CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Solving the Large-Scale Min-Max K-Rural Postman Problem for Snow Plowing

Olivier Quirion-Blais
André Langevin
Fabien Lehuédé
Olivier Péton
Martin Trépanier

October 2016

CIRRELT-2016-56

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Solving the Large-Scale Min-Max K-Rural Postman Problem for Snow Plowing

Olivier Quirion-Blais[1,*], André Langevin[1], Fabien Lehuédé[2], Olivier Péton[2], Martin Trépanier[1]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Mathematics and Industrial Engineering, Polytechnique Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

[2] Equipe Systèmes Logistiques et de Production, IRCCyN - Ecole des Mines de Nantes, La Chantrerie - BP20722, 44307 Nantes, France

Abstract. This paper studies the snow plow routing problem, which is a modified version of the min-max problem with k-vehicles for arc routing on a mixed graph with hierarchy. Each arc or edge is given a priority and instead of minimizing the overall finishing time, we minimize the latest finishing time for each priority class. We consider turn restrictions, route balancing, and variable vehicle speeds in a real large-scale network. To solve the problem, we present a graph transformation from a directed rural postman problem with turn penalties (DRPP-TP) to an asymmetric traveling salesman problem (ATSP). We then make the following modifications to the adaptive large-neighborhood search (ALNS) metaheuristics to better handle the constraints: development of new neighborhood operators, several applications of the same destruction operators before repair of the solution, and a dynamic arcgrouping procedure when arcs are removed or inserted. We tested our methodology on three real networks with from 1626 to 2146 street segments and 613 to 723 intersections. The results show that our approach can improve the solution, and the grouping procedure is helpful. The results also show that some operators perform better than others; the network topology seems to explain these variations.

Keywords: Winter maintenance, adaptive large neighborhood search (ALNS), rural postman problem (RPP), large-scale real-life applications, arc grouping, network topology.

Acknowledgements. This work was supported by the Natural Sciences and Engineering Research Council of Canada, the Fonds de recherche Nature et technologies du Québec (FRQNT) and Transports, Mobilité durable et Électrification des transports Québec. This support is gratefully acknowledged.

---

* Corresponding author: Olivier.Quirion-Blais@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2016

1 Introduction

Northern countries experience significant snowfall during the winter season, and considerable effort is needed to keep roads passable. For example, the Ministère des Transports du Québec spent more than CAD 260M to maintain 30,552 km of roadways during the winter of 2011–2012 [31]. For the city of Montreal in Quebec, Canada, the costs of winter maintenance were evaluated at CAD 164.1M for 11,560 km of roadway in 2014 [33]. In this paper we investigate the optimization of real instances of the snow plow routing problem (SPRP). In practice, this problem is often handled by sending vehicles to service predefined sectors of the city. Usually the drivers choose the order in which to clear the streets. Both the definition of the sectors and the drivers' choices may be far from optimal.

The objective of the problem is to complete the operations as soon as possible given the limited number of vehicles available. Therefore, it is related mathematically to the min-max K-rural postman problem (MM K-RPP). Some streets must be serviced before others. To ensure this, the objective should consider the finishing time of each priority class. We assign weights to the various ending times according to the local authority's requirements. Moreover, we consider a mixed network since most of the streets must be serviced in both directions, whereas a few back alleys are serviced in just one direction. Turn restrictions must be considered since when a snow plow crosses an intersection or turns left, it must be careful not to leave a windrow across the intersection. For this reason and to reduce risk, some U-turns are prohibited. However, it is considered good practice to finish plowing one street before starting another. Therefore, the penalties can be reduced or removed when the vehicles travel straight ahead to stay on a street with the same name. Finally, we consider heterogeneous vehicles because some are better

suited to certain street types. The speed of the vehicles depends on the vehicle type and the street type. To the best of our knowledge, this version of the problem has never been studied except by Quirion-Blais et al. [24]. We will present a graph transformation and an improved algorithm using grouping mechanisms that better handles the k-vehicle min-max objective and the constraints. In summary, the constraints that we consider are: route balancing, prohibited turns and turn penalties, street hierarchy, and heterogeneous vehicle speeds. We assess the efficiency of the different neighborhood exchange operators using three case studies based on different real road networks.

The rest of the article is organized as follows. Section 2 reviews related work, and Section 3 introduces a mathematical formulation for the SPRP. Section 4 discusses the data processing and Section 5 presents the metaheuristic. Section 6 presents the results, and Section 7 provides concluding remarks.

2 Related Work

We first present problems that are directly related to snow plowing and then discuss other relevant problems. Most of these problems have been reviewed recently by Corberán et al. [11] and Benavent et al. [3]. We do not discuss the problem of salt spreading because it is generally considered on an undirected graph where both sides of the street can be serviced at once, so it differs from the SPRP on a mixed graph.

We use the following definitions: an edge is a link between two nodes in a graph, and it can be traversed in either direction. An arc is a directed street segment with a starting and an ending node. Deadheading occurs when a snow plow travels along a street without servicing it.

2.1 The Snow Plow Routing Problem

Perrier et al. [21] and Campbell et al. [9] have provided thorough literature reviews of the SPRP. The problem has attracted increasing attention in recent years, but there are only a few mathematical formulations. The formulations vary substantially, depending on the constraints that are considered. The following articles are the most relevant to this work. Golbaharan [14] considers service of the required edges, a restricted number of vehicles, and time windows. Razmara [25] considers heterogeneous vehicles and periodic coverage of the network. Perrier et al. [20] consider hierarchical classes with upgrading possibilities, street–vehicle dependency, and different deadhead and service speeds. Finally, Salazar-Aguilar et al. [28] use mixed integer programming (MIP) for the case where several vehicles must be synchronized to service multiple lanes on a given road.

2.2 Problems Related to the Objective and the Constraints Considered

The objective of the SPRP is to minimize the total duration of the operations, from the moment when the vehicles leave the depot to the latest finishing time of each priority class. Moreover, the routes should be balanced. There are two related problems in the literature. The priorities, also known as hierarchies, are often handled via a lexicographic objective where various weights are given to the finishing time of each priority class according to the importance defined by the clients. In our study, these weights will be determined by the local authorities.

The other related problem is the k-vehicle min-max problem where the objective is to minimize the longest route among a set of k vehicles. Benavent et al. [2] develop an integer linear programming (ILP) formulation and implement a branch-and-cut algorithm to

solve the windy version of the problem. Benavent et al. [4] present a metaheuristic that can handle instances with up to 50 nodes and 184 edges for 5 vehicles. A tabu search algorithm for the routing of security guards was developed by Willemse and Joubert [34]. They apply their methodology to a real instance with 68 nodes and 126 arcs (74 required) and to benchmark instances with up to 140 vertices and 190 required edges.

Forbidden turns and turn penalties must be considered because turning left, doing a U-turn, or even continuing straight ahead may leave a snow windrow in the intersection. Early attempts to consider turn penalties used heuristic methods [6, 27]. In 1999, the directed rural postman problem with turn penalties (DRPP-TP) was explicitly defined by Benavent and Soler [5]. It was also studied by Corbéran et al. [10] in the context of mixed graphs. In 2004 Bautista and Pereira [1] showed that when the restricted turns are numerous, they should be considered during the construction phase. Later, Soler et al. [30] and Bräysy et al. [7] studied the mixed general routing problem with turn penalties (MGRP-TP). Lacomme et al. [16, 17] adapted Dijkstra's algorithm to compute the shortest paths between all pairs of edges. They used the resulting distance matrix with a memetic algorithm to solve the capacitated arc routing problem with turn penalties (CARP-TP).

Some streets must be serviced before others. The heuristics developed by Perrier et al. [20] tackle this constraint with the possibility of *promotion*. Other researchers handle the constraint within the Chinese postman problem (CPP) [13, 8, 29, 15].

For the DRPP-TP, a graph transformation described in [5] allows us to change the problem to an asymmetric traveling salesman problem (ATSP). Using the procedure by Noon and Bean [19], we can also transform the mixed rural postman problem with turn penalties (MRPP-TP) into an ATSP. Laporte [18] states that this last transformation can

introduce considerable degeneracy. However, no major impacts on heuristic methodologies have been reported in the literature.

This article focuses on large instances, for which various heuristics and metaheuristics have been developed [23]. Among these, the ALNS performs a local search among different neighborhoods to handle more complex problems [26, 22]. This metaheuristic seems to be particularly successful for large problems with various real-world constraints.

The SPRP on real road networks combines various problems studied in the literature. The goal of this article is to develop new approaches within a unified network to deal with all the constraints in the context of large real-world networks.

3 Mathematical Formulation

We now present a MIP formulation for the snow plow problem. The problem is defined on a mixed graph $G = (V, A)$ where V is the set of vertices and A is the set of directed arcs including the set of edges. We introduce a set of artificial arcs $A(end) \notin A$ to allow the vehicles to exit the network. Similarly, $a_0 \notin A$ is an arc connected to the depot that allows the vehicles to enter the network. The deadheading time for the arcs in $A(end)$ and for a_0 is set to 0.

We also require the following subsets of A. $A^+(S)$ is the set of arcs leaving the set of arcs S, excluding the artificial arcs from $A(end)$. $A^-(S)$ is the set of arcs entering the set of arcs S excluding a_0. For simplicity we write $A^+(i)$ $(A^-(i))$ instead of $A^+(\{i\})$ $(A^-(\{i\}))$ for the set of arcs leaving (entering) a set including only arc i. A_r^p is the subset of A that contains the required arcs (edges) of priority p. The arcs and edges of A_r^1 must be serviced first, followed by the arcs of A_r^2, and so on until all the priorities in the set P of priorities are covered. An artificial priority p_0 is introduced to define the initial time.

A binary variable $x_{ij}^{kp} = 1$ if arc i is serviced immediately prior to the traversal or service of j in the route of vehicle $k \in K$ during priority class $p \in P$. If more than one passage is required on a given street, then we introdude an arc for each passage required. y_{ij}^{kp} is an integer variable representing the number of deadheading passages on arc i immediately prior to the traversal or service of arc j by vehicle k during priority p. z^p represents the latest finishing time of all vehicles in priority class p. In the objective function the finishing times are weighted by M^p to assign different levels of importance to the various priority classes. We introduce a set of variables t^{kp} to represent the finishing time of priority class p for vehicle k.

Finally, s_{ij}^{kp} and d_{ij}^{kp} are respectively the time required to service and to deadhead arc i immediately prior to arc j by vehicle k during priority class p. These times depend on arc j because the turning penalty is included in the time.

$$\text{Minimize} \quad \sum_{p \in P} (z^p M^p) \tag{1}$$

$$\text{s.t.} \quad z^p \geq t^{kp} \qquad k \in K, p \in P \tag{2}$$

$$t^{kp} - t^{k,p-1} = \sum_{i \in A} \sum_{j \in A^+(i)} (x_{ij}^{kp} s_{ij}^{kp} + y_{ij}^{kp} d_{ij}^{kp}) \qquad k \in K, p \in P \tag{3}$$

$$t^{kp_0} = 0 \qquad k \in K \tag{4}$$

$$\sum_{p^\star \in \{1,\ldots,p\}} \sum_{i \in A^-(j)} (x_{ij}^{kp^\star} + y_{ij}^{kp^\star}) \geq$$

$$\sum_{p^\star \in \{1,...,p\}} \sum_{m \in A^+(j)} (x_{jm}^{kp^\star} + y_{jm}^{kp^\star}) \qquad p \in \{1, ..., |P-1|\},$$

$$j \in A \cup E \cup a_0 \cup A^+(end) \cup E^-(end), k \in K$$

(5)

$$\sum_{p \in P} \sum_{i \in A^+(j)} (x_{ij}^{kp} + y_{ij}^{kp}) =$$

$$\sum_{p \in P} \sum_{mk \in A^+(j)} (x_{jm}^{kp} + y_{jm}^{kp}) \qquad j \in A \cup a_0 \cup A^-(end), k \in K$$

(6)

$$\sum_{k \in K} \sum_{j \in A^+(i)} x_{ij}^{kp} = 1 \qquad i \in A_r^p, p \in P$$

(7)

$$\sum_{p \in P} \sum_{j \in A^+(a_0)} x_{a_0,j}^{kp} = 1 \qquad k \in K$$

(8)

$$\sum_{i \in A} \sum_{j \in A^-(end)} x_{ij}^{kp} = 1 \qquad k \in K, p \in P$$

(9)

$$\sum_{i \in A^-(S)} \sum_{j \in A^+(i)} \left(x_{ij}^{kp} + y_{ij}^{kp}\right) \geq$$

$$\sum_{m \in S \cap A_r \cup A^-(S)} \sum_{l \in A^-(i)} x_{ml}^{kp} \qquad k \in K, p \in P, \forall S \subset A$$

(10)

$$x_{ij}^{kp} = \{0, 1\} \qquad i \in A \cup a_0 \cup A(end), k \in K, p \in P$$

(11)

$$y_{ij}^{kp} \geq 0, integer \qquad i \in A \cup a_0 \cup A(end), k \in K, p \in P$$

(12)

$$t^{kp} \geq 0 \qquad k \in K, p \in P \tag{13}$$

$$z^{p} \geq 0 \qquad p \in P \tag{14}$$

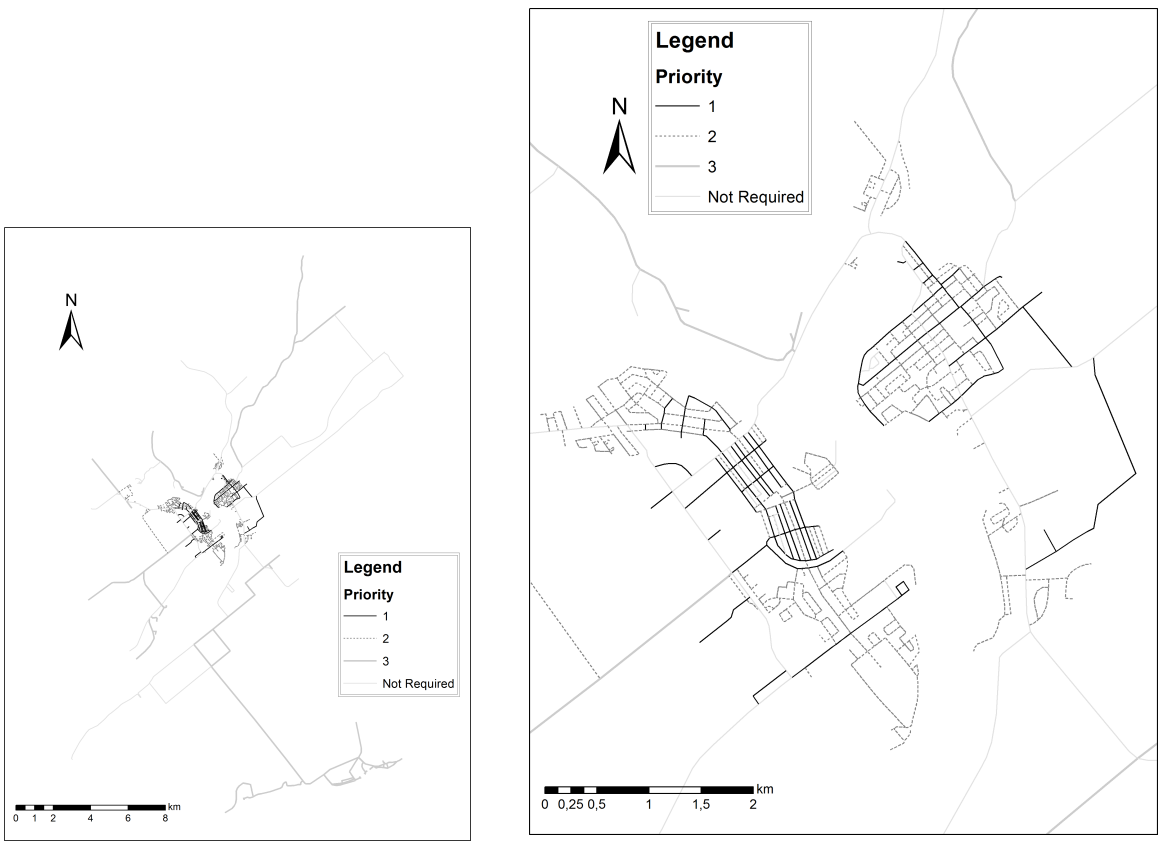
The objective function (1) together with inequalities (2) minimizes the weighted sum of the latest finishing times of all vehicles for all priority classes p. Doing so also allows to balance the workload among the vehicles since we need to transfer as many arcs as possible from the latest finishing vehicle to the others. Constraints (3) ensure that the time spent in priority class p is equal to the time difference between the finishing time of class p and the finishing time of class $p-1$. Constraints (4) set the initial time to 0. The flow conservation constraints (5) and (6) ensure that each vehicle entering arc j is coming from a connected arc i and leaving through another connected arc m. Constraints (5) and (6) ensure that the vehicles do not return to a higher priority class once they have started a lower class. Specifically, constraints (5) ensure that, for a given priority class p that is not the lowest, the total number of times a vehicle enters an arc at class p or higher must be greater than or equal to the number of times it leaves for the same class. Constraints (6) ensure that for all the priority classes, the number of times a vehicle enters an arc is equal to the number of times it leaves. Constraints (7) ensure that all the required arcs are serviced during the corresponding priority class. Constraints (8) and (9) ensure that each vehicle starts at the depot and exits the network when its route is completed. Constraints (10) ensure the connectivity of each tour. Finally, (11), (12), (13), and (14) ensure that x and y are binary and that z and t are non-negative.

4 Graph Transformation

This section details the steps that transform the data from geographical information system (GIS) format to a network.

4.1 Data Processing and Graph Transformation

The primary data are obtained from free-access GIS repositories in the form of an undirected graph, as shown in Figure 1. Some secondary data are obtained from the local authority. These data, including the attributes and projected coordinates, are then fetched directly from the GIS file for further processing.



(a) The rural part has long stretches of road.

(b) The urban part has a grid pattern.

Figure 1: Q1: The network has different topologies.

The graph is transformed from a DRPP-TP to an ATSP based on a transformation

described in [5]. This transformation was chosen because it can prohibit some turns instead of penalizing them, and it can handle both arcs and edges. Moreover, the shortest paths given the turn penalties can be computed in the transformed graph. Therefore, during the improvement phase the algorithm does not have to order the traversed arcs between the serviced arcs.

The transformation has two steps. The first step splits the nodes and adds arcs for every possible turn. The weights of the added arcs correspond to the turn penalties. The types of turns are determined by the angle of the arc leaving the node, and the values used for the penalties are proportional to those in [12] for block design, as shown in Table 1.

Situation	Leaving Angle	Penalty
Right-turns	$20° < \theta < 160°$	0
Left-turns	$200° < \theta < 340°$	2
U-turns	$-20° < \theta < 20°$	12
Straight ahead, street changes	$160° < \theta < 200°$	1
Straight ahead, street crossings	$160° < \theta < 200°$	0

Table 1: Left turns and U-turns receive a higher penalty

The second step transforms the required arcs and edges to nodes to obtain an ATSP, as shown in Figure 2. This is carried out as follows:

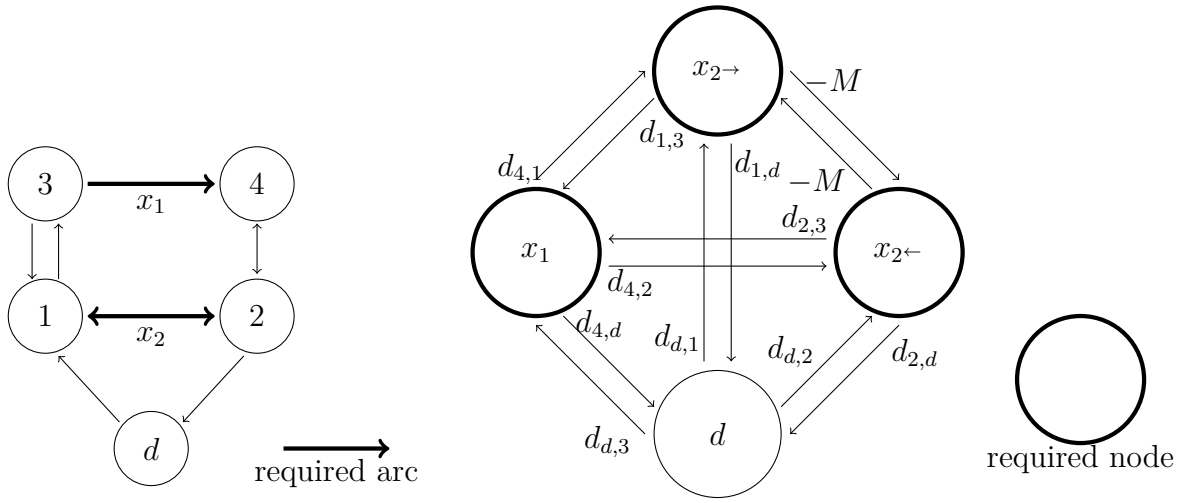
1. Each required arc is transformed into a node-arc, and each required edge is transformed into a node-edge pair, one for each direction. A node is added to represent the depot. The depot d, the arc x_1 and the edge x_2 in Figure 2 (a) are respectively transformed into the nodes $d, x_1, x_{2\rightarrow}$ and $x_{2\leftarrow}$ of Figure 2 (b). "$\rightarrow$" and "$\leftarrow$" represent the two directions of the edge x_2.

2. Arcs with the dual weights 0 and $-M$, where M is a large number, are inserted into the graph between the node-edge corresponding to opposite directions of the same edge. The weight $-M$ is used when designing individual routes, and the weight 0

is used in the evaluation of the objective function.

3. A vehicle leaving a node-edge should be considered to leave from the opposite edge. Therefore, arcs are added from all the node-edges to all the other node-arcs/node-edges, and the distance used as a weight is calculated from the opposite node-edge.

4. Other arcs are added to fully connect the graph. The weights of these arcs correspond to the length of the shortest path going from the end of the arc to the end of the arc/edge in the expanded graph.



(a) Expanded graph (b) ATSP graph

Figure 2: Arcs are transformed into a single node, and edges are transformed into pairs of nodes.

We computed the distances between the arcs and edges using Dijkstra's algorithm. We kept traces so that we could rebuild the tours. Indeed, the solution of the ATSP in the transformed graph gives the order of passage of the required arcs or edges in the original graph. For example, if the solution to the ATSP in Figure 2b) is $d \rightarrow x_{2\rightarrow} \rightarrow x_{2\leftarrow} \rightarrow x_1$, then in the original graph the vehicle should travel from the depot, service arc $x_{2\rightarrow}$ from left to right, and then service x_1. The second arc $x_{2\leftarrow}$ is ignored since it is the counterpart of the edge. The shortest path calculated via Dijkstra's algorithm is used

between the required arcs. Therefore, the complete solution in the original graph is $d \to 1 \to 2 \to 1 \to 2 \to 4$.

5 Improved Adaptive Large-Neighborhood Search

This section describes our algorithm for the SPRP. Figure 3 outlines the general scheme followed by the ALNS, which is performed once for each priority class. It builds an initial solution using three simple construction heuristics and then improves it to obtain a good partial solution for each class. At the end, the priorities are merged to create a complete route for each vehicle. A further improvement step is then performed on the global solution to take into account the relative importance given by the weights of each class.

The improvement steps are based on the ALNS metaheuristic introduced by Ropke and Pisinger [26] where different neighborhoods, called operators, are applied iteratively. A dynamic adaptation of the weights given to each operator increases the chances of selecting operators that perform well. This feature is particularly well suited for situations with varying network topologies.

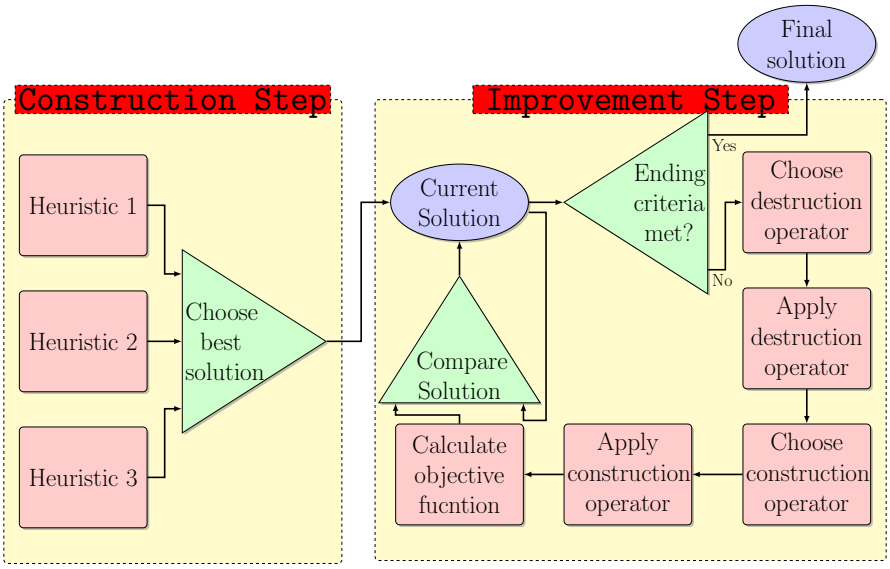


Figure 3: ALNS: An initial solution is built and improved.

5.1 Initial Solution

The algorithm starts by building a feasible solution that is characterized by a set of routes, one per vehicle, with strictly enforced hierarchy constraints. The multi-start (MS) scheme used builds three solutions using simple heuristics and keeps the best one for further processing. All the heuristics use the required nodes and the starting nodes as input. Since an initial solution is found for each priority class, the required nodes correspond to the nodes of the current class. The starting node is the depot for the first class and the last node of the previous class subsequently. The distance between the last node visited and the depot is assumed to be 0 since the vehicles do not need to return to the depot. We now discuss the three heuristics in detail.

The first heuristic is a route-first, cluster-second approach. The first step uses the Lin-Kernighan heuristic (LKH) to build a giant tour servicing all the nodes. The segmentation of this tour is done using a split methodology [32] with a makespan objective. A version of the split procedure minimizing the makespan subject to a limited fleet provides an upper bound [16]. With this bound, the problem can readily be solved with a commercial optimizer using the formulation in Appendix A.

The second heuristic is a cluster-first, route-second approach. We select a number of cluster seeds corresponding to the number of vehicles, and they should be evenly distributed on the network. We chose the seeds based on input from an expert in winter maintenance. We then assign the arcs to their nearest seed. We apply a repair function to ensure that each pair of nodes corresponding to an edge is assigned to the same cluster. Finally, we apply the LKH algorithm to each cluster to obtain a route.

The third heuristic is based on the fact that LKH can build good routes quickly. It is again used to build a giant tour. The tour is split where the duration of the deadhead ex-

ceeds 3% of the total duration. Then all the short segments are inserted into the vehicles using a standard best insertion.

5.2 Adaptive Large-Neighborhood Search Improvement Phase

The main steps of the improvement phase are outlined in Figure 3. The algorithm cycles through the selection and application of destruction and repair operators until the ending criterion is met. At every iteration the solution is evaluated using the objective function. We now describe the main features of the algorithm.

5.2.1 Acceptance and Ending Criterion

The acceptance of new solutions is based on a simulated annealing pattern [26]. Therefore, improving solutions are always accepted, and inferior solutions can also be accepted since they may help escape local minima.

The probability of accepting a new solution is given by $\alpha = e^{-(f(s')-f(s))/T}$, where s' is the incumbent solution, s is the current solution, and T is the temperature. At each iteration, T is updated by multiplying its value by a cooling rate $c = 0.9999$. The initial temperature is given by $T_{init} = \frac{-\omega * f(s_{init})}{ln(\alpha_{init})}$, where $\alpha_{init} = 0.1$ is the probability of accepting a solution that worsens the initial solution by at most $\omega = 0.05$. The improvement phase is run for 7000 iterations, which provides a good compromise between computational time and solution quality.

5.2.2 Choosing an Operator

The selection mechanism described here applies to both the destruction and construction operators. It is based on a random selection biased by the weights μ_ω, where ω corresponds to the operator. At the beginning of the improvement step, all the μ_ω are equal

to 1. As the algorithm proceeds, a system of points increases the chances of choosing operators that perform well. Every time an operator improves the incumbent solution, improves the best solution, or worsens the solution, it is awarded five points, ten points, or one point respectively. The points are added to the corresponding μ_ω, and the probability of choosing ω is $\frac{\mu_\omega}{\sum \mu_\omega}$. These probabilities are updated every 150 iterations and reinitialized every 2000 iterations.

5.2.3 Destruction Operators

The operators are divided into two groups: the destruction operators remove nodes from the solution and the construction operators repair the solution. Instead of developing stand-alone operators, we describe different attributes that can be combined to create the operators. The attributes are grouped into five classes as shown in Appendix B. Using this procedure, we design 22 destruction operators. We perform tests to select the best.

Number of nodes to be removed

The first class determines a maximum number of nodes to be removed. We select a random number between one and n_{max_ω} and update n_{max_ω} at each iteration ω via $n_{max\_\omega} = n_{max\_init} * (1 - \%progess)^{\phi}$, where n_{max_init} is the initial maximum number of nodes (set to 150) and ϕ is a shortening parameter (set to 0.5).

Number of applications

The second class of attributes concerns the number of times the operator is applied before we repair the solution. We introduced this class of attributes because the min-max objective tends to equalize all the finishing times of the routes. Therefore, it is better to remove a few nodes from each route rather than removing many nodes from a

single route. We select the number of applications via a random mechanism with values between 1 and a maximum number set equal to twice the number of vehicles. After 4000 iterations without improvement, we reduce the maximum number to half the number of vehicles until another improvement is achieved. To determine the number of nodes to be removed per application, we divide the maximum total number of nodes to be removed per iteration obtained at the previous step by the number of applications of the operator.

Selection of the route

We use four choices for this group of attributes: a) the worst cost, b) the longest route, c) the most empty route, or d) a random route. The first two allow us to perform local search, and the other two are more likely to broaden the solution.

Option a) seeks the route containing the node that incurs the highest cost in the objective function. To do this, the nodes are removed iteratively, the objective function is evaluated, and the values obtained are listed in ascending order. We use the following function to determine which value to select from the list: $nb\_of\_insertions * Random^{P_{worst}}$ where $P_{worst} = 0.02$. This prevents the metaheuristic from looping over the same solutions. Option b) finds the route with the longest duration given the turn penalties and the time spent in the lower priority classes, but without considering the weights M^p of the objective function.

Selection of the first node

This group of components selects the first node to be removed from the route. We use four choices: a) worst cost, b) random node, c) no selection, and d) isolated nodes. Option a) seeks the node that incurs the highest cost. Option b) selects a random node. Option c) makes no selection: this option is used specifically in the case of the split methodology for grouping. Option d) tries to find isolated nodes. The idea is to catch remote nodes that

could be transferred to another route. To do this, the distances between the subsequent nodes in the routes are sorted by descending order. Then we inspect the sequence between the first largest distance and the second in the list to determine if the number of nodes is lower than a maximum number of nodes to be removed (fixed by the user). If it is not, we inspect the sequence between the first largest distance and the third in the list. We continue until a sequence respecting the maximum number of nodes is found.

Grouping methodology

This class of attributes is introduced because we are dealing with real cases of arc routing problems. Therefore, we wish to remove groups of arcs. This allows us to combine arc sequences that should be serviced by a single vehicle. It also allows us to retain the order of passage of the current solution. We use six options: a) until objective function decreases, b) minimize the distance between the end and the start of the removed sequence, c) continuous sequence, d) split, e) random, and f) no grouping.

Option a) is based on the observation that removing one node in the middle of a sequence often does not result in an improvement in the objective function. Instead, we select nodes before and after the first selected node until the objective function decreases or the maximum number of nodes to be removed is attained. Figure 4 shows an example of a routing in the original graph; the arcs are represented by nodes in the transformed graph. The first selected node corresponds to arc $n2$. However, removing $n2$ will not result in a decrease in the objective function since the vehicle must traverse $n2$ after $n1$ to reach $n3$. The algorithm then tries to remove $n1$ as well. However, the value of the objective function stays the same since the vehicle has to traverse $n1$ and $n2$ to reach $n3$. Finally, when $n3$ is removed, a new routing is possible through $n4$, which reduces the value of the objective function.

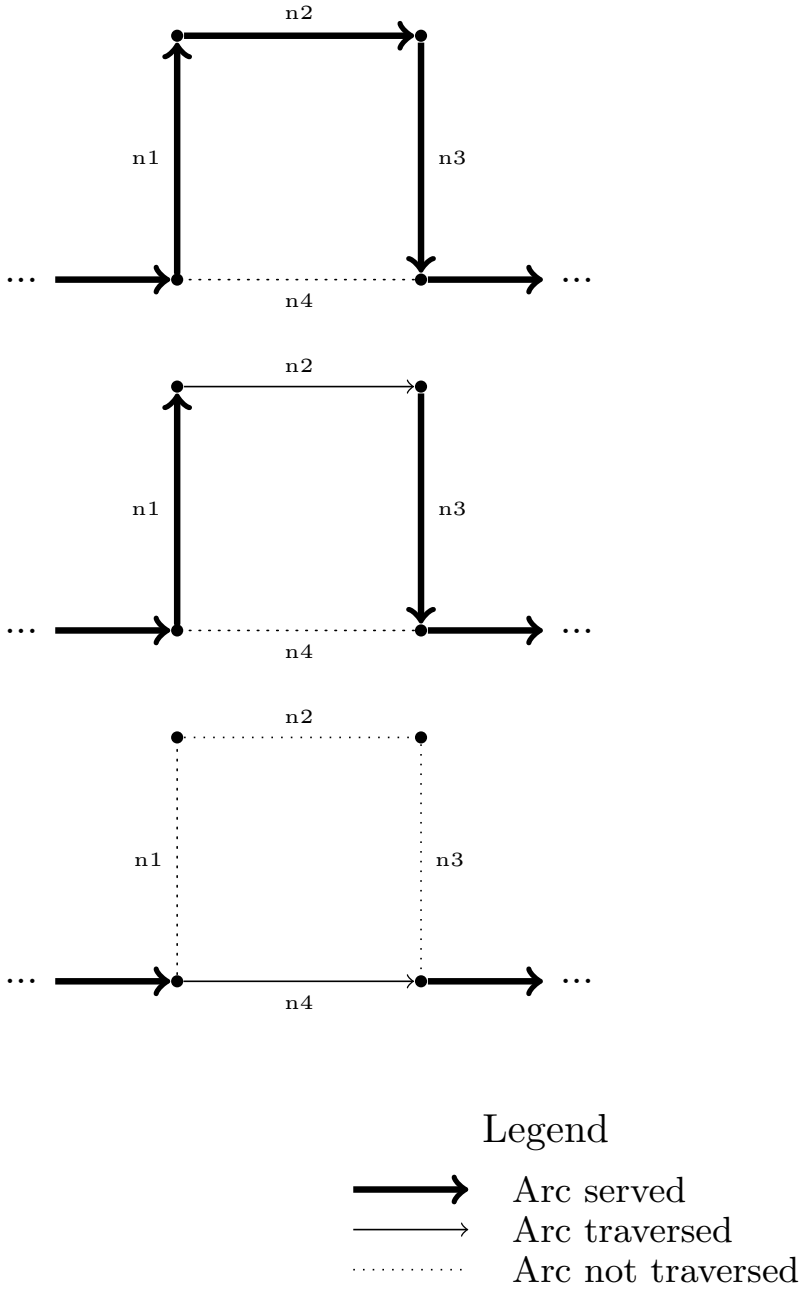


Figure 4: It is often necessary to remove a group of arcs from a given route to improve the solution.

Option b) tries to extract good sequences by minimizing the distance between the end and the start of the removed sequence. To do this, we inspect all the combinations of nodes before and after the first selected node and with respect to the maximum number of nodes. We remove the sequence that provides the minimum distance between the end

and the beginning of the sequence. In Figure 4, if $n2$ is removed, the distance between the end of the sequence (end of $n2$) and its start (start of $n2$) includes a penalty for a U-turn and the time to traverse $n2$. If $n1$ and $n3$ are also removed a smaller distance can be found between the end of the sequence removed (end of $n3$) and its start (start of $n1$) where there is a right turn and the length of $n4$.

Option c) selects a continuous sequence of arcs. Starting from the selected nodes, the algorithm finds the node that is directly after. If the distance between the current node and the added node is 0, then it is added to the group. This is done before and after the initial node until we achieve a distance greater than 0 or the maximum number of nodes to be removed. In Figure 5 the first node selected corresponds to arc $n1$. Then the algorithm examines the following arc $n2$. If there is no turn penalty between the arcs, $n2$ is added to the removed group. If there is a penalty, then $n2$ is rejected.

Figure 5: Two arcs forming a continuous sequence.

Option d) is based on the split process. We calculate the average length of all the routes in the solution. This is used as an upper bound to remove all the longer links from the auxiliary graph. Then the shortest sequence obtained from the split is removed from the tour.

Option e) selects a random number of nodes, and option f) allows no grouping, so only one arc is selected.

In our first version of the ALNS, we built an exhaustive list of operators to test the different features. We describe the features of each operator in Table 6 of Appendix B. When a choice has been made for all the features, the operator is applied and the nodes

are removed. When the nodes are removed, they are grouped by priority class so that each group can be inserted into the same class. We also apply a repair function to ensure that the node pairs corresponding to edges are kept together.

5.2.4 Construction Operators

The construction operators were also designed based on their main components, the grouping methodology and the insertion position, as shown in Table 7 of Appendix C. We obtained four construction operators by combining the components.

Grouping methodology

The first grouping option is to retain the sequences of nodes defined by the destruction operators. The second option is to use the LKH as follows. The nodes are first sorted by priority. Assuming that the distance between the beginning of the route and the first node and that between the end and the last node are 0, we use LKH to build a giant tour visiting all the nodes in the insertion list for each priority class. Then we cut the route at every place where the duration between two nodes is greater than 3% of the total duration of the route. These node sequences are sent to the next step of the construction operator.

Insertion position

The second class of features is the selection of the insertion position. We used two standard insertion heuristics: best insertion and k-regret, where $k = 3$. To prohibit the promotion of the arcs, a group of nodes must be inserted into the priority class to which it belongs.

6 Case Study and Results

We now present three case studies based on real data.

6.1 Case Studies

The first case study (Q1) is a small city in Northern Quebec, Canada. It has been described in [9, 24], and Figure 1 shows a representation of the road network. It is composed of 1609 single street segments, 17 back alley streets, and 613 intersections. The topology of the network varies depending on the area. Rural areas have long street segments with a predominance of three-way intersections. Urban and residential areas have a grid pattern with short street segments, four-way intersections, and a few one-way alleys.

The network is serviced by two graders, three front-end loaders, and two ten-wheel trucks; Table 2 gives their speeds. Initially, each vehicle had a predefined sector and the drivers decided the order in which to plow the streets. Some of the vehicles can either plow or grit, but we do not consider gritting. We assume that all the vehicles should be assigned similar route durations.

Table 2: Vehicle speeds (in km/h)

	Grader	Front-end loader	Ten-wheel truck
Priority 1	20	25	35
Priority 2	20	25	35
Priority 3	25	25	50
Deadhead	25	25	50

The second case study (Q2), illustrated in Figure 6, is a small city in Northeastern Quebec and the third (Q3), illustrated in Figure 7, is a sector of a major city in the province of Quebec. Table 3 gives the total road length for each priority class and the number of road

segments. Q2 has just a few roads of priority 3 and almost no grid pattern. Q3 has no roads of priority 3 and an extensive grid pattern. Table 4 summarizes the topologies of the networks. Two graders, four front-end loaders, and one ten-wheel truck are available for Q2, and four graders and two front-end loaders are available for Q3. The road classes and the number of trucks for Q2 and Q3 were estimated based on the data from Q1.

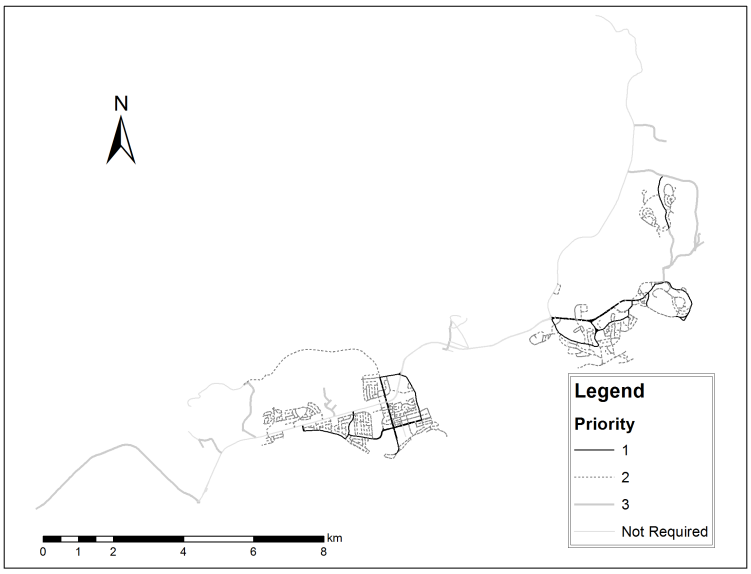


Figure 6: Q2: This network has many three-way intersections.

Table 3: Longer lengths with a smaller number of road segments result in a smaller density

	Length (in km)				Number of road segments				Number of
Case	Prio 1	Prio 2	Prio 3	Total	Prio 1	Prio 2	Prio 3	Total	Intersections
Q1	52.46	130.28	163.91	527.14	357	873	148	1626	613
Q2	55.59	202.60	41.89	360.95	515	1208	72	1971	687
Q3	89.16	93.54	0	254.41	894	844	0	2146	723

6.2 Results

The metaheuristic was coded in Visual Basic .NET (VB.NET). We performed two series of tests for the destruction and construction operators. The first series tested the individual operators, and we used the results obtained to group the operators for the second series

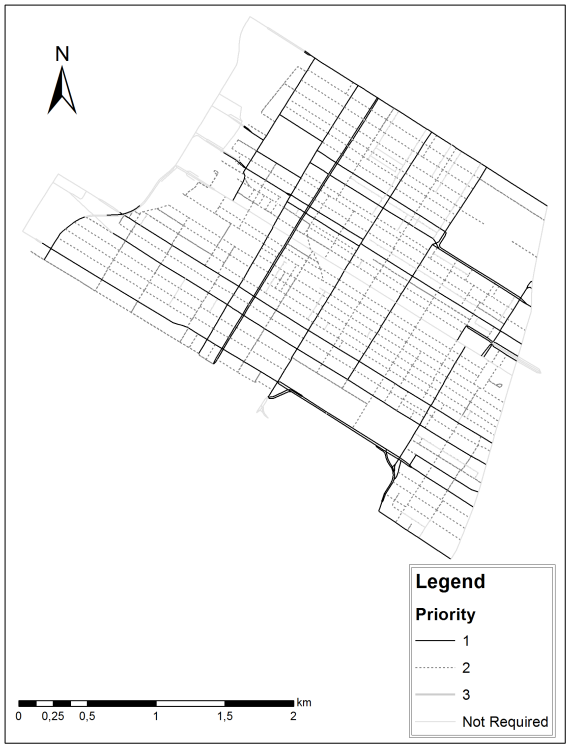


Figure 7: Q3: This network has an urban layout characterized by a compact grid pattern.

Table 4: The networks show great diversity in their topology

Case	Prio 1	Prio 2	Prio 3
Q1	Sparse, close to grid network	High-density network, partial grid pattern	Sparse network, long roads, predominance of 3-way intersections
Q2	Sparse network with predominance of 3-way intersections	High density with predominance of 3-way intersections	Few roads, some long
Q3	Sparse grid network	High-density grid network	Not applicable

of tests. We did not compare our results to any benchmarks since we could not find any with similar constraints and objective. Unless otherwise specified, we set all the parameters to the values given in Section 5. For Q1, the weights of the objective function are: $M_1 = 150, M_2 = 100, M_3 = 100$, and $M_{deadhead} = 1$. These weights were chosen so that the finishing time of each priority class has about the same weight in the objective and the deadhead time has a smaller impact. We limited the total number of iterations to 7000 to obtain good results in a reasonable time. Because there is a random process

in the metaheuristic, we carried out 10 replications for each test.

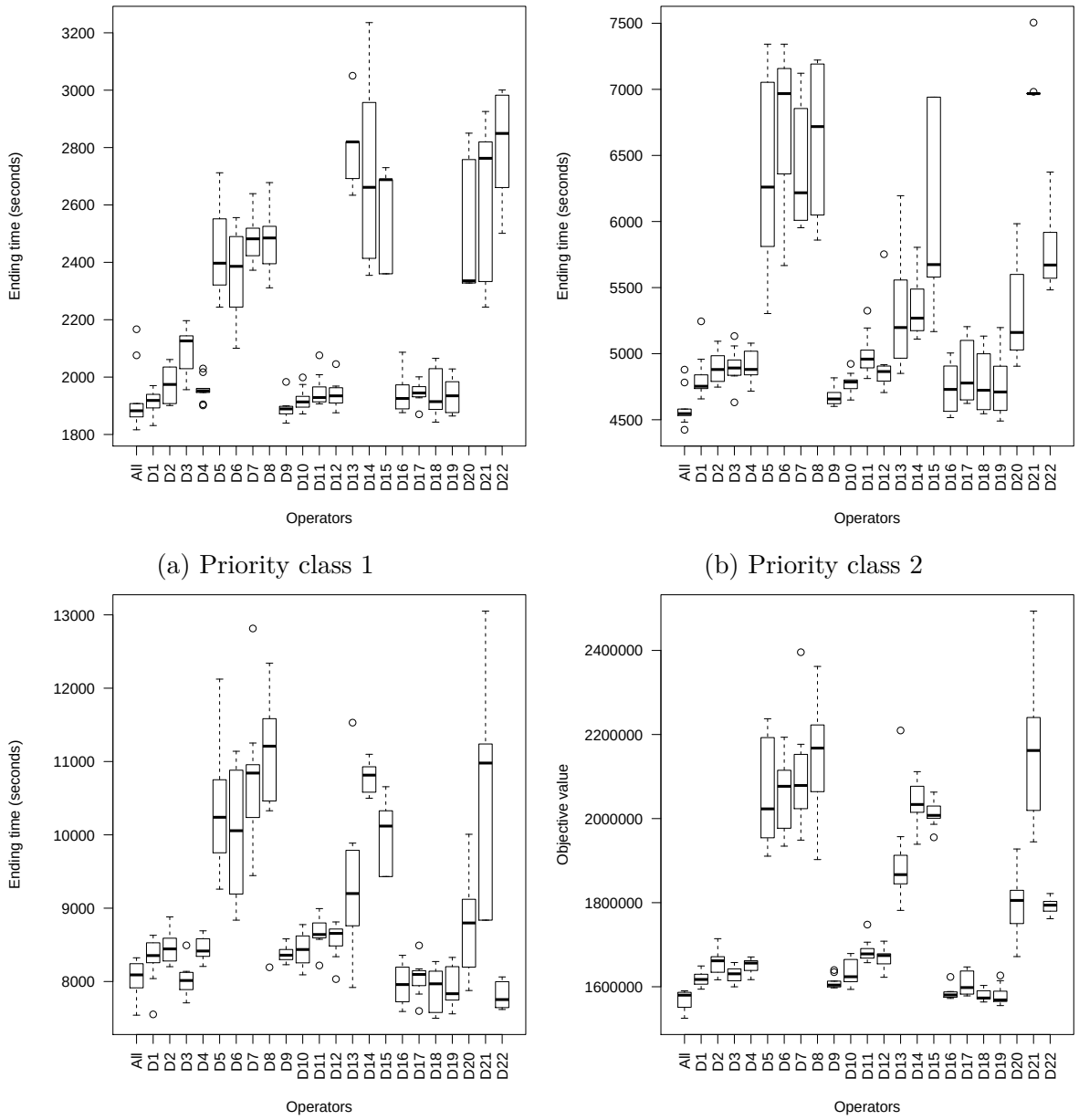
We set the maximum number of nodes to be removed to 130 for Q2 and 170 for Q3, and we set the simulated annealing parameters to $\omega = 0.075$, $\alpha_{init} = 0.1$, and $c = 0.999$ for Q2 and $\omega = 0.075$, $\alpha_{init} = 0.05$, and $c = 0.999$ for Q3. We chose these values based on empirical tests. The length of the streets and the proportion of each priority class is different, so we adjusted the weights to maintain a similar order of importance for each part of the objective function: $M_1 = 220, M_2 = 65, M_3 = 55$, and $M_{deadhead} = 1$ for Q2 and $M_1 = 150, M_2 = 70, M_3 = 70$, and $M_{deadhead} = 1$ for Q3.

6.2.1 Testing the Destruction Operators

In the first series of tests, we assessed the performance of the destruction operators by using them one at a time while using all the construction operators. Figure 8 shows the finishing times for each priority class and the final value of the objective. Some operators seem to perform better for certain classes. For example, operator D22 gives good results for the third class. However, it is clearly not the best for the finishing time of the first class. This could be due to the different topologies of the network for each class. Since the importance of the finishing time of each class is user-dependent, there is no obvious choice for the operators. However, the variable bias in the selection of the operators in the ALNS helps to favor good operators.

Figure 8 also shows that the operators can be grouped based on the selection of the first node and the grouping methodology. The second series of tests was based on these groups. Table 5 shows the grouping of the operators.

We applied the second series of tests to all three case studies. To consider interactions between the operators, we removed a group of operators at a time. Figure 9 shows the



(a) Priority class 1

(b) Priority class 2

(c) Priority class 3

(d) Objective

Figure 8: The finishing times show that some operators perform better on a single priority class.

Table 5: Grouping of operators based on their features

	Random seq	Loop	Decreasing obj	Cont Seq	Isolated Arcs	Split
Random tour	D1	D2	D3	D4	D13	D22
Most empty	D5	D6	D7	D8	D15	D21
Longest	D9	D10	D11	D12	D14	D20
Worst cost	D16	D18	D17	D19	X	X

finishing times and the value of the objective for each group removed.

Removing groups of operators is beneficial in some cases. We think that this is due to

the topology of the network. The ranges between the highest and the lowest medians in the plots indicate that the regular networks (Q3 and class 1 of Q1) provide more stable results and all the operators can be used. In contrast, the ranges are larger when the network has an irregular topology (class 1 of Q2 and the objective of Q1). Therefore, it is important to select the strategy carefully when the network is irregular. Since the adaptive mechanism tends to choose better operators, the ALNS metaheuristic seems to be appropriate for this situation.

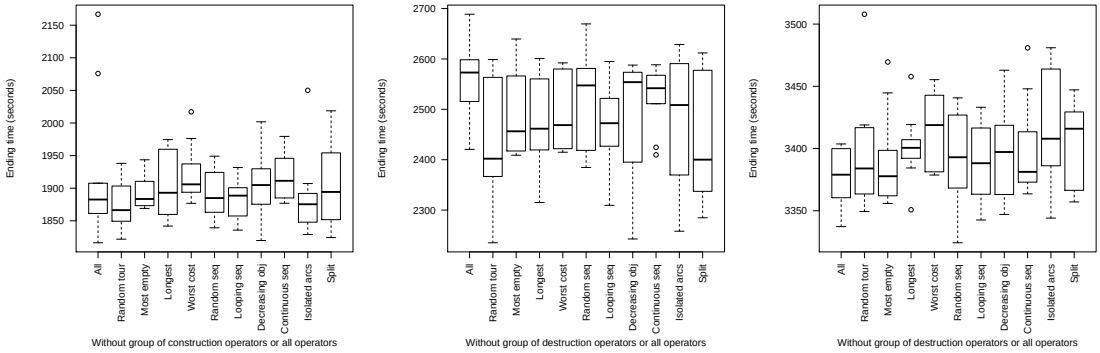
The plots showing the objective value in Figure 9 indicate that some groups of operators improve the solution when they are removed. This suggests that the selection mechanism of the ALNS could be improved to reduce the impact of these operators.

On the same plots, it can be seen that removing more than one group can improve the solution. Therefore, we explored removing combinations of poorly performing groups of operators. We found that removing too many operators does not lead to worthwhile improvements.

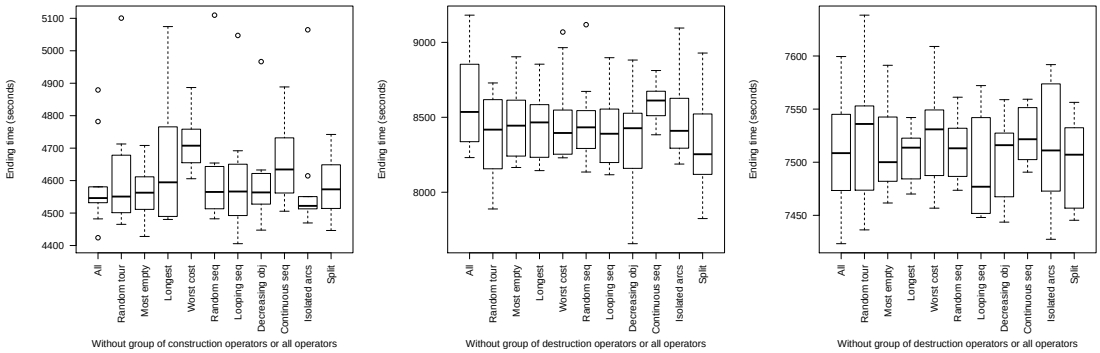
Is the grouping methodology helpful? To explore this, we tried removing just one node. However, not all the groups of operators can be limited to a single node. The relevant groups are: random tour selection (D1, D2, D3, D4, D13, D22), most empty tour (D5, D6, D7, D8, D15, D21), longest tour (D9, D10, D11, D12, D14, D20), and worst cost (D16, D17, D18, D19). Figure 10 gives the results with and without grouping. The grouping methodology generally improves the results.

6.2.2 Testing the Construction Operators

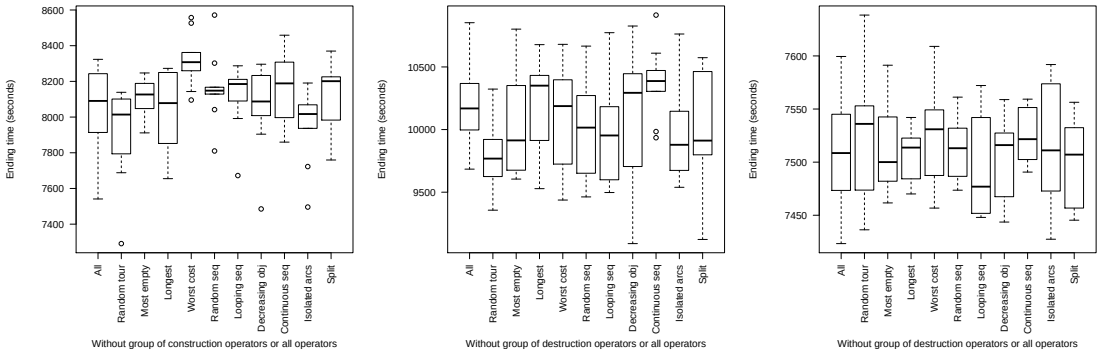
We similarly assessed the performance of the construction operators while using all the destruction operators. Figure 11 shows the application of the construction operators one



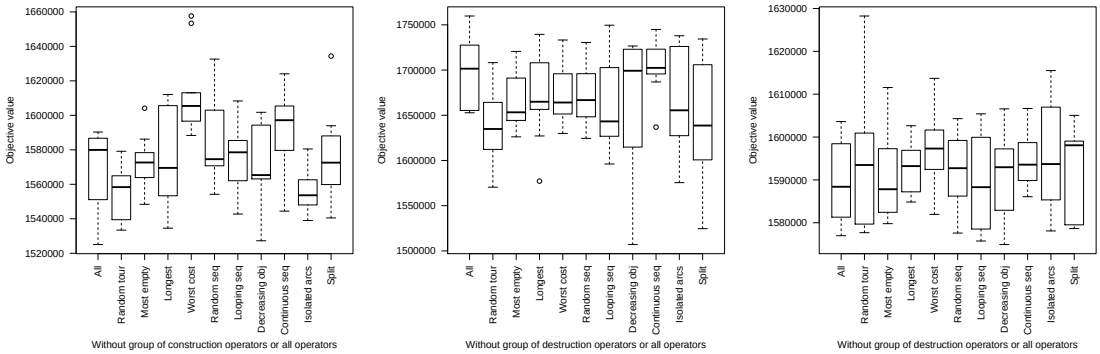
(a) Q1-Priority 1 (b) Q2-Priority 1 (c) Q3-Priority 1



(d) Q1-Priority 2 (e) Q2-Priority 2 (f) Q3-Priority 2

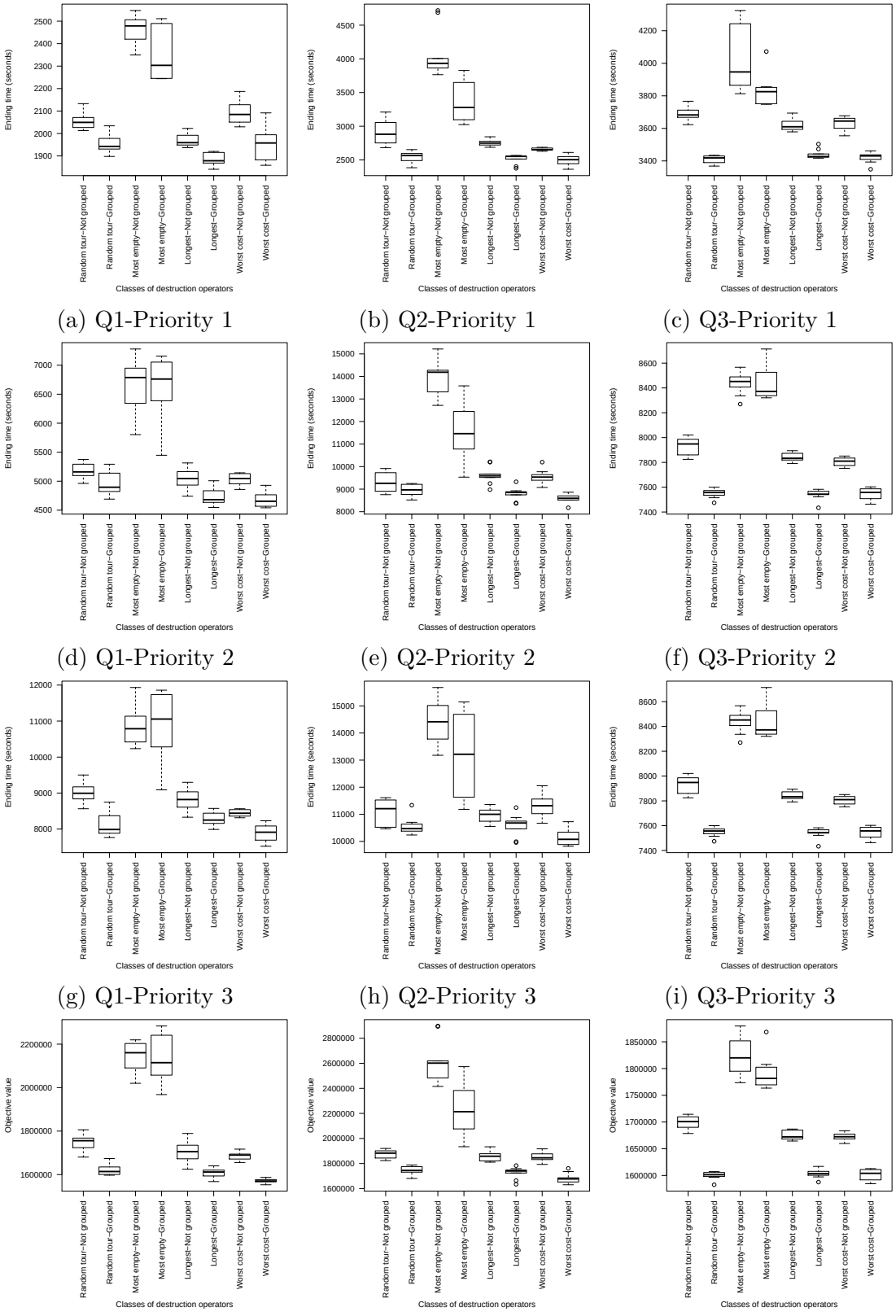


(g) Q1-Priority 3 (h) Q2-Priority 3 (i) Q3-Priority 3



(j) Q1-Objective (k) Q2-Objective (l) Q3-Objective

Figure 9: A large finishing time or objective value indicates that the removal of the group of operators has had a negative impact.



(a) Q1-Priority 1 (b) Q2-Priority 1 (c) Q3-Priority 1

(d) Q1-Priority 2 (e) Q2-Priority 2 (f) Q3-Priority 2

(g) Q1-Priority 3 (h) Q2-Priority 3 (i) Q3-Priority 3

(j) Q1-Objective (k) Q2-Objective (l) Q3-Objective

Figure 10: The grouping methodology improves the solution in all cases.

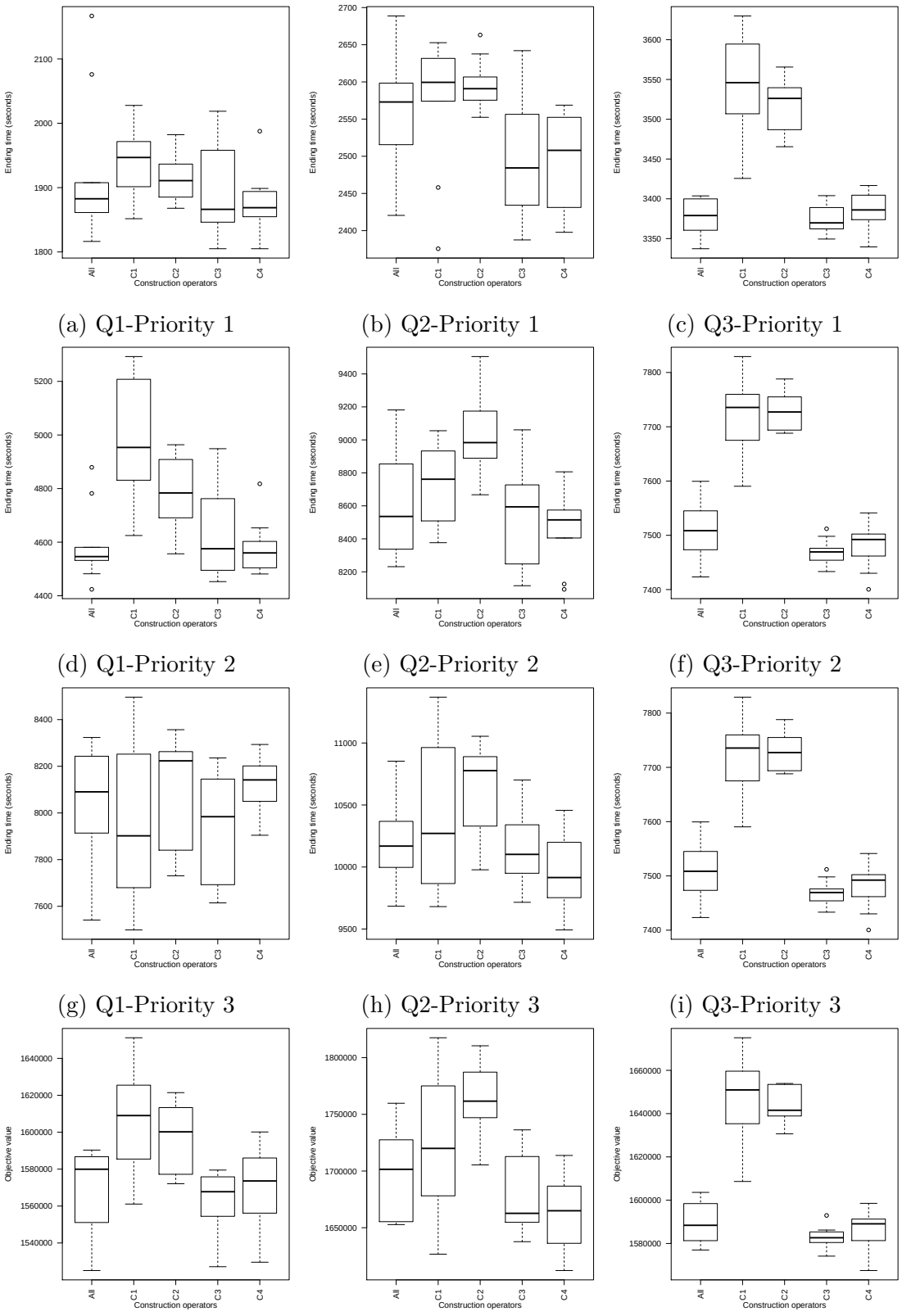
at a time for all three case studies. The grouping of the construction operators is not as obvious as it is for the destruction operators. However, plots (b), (c), (f), (i), and (l) seem to show that C1 can be grouped with C2 and C3 with C4. The first group corresponds to the cases where the sequences removed from the destruction step are kept unchanged, and the second group corresponds to the case where the groups are modified in the construction phase. We performed a series of tests with combinations of construction operators. Two combinations were based on the insertion technique (best insertion or k-regret) and two were based on the grouping methodology of the construction step (keep the groups from the destruction step or create new groups).

For the insertion combinations, Figure 12 shows that there is no significant difference between the strategies when we use both grouping methodologies. However, for Q2, it appears better to choose one or the other instead of using both. The construction combinations seem to give an improvement in all cases, but it is not always significant. It is especially good for Q3. However, the calculation for the insertion is three times longer when the groups are rearranged.

Finally, the transformation of the network allows us to use a powerful heuristic designed for the ATSP. This creates good individual routes for the vehicles. The challenge of the SPRP lies in the fact that several vehicles must be coordinated. The grouping methodologies retain the good sequences produced by LKH instead of transferring single arcs at a time.

7 Conclusion

We have presented a graph transformation from a DRPP-TP to an ATSP to tackle the SPRP. The transformation of the graph allowed us to take into account the turn



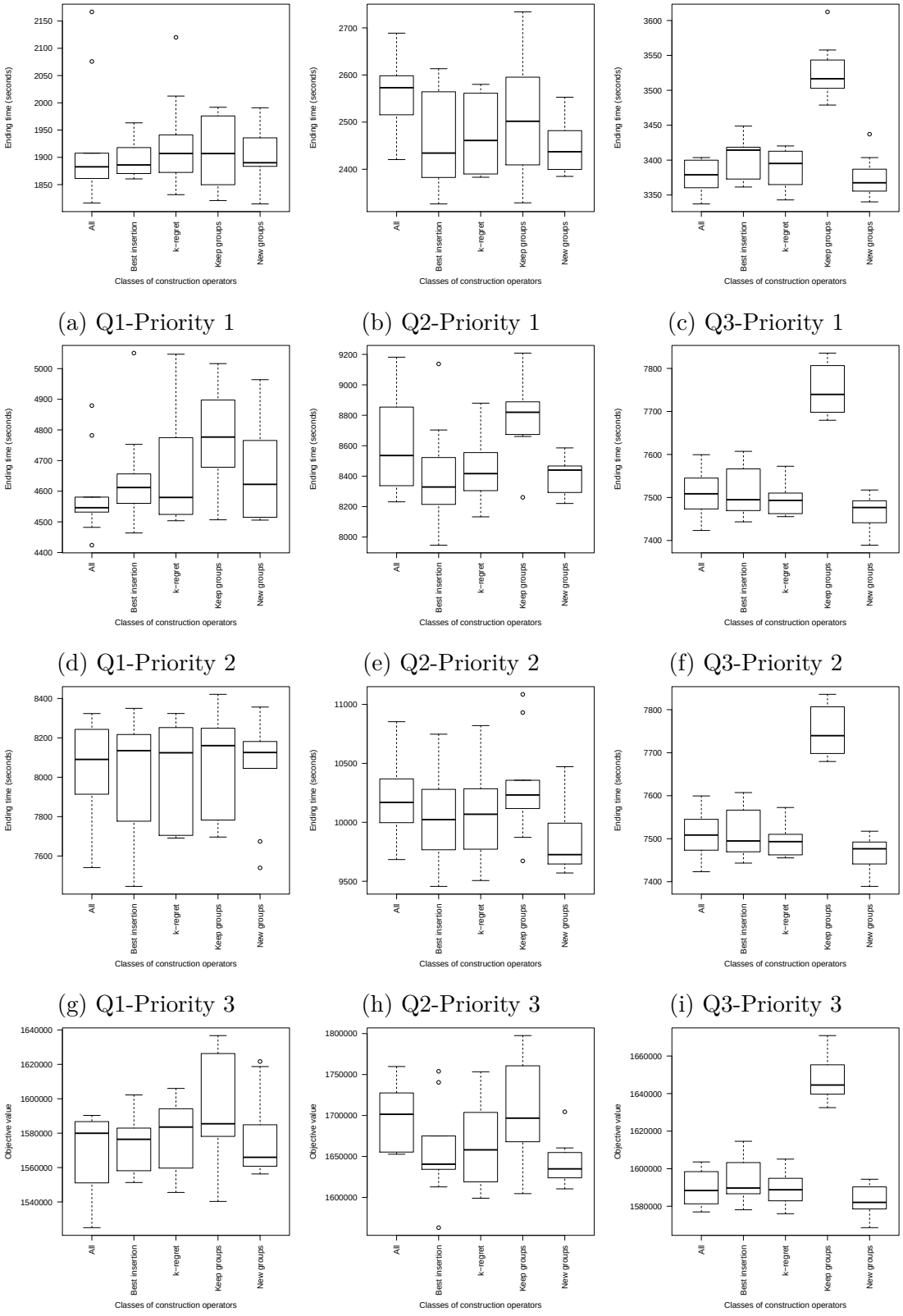
(a) Q1-Priority 1 (b) Q2-Priority 1 (c) Q3-Priority 1

(d) Q1-Priority 2 (e) Q2-Priority 2 (f) Q3-Priority 2

(g) Q1-Priority 3 (h) Q2-Priority 3 (i) Q3-Priority 3

(j) Q1-Objective (k) Q2-Objective (l) Q3-Objective

Figure 11: Operators C3 and C4, which modify the groups from the destruction step, seem to provide earlier finishing times.



(a) Q1-Priority 1 (b) Q2-Priority 1 (c) Q3-Priority 1

(d) Q1-Priority 2 (e) Q2-Priority 2 (f) Q3-Priority 2

(g) Q1-Priority 3 (h) Q2-Priority 3 (i) Q3-Priority 3

(j) Q1-Objective (k) Q2-Objective (l) Q3-Objective

Figure 12: For Q2, the algorithm seems to perform significantly better when selecting a single group.

penalties without slowing down the search phase with back-and-forth movements in the metaheuristic. We use a modified version of the ALNS metaheuristic to solve the problem. We chose the ALNS framework because it takes advantage of various neighborhoods to handle the different topologies that are often found in real networks. We developed new neighborhood operators to better handle the constraints. The grouping methodology is especially promising since it helps to retain good sequences and decreases the number of insertion requests. Finally, we apply a destruction operator several times before we repair the solution. This procedure handles the min-max objectives with several vehicles and the route balancing constraint.

We tested our methodology on three real networks. The tests showed that the metaheuristic was successful. Some operators performed better on some networks; this is probably due to the network topology. However, we cannot yet determine which operators perform better on a given topology. It would be interesting to explore this by applying the metaheuristic to many network topologies. Also, it appears that even within a network the topology can vary. It would be interesting to develop a way to choose the operators based on the local network topology. We used the same weights for the different priority classes. However, in the case presented, the different classes have different topologies. It would therefore be interesting to investigate the performance of the algorithm with modified weights.

Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council of Canada, the Fonds de recherche Nature et technologies du Québec and Transports, Mobilité durable et Électrification des transports Québec. This support is gratefully

acknowledged.

References

[1] J. Bautista and J. Pereira, Ant algorithms for urban waste collection routing, Ant Colony Optimization and Swarm Intelligence, Proceedings 3172 (2004), 302–309.

[2] E. Benavent, Á. Corberán, I. Plana and J. M. Sanchis, Min-max k-vehicles windy rural postman problem, Networks 54.4 (2009), 216–226.

[3] E. Benavent, Á. Corberán, I. Plana and J. M. Sanchis, "Arc routing problems with min-max objectives", Arc Routing: Problems, Methods and Applications, Á. Corberán and G. Laporte (Editors), Society for Applied Mathematics and the Mathematical Optimization Society, Philadelphia, PA, 2014, chap. 11, pp. 255–280, eprint: http://epubs.siam.org/doi/pdf/10.1137/1.9781611973679.ch11.

[4] E. Benavent, Á. Corberán and J. M. Sanchis, A metaheuristic for the min-max windy rural postman problem with k vehicles, Computational Management Science 7.3 (2010), 18.

[5] E. Benavent and D. Soler, The directed rural postman problem with turn penalties, Transportation Science 33.4 (1999), 408–418.

[6] L. Bodin, G. Fagin, R. Welebny and J. Greenberg, The design of a computerized sanitation vehicle routing and scheduling system for the town of Oyster Bay, New York, Computers & Operations Research 16.1 (1989), 45–54.

[7] O. Bräysy, E. Martinez, Y. Nagata and D. Soler, The mixed capacitated general routing problem with turn penalties, Expert Systems with Applications 38.10 (2011), 12954–12966.

[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 (2004), 44–50.

[9] J. F. Campbell, A. Langevin and N. Perrier, "Advances in vehicle routing for snow plowing", Arc Routing: Problems, Methods and Applications, Á. Corberán and G. Laporte (Editors), Society for Applied Mathematics and the Mathematical Optimization Society, Philadelphia, PA, 2014, chap. 14, pp. 321–350, eprint: `http://epubs.siam.org/doi/pdf/10.1137/1.9781611973679.ch14`.

[10] Á. Corbéran, R. Marti, E. Martinez and D. Soler, The rural postman problem on mixed graphs with turn penalties, Computers & Operations Research 29.7 (2002), 887–903.

[11] Á. Corberán, I. Plana and J. M. Sanchis, "The rural postman problem on directed, mixed, and windy graphs", Arc Routing: Problems, Methods and Applications, Á. Corberán and G. Laporte (Editors), Society for Applied Mathematics and the Mathematical Optimization Society, Philadelphia, PA, 2014, chap. 6, pp. 101–127, eprint: `http://epubs.siam.org/doi/pdf/10.1137/1.9781611973679.ch6`.

[12] M. Gendreau, G. Laporte and S. Yelle, Efficient routing of service vehicles, Engineering Optimization 28.Compendex (1997), 263–271.

[13] G. Ghiani and G. Improta, An algorithm for the hierarchical Chinese postman problem, Operations Research Letters 26.1 (2000), 27–32.

[14] N. Golbaharan, "An Application of Optimization to the Snow Removal Problem: A Column Generation Approach, Thesis No. 886", PhD Thesis, Linköping, Sweden: Linköping University, 2001.

[15] P. Korteweg and T. Volgenant, On the hierarchical Chinese postman problem with linear ordered classes, European Journal of Operational Research 169.1 (2006), 41–52.

[16] P. Lacomme, C. Prins and W. Ramdane-Chérif, Competitive memetic algorithms for arc routing problems, Annals of Operations Research 131.1-4 (2004), 159–185.

[17] P. Lacomme, C. Prins and W. Ramdane-Chérif, Evolutionary algorithms for periodic arc routing problems, European Journal of Operational Research 165.2 (2005), 535–553.

[18] G. Laporte, Modeling and solving several classes of arc routing problems as traveling salesman problems, Computers & Operations Research 24.11 (1997), 1057–1061.

[19] C. E. Noon and J. C. Bean, An efficient transformation of the generalized traveling salesman problem, INFOR 31.1 (1993), 39.

[20] N. Perrier, A. Langevin and C.-A. Amaya, Vehicle routing for urban snow plowing operations, Transportation Science 42.Compendex (2008), 44–56.

[21] 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 and Operations Research 34.Compendex (2007), 258–294.

[22] D. Pisinger and S. Ropke, A general heuristic for vehicle routing problems, Computers and Operations Research 34.8 (Aug. 2007), 2403–2435.

[23] C. Prins, "The capacitated arc routing problem: Heuristics", Arc Routing: Problems, Methods and Applications, Á. Corberán and G. Laporte (Editors), Society for Applied Mathematics and the Mathematical Optimization Society, Philadel-

phia, PA, 2014, chap. 7, pp. 131–157, eprint: http://epubs.siam.org/doi/pdf/10.1137/1.9781611973679.ch7.

[24] O. Quirion-Blais, M. Trépanier and A. Langevin, A case study of snow plow routing using an adaptive large hood search metaheuristic, Transportation Letters-the International Journal of Transportation Research 7.4 (2015).

[25] G. Razmara, "Snow Removal Routing Problems: Theory and Applications", PhD Thesis, Linköping, Sweden: Linköping University, Linköping Studies in Science and Technology, 2004.

[26] S. Ropke and D. Pisinger, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, Transportation Science 40.4 (2006), 455–472, eprint: http://dx.doi.org/10.1287/trsc.1050.0135.

[27] S. Roy and J.-M. Rousseau, The capacitated Canadian postman problem, INFOR 27 (1989), 58–73.

[28] M. A. Salazar-Aguilar, A. Langevin and G. Laporte, Synchronized arc routing for snow plowing operations, Computers & Operations Research 39 (2012), 1432–1440.

[29] U. B. Sayata and N. P. Desai, "An algorithm for hierarchical Chinese postman problem using minimum spanning tree approach based on Kruskal's algorithm", Advance Computing Conference (IACC), 2015 IEEE International, June 2015, pp. 222–227.

[30] D. Soler, E. Martinez and J. C. Micó, A transformation for the mixed general routing problem with turn penalties, Journal of the Operational Research Society 59.4 (2008), 540–547.

[31] Technical Committee 2.4 - Winter Service, Snow and Ice Databook 2014, tech. rep. SIDB 2014EN, La Défense, France: World Road Association (PIARC), 2015.

[32] G. Ulusoy, The fleet size and mix problem for capacitated arc routing, European Journal of Operational Research 22.3 (1985), 329–37.

[33] Ville de Montréal, Coût de l'entretien hivernal des routes par kilomètre de voie entretenue l'hiver, `http://ville.montreal.qc.ca/vuesurlesindicateurs/index.php?kpi=1145`.

[34] E. J. Willemse and J. W. Joubert, Applying min-max k postmen problems to the routing of security guards, Journal of the Operational Research Society 63.2 (2012), 245–260.

Appendix A Formulation for the split problem with makespan objective

$$\text{Minimize} \quad U \tag{15}$$

$$\text{s.t.} \quad U \geq d_{ij}x_{ij} \qquad i \in V, j \in V \tag{16}$$

$$\sum_{i \in V}\sum_{j \in V} x_{ij} = |K| \qquad i \in V, j \in V \tag{17}$$

$$\sum_{i \in V} x_{ij} = \sum_{m \in V} x_{jm} \qquad j \in V \tag{18}$$

$$\sum_{j \in V} x_{dj} = 1 \tag{19}$$

$$\sum_{i \in V} x_{ie} = 1 \tag{20}$$

$$x_{ij} = \{0, 1\} \qquad i \in V, j \in V \tag{21}$$

This formulation seeks to find the shortest path traversing the auxiliary graph. An example of such graph is presented in Figure 13. The variables x_{ij} indicate if the arc going from node i to node j is traversed. The constants d_{ij} correspond to the distance between the nodes i and j. The objective (15) together with constraints (16) seek to minimize the

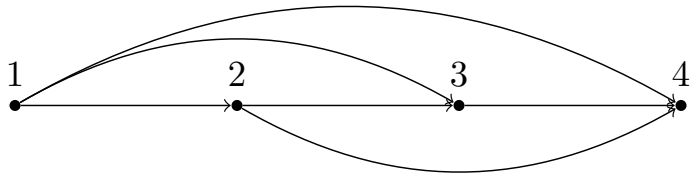


Figure 13: Example of an auxiliary graph.

longest arc traversed in the graph. Constraints (17) insure that the maximum number of vehicles $|K|$ is respected. Constraints (18) insure continuity in the graph. Constraints

(19) and (20) allow to come in and go out of the graph.

Appendix B Features of Destruction Operators

Table 6: The operators share the same features but they follow different arrangements.

Name of operator	Number of nodes to be removed	Number of applications	Route selection	First-node selection	Grouping methodology
D1	Select number of nodes	Select number of applications	Random	Random	Random
D2	Select number of nodes	Select number of applications	Random	Random	Min distance start/end
D3	Select number of nodes	Select number of applications	Random	Random	Until f(sol) improves
D4	Select number of nodes	Select number of applications	Random	Random	Continuous
D5	Select number of nodes	Select number of applications	Most empty	Random	Random
D6	Select number of nodes	Select number of applications	Most empty	Random	Min distance start/end
D7	Select number of nodes	Select number of applications	Most empty	Random	Until f(sol) improves
D8	Select number of nodes	Select number of applications	Most empty	Random	Continuous
D9	Select number of nodes	Select number of applications	Longest	Random	Random
D10	Select number of nodes	Select number of applications	Longest	Random	Min distance start/end

continued ...

... continued

Name of operator	Number of nodes to be removed	Number of applications	Route selection	First-node selection	Grouping methodology
D11	Select number of nodes	Select number of applications	Longest	Random	Until f(sol) improves
D12	Select number of nodes	Select number of applications	Longest	Random	Continuous
D13	Select number of nodes	Select number of applications	Random	Isolated nodes	
D14	Select number of nodes	Select number of applications	Longest	Isolated nodes	
D15	Select number of nodes	Select number of applications	Most empty	Isolated nodes	
D16	Select number of nodes	Select number of applications	Worst cost		Random
D17	Select number of nodes	Select number of applications	Worst cost		Until f(sol) improves
D18	Select number of nodes	Select number of applications	Worst cost		Min distance start/end
D19	Select number of nodes	Select number of applications	Worst cost		Continuous
D20	Select number of nodes	Select number of applications	Longest	None	Split
D21	Select number of nodes	Select number of applications	Most empty	None	Split

continued ...

...continued

Name of operator	Number of nodes to be removed	Number of applications	Route selection	First-node selection	Grouping methodology
D22	Select number of nodes	Select number of applications	Random	None	Split

Appendix C Description of Features of Construction Operators

Table 7: As for the destruction operators, the construction operators share the same features but they follow different arrangements.

Name of operator	Grouping methodology	Insertion position
C1	Removed sequences	Best insertion
C2	Removed sequences	k-regret
C3	LKH	Best insertion
C4	LKH	k-regret