Snowplow Route Optimization Using Chinese Postman Problem and Tabu Search Algorithm

Abdullah Rasul[a], Jaho Seo[a], Shuoyan Xu[b], Tae J. Kwon[b], Justin MacLean[c], and Cody Brown[c]

[a]Department of Automotive and Mechatronics Engineering, Ontario Tech University, Canada
[b]Department of Civil and Environmental Engineering University of Alberta, Canada
[c]Office of the Chief Administrative Officer, Municipality of Clarington, Canada
E-mail: abdullah.rasul@ontariotechu.net, jaho.seo@ontariotechu.ca, shuoyan@ualberta.ca, tjkwon@ualberta.ca, jmaclean@clarington.net, cbrown@clarington.net

Abstract –

Snowplowing is critical to winter road operation and maintenance since it can improve driver's safety and mobility. The goal of this study is to generate optimal routes for snowplowing that can reduce travel distance and improve efficiency by considering operational constraints. To achieve this goal, we first adopted the Chinese Postman Problem to generate initial routes to be Euler circuits, and then the shortest path was generated using Dijkstra's algorithm. For an optimization process, the tabu search algorithm as a meta-heuristic approach was applied to find near-optimal routes by optimizing the order of precedence of snowplow routes, and the minimum maintenance standards and turn directions were considered as a constraint of the defined objective function.

Through a simulation study, we compared routes generated by different approaches in terms of total travel distance, turning restriction, and road maintenance priority.

Keywords –

Snowplow optimization; Chinese Postman Problem; Tabu search algorithm; MMS; Dijkstra's algorithm

1 Introduction

In many Canadian municipalities, the snowfall is sufficiently heavy to require the use of large, costly snowplowing vehicles to remove it. A slight delay in the schedule can result in the formation of thick sheets of ice that worsen public travel safety. In extreme circumstances, storm drains become blocked, and transportation is brought to a halt. However, municipalities need to spend a considerable amount on annual expenditures to provide an acceptable level of snowplowing service. One of the promising solutions to deal with this problem is to identify optimal routes for snowplow trucks that can allow the desired level of service quality to be maintained while improving efficiency and reducing costs.

Since snowplowing is a complex process, there are various factors to be considered that have a direct influence on formulating optimal routes to reduce operational costs. The representative examples of these factors include the number of trucks, deadhead, fuel consumption, and time travel [1].

To deal with this combinatorial optimization problem, mathematical optimization techniques can be a tool to design efficient and optimal routes for snow plowing [1, 2]. In the earlier studies on snowplow routing, Malandraki and Daskin formulated the maximum Chinese Postman Problem, which allowed each arc to be visited multiple times in order to derive a snowplow route [3]. The authors in [4] proposed a hierarchical optimization method in which the high-priority roads are served first.

Kinable et al. compared mixed-integer programming with constraint programming to evaluate the effectiveness of the heuristic algorithm that was used for the problem of optimizing the routes [5]. A heuristic approach based on network clustering and capacitated vehicle routing was proposed by [6], which deals with ice and snow control.

These studies have not included complex real-world constraints fully, so a more sophisticated approach is needed to optimize routes while accurately representing actual operational conditions. As an actual constraint that needs to be further considered, Minimum Maintenance Standards (MMS) [7] is the required service time that municipalities in Ontario must meet to remove ice and snow according to the defined roadway class. The number of ways (e.g., 1- or 2-way street) and turning direction are also key constraints that can restrict the generation of optimal routes and affect efficiency in snowplowing. Therefore, this project aims to propose an advanced and practical optimal routing strategy for residential snowplowing services that can identify optimal routes, and thus generate the least costs while considering various operational constraints that have not been fully considered in the existent studies.

For this, Chinese Postman Problem (CPP) [8] and Dijkstra's algorithm were applied to generate initial routes and shortest paths, respectively. The tabu search algorithm [9] was used to find near-optimal routes by optimizing the order of precedence of snowplow routes that can satisfy the MMS and turn directions as a constraint for the optimization process.

This paper is divided into five sections. In Section 1, the complex process of snow plowing is described and existing studies are discussed. Section 2 discusses data gathering and processing for route generation. Section 3 elaborates on the applied methodologies, which include the defined objective function, CPP, and tabu search algorithm. The simulation results are presented in Section 4 along with a comparison between the generated routes. Section 5 includes a summary of the findings and recommendations for future research.

2 Data Gathering and Processing

This section describes the process of gathering and processing the geographic data necessary to generate the initial route for each truck.

For the data gathering, OpenStreetMap (OSM) [10] was selected, which is a free editable map of the world, created by a community of mappers. To begin gathering data, a map of the targeted municipality (Clarington) was extracted from OSM using the QGIS [11] software, as seen in Figure 1.

Figure 1. Map of the targeted municipality

Since trucks have designated areas, the next step is to extract the individual truck routes on Clarington's network using the map in Figure 1.

The routes for one of the operational trucks in the OSM and Clarington's network are shown below as an example.

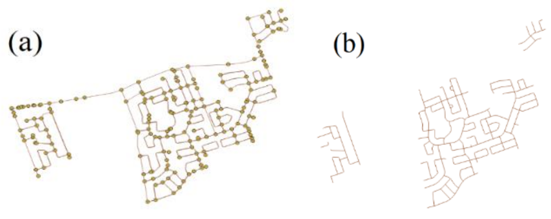


Figure 2. Routes of a truck in OpenStreetMap (a) and corresponding ones in the Clarington's network (b)

The attributes from the city's network, including roadway class, number of ways, etc., were transferred to the OSM routes. Using a complete set of initial data (OSM routes) for each truck that is required for optimal routing, shapefiles were generated to apply the optimization algorithm.

3 Methodology

3.1 Problem Description and Objective Function

The route optimization problem considered in this study is a classical route inspection problem, which can be described using several parameters as follows.

Let $G(V,E)$ be a strongly connected multigraph where $V=\{v_0,v_1 \dots, v_n\}$ is a set of nodes including the depot location, v_0 and $E=\{(v_i,v_j): v_i, v_j \epsilon\, V, i<j\}$ is a set of directed edges. Nodes represent intersections in the road network, while edges represent road segments.

Each edge $e=\ (v_i,v_j)\ \epsilon\ E$ has a non-negative travel cost, c_{ij} interpreted as the travel distance. t_{ij} is the travel time for each road segment.

Another variable introduced in this study is the accumulated time, $T_{ij}=\ \sum t_{ij}$ that sums up each t_{ij}. Each edge should be traversed at least once and deadheading (i.e., act of traversing an edge without service) should be minimized.

Each node contains the geographical information of longitude and latitude that is used to calculate the heading angle and the number of turns.

Each road segment is given the priority of road class, $X_{ij}\ \epsilon\ P$ that is denoted by the range of $P=\{1,\dots,p\}$ where the roads in class 1 have a higher priority than the rest of the roads. Often, the priority class of a road is determined by its traffic volume. As a result, highways have a higher class than residential roads due to their higher traffic volume.

To meet the MMS, we defined an objective function that yields a higher value for servicing higher-class roads first as compared to servicing in reverse order. Figure 3 demonstrates how to calculate the values for the two opposite cases to deal with the MMS. Since a higher value is desired to satisfy the MMS, our objective function for the optimization process was set to provide an optimal solution when it maximizes its output. In the objective function, we also penalize it whenever left-, U-, and sharp right turns occur since these types of turns need to be avoided for safety and efficiency. In this paper, the calculated value of the objective function is referred to as the fitness value. The finalized objective function is given in Eq. (1).

$$\max\left(\sum X_{ij}\ T_{ij} - a * \sum Left_{turns} - b * \sum U_{turns} - c * \sum SharpRight_{turns}\right) \quad (1)$$

where a, b, c are coefficients.

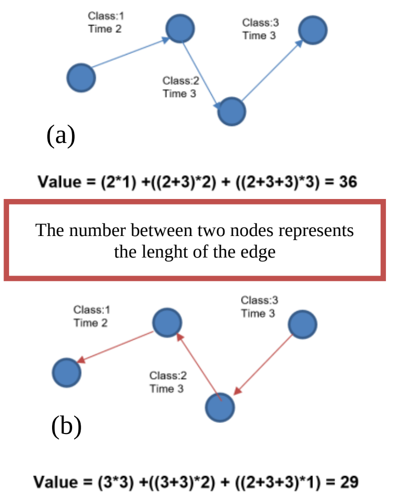
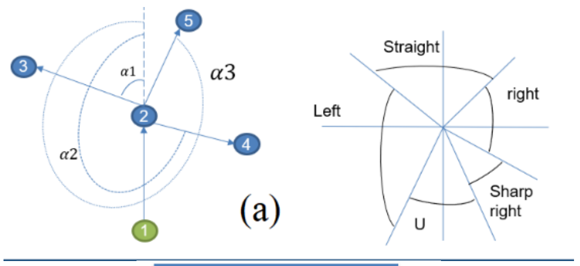

Class:1
Time 2
Class:3
Time 3
Class:2
Time 3
(a)
Value = (2*1) +((2+3)*2) + ((2+3+3)*3) = 36
The number between two nodes represents the lenght of the edge
Class:1
Time 2
Class:3
Time 3
Class:2
Time 3
(b)
Value = (3*3) +((3+3)*2) + ((2+3+3)*1) = 29

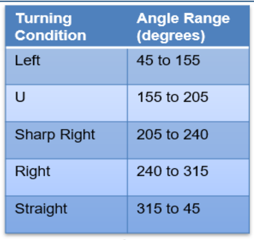
Figure 3. Comparison of routing sequence for the MMS: Routing sequence for higher class roads (a) and lower class roads (b)

In order to count the left-, U- and sharp right turns, we distinguish each type of turn by defining its angle range (Figure 4). This angle is calculated relative to a heading. When the truck reaches an intersection, the angle between this intersection and the next one can be calculated using cosine laws.

3.2 Chinese Postman Problem

After translating the snowplow problem into the language of graph theory and defining our objective function and constraints, the next step is to address the Chinese Postman Problem (CPP), a well-known problem in graph theory. When given a connected undirected weighted graph G, the CPP computes a minimum cost-closed path that contains all edges at least once.





Turning Condition	Angle Range (degrees)
Left	45 to 155
U	155 to 205
Sharp Right	205 to 240
Right	240 to 315
Straight	315 to 45

(b)

Figure 4. The defined angle ranges for each type of turn

The pseudocodes for the CPP algorithm are presented in Figure 5 and each step will be explained with the results presented in Section 4.2.

```
Chinese Postman Problem - Pseudocodes
Input: An undirected connected graph $G = (V, E)$
Output: A (CPP) tour $T = (n_1, e_1, n_2, e_2, \ldots, n_1, e_1, n_{i+1} = n)$
Step 1: Determine if $G$ is Eulerian
Step 2: If G is Eulerian, obtain an Euler − tour $T$
Step 3: If $G$ is not Eulerian, find a minimum − cost augmentation of $G$ to construct an Eulerian multigraph $G'$, and let $T$ be and Euler − tour of $G'$
Stop
```

Figure 5. A pseudo algorithm for the Chinese Postman Problem

An Eulerian graph is a closed path that uses every edge of a graph exactly once. Figure 6 shows a simple example of a conversion of a non-Eulerian graph to an Eulerian. Figure 6 (a) is a non-Eulerian graph while Figure 6 (b) is the minimum-weight expansion of the non-Eulerian graph to make Eulerian. In the figure, every double edge in the graph represents a backtrack. From the assumption that node 1 is the starting and ending node for the closed tour, a possible solution obtained by applying the CPP algorithm is $T = (1, 2, 4, 5, 6, 5, 7, 3, 4, 3, 2, 1)$ with a total length of 24 by summing up the length of edges (i.e., the distance between nodes) to travel.

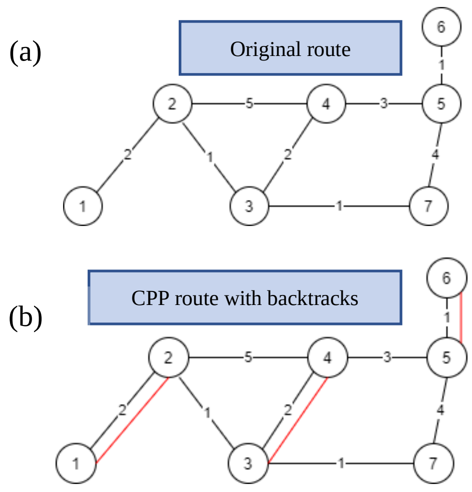


Figure 6. Non-Eulerian graph (a) and minimum-weight expansion of the non-Eulerian graph (b).

3.3 Tabu Search Algorithm

Tabu search is a meta-heuristic optimization technique for optimizing model parameters. Similar to other metaheuristic algorithms, the tabu search algorithm starts with an initial solution and progresses iteratively by searching immediate neighborhoods. During the search process, a set of predefined local search schemes and unique memory structures are used to prevent a solution from getting stuck in a suboptimal trap. The pseudocodes of the tabu search algorithm implemented in this study are provided in Figure 7 [12].

The tabu search algorithm relies on three main factors, namely, short-term memory, long-term memory, and tabu list. The short-term memory prevents the algorithm from generating the previously generated solution and hence intensifies a search. The long-term memory performs diversification to search into new regions and prevent a solution from being stuck in a suboptimal dead-end. By containing all the previous solutions, the tabu list facilitates the short-term memory to track the previously visited solutions.

The next step was to generate efficient neighborhoods with the help of the permutation process after finding the initial route. Once an optimal solution is selected among the newly generated neighborhoods, it is verified whether the generated solution is better than the initial one. If a better solution is found, the permutation process is repeated to find another set of different neighborhoods; otherwise, a different combination of routes is searched through the procedure of merging and separating edges.

```
Algorithm 1 Tabu Search Algorithm
    Input: An initial solution of truck routes, maximum iteration number, length of tabu
    Output: The possible improved truck routes

    Step 1: Initialization
    Set
Current solution=Inital solution
Current value=Initial value
Best solution=Initial solution
Best value=a large number
Maximum number of iteration=a large number

    Step 2: Forward updating objective function value

  for $iteration\ number = 1 : maximum\ iteration\ number$ do
      while $Best\ value \geq Current\ value$ do
          Randomly generate 24 different neighborhood based on the current solution using permutation
          Permutation Process:
          Current solution=Best found solution not in tabu
          Current value=Objective function (current solution)
          if Best value $\geq$ Current value & current solution not in tabu list then
              Best solution=Current solution
              Best value=Current value
              add current solution to tabu list
          end if
      end while
  end for
```

Figure 7. The logic flow of the tabu search algorithm

3.4 Dijkstra's algorithm

To find the shortest path between the nodes (intersections), Dijkstra's algorithm was implemented [13]. Dijkstra's algorithm maintains a set of vertices whose final shortest-path weights from the source (depot location in our case) have already been determined. The algorithm repeatedly selects a vertex with the minimum shortest path estimate and relaxes all edges leaving the selected vertex. Since Dijkstra's algorithm is a single-source shortest-path problem on a weighted and directed graph, it matches exactly with the snowplowing routing problem described in this study.

4 Results and Discussions

This section provides an explanation of the results along with an analysis of them. Specifically, the generation of a network topology from initial routes will be discussed first, and then, the results obtained by applying the CPP algorithm are presented. Finally, the results obtained with the optimization algorithm are discussed.

4.1 Network Topology Generation

Using the NetworkX library [14], the initial routes generated with OSM and QGIS were converted into a network topology in Python.

A network topology of one of the service trucks is shown in Figure 8 in the Python graphing format, where the nodes are labeled with their node IDs.

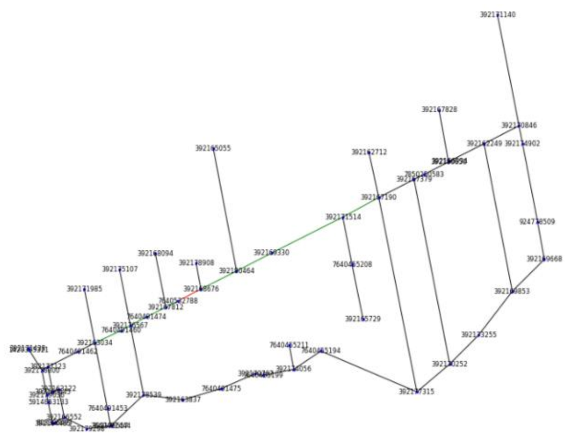


Figure 8. Network typology in the Python graphing format.

4.2 Results Obtained Using the CPP Algorithm

Following the generation of the network topology, the CPP algorithm was implemented to create Euler circuits for each truck. The CPP algorithm first identifies all odd degree nodes, then and adds backtracks (edges) to make them even nodes in order to generate an Eulerian graph (circuit).

To achieve an Eulerian graph, the CPP follows three steps. The first step is to compute all possible pairs of odd degree nodes as seen in Figure 9. The second step is to calculate the shortest path between nodes that can be achieved by applying Dijkstra's algorithm to minimize the distance for backtracking [15]. Hence, this step plays a critical role in achieving optimal routes. Figure 10 shows a complete graph connecting every node with the shortest path through the second step.

The final step in the CPP algorithm is to add backtracks to the original network. Once we have the information about the roads entailing backtracking, the original map is updated by including the backtracked edges that are highlighted in blue in Figure 11.

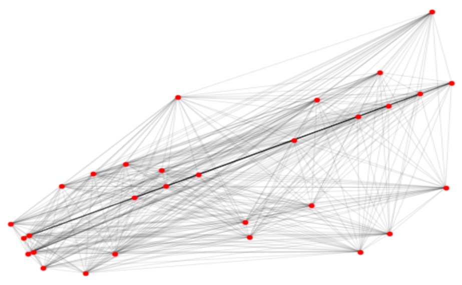


Figure 9. Pairs of odd degree nodes

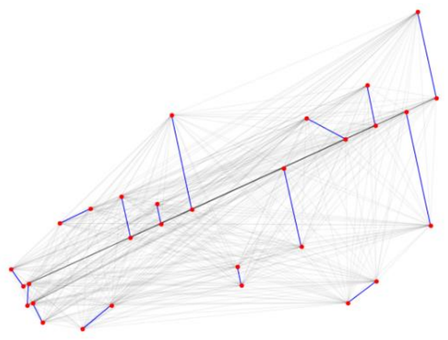


Figure 10. The shortest path among pairs of odd degree nodes

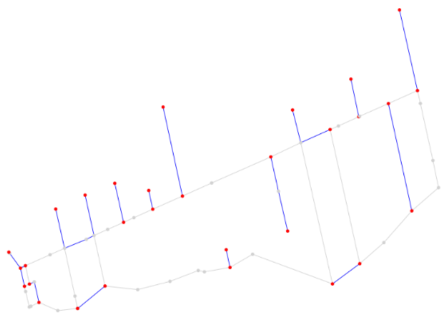


Figure 11. Backtracks added to the original network

4.3 Results of Optimization Algorithms

Table 1 presents the values for major variables assumed during the optimization process. For the MMS, the roadway should be serviced within the required time set out in the table (i.e., original time in a parenthesis for each road's class). However, since the service area covered by only one truck was considered for simulations, the shortened time was applied instead of the original application time. For example, the original applicable time of 4 hours for road class 1 was assumed to be 1 hour instead.

To administer the optimization process, it is required to calculate the fitness value of each route. The routes with the highest fitness value qualify for the next iteration of optimization. The optimization was conducted for 10 iterations.

In Table 2 presenting simulation results, the route in the first column (Route 1) was obtained as a reference route by the application of the CPP and Dijkstra's algorithm. The route in the second column was generated using a combination of the CPP with Dijkstra's algorithm and the tabu search optimization.

The tabu search algorithm can generate optimal solutions by swapping the neighboring nodes in the permutation process. Therefore, for the second column route (Route 2), neighboring nodes were swapped at nodes where at least three roads (edges) are connected.

Table 1. Assumptions for the optimization process

Matric	Assumed Value
Vehicle speed	35 km/h
Truck areas	Clarington
Number of trucks	1
Applicable time for road class 1	1 hour for simulation (original time: 4 hours)
Applicable time for road class 2	2 hours for simulation (original time: 6 hours)
Applicable time for road class 3	3 hours for simulation (original time: 12 hours)
Applicable time for road class 4	4 hours for simulation (original time: 16 hours)
Applicable time for road class 5	5 hours for simulation (original time: 24 hours)
Number of lanes	Both single and multi-lanes covered

The fitness value is affected by the number of left-, U-, and sharp right turns, and MMS (see Eq. (1)). If a route has the highest value of $\sum X_{ij}\ T_{ij}$ regarding MMS with the fewest number of these turns, this is considered the best optimal route.

In the table, while Route 2 has one more left turn than Route 1, there are fewer U-turns and roads failing to meet the MMS in Route 2 than in Route 1.

Therefore, Route 2 provides a safe and practical solution by minimizing undesirable turns and considering the required service time based on MMS. In addition, both the total travel time and distance of Route 2 are slightly shorter than those of Route 1.

The fitness values in Table 2 take into consideration of both the above operational constraints and traditional issues of travel distance and time. Hence, the highest fitness value of Route 2 implies that this is the optimal route by providing the best overall performance that eliminates unfavorable turning directions and achieves the shortest total travel time (i.e., the shortest distance for backtracking with Dijkstra's algorithm) simultaneously.

In Figure 12, the roads that are serviced within the designated service time are labeled as 1, and those that do not meet this requirement are labeled as 0.

In the optimal route, a starting node for service was selected based on the shortest distance from the depot location. This will allow a truck to travel the shortest distance to reach its designated service area. Figure 13 indicates the location of a starting node for the truck considered in this study that guarantees the shortest distance to the depot.

Table 2. Comparison between the routes obtained from simulations

Routes	Route 1: CPP	Route 2: CPP + Optimization
Fitness	-20	60
Total travel Time	1h 17min	1h 15 min
Total travel distance (km)	63.4	62.9
Left Turns	21	22
U-turns	12	9
Sharp Right Turns	0	0
Number of roads (edges) failing to the MMS	4	3

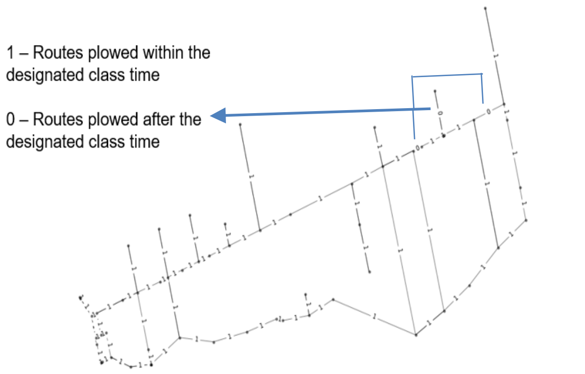

1 – Routes plowed within the designated class time
0 – Routes plowed after the designated class time

Figure 12. The satisfaction of MMS is indicated on the map

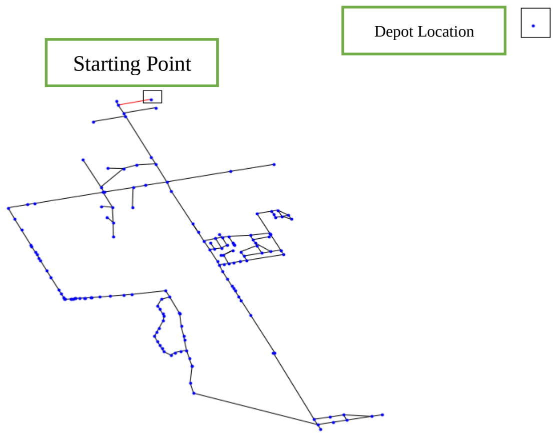


Figure. 13. Locations of the starting node and depot for the truck under consideration

5 Conclusions

The main objective of this project is to optimize the routes of residential snowplowing services. For this, an initial network topology was generated using the GIS data from the OSM that includes the city network's attributes. Then, the CPP with Dijkstra's algorithm and the optimization process with the tabu search algorithm were applied to generate an optimal route.

In the defined objective function for the optimization process, the MMS and penalty for unwanted turning directions for safety and efficiency were considered. Therefore, the generated optimal solution represents the route that can satisfy the shortest travel time and the MMS requirement as well as minimize the number of deadheads and unpreferred turns.

Since our study was limited to simulation results, it is required to validate the proposed method through field tests by considering the real-time dynamic routing situations and on-site weather conditions.

The project focuses on the improvement of the current service efficiency by optimizing the route sequence. Therefore, as future work, we can create new routes by reassigning roads to further reduce the total travel distance in the entire service area and to balance the workload between truck drivers.

References

[1] Rao, T., Mitra, S., J. Zollweg, J., and Sahin. F. Computing optimal snowplow route plans using genetic algorithms. *Systems, Man, and Cybernetics (SMC). IEEE*, 2785–2790. 2011.

[2] Rao, T., Mitra, S., and Zollweg, J. Snow-plow route planning using AI search. *Systems, Man, and Cybernetics (SMC). IEEE*, 2791–2796. 2011.

[3] Malandraki, C. and Daskin, M. The maximum benefit Chinese postman problem and the maximum benefit traveling salesman problem. *European Journal of Operational Research*, 65:218–234, 1993.

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

[5] Kinable, J., van Hoeve, W., and Smith, S. Optimization models for a real-world snowplow routing problem. *In Proceedings of International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 229-245, 2016.

[6] Sullivan, J., Dowds, J., Novak, D., Scott, D., and Ragsdale, C. Development and application of an iterative heuristic for roadway snow and ice control. *Transportation research part A: policy and practice*, 127:18-31, 2019.

[7] Ontario. Minimum maintenance standards for municipal highways. On-line: https://www.ontario.ca/laws/regulation/020239, Accessed: 10/11/2020.

[8] Minieka, E. The Chinese postman problem for mixed networks. *Management Science*, 25:643-648. 1979.

[9] Ahr, D., Reinelt, A Tabu search algorithm for the Min–Max k-Chinese postman problem. *Computers & Operations Research*, 33(12):3403–3422, 2006.

[10] OpenStreetMap. OpenStreetMap data. On-Line: https://www.openstreetmap.org, Accessed: 01/10/2020.

[11] Qgis.org. QGIS. On-line: https://www.qgis.org/en/site, Accessed: 10/09/2020.

[12] Xu, S. and Kwon, T. Optimizing snowplow routes using all-new perspectives: road users and winter road maintenance operators. *Canadian Journal of Civil Engineering*, 2020.

[13] Cormen, T., Leiserson, C., Rivest, R., and Stein, C. *Introduction to Algorithms*, 2nd ed. The MIT Press, 2001.

[14] NetworkX. NetworkX documentation. On-Line: https://networkx.org, Accessed: 30/10/2020.

[15] Dijkstra, E., A note on two problems in connexion with graphs. *Numerische mathematik*, 1:269–271, 1959.