for k = 1, ..., p do
    Set up the MILP for the kth sub-problem;
    Solve the MILP with the branch-and-cut algorithm in Section 3.5;
    Update t_k^m and l_k^m for all m ∈ M;
    Update T_k;
end
output: T_k for all k ∈ P
```

4.1. Sub-problems for the HDCARP-P

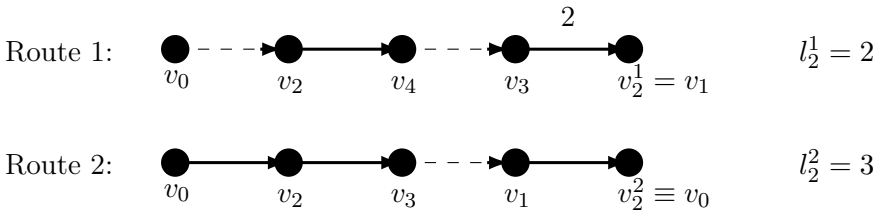
The HDCARP-P is decomposed into p sub-problems, each aimed at finding a set of partial routes for $|M|$ vehicles to service each class. Specifically, let's consider the scenario where $k-1$ sub-problems have already been solved ($1 \leq k \leq p$), and $|M|$ partial routes have been created for $|M|$ vehicles to service all required arcs in $A_r^1 \cup \ldots \cup A_r^{k-1}$. Assuming each partial route m takes t_{k-1}^m units of time and vehicle m is currently at node v_{k-1}^m with a load of l_{k-1}^m. The goal of the kth sub-problem is to identify a set of $|M|$ new partial routes that service each required

arc in A_r^k exactly once and satisfy the following conditions: (1) The partial route for vehicle m must begin from node v_{k-1}^m, and (2) The total load of the partial route for vehicle m must not exceed the maximum remaining load capacity, which is $Q - l_{k-1}^m$. These new partial routes are then connected to the existing ones based on linear precedence relations. As mentioned earlier, this sub-problem consists of two objective functions. The primary objective is to minimise the service completion time T_k for class k, while the secondary objective is to minimise the total travelling time of the partial routes ($\sum_{m \in M} t_k^m$).

Figure 6 shows a solution obtained by solving the two sub-problems of the HDCARP-P instance in Figure 1. Figure 6(a) illustrates one feasible solution for the first sub-problem. It consists of two partial routes assigned to two vehicles, servicing all required arcs in class 1. The first vehicle is currently located at node v_4 after $t_1^1 = 2$ units of time, carrying a load of $l_1^1 = 1$ unit. The second vehicle returns to node v_0 after $t_1^2 = 4$ units of time, carrying a load of $l_1^2 = 3$ units. Figure 6(b) presents a solution after solving the first two sub-problems for classes 1 and 2. The single required arc in the second class is serviced by the first vehicle. After an additional two units of time ($t_2^1 = 4$), the first vehicle is currently located at node v_1 and carries a demand of $l_2^1 = 2$ units.



(a) Two partial routes after solving 1st sub-problem for class 1.



(b) Two partial routes obtained after solving two sub-problems for classes 1 and 2.

Figure 6: An illustration of sub-problems for the HDCARP-P instance in Figure 1.

We will now present the MILP formulation for the kth sub-problem, using the same notation as in the MILP model for the HDCARP-P described in Section 3.3. However, since in the kth sub-problem only arcs in A_r^k are required to be serviced, we will drop the "class" index from the variables y, t, r, and T for simplicity. Note that t_{k-1}^m and t_{k-1}^m are fixed values from the solution of the $(k-1)$th sub-problem.

The MILP for the kth sub-problem is as follows:

$$\text{lex-min } T, \sum_{m \in M} t^m \tag{35}$$

$$\text{st. } T \geq t^m - \beta(1 - r^m) \qquad m \in M \qquad (36)$$

$$t^m = t^m_{k-1} + \sum_{a \in A^k_r} s_a x^m_a + \sum_{a \in A} d_a y^m_a \qquad m \in M \qquad (37)$$

$$\sum_{a \in A^k_r} x^m_a \leq |A^k_r| r^m \qquad m \in M \qquad (38)$$

$$y^m_{\{v'_0, v^m_{k-1}\}} = 1 \qquad m \in M \qquad (39)$$

$$\sum_{a \in A_f} y^m_a = 1 \qquad m \in M \qquad (40)$$

$$\sum_{m \in M} x^m_a = 1 \qquad a \in A^k_r \qquad (41)$$

$$\sum_{a \in A^k_r} q_a x^m_a \leq Q - l^m_{k-1} \qquad m \in M \qquad (42)$$

$$\sum_{a \in \delta^+_{r,k}(v)} x^m_a + \sum_{a \in \delta^+(v)} y^m_a = \sum_{a \in \delta^-_{r,k}(v)} x^m_a + \sum_{a \in \delta^-(v)} y^m_a \qquad m \in M, v \in V' \qquad (43)$$

$$\sum_{a \in \delta^+_{r,k}(S)} x^m_a + \sum_{a \in \delta^+(S)} y^m_a \geq x^m_b \qquad m \in M, S \subseteq V, b \in A^k_r(S) \qquad (44)$$

$$x^m_a \in \{0, 1\} \qquad m \in M, a \in A^k_r \qquad (45)$$

$$y^m_a \in \mathbb{N} \qquad m \in M, a \in A' \qquad (46)$$

$$t^m \in \mathbb{R}^+ \qquad m \in M \qquad (47)$$

$$r^m \in \{0, 1\} \qquad m \in M \qquad (48)$$

$$T \in \mathbb{R}^+ \qquad (49)$$

The hierarchical objective (35) minimises the maximum completion time of class k and then minimises the total duration of the partial routes. Constraints (36) and (37) define the maximum completion time of class k and the completion time of class k on each route. Constraints (39-40) ensure that each vehicle continues its route from node v^m_{k-1}. Constraints (42) ensure that the remaining capacity of the vehicles is not exceeded. The remaining constraints are trivial.

It can be observed that the MILP formulation for the kth sub-problem contains at most $|M|(|A^k_r| + |A| + 2|V_t| + 4) + 1$ variables, while the MILP formulation for the HDCARP-P in Section 3.3 has around p times the number of variables, at least $|M|\Big(|A_r| + p(|A| + 2|V_t| + 2)\Big) + p$ variables. Moreover, the hierarchical objective in each sub-problem objective consists of 2 objective functions, while that in the MILP-P formulation consists of p objectives. This suggests that the proposed matheuristic, denoted as MB1, is expected to be faster than solving optimally the MILP formulation for the HDCARP-P, especially when p is large, as solving MILP takes exponential time.

It is worth noting that each sub-problem may have different optimal solutions, but only one solution is required to generate the solution for the original problem.

4.2. Sub-problems for the HDCARP-U

Similarly, the HDCARP-U is decomposed into p sub-problems. However, unlike in the HDCARP-P, we cannot treat each class separately in the HDCARP-U, as the order of classes on each route is unknown. Therefore, we need to redefine $p - 1$ sub-problems, except for the first one. The resulting matheuristic is denoted as MB2-U, and as explained later, it can be adapted to HDCARP-P, referred as MB2-P matheuristic.

Consider the scenario where $k - 1$ sub-problems have been solved ($1 \leq k \leq p$), and all $|M|$ vehicles have successfully completed servicing the required arcs from classes 1 to $k-1$. Therefore,

we have detailed information about the assignment of these required arcs to hierarchy levels on each route. For each vehicle $m \in M$ and each hierarchy level $1 \leq h \leq k-1$, let $A_r^{h,m}$ denote the set of required arcs serviced by vehicle m in hierarchy level h. Additionally, we also know the completion time of hierarchy level h on route m (assigned to vehicle m), denoted as τ_h^m, and the current load of each vehicle m at time τ_h^m, denoted as l_h^m. We define the maximum completion time of hierarchy level h across all routes as C_h, given by $C_h = \max_{m \in M}\{\tau_h^m\}$.

The kth sub-problem aims to determine a set of $|M|$ partial routes that serve classes from 1 to k. This is subject to several conditions. Firstly, the total load of each partial route must not exceed the vehicle capacity. Secondly, required arcs from class 1 to $k-1$ will only be allowed to change their position within the already assigned hierarchy level and route. Thirdly, for $1 \leq h \leq k-1$, the maximum completion time of hierarchy level h on each route should not exceed C_h. The first condition is straightforward. The second condition restricts the solution space, thereby improving the running time of MB2-U. However, this restriction may increase the value of the hierarchical objective. The third constraint ensures that the maximum completion time for higher-priority classes does not increase when assigning required arcs in class k to existing hierarchy levels.

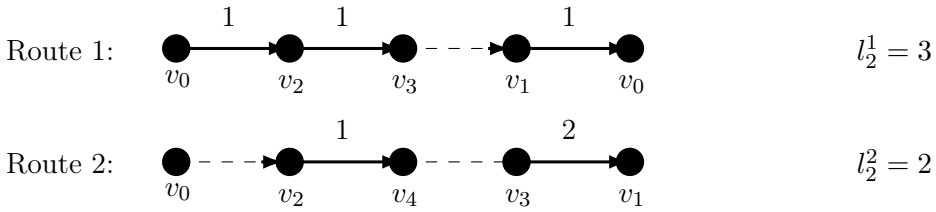
It can be seen that the kth sub-problem not only assigns the required arcs in class k to the existing partial routes but also offers limited flexibility in rearranging the order of required arcs to trade off between solution quality and running time. Similar to the HDCARP-P, the hierarchical objective in each sub-problem consists of two objective functions. The primary objective is to minimise the upper bound on the maximum completion time for class k, while the secondary objective is to minimise the total travel time of the partial routes. Note that after solving the kth sub-problem, the set of required arcs in each hierarchy level h on each route can change, as well as C_h $(1 \leq h \leq k-1)$. Therefore, they need to be re-updated for later sub-problems.

Figure 7 shows the solutions for two sub-problems of the HDCARP instance in Figure 1. Figure 7(a) presents one feasible solution after solving the second sub-problem. It contains two partial routes servicing all required arcs in $A_r^1 \cup A_r^2$. The first partial route currently ends at node v_0 and carries a load of $l_2^1 = 3$. It can be checked that $A_r^{1,1} = \{\{v_0, v_2\}, \{v_2, v_3\}, \{v_1, v_0\}\}$, $A_r^{2,1} = \emptyset$, $\tau_1^1 = \tau_2^1 = 4$. The second partial route ends at node v_1 and carries a load of $l_2^2 = 2$. Similarly, it can be checked that $A_r^{1,2} = \{\{v_2, v_4\}\}$, $A_r^{2,2} = \{\{v_3, v_1\}\}$, $\tau_1^2 = 2, \tau_2^2 = 4$. Therefore, $C_1 = C_2 = T_1 = T_2 = 4$.

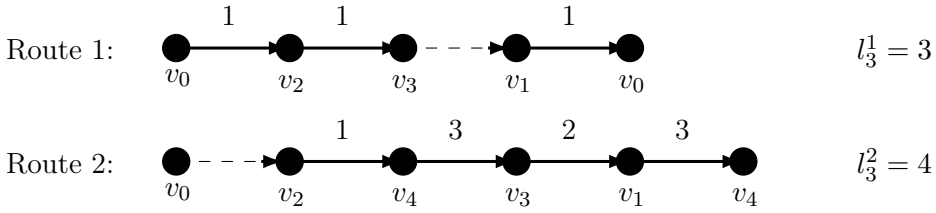
Figure 7(b) illustrates one feasible solution for the third sub-problem. For the first partial route, parameters for the first two hierarchy levels remain unchanged, and parameters for hierarchy level 3 are similar to those for hierarchy level 2 ($A_r^{3,1} = \emptyset, \tau_3^1 = 4$). For the second partial route, one can check that $A_r^{1,2} = \{\{v_2, v_4\}\}, A_r^{2,2} = \{\{v_4, v_3\}, \{v_3, v_1\}\}, A_r^{3,2} = \{\{v_1, v_4\}\}, \tau_1^2 = 2, \tau_2^2 = 4, \tau_3^2 = 5$. Therefore, $T_3 = C_3 = 5$. It can be observed that, after solving the third sub-problem, the second hierarchy level in the second partial route has been re-updated by incorporating the additional required arc $\{v_4, v_3\}$ of class 3.

We will now present the MILP formulation for the sub-problem k, using the same set of variables as in the MILP model for the HDCARP-U described in Section 3.4. However, since only arcs in $A_r^1 \cup \ldots \cup A_r^k$ are considered in the kth sub-problem, the range of the h index in variables x, y, and t has been limited to $\{1, \ldots, k\}$. Similarly, we will drop the "class" index from the variables r and T and the "hierarchy" index from the variables t for simplicity.

The formulation of sub-problem k is then as follow:



(a) Two partial routes after solving the second sub-problem.



(b) Two partial routes obtained after solving the third sub-problem

Figure 7: An illustration of sub-problems for the HDCARP-U instance in Figure 1.

$$\text{lex-min } T, \sum_{m \in M} t_k^m \tag{50}$$

$$\text{st. } T \geq t_h^m - \beta(1 - r_h^m) \qquad m \in M; h = 1, ..., k \tag{51}$$

$$t_h^m = t_{h-1}^m + \sum_{a \in A_r^{h,m}} s_a + \sum_{a \in A_r^k} s_a x_{ah}^m + \sum_{a \in A} d_a y_{ah}^m \qquad m \in M; h = 1, ..., k \tag{52}$$

$$\sum_{a \in A_r^k} x_{ah}^m \leq |A_r^k| r_h^m \qquad m \in M; h = 1, ..., k \tag{53}$$

$$y_{\{v_0', v_0\},1}^m = 1 \qquad m \in M \tag{54}$$

$$\sum_{a \in A_f} y_{ah}^m = 1 \qquad m \in M; h = 1, ..., k \tag{55}$$

$$y_{\{v_0', v\},h}^m = y_{\{v, v_0'\},h-1}^m \qquad m \in M; h = 2, ..., k; v \in V_t \cup \{v_0\} \tag{56}$$

$$\sum_{m \in M} \sum_{h=1}^{k} x_{ah}^m = 1 \qquad a \in A_r^k \tag{57}$$

$$\sum_{h=1}^{k} \sum_{a \in A_r^k} q_a x_{ah}^m \leq Q - l_{k-1}^m \qquad m \in M \tag{58}$$

$$\sum_{a \in \delta_r^+(v)} x_{ah}^m + \sum_{a \in \delta^+(v)} y_{ah}^m = \sum_{a \in \delta_r^-(v)} x_{ah}^m + \sum_{a \in \delta^-(v)} y_{ah}^m \qquad m \in M; h = 1, ..., k; v \in V' \tag{59}$$

$$\sum_{a \in \delta_r^+(S)} x_{ah}^m + \sum_{a \in \delta^+(S)} y_{ah}^m \geq x_{bh}^m \qquad m \in M; h = 1, ..., k; S \subseteq V; b \in A_r(S) \tag{60}$$

$$C_h \geq t_h^m \qquad m \in M; h = 1, ..., k-1 \tag{61}$$

$$x_{ah}^m = 1 \qquad m \in M; h = 1, ..., k-1; a \in A_r^{h,m} \tag{62}$$

$$x_{ah}^m \in \{0, 1\} \qquad m \in M; h = 1, ..., k; a \in A_r \tag{63}$$

$$y_{ah}^m \in \mathbb{N} \qquad m \in M; h = 1, ..., k; a \in A' \tag{64}$$

$$t_h^m \in \mathbb{R}^+ \qquad m \in M; h = 1, ..., k \tag{65}$$

$$r_h^m \in \{0, 1\} \qquad m \in M \tag{66}$$

$$T \in \mathbb{R}^+ \tag{67}$$

Constraints (51) define the upper bound on the maximum completion time of class k. Constraints (52) define the completion time of hierarchy level $h+1$ ($0 \leq h \leq k-1$) on each route, given that the set $A_r^{h+1,m}$ could be extended by including some required arcs in A_r^k. (Note that $A_r^{k,m}$ is initially set to be empty.) Constraints (53) indicate whether route m services any required arcs of class k in hierarchy level h. Constraints (61) set an upper bound on the servicing time of each vehicle for each hierarchy level from 1 to $k-1$. Constraints (62) guarantee that required arcs in $A_r^{h,m}$ must be serviced in hierarchy h by vehicle m. The objective function and the remaining constraints are trivial.

As previously mentioned, it is necessary to re-update the set of required arcs in each hierarchy level and its completion time after each sub-problem. The maximum completion time for each class can be determined after solving all k sub-problems. These updates and determinations can be carried out in a greedy manner.

The MB2-U matheuristic can be adapted to HDCARP-P. In HDCARP-P, linear precedence relations prohibit servicing required arcs of class k in any hierarchy level $h \leq k-1$. Therefore, after solving $k-1$ sub-problems, we only need to keep track of which required arcs of class $k-1$ were serviced on which routes, rather than for all classes from 1 to $k-1$. In other words, this restriction narrows down the set of hierarchy levels to $\{k\}$ in constraints (51-53), to $\{k-1\}$ in constraints (61-62), and to $\{k-1, k\}$ in constraints (52 - 60) as well as (63 - 65). As a result, the number of variables is reduced to at most $|M|(|A_r^k| + |A_r^{k-1}| + 2|A| + 4|V_t| + 8) + 1$. In other words, the objective of the kth sub-problem is to assign required arcs of class k to existing partial routes, while allowing for the rearrangement of required arcs of class $k-1$. The latter aims to reduce deadheading time between class $k-1$ and class k, potentially lowering T while keeping the maximum completion time of class $k-1$ unchanged. Consequently, we can eliminate the secondary objective in each sub-problem. This methodology is referred to as MB2-P.

5. Computational Experiments

In this section, we present the results of some computational experiments. For all of our computational experiments, we used a laptop with an Intel Core i5-13500 processor and 64GB RAM. All algorithms were implemented in C/C++. The MILP models and the sub-problems in the heuristic algorithms were solved by using CPLEX 22.1.2. We turn off all cuts in the CPLEX solver, reset CPLEX parameters to default, and set parallel mode to use eight threads.

To compare the performance of the different approaches, we set the time limit of an hour for solving the full MILP-U or MILP-P models in one pass, and 10 minutes for solving each sub-problem in matheuristic MB1, MB2-P, or MB2-U for every instance. Lexicographic objectives are handled by the default CPLEX solvers. To be more specific, distinct priorities are assigned to each objective, and the solver optimises them in decreasing order of priority. At each iteration, the solver chooses the best solution for each objective, considering those that do not degrade the value of higher-priority objectives.

5.1. Instance generation

Now, we explain how we created our artificial HDCARP instances to closely imitate real road network structures. Note that road networks are often planar and, even if not, they could

be made planar by deleting a very small number of road segments. In most of the planar road networks, the average number of road segments incident to each node usually ranges from 1.5 to 3 [6, 7]. Moreover, distances in the road network can be reasonably approximated by multiplying Euclidean distances by a suitable constant [5].

Our procedure to construct the graph $G = (V, A)$ is inspired by methods introduced in [23, 27], as follows:

- We generate randomly n vertices inside a unit square, where n is a multiple of 10 between 10 and 100. We denote the set of these vertices by V. These vertices have known coordinates. The first vertex is set to be the depot.
- Then, we construct the minimum Hamiltonian circuit passing through each node exactly once. We add all arcs in the giant circuit to the set A according to the direction in which they are traversed. The resulting graph G is now strongly connected.
- Additional arcs are introduced randomly into the graph following these conditions: (1) The total number of arcs is equal to $n \times d$ ($|A| = nd$), where $d \in \{1.5, 2, 2.5, 3\}$ represents the average number of arcs incident on each node in graph G; (2) The length of the added arcs are not excessively long; (3) None of the arcs intersect or cross each other.
- For each combination of n and $|A|$, five different version of graphs were generated.
- The arc set A is randomly divided into four separate sets: A_r^1, A_r^2, A_r^3, and A_{nr}. Each of the first three sets contains $\left\lfloor \frac{|A|}{4} \right\rfloor$ required arcs and corresponds to priority classes. The last set, A_{nr}, consists of non-required arcs. This means $A_r = A_r^1 \cup A_r^2 \cup A_r^3$ and $P = \{1, 2, 3\}$.
- For each arc $a \in A$, let d'_a be the Euclidean distance multiplied by a constant factor of 100. The deadheading cost d_a is set as $\max\{1, [d'_a + 0.5]\}$ to ensure it is always at least 1, regardless of the value of d'_a.
- For each arc $a \in A_r$, its demand q_a is calculated as $\lfloor d_a \times 0.5 + 0.5 \rfloor$, and its servicing cost s_a is set to twice its deadheading cost, i.e., $s_a = 2d_a$.
- There are three homogeneous vehicles with capacity $Q = \sum_{a \in A_r} q_a/3 + 0.5$, based at a depot.

To facilitate transparency, reproducibility, and further exploration, all benchmark instances used in our experimental evaluation - as well as the corresponding detailed computational results - have been made publicly available at: https://orlab.com.vn/materials/.

5.2. Experimental results on the small instances

This subsection starts with comparing the exact algorithm presented in Section 3 with two matheuristics, MB1 and MB2-P, for the HDCARP-P on 100 instances with up to 50 nodes. Table 2 presents the aggregated results for each combination of $|V|$ and $|A|$. The results shown are averaged over five versions. The first two columns show the number of vertices and arcs. For the exact algorithm, we report the number of instances solved to optimality (out of five versions) (#opt) and the average optimality gap (Gap (%)) for each objective in the hierarchical model (i.e., the maximum completion times for the first, second, and third priority classes), for instances solved within the time limit. In the case of each matheuristics, we present the number of instances where it produces better (B), equal (E), or worse (W) solutions compared to the

exact algorithm. Furthermore, we are interested in comparing two matheuristics. Hence, we show the number of instances in which MB2-P performs better (B_{MB1}), the same (E_{MB1}), or worse (W_{MB1}) than MB1. Finally, for each approach, we provide the average running time (Time) in seconds.

Instance		MILP-P			MB1				MB2-P						
|V|	|A|	#opt	Gap (%)	Time	B	E	W	Time	B	E	W	B_{MB1}	E_{MB1}	W_{MB1}	Time
10	15	5	(0.0, 0.0, 0.0)	0.07	0	4	1	0.04	0	4	1	0	4	1	0.03
	20	5	(0.0, 0.0, 0.0)	0.22	0	2	3	0.05	0	3	2	2	2	1	0.04
	25	5	(0.0, 0.0, 0.0)	0.40	0	2	3	0.06	0	2	3	0	4	1	0.04
	30	5	(0.0, 0.0, 0.0)	1.58	0	3	2	0.08	0	1	4	0	1	4	0.05
20	30	5	(0.0, 0.0, 0.0)	0.72	0	2	3	0.06	0	1	4	1	0	4	0.04
	40	5	(0.0, 0.0, 0.0)	2.26	0	4	1	0.11	0	5	0	1	4	0	0.07
	50	5	(0.0, 0.0, 0.0)	5.35	0	4	1	0.14	0	2	3	1	2	2	0.11
	60	5	(0.0, 0.0, 0.0)	46.44	0	5	0	0.41	0	2	3	0	2	3	0.26
30	45	5	(0.0, 0.0, 0.0)	1.35	0	5	0	0.08	0	3	2	0	3	2	0.06
	60	5	(0.0, 0.0, 0.0)	6.29	0	4	1	0.22	0	3	2	0	3	2	0.16
	75	5	(0.0, 0.0, 0.0)	147.10	0	4	1	0.64	0	1	4	0	1	4	0.51
	90	5	(0.0, 0.0, 0.0)	923.31	0	1	4	10.32	0	1	4	1	2	2	1.92
40	60	5	(0.0, 0.0, 0.0)	7.86	0	3	2	0.18	0	4	1	2	3	0	0.14
	80	5	(0.0, 0.0, 0.0)	50.23	0	3	2	0.45	0	2	3	0	3	2	0.37
	100	3	(0.0, 0.0, 0.6)	2317.21	0	5	0	6.89	0	4	1	0	4	1	4.65
	120	0	(0.0, 7.3, 10.9)	3601.17	3	0	2	56.49	3	0	2	1	2	2	40.74
50	75	5	(0.0, 0.0, 0.0)	15.69	0	3	2	0.30	0	3	2	1	4	0	0.20
	100	5	(0.0, 0.0, 0.0)	125.14	0	1	4	1.40	0	1	4	1	2	2	1.25
	125	2	(0.0, 0.0, 10.4)	2828.95	3	1	1	12.57	3	0	2	0	3	2	7.27
	150	1	(0.0, 4.0, 12.3)	3018.11	4	0	1	167.49	3	1	1	2	1	2	157.49
Total		86			10	56	34		9	43	48	13	50	37	

Table 2: HDCARP-P's results on small instances

The exact method solves optimally 86 out of 100 instances, including all with up to 90 arcs and 11 out of 25 instances with at least 100 arcs. Among the remaining instances, three could not be solved within the time limit. It is worth noting that our MILP formulation successfully captures a special case highlighted in Subsection 3.2. For example, in the optimal solution of one of the five instance versions with $|V| = 10$ and $|A| = 15$, the maximum completion times for classes 1, 2, and 3 are 249, 444, and 441, respectively.

The MB1 and MB2-P matheuristics have found solutions that are at least as good as the exact algorithm for 66 and 52 instances, respectively. Among these instances, MB1 and MB2-P are able to match the optimal solutions found by the exact method in 54 and 42 cases, respectively. The improved solutions obtained by these matheuristics were observed in instances with a minimum of 120 arcs. Additionally, the running time of both matheuristics was significantly shorter compared to the exact algorithm. These findings suggest that the MB1 and MB2-P matheuristics are especially promising for large-scale instances.

The MB2-P outperforms the MB1 in terms of solution quality for 13 instances. This confirms that the MB2-P has succeeded in reducing deadheading time between consecutive classes on each route. Additionally, the MB2-P runs slightly faster than the MB1 in all instances. A possible explanation for this is that the MB2-P has a single objective in each sub-problem, while the MB1 has two objectives that are considered in lexicographic order.

We now present results for the HDCARP-U variant. Table 3 reports the performance of the exact algorithm, MILP-U, and MB2-U on the same set of instances. Results are averaged over five versions and grouped by each $(|V|, |A|)$ combination. Table 3 is similar to Table 2, but

it includes detailed comparisons of MB2-U with MILP-U, MILP-P, and both MB1 and MB2-P. Specifically, for the comparison with MILP-U, the table reports the number of instances where MB2-U performs better (B), equal (E), or worse (W). For the comparison with MILP-P, it reports the number of instances where MB2-U performs better (B_P), equal (E_P), or worse (W_P). Finally, for the comparison with both MB1 and MB2-P, it shows the number of instances where MB2-U performs better than both (B'$_P$), at least as good as both (AE'$_P$), or worse than both (W'$_P$).

The exact algorithm found optimal solutions for 64 instances, all of which have at most 90 arcs. Among these, MB2-U was able to match the optimal solution in only 18 cases. This limited performance may be attributed to the constraints imposed in each sub-problem, which reduce the overall solution space. For the remaining instances that could not be solved to optimality by the exact method, MB2-U produced better solutions in 30 cases. Notably, for instances with $(|V|, |A|) \in (40, 100), (40, 120), (50, 125), (50, 150)$ that were unsolved within the time limit (TL), MB2-U successfully generated feasible solutions within a short computation time.

MB2-U improved upon the hierarchical objective obtained by the exact algorithm for HDCARP-P in only 12 out of 100 instances but achieved the same objective value in 78 instances. This demonstrates its effectiveness in maintaining the maximum completion time for high-priority classes.

Among the 53 instances where MB2-U performed at least as well as both MB1 and MB2-P, it strictly improved the hierarchical objective in 34 cases. The running time of MB2-U is similar to that of MB1, even though the model size in each sub-problem of the MB2-U heuristic is typically much larger than in MB1. This may be due to the small instance size, which results in relatively small class sizes.

Instance		MILP-U			MB2-U									
|V|	|A|	#opt	Gap	Time	B	E	W	B_P	E_P	W_P	B'$_P$	AE'$_P$	W'$_P$	Time
10	15	5	(0.0, 0.0, 0.0)	10.09	0	3	2	0	5	0	1	4	0	0.05
	20	5	(0.0, 0.0, 0.0)	36.22	0	1	4	1	4	0	1	3	1	0.28
	25	5	(0.0, 0.0, 0.0)	52.31	0	0	5	1	3	1	1	3	1	0.16
	30	5	(0.0, 0.0, 0.0)	32.53	0	0	5	0	4	1	3	2	0	0.10
20	30	5	(0.0, 0.0, 0.0)	90.43	0	0	5	1	4	0	4	0	1	0.14
	40	5	(0.0, 0.0, 0.0)	258.68	0	4	1	1	4	0	0	5	0	0.57
	50	5	(0.0, 0.0, 0.0)	502.31	0	3	2	0	5	0	2	2	1	1.53
	60	3	(0.0, 0.0, 10.3)	2098.01	2	1	2	0	4	1	2	3	0	1.62
30	45	5	(0.0, 0.0, 0.0)	187.68	0	1	4	0	3	2	2	2	1	0.73
	60	5	(0.0, 0.0, 0.0)	689.04	0	1	4	0	5	0	2	3	0	0.73
	75	4	(0.0, 0.0, 0.6)	2482.37	0	2	3	0	3	2	2	3	0	1.92
	90	1	(0.4, 2.6, 16.4)	3347.74	4	0	1	1	4	0	3	1	1	21.49
40	60	5	(0.0, 0.0, 0.0)	1588.77	0	0	5	0	5	0	0	3	2	0.65
	80	4	(0.0, 0.0, 8.8)	1655.23	1	1	3	1	4	0	2	3	0	1.94
	100	0	(0.8, 0.6, TL)	3600.34	4	0	1	0	5	0	1	4	0	9.64
	120	0	(12.0, 3.9, TL)	3600.53	5	0	0	1	3	1	1	3	1	115.69
50	75	2	(0.0, 0.0, 11.7)	2552.86	2	1	2	1	3	1	1	3	1	1.14
	100	0	(0.4, 1.1, 58.6)	3602.94	3	0	2	2	2	1	1	3	1	4.56
	125	0	(3.4, TL, TL)	3600.84	5	0	0	1	4	0	3	2	0	21.12
	150	0	(21.1, 1.8, TL)	3600.71	4	0	1	1	4	0	2	1	2	176.05
Total		64			30	18	52	12	78	10	34	53	13	

Table 3: HDCARP-U's results on small instances

In summary, the exact approaches are only effective for small-scale instances and require substantial computational time, often failing to return solutions within the time limit for larger instances. In contrast, the matheuristics consistently produce reasonably good solutions within a short computing time across all instances in the experiment.

5.3. Comparing matheuristics on the large instances

We conducted experiments with our matheuristics on larger instances, ranging from 60 to 100 nodes. The results are presented in Tables 4 and 5. The first three columns show the number of nodes $|V|$, the number of arcs $|A|$, and the version of each instance. The next four columns report the maximum completion time for each class (T_1, T_2, and T_3), as well as the running time (in seconds) for MB1. For MB2-P or MB2-U, we also report the percentage gap in the maximum completion time for each class, denoted as Gap_1, Gap_2, and Gap_3, along with the running time in seconds. For $k \in \{1, 2, 3\}$, the gap Gap_k is calculated as $100 \times (T'_k - T_k)/T_k$, where T_k and T'_k represent the maximum completion times of class k obtained by MB1 and MB2-P (either MB2-P and MB2-U), respectively. A negative value of Gap_k indicates an improvement, while a positive value indicates a deterioration in solution quality.

The performance of MB1 and MB2-P for solving HDCARP-P is comparable in both runtime and solution quality. In terms of solution quality, MB1 outperforms MB2-P in 38 instances, while MB2-P provides better solutions in 31 instances.

Although MB2-U fails to produce feasible solutions for four instances, it significantly improves the hierarchical objective in 26 and 41 instances compared to MB1 and MB2-P, respectively. Furthermore, MB2-U outperforms the better of the two matheuristics in 19 instances. This experiment on the larger instance set further supports the effectiveness of the upgrading variant (HDCARP-U) in reducing the maximum completion time of high-priority classes in many instances.

MB2-U generally runs slightly slower than MB2-P due to the larger size of its sub-problems. In contrast, MB1 is significantly slower than MB2-P. This observation is consistent with the results from the small-instance experiments and further supports the explanation that MB1's slower performance is due to its use of two lexicographically ordered objectives, whereas MB2-P addresses sub-problems with a single objective.

5.4. Total time comparison of the HDCARP-P and HDCARP-U

To better understand the effect of class upgrading, we also examined the total time across all routes for each HDCARP variant. Specifically, we compared the best solutions obtained from MB1 or MB2-P (for HDCARP-P) with the solutions from MB2-U (for HDCARP-U) to evaluate how upgrading impacts overall routing efficiency.

Table 6 reports the average total time for the HDCARP-P and HDCARP-U variants, across different combinations of $|V|$ and d. It also shows the percentage improvement (Gap) achieved by allowing class upgrading in HDCARP-U. The Gap is calculated as $\text{Gap} = 100 \times (\overline{T_P} - \overline{T_U})/\overline{T_P}$, where $\overline{T_P}$ and $\overline{T_U}$ are the average total times in HDCARP-P and HDCARP-U, respectively.

It can be seen that HDCARP-P performs slightly better, although the difference between the two variants is relatively small. This is due to the following reasons: (1) the hierarchical objective focuses on minimising the maximum completion time of high-priority classes, and (2) the total time of each route includes both the time spent on serving tasks and the deadheading time to return to the depot. However, the hierarchical objective does not take this deadheading time into account.

Instance			MB1				MB2-P				MB2-U			
$\|V\|$	$\|A\|$	i	T_1	T_2	T_3	Time	Gap_1	Gap_2	Gap_3	Time	Gap_1	Gap_2	Gap_3	Time
60	90	1	455	897	1252	0.26	0.00	0.00	-0.32	0.18	0.00	-0.67	0.24	0.75
		2	478	761	1082	0.33	0.00	0.53	0.37	0.20	0.00	0.00	0.00	1.24
		3	525	886	1277	0.24	0.00	0.00	0.00	0.23	0.00	0.00	0.00	1.10
		4	547	1066	1518	0.31	0.00	0.00	0.00	0.19	0.00	0.00	0.00	0.69
		5	507	916	1296	0.64	0.00	0.00	1.31	0.51	0.00	0.00	0.00	1.33
	120	1	616	1065	1471	6.18	0.00	0.00	0.00	3.73	0.00	0.00	0.00	6.69
		2	466	1132	1750	4.50	0.00	0.00	1.09	4.16	0.00	0.00	0.00	8.33
		3	596	1279	1702	5.37	0.00	0.00	0.00	4.48	0.00	0.00	0.00	78.87
		4	439	916	1322	14.05	0.00	0.00	0.15	10.51	0.00	0.00	0.00	11.98
		5	478	871	1279	4.09	0.00	-0.11	-0.08	5.25	0.00	0.00	0.00	18.13
	150	1	580	1115	1584	603.25	-0.17	-0.09	0.00	607.82	0.00	0.00	-0.06	618.75
		2	556	1123	1594	6.36	0.00	0.00	0.13	10.18	0.00	0.00	0.06	13.11
		3	548	1128	1715	77.31	0.00	0.00	-0.06	31.51	0.00	0.00	0.00	79.99
		4	456	1090	1591	8.16	0.00	-0.09	-0.06	10.52	0.00	0.00	0.00	18.34
		5	621	1131	1714	43.52	0.00	0.00	0.06	88.30	0.00	0.00	0.00	69.97
	180	1	698	1311	1877	1489.68	1.15	2.36	0.80	1205.59	-0.29	1.14	-0.05	1206.67
		2	593	1186	1787	657.51	0.00	0.00	0.45	727.63	0.00	0.00	0.00	654.44
		3	630	1219	1775	91.05	0.00	0.00	0.00	88.51	0.00	0.00	0.00	103.74
		4	593	1202	1689	249.35	0.00	0.00	-0.06	485.95	0.00	1.41	0.47	772.03
		5	573	1214	1833	1323.16	0.00	0.00	0.00	1287.86	0.00	0.00	6.60	1256.56
70	105	1	531	998	1580	1.12	0.00	0.00	0.00	0.72	0.00	0.00	0.00	4.67
		2	547	1120	1555	1.11	0.00	0.00	0.00	0.45	0.00	0.00	0.00	1.63
		3	492	1018	1515	0.57	0.00	0.20	0.20	0.31	0.00	0.29	0.13	5.85
		4	525	946	1491	0.56	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.83
		5	537	966	1281	0.39	0.00	0.00	0.08	0.23	0.00	0.00	-0.08	2.58
	140	1	682	1223	1680	18.29	0.00	0.00	0.00	23.01	0.00	0.00	0.00	19.47
		2	425	831	1272	11.58	0.00	0.00	0.00	8.33	0.00	0.00	0.00	15.46
		3	580	1162	1743	7.83	0.00	0.00	0.00	6.48	0.00	0.00	0.00	7.22
		4	469	974	1460	1.96	0.00	8.01	6.03	10.24	0.00	0.00	0.00	5.99
		5	522	980	1509	3.29	0.00	0.00	0.00	3.30	0.00	0.00	0.00	6.05
	175	1	595	1151	1816	19.11	0.00	0.00	0.11	14.32	0.00	0.00	0.00	20.20
		2	527	1021	1568	14.13	0.00	0.00	0.00	16.92	0.00	0.00	0.00	12.84
		3	523	995	1601	157.38	0.00	0.40	0.19	139.45	0.00	0.00	0.00	513.85
		4	620	1234	1753	67.05	0.00	0.00	0.06	48.36	0.00	0.00	0.00	82.56
		5	565	1092	1559	20.77	0.00	0.00	0.00	12.66	0.00	0.00	0.00	31.12
	210	1	691	1419	2347	620.51	0.00	0.00	-0.55	624.95	0.00	0.00	18.41	674.82
		2	633	1242	1896	283.57	0.00	0.00	-0.05	187.21	0.00	-0.16	0.00	347.65
		3	638	1226	1792	680.27	0.00	0.00	0.00	97.95	0.00	1.88	4.19	1580.42
		4	575	1144	1825	1208.93	0.00	0.00	-0.66	1207.62	0.17	-0.09	1.59	1235.60
		5	719	1355	2017	463.84	0.00	0.00	0.05	671.94	0.00	0.00	0.05	238.41
80	120	1	692	1368	1768	1.66	0.00	-0.07	1.02	1.90	0.00	-0.07	0.11	4.11
		2	649	1159	1592	3.18	0.00	0.26	-0.06	4.97	0.00	0.09	-0.06	3.54
		3	556	991	1467	5.05	0.00	0.00	0.00	3.57	0.00	-0.10	0.00	7.28
		4	576	967	1450	1.44	0.00	0.00	0.00	0.90	0.00	0.00	0.00	2.61
		5	520	1073	1574	2.62	0.00	0.00	0.00	1.75	0.00	0.00	0.00	5.62
	160	1	574	1068	1710	65.99	0.00	-0.09	0.06	22.80	0.00	0.00	0.00	57.38
		2	644	1109	1686	2.72	0.00	-0.09	0.00	2.65	0.00	0.00	0.00	5.50
		3	528	1031	1549	1213.23	0.19	0.29	0.19	613.11	0.00	0.10	0.19	615.37
		4	589	1062	1497	606.51	0.00	0.09	3.94	108.47	0.00	0.09	0.00	613.55
		5	575	1047	1513	27.44	0.00	0.19	0.33	11.06	0.00	-0.10	0.00	18.57

Table 4: The results of matheuristics on large instances

Instance			MB1				MB2-P				MB2-U			
$\|V\|$	$\|A\|$	i	T_1	T_2	T_3	Time	Gap_1	Gap_2	Gap_3	Time	Gap_1	Gap_2	Gap_3	Time
80	200	1	683	1373	2027	1204.33	0.00	1.02	-1.87	620.10	0.00	-1.24	-2.12	1204.27
		2	809	1406	2107	673.52	-0.87	-0.92	-0.62	698.47	-1.61	-0.36	-1.80	1800.32
		3	634	1276	1874	33.09	0.00	0.00	0.11	45.79	0.00	0.00	0.00	36.89
		4	624	1142	1652	170.17	0.00	0.00	0.00	321.81	0.00	0.00	0.00	500.80
		5	880	1484	2233	620.63	0.00	0.00	-0.04	612.65	0.00	-0.07	-0.04	615.54
	240	1	651	1401	2001	TL	0.00	9.71	5.10	1254.63	0.00	1.07	0.80	1253.49
		2	614	1222	1833	1172.19	0.00	0.00	-0.87	578.90	6.35	7.28	6.93	TL
		3	746	1400	2141	862.76	2.68	1.29	2.80	1239.60	-4.29	-2.79	0.70	1430.52
		4	718	1325	2023	716.13	11.28	6.11	3.71	658.49	5.99	TL	TL	1200.50
		5	675	1318	2105	626.46	0.00	0.00	1.71	634.15	0.00	0.08	2.19	643.63
90	135	1	507	962	1335	1.79	0.00	-0.83	-0.90	0.84	0.00	0.00	-0.67	4.30
		2	610	1160	1642	51.56	0.00	0.00	-0.30	18.04	0.00	0.00	0.00	39.89
		3	655	1109	1570	601.43	0.00	0.00	0.00	601.64	0.00	0.00	0.00	604.51
		4	571	1034	1553	2.87	0.00	0.58	0.71	3.18	0.00	0.58	-0.13	6.40
		5	472	1062	1665	7.65	0.00	-7.06	-6.61	3.26	0.00	0.00	0.00	13.97
	180	1	666	1166	1800	611.63	0.00	0.00	-0.06	397.67	0.00	0.00	0.00	192.77
		2	631	1135	1705	437.37	0.00	0.00	0.00	623.38	0.00	0.00	0.06	668.01
		3	564	1109	1748	12.34	0.00	0.00	0.00	18.39	0.00	0.00	0.00	23.71
		4	608	1129	1612	18.38	0.00	0.00	0.25	6.67	0.00	0.00	0.00	21.79
		5	674	1299	1860	12.13	0.00	0.08	0.32	5.96	0.00	0.00	0.00	11.59
	225	1	730	1310	1909	651.05	-19.18	-8.85	-4.92	738.33	-20.96	-9.69	-6.55	763.77
		2	663	1267	1925	29.79	0.00	0.00	0.00	18.26	0.00	0.00	0.00	120.41
		3	708	1436	2039	833.75	0.00	-0.91	2.89	1270.65	0.00	0.14	0.88	839.84
		4	792	1418	2041	1417.01	4.04	2.05	1.57	1452.19	-0.63	5.85	17.98	TL
		5	648	1449	2177	1182.83	0.00	0.00	-0.09	1213.79	0.00	0.00	-0.14	717.38
	270	1	729	1455	2169	1300.09	4.39	2.75	6.41	1208.91	-3.16	-3.30	0.28	1334.99
		2	832	1643	2468	TL	6.13	-3.59	-4.46	711.32	12.38	3.35	0.24	TL
		3	717	2054	2708	1345.98	0.28	-22.69	-3.06	TL	1.26	-27.75	-12.00	TL
		4	738	1431	2229	TL	-1.36	1.05	3.23	TL	0.14	TL	TL	1200.52
		5	753	1471	2236	887.78	0.00	0.00	-0.22	645.45	0.00	0.00	0.00	941.60
100	150	1	548	1146	1694	0.80	0.00	-0.09	0.06	0.75	0.00	0.00	-0.18	3.00
		2	593	1163	1677	3.68	0.00	0.09	0.00	4.78	0.00	0.09	-0.06	4.01
		3	653	1543	1998	0.55	0.00	0.00	0.00	0.45	0.00	0.00	0.00	1.87
		4	644	1178	1857	492.85	0.00	0.00	0.00	815.67	0.00	0.08	0.00	903.86
		5	645	1091	1686	2.34	0.00	0.00	0.00	1.63	0.00	0.00	0.00	6.14
	200	1	770	1638	2664	666.43	0.00	0.00	-3.87	729.20	0.00	0.00	-2.44	1202.10
		2	671	1378	1955	741.43	0.00	11.68	9.10	825.62	0.00	0.00	0.05	652.17
		3	635	1200	1764	8.87	0.00	0.00	0.06	10.64	0.00	0.00	0.06	26.79
		4	623	1304	1903	616.37	0.00	-3.76	-3.36	666.18	0.00	-3.76	-2.42	652.75
		5	675	1341	1990	179.96	0.00	0.00	0.00	35.63	0.00	0.00	0.00	380.32
	250	1	657	1223	1893	144.16	0.00	0.00	0.11	138.93	0.00	0.00	-0.16	196.65
		2	671	1246	1969	308.43	0.00	-0.08	0.00	277.72	0.00	0.00	0.00	359.22
		3	693	1314	2088	1090.03	0.00	0.00	0.00	667.07	0.00	0.00	0.05	756.81
		4	753	1667	2309	1220.37	-1.46	-11.34	-6.19	1475.28	-1.46	-9.06	3.85	TL
		5	740	1380	2102	703.60	0.00	0.00	0.00	627.05	0.00	0.07	0.00	642.79
	300	1	791	1561	2490	1356.55	-3.16	2.75	-3.86	TL	-3.16	-0.64	-4.58	1301.26
		2	763	1578	2298	719.48	0.00	0.00	0.00	1346.59	0.00	0.06	TL	1435.15
		3	831	1568	2433	1257.10	0.48	0.19	0.12	1281.07	-0.36	3.89	TL	TL
		4	863	2028	2588	TL	0.00	4.44	11.36	TL	0.00	-1.58	-8.15	TL
		5	904	1750	2580	1621.69	-10.51	-5.43	-1.32	1783.42	-9.62	-5.03	-1.98	TL

Table 5: The results of matheuristics on large instances (continued)

$\|V\|$	d	Average Total Time		Gap
		HDCARP-P	HDCARP-U	
10	1.5	1074.0	1074.0	0.00
	2	1298.6	1303.2	-0.35
	2.5	1268.0	1264.6	0.27
	3	1242.6	1246.4	-0.30
20	1.5	1560.6	1563.0	-0.15
	2	1780.6	1780.6	0.00
	2.5	1842.6	1837.6	0.27
	3	2024.4	2024.6	-0.01
30	1.5	1772.0	1750.2	1.25
	2	2158.4	2158.4	0.00
	2.5	2106.8	2107.6	-0.04
	3	2643.2	2644.8	-0.06
40	1.5	2196.8	2192.8	0.18
	2	2235.6	2235.2	0.02
	2.5	2777.0	2777.0	0.00
	3	2894.2	2894.6	-0.01
50	1.5	2754.2	2754.4	-0.01
	2	2638.2	2640.0	-0.07
	2.5	2854.0	2853.8	0.01
	3	3176.6	3178.0	-0.04
60	1.5	2691.8	2692.0	-0.01
	2	3076.0	3076.4	-0.01
	2.5	3308.2	3309.2	-0.03
	3	3635.8	3667.6	-0.87
70	1.5	3020.4	3021.2	-0.03
	2	3102.4	3102.4	0.00
	2.5	3324.0	3324.0	0.00
	3	3898.6	4015.4	-2.91
80	1.5	3283.8	3280.4	0.10
	2	3236.2	3237.2	-0.03
	2.5	4034.0	4017.2	0.42
	3	4031.4	4096.0	-1.58
90	1.5	3139.4	3180.4	-1.29
	2	3541.0	3541.4	-0.01
	2.5	4041.2	4113.8	-1.76
	3	4741.0	4612.0	2.80
100	1.5	3623.2	3622.8	0.01
	2	4059.0	4070.6	-0.28
	2.5	4072.2	4126.2	-1.31
	3	4944.8	4979.0	-0.69

Table 6: Comparison of average total time between HDCARP-P and HDCARP-U

5.5. *Effect of increasing the number of classes*

For completeness and to enhance understanding, we investigate the effect of increasing the number of priority classes. While our default setting involves three priority classes, we extend the analysis to scenarios with five classes.

For HDCARP-P, the size of the MILP-P formulation increases approximately linearly with the number of classes p. In contrast, the MILP-U formulation for HDCARP-U grows roughly quadratically with p, which poses significant challenges for exact approaches. As a result, this experiment focuses solely on the matheuristic methods. For brevity, we restrict this analysis to instances with 50 nodes. Results are presented in Tables 7 and 8.

Table 7 presents the average running time (measured in seconds) across instance versions for each value of $|A|$. It can be observed that MB2-P is only slightly slower than MB1, which contrasts with the results from instances with three priority classes. This indicates that, although MB2-P handles sub-problems with one fewer objective than MB1, the increased size of its sub-problem formulations offsets this computational benefit.

It can be observed that the running times of both MB1 and MB2-P on instances with five classes are shorter than those on instances with three classes. This phenomenon can be explained by the fact that, although the number of sub-problems increases with p, the size of each sub-problem decreases when $|A|$ is held constant. As a result, each sub-problem becomes easier to solve, leading to shorter overall running times. However, this phenomenon is not observed for MB2-U, as its formulation size grows approximately quadratically with the number of classes p, resulting in increased computational effort despite smaller priority class sizes.

$\|A\|$	MB1	MB2-P	MB2-U
75	0.21	0.73	4.76
100	1.13	3.00	30.18
125	14.23	18.86	147.90
150	12.28	18.15	208.45

Table 7: Average running times of matheuristics on 50-node, 5-class instances

Table 8 follows the same structure as Tables 4-5, but omits the columns for $|V|$ and Gap_1, as the number of nodes is fixed at 50 and Gap_1 is consistently 0. One can check that MB2-P improves the hierarchical objective in 16 out of 20 instances compared to MB1, and performs worse in only 3 instances. This is expected, as allowing the rearrangement of required arcs within a priority class helps reduce deadheading time between successive priority classes. This effect becomes more evident as the number of classes increases. However, we did not observe a significant improvement from the upgrading possibility in reducing the completion time of priority classes. This may be due to the time limit imposed on solving each subproblem, especially since the subproblem formulations for lower-priority classes in instances with 5 classes are significantly larger than those in instances with 3 classes.

6. Conclusions

In this paper, we have introduced the HDCARP, which is a generalization of the CARP and is particularly suitable for modeling routing problems that involve the categorisation of roads into priority classes. We have studied two HDCARP variants, namely HDCARP-P and HDCARP-U.

Instance		MB1					MB2-P				MB2-U			
$\|A\|$	i	T_1	T_2	T_3	T_4	T_5	Gap_2	Gap_3	Gap_4	Gap_5	Gap_2	Gap_3	Gap_4	Gap_5
	0	460	799	1066	1352	1594	3.0	5.0	5.0	5.8	1.5	1.2	-2.3	-0.4
	1	487	795	1038	1385	1629	0.0	-0.6	0.4	-0.7	-1.0	-2.6	-12.9	-6.3
75	2	445	766	1066	1371	1716	0.0	-0.4	-0.1	-2.0	0.0	-0.7	-0.1	-1.7
	3	564	916	1221	1408	1641	-0.3	-0.9	-0.9	-0.7	0.0	0.0	0.0	0.0
	4	433	676	935	1222	1584	0.0	2.8	2.3	-2.4	0.0	0.0	0.0	0.0
	0	417	778	1078	1362	1643	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	1	400	861	1226	1565	1962	-0.8	-0.6	-0.4	-0.4	0.0	-0.1	0.0	0.0
100	2	349	637	839	1192	1532	0.0	-0.5	-5.4	-1.6	0.0	0.0	-0.3	0.0
	3	391	640	902	1241	1443	0.0	-0.4	-0.1	2.9	0.0	0.0	0.0	0.0
	4	516	930	1326	1762	2235	0.0	-1.7	1.0	0.6	0.0	0.1	-0.1	0.0
	0	423	774	1103	1440	1721	-6.1	-1.7	-1.3	-1.5	-0.6	0.4	0.6	0.4
	1	356	692	1022	1391	1690	0.0	0.0	0.0	-0.7	0.0	0.0	0.0	0.0
125	2	466	751	1033	1335	1717	0.0	0.0	0.0	-0.8	0.0	0.0	0.0	0.0
	3	346	789	1189	1552	1894	-0.6	0.8	1.3	1.7	0.0	0.0	0.0	0.0
	4	325	735	1114	1468	1797	0.0	0.0	-1.0	-1.1	0.0	0.0	0.0	0.0
	0	427	805	1153	1547	1933	0.0	-0.3	-0.9	-1.6	0.0	0.0	0.0	0.1
	1	509	911	1275	1659	2088	0.0	0.3	0.2	-1.2	0.0	0.0	0.0	0.0
150	2	458	857	1199	1598	1918	0.0	-0.2	-0.2	1.9	0.0	0.0	0.0	0.0
	3	366	747	1130	1516	1917	-0.4	-1.0	-1.9	-1.7	-0.1	1.1	0.4	0.3
	4	366	758	1119	1469	1854	0.0	0.0	-0.7	-0.4	0.0	0.0	0.0	0.0

Table 8: Matheuristic solutions on 50-node, 5-class instances

Both variants share the same hierarchial objective. However, HDCARP-P gives strict priority to higher-priority roads before servicing lower-priority ones, whereas HDCARP-U allows for more flexibility, enabling lower-priority roads to be serviced before some higher-priority ones. We proposed the mathematical models and matheuristics for both variants and conducted extensive computational experiments. The results indicate that the exact methods are only suitable for small instances, while the proposed matheuristics work rather well in most cases. Furthermore, a comparison between the two variants was performed to evaluate the impact of class upgrading option. Despite the matheuristic's restriction, the HDCARP-U still illustrates the benefits of class upgrading in improving completion times for higher-priority classes in many cases.

Our approaches could be fairly easily adapted to other HDCARPs, such as HDCARPs with multiple depots, heterogeneous vehicles or HDCARPs on mixed graphs.

We can think of three possible topics for future research. The first topic involves the development of fast heuristics to efficiently handle real-world instances because matheuristics proposed in this study are still time-consuming. The second is the development of a local search heuristic to further improve the solutions. Finally, the third topic involves the development of a matheuristic approach specifically designed for the HDCARP-U variant to improve the total completion time without any restrictions.

References

References

[1] V.A. Afanasev, R. van Bevern, and O.Y. Tsidulko. The hierarchical chinese postman problem: the slightest disorder makes it hard, yet disconnectedness is manageable. *Operations Research Letters*, 49(2):270–277, 2021.

[2] C. Ahabchane, A. Langevin, and M. Trépanier. Robust optimization for the hierarchical mixed capacitated general routing problem applied to winter road maintenance. *Computers & Industrial Engineering*, 158:107396, 05 2021.

[3] C. Ahabchane, M. Trépanier, and A. Langevin. Street-segment-based salt and abrasive prediction for winter maintenance using machine learning and GIS. *Transactions in GIS*, 23(1):48–69, 2019.

[4] A.S. Alfa and D.Q. Liu. Postman routing problem in a hierarchical network. *Engineering Optimization*, 14(2):127–138, 1988.

[5] B. Boyacı, T.H. Dang, and A.N. Letchford. Vehicle routing on road networks: how good is Euclidean approximation? *Computers & Operations Research*, 129:105197, 2021.

[6] B. Boyacı, T.H. Dang, and A.N. Letchford. On matchings, T-joins and arc routing in road networks. *Networks*, 79(1):20– 31, 2022.

[7] B. Boyacı, T.H. Dang, and A.N. Letchford. Fast upper and lower bounds for a large-scale real-world arc routing problem. *Networks*, 81(1):107–124, 2023.

[8] E.A. Cabral, M. Gendreau, G. Ghiani, and G. Laporte. Solving the hierarchical chinese postman problem as a rural postman problem. *European Journal of Operational Research*, 155(1):44–50, 2004.

[9] J.F. Campbell and A. Langevin. Operations management for urban snow removal and disposal. *Transportation Research Part A: Policy and Practice*, 29(5):359–370, 1995.

[10] N. Christofides. The optimum traversal of a graph. *Omega*, 1(6):719–732, 1973.

[11] M.K. Çodur and M. Yılmaz. A time-dependent hierarchical chinese postman problem. *Central European Journal of Operations Research*, pages 1–30, 2018.

[12] M. Colombi, A. Corberán, R. Mansini, I. Plana, and J.M. Sanchis. The hierarchical mixed rural postman problem. *Transportation Science*, 51(2):755–770, 2017.

[13] M. Colombi, A. Corberán, R. Mansini, I. Plana, and J.M. Sanchis. The hierarchical mixed rural postman problem: Polyhedral analysis and a branch-and-cut algorithm. *European Journal of Operational Research*, 257(1):1–12, 2017.

[14] Á. Corberán, R. Eglese, G. Hasle, I. Plana, and J. M. Sanchis. Arc routing problems: A review of the past, present, and future. *Networks*, 77(1):88–115, 2021.

[15] P. Damodaran, M. Krishnamurthi, and K. Srihari. Lower bounds for hierarchical chinese postman problem. *International Journal of Industrial Engineering: Theory, Applications and Practice*, 15(1):36–44, 2008.

[16] M. Dror, H. Stern, and P. Trudeau. Postman tour on a graph with precedence relation on arcs. *Networks*, 17(3):283–1294, 1987.

[17] J. Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of Research of the National Bureau of Standards B*, 69:125–130, 1965.

[18] J. Edmonds and E.L. Johnson. Matching, Euler tours and the Chinese postman. *Mathematical Programming*, 5(1):88–124, 1973.

[19] H.A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems, part i: The Chinese postman problem. *Operations Research*, 43(2):231–242, 1995.

[20] É. Gélinas. *Le probleme du postier chinois avec contraintes générales de préséance. M. Sc. A*. PhD thesis, Dissertation, École Polytechnique de Montréal, Canada, 1992.

[21] G. Ghiani and G. Improta. An algorithm for the hierarchical chinese postman problem. *Operations Research Letters*, 26(1):27–32, 2000.

[22] M.G. Guan. Graphic programming using odd or even points. *Chinese Mathematics*, 1:273–277, 1962.

[23] M.H. Ha, N. Bostel, A. Langevin, and L.M. Rousseau. Solving the close-enough arc routing problem. *Networks*, 63(1):107–118, 2014.

[24] E. Haslam and J.R. Wright. Application of routing technologies to rural snow and ice control. *Transportation Research Record*, 1304, 1991.

[25] P. Korteweg and T. Volgenant. On the hierarchical chinese postman problem with linear ordered classes. *European Journal of Operational Research*, 169(1):41–52, 2006.

[26] P.F. Lemieux and L. Gampagna. The snow ploughing problem solved by a graph theory algorithm. *Civil Engineering Systems*, 1(6):337–341, 1984.

[27] L.C. Lu, M.H. Ha, A. Langevin, and M. Gendreau. Optimizing road network daily maintenance operations with stochastic service and travel times. *Transportation Research Part E: Logistics and Transportation Review*, 64:88–102, 2014.

[28] U. Manber and S. Israni. Pierce point minimization and optimal torch path determination in flame cutting. *Journal of Manufacturing Systems*, 3(1):81–89, 1984.

[29] N. Perrier, A. Langevin A.Ãmaya, and G.Cormier. Improving snow removal operations using operations research: A case study. In *Proceedings of International Conference on Information Systems, Logistics and Supply Chain*. INSA de LYON Lyon, France, 2006.

[30] N. Perrier, A. Langevin, and C.A. Amaya. Vehicle routing for urban snow plowing operations. *Transportation Science*, 42(1):44–56, 2008.

[31] N. Perrier, A. Langevin, and J.F. Campbell. A survey of models and algorithms for winter road maintenance. part I: system design for spreading and plowing. *Computers & Operations Research*, 33(1):209–238, 2006a.

[32] N. Perrier, A. Langevin, and J.F. Campbell. A survey of models and algorithms for winter road maintenance. part II: system design for snow disposal. *Computers & Operations Research*, 33(1):239–262, 2006b.

[33] N. Perrier, A. Langevin, and J.F. Campbell. A survey of models and algorithms for winter road maintenance. part III: Vehicle routing and depot location for spreading. *Computers & Operations Research*, 34(1):211–257, 2007a.

[34] N. Perrier, A. Langevin, and J.F. Campbell. A survey of models and algorithms for winter road maintenance. part IV: Vehicle routing and fleet sizing for plowing and snow disposal. *Computers & Operations Research*, 34(1):258–294, 2007b.

[35] O. Quirion-Blais, A. Langevin, F. Lehuédé, O. Péton, and M. Trépanier. Solving the large-scale min-max K-rural postman problem for snow plowing. *Networks*, 70, 08 2017.

[36] O. Quirion-Blais, M. Trépanier, and A. Langevin. A case study of snow plow routing using an adaptive large hood search metaheuristic. *Transportation Letters*, 7:201–209, 09 2015.

[37] U.B. Sayata and N.P. Desai. An algorithm for hierarchical chinese postman problem using minimum spanning tree approach based on kruskal's algorithm. In *2015 IEEE International Advance Computing Conference (IACC)*, pages 222–227. IEEE, 2015.

[38] J.L. Sullivan, J. Dowds, D.C. Novak, D.M. Scott, and C. Ragsdale. Development and application of an iterative heuristic for roadway snow and ice control. *Transportation Research Part A: Policy and Practice*, 127:18–31, 2019.

[39] Jin-Yuan Wang and Jeff R Wright. Interactive design of service routes. *Journal of Transportation Engineering*, 120(6):897–913, 1994.