Received: 8 November 2022 | Revised: 4 September 2023 | Accepted: 15 September 2023

DOI: 10.1002/net.22189

SPECIAL ISSUE ARTICLE

WILEY

A rolling horizon framework for the time-dependent multi-visit dynamic safe street snow plowing problem

Georg E. A. Fröhlich[1] | **Margaretha Gansterer**[2] | **Karl F. Doerner**[1,3]

[1]Department of Business Decisions and Analytics, University of Vienna, Vienna, Austria
[2]Department for Operations Management and Logistics, University of Klagenfurt, Klagenfurt, Austria
[3]Data Science, University of Vienna, Vienna, Austria

Correspondence
Georg E. A. Fröhlich, Department of Business Decisions and Analytics, University of Vienna, Oskar-Morgenstern-Platz 1, 1090 Vienna, Austria.
Email: georg.erwin.adrian.froehlich@univie.ac.at

Abstract

As a major real-world problem, snow plowing has been studied extensively. However, most studies focus on deterministic settings with little urgency yet enough time to plan. In contrast, we assume a severe snowstorm with little known data and little time to plan. We introduce a novel time-dependent multi-visit dynamic safe street snow plowing problem and formulate it on a rolling-horizon-basis. To solve this problem, we develop an adaptive large neighborhood search as the underlying method and validate its efficacy on team orienteering arc routing problem benchmark instances. We create real-world-based instances for the city of Vienna and examine the effect of (i) different snowstorm movements, (ii) having perfect information, and (iii) different information-updating intervals and look-aheads for the rolling horizon method. Our findings show that different snowstorm movements have no significant effect on the choice of rolling horizon settings. They also indicate that (i) larger updating intervals are beneficial, if prediction errors are low, and (ii) larger look-aheads are better suited for larger updating intervals and vice versa. However, we observe that less look-ahead is needed when prediction errors are low.

KEYWORDS

arc routing, rolling horizon, safe and secure, snow plowing, team orienteering, vehicle routing

1 | INTRODUCTION

Snow plowing is a major real-world problem. It affects street safety, travel times, and traffic jams. Furthermore, some regions suffer considerable costs for efficient plowing and often go over budget, for example, total costs of $ 192 million in Montreal in 2018 [18]. To deal with such problems, snow plowing has been studied extensively in the literature. Most studies, however, consider deterministic problems where arcs do not need repeated plowing and priorities between sets of arcs do exist. These models are good representations of situations after typical snowstorms with no major impact. In our opinion, however, these models are not too well suited for severe snowstorms that threaten to shut down the whole infrastructure and that require immediate care. Such a scenario was considered by Dussault et al. [4]. The snowstorm of their model warranted a shutdown and imposed impassable streets, even for a snow plowing truck. A problem as in the aforementioned paper could be evaded or mitigated, should effective plowing be performed before the accumulation of masses of snow.

We want to address this research gap by modeling a novel time-dependent multi-visit dynamic safe street snow plowing problem called the time-dependent multi-visit dynamic team orienteering arc routing problem (TDMVD-TOARP), which aims to keep the streets as safe as possible. The roads are deemed safe if the accumulated snow is below a specified threshold. Different priorities are assigned to streets according to classical characteristics (e.g., streets leading to critical infrastructure,

streets located in heavily populated areas). In doing so, however, we do not create classes of priorities (e.g., Perrier et al. [13]), where higher ones must be serviced before the lower ones; instead, we transform them into weights and include them in the objective function. Further, fleet costs are not considered, as the fleet is assumed to be available anyways, and we want to maximize the efficacy of the available resources. Snowfall intensity is not assumed to be constant or known ahead of time. This can result in some streets not requiring any service, while others require multiple plowings to be deemed safe over the entire planning horizon. These characteristics lead to a time-dependent problem wherein an earlier plowing can turn detrimental (see Section 3.5). A time-discretized mixed integer program (MIP) would be very challenging to solve, even if snowstorm movement is predictable; therefore, we address the problem on a rolling planning horizon. The proposed framework splits the problem into smaller problems and predicts the future snowfall and the state of the system before solving each subproblem. The subproblems are, hence, considered to be deterministic and solutions are generated using adaptive large neighborhood search (ALNS). Due to considering a real-world problem with dynamic changing data, the rolling horizon framework has only a very limited computation time to calculate a solution for a given subproblem, therefore a metaheuristic solution technique is used. During this time the solution for the previous subproblem is executed (e.g., the solution for the subproblem starting at minute 31 is determined during minutes 30 to 31; then, for minute 32, the solution will be determined during minutes 31 to 32). In the following, this available computation time will be called *updating interval*. In order to not be too myopic, subproblems might plan a little ahead of time, even if the latter part of the solution will not be used (e.g., the subproblem lasts from minute 31 to 32.5). This additional time (e.g., 0.5 min) will be called *look-ahead*.

The contribution of our study is fourfold:

1. We introduce a novel and challenging dynamic safe street snow plowing problem, model it as a TDMVD-TOARP, and tackle it in a rolling horizon framework.
2. To generate solutions, we propose an effective ALNS-based method with tailored destroy and repair operators.
3. In an extensive computational study, we examine the effects of different snowstorm movements, different levels of information, and forecasting quality.
4. We perform an in-depth analysis of the most important parameters for the rolling horizon frameworks (e.g., updating interval, look-ahead).

The remaining paper is structured as follows: Section 2 provides an overview of the related literature. Section 3 defines the problem and describes the transformation to the rolling horizon setting. Section 4 explains the proposed solution approach, while Section 5 presents the computational study. There, we first show the efficacy of the algorithm and then apply it to real-world-based instances. Finally, Section 6 includes concluding remarks and future research.

2 | LITERATURE REVIEW

An extensive overview of the characteristics and problems related to snowfall can be found in the four-part review by Perrier et al. [9-12]. The authors classify the problems on strategic (e.g., facility location, fleet replacement scheduling), tactical (e.g., sector assignment to snow disposal sites, fleet size), and operational (e.g., vehicle routing and crew management) levels and extend them to real-time problems. The snow plowing problem is closely related to anti-icing, deicing, and abrasive spreading operations. All these problems typically belong to the operational or real-time class and are often closely related to the Chinese postman problem, rural postman problem, or capacitated arc routing problem. Commonly added characteristics are priorities or hierarchies for subsets of streets, turn penalties, variable speeds, and heterogeneous fleets. Priorities are deemed necessary as some streets (e.g., those leading to a hospital) are more important than others. Priorities can be included by weights or the requirement to serve arcs with a higher priority before those with lower ones. Turn penalties can be due to (i) street conditions leading to risky turns, or (ii) turns leading to snow accumulation on crossings. Within the models, these penalties can be tackled by forbidding very risky turns, adding time for turns, or adding penalties in the objective functions. Variable speeds are needed to carry out deadheading, which is typically faster than plowing. Regarding heterogeneous fleets, authors typically assume that companies use vehicles of different sizes (e.g., smaller ones for alleys), and vehicles for plowing or spreading.

Perrier et al. [13] study a hierarchical rural postman problem with lexicographic objectives; the problem minimizes the makespan for each subset of streets of the same priority and includes heterogeneous vehicles, turn restrictions, and variable speeds. Different decompositions are applied. Aguillar et al. [17] focus on a synchronized arc routing problem with the aim to minimize the makespan. They require synchronization for streets that have multiple lanes in the same direction. This is due to the fact that a single plow would only create a large amount of snow on the neighboring lane. This problem is tackled by using a synchronized convoy of plows, with each plow pushing the snow further to the edge. They tackle it by solving a relaxed MIP and then improving it via ALNS. Dussault et al. [4], in their windy postman problem, forbid the deadheading of non-plowed arcs, as their scenario assumes a snowstorm severe enough to warrant a shutdown. Their objective is cost minimization and they tackle the problem using a heuristic, which delivers results close to the lower bounds. Hajibabai and Ouyang [6] study a stochastic

snow plow fleet management problem with a weighted sum of deadheading and repositioning costs and rewards depending on service levels. The authors split the problem into multiple periods. They assign vehicles to snow routes such that the tasks can be served within one period. In each period, a truck can either (i) serve tasks in its current snow route, (ii) move to a different snow route, or (iii) remain idle. The tasks can arise randomly (following a Poisson process), and vehicles can fail randomly. In case of vehicle failure, the tasks that should have been serviced are added to the next period. To generate solutions, the authors propose adaptive dynamic programming, which is used to evaluate the effect of taking uncertainty into account. Quirion et al. [15] study a hierarchical plowing and spreading problem with heterogeneous fleets, variable speeds, and turn penalties. They use ALNS to solve it for varying fleet compositions and deduce bottlenecks. Quirion et al. [16] focus on a hierarchical rural postman problem with turn penalties, variable speeds, and heterogeneous vehicles with the objective to minimize a weighted sum of the makespan for each subset of streets of the same priority. The problem is transformed into an asymmetric traveling salesman problem and is solved using ALNS.

Holmberg [8] investigates a snow plowing problem for a single snow remover, where streets need to be plowed up to four times. The reasoning is that some cities with broader streets require additional plowing between the lanes, as plows are too narrow. While some types of plowing need to be performed in a specific direction, others can be performed in either direction. The problem is reformulated into an asymmetric traveling salesman problem and solved using a heuristic. Hajizadeh and Holmberg [7] extend this study by creating a hybrid heuristic and a MIP relaxation. The authors use branching techniques in both methods, which allows sharing of information between the hybrid heuristic and the MIP relaxation. Castro et al. [3] prove that the problem in [4] can be solved in polynomial time when costs are symmetric or metric, that is, uphill service costs are, at most, equal to downhill service costs and two uphill deadheadings. This assumption applies to several real-world problems. Additionally, the authors also create heuristics for this problem, which make use of several of their theoretical findings.

Xu and Kwon [19] address service levels based on users' driving times. The objective is to minimize the weighted sum of travel times and the makespan of the snow plowing problem. Travel times are time-dependent, as cleared streets can be traversed faster. Accordingly, the following two user behaviors are tested in this problem: (i) users follow their usual path and (ii) users adapt to street conditions and make use of alternative street segments. The authors propose to tackle the problem using tabu search. Ahabchane et al. [1] address a hierarchical salt spreading model, where priorities are included in the objective function. The latter consists of the total routing costs and the sum product of weights given according to the priorities and the times they are serviced. While the original problem is a mixed capacitated general routing problem, the authors transform it into a node routing problem. Furthermore, the demands are not deterministic, which they tackle using a robust formulation by accounting for demand variation. The authors create a heuristic based on simulated annealing (SA) and evaluate the performance using Monte Carlo simulations.

Through this work, we extend the research on dynamic *real-time* snowplowing problems, where complexity and, particularly, the underlying time-dependencies require a new solution approach. We propose to tackle the problem within a rolling horizon framework and evaluate the updating intervals as well as look-aheads based on prediction accuracy.

3 | PROBLEM FORMULATION

Our problem formulation is split into multiple parts. First, we define the main problem of keeping the streets as safe as possible. However, as it is too large to be solved and, for the rolling framework, is later decomposed into smaller problems, a formal definition for these subproblems is given. Due to the subproblems' limited time frame lengthy arcs might pose problems and require special consideration, explained in Section 3.3. Afterwards the information handling—what information is available, what is only assumed—for the rolling horizon framework is explained. Finally, a section is dedicated to the calculation of changes to the solution.

3.1 | Integrative model

The TDMVD-TOARP can be represented on a graph $G = (V, A)$, where V is the set of vertices and A is the set of arcs in which each arc a is of different priority w_a. Arcs with high w_a typically include locations that are (i) critical infrastructure (e.g., hospitals), (ii) situated within highly populated areas, or (iii) considered arterial roads. The time horizon is denoted as $\mathcal{T}$. During $\mathcal{T}$, a snowstorm passes through the system (the city), causing snow to accumulate, while a fleet of vehicles K, which starts at a depot, clears the streets and tries to keep them as safe as possible.

When an arc a is affected by the snowstorm at time t, its level of snow l_{at} increases with the storm's intensity i_t ($l_{at} = l_{a,t-1} + i_t$). Should an arc a be serviced by a vehicle at time t, the level is reset to 0 ($l_{at} = 0$). Whenever an arc's level of snow is below a threshold H, it is deemed safe ($s_{at} = 1$); otherwise, it is deemed not safe ($s_{at} = 0$).

The objective is to maximize the weighted time streets are safe. Here, cost minimization is not an objective, as the fleet of vehicles is considered to be available.

TABLE 1 Decision variables and constants.

Sets	
A_S	Set of arcs, where traversal includes service
A_T	Set of arcs, where only traversal occurs
D	Set of nodes, where depots are located
K	set of vehicles
T	Set of discretized time units
Parameters	
w_a	Weight (priority) of arc a
d_a	Duration for arc a (depending on the arc it can either be service and traversal If $a \in A_S$, or only traversal, if $a \in A_T$)
i_{at}	Intensity of snowfall on arc a at time t
σ_a	Start node of arc a
ω_a	End node of arc a
e_k	Node from which vehicle k starts
t_k^s	Time at which vehicle k starts
H	Threshold of snow that has to be met
M	Large number
Decision variables	
x_{atk}	Equals 1 if arc a is traversed by vehicle k starting at time t; otherwise, 0
s_{at}	Equals 1, if arc a is unsafe at time t; otherwise, 0
p_{at}	Equals 1, if arc a is serviced starting at time t; otherwise, 0
l_{at}	Level of snow on arc a at time t

To depict the problem as a MIP (cf. Table 1 for naming of decision variables and constants), each arc is represented twice: once in set $A_S = \{a_1^s, \dots, a_n^s\}$ and once in set $A_T = \{a_1^t, \dots, a_n^t\}$, where every arc corresponds to (i) traversing and servicing or (ii) only traversing the arc, respectively. The duration needed for each arc is denoted by d_a. Traversal and service of an arc takes longer than just traversal ($d_{a_i^s} > d_{a_i^t}$). The starting and ending nodes for an arc are given by σ_a and ω_a. The decision variable x_{atk} equals 1 if an arc a's traversal (and possibly service) starts at time t by vehicle k. Similarly, p_{at} equals 1 if arc a's service starts at time t.

$$max \sum_{a \in A_S} \sum_{t \in \mathcal{T}} s_{at} w_a, \tag{1}$$

$$\sum_{a \in A_S \cup A_T | \sigma_a \in D} x_{a,0,k} = 1, \qquad \forall k \in K, \tag{2}$$

$$\sum_{a \in A_S \cup A_T} x_{atk} \leq 1, \qquad \forall t \in T, \quad \forall k \in K, \tag{3}$$

$$\omega_a x_{atk} = \sum_{j \in A_S \cup A_T} x_{j,t+d_a} \sigma_j, \qquad \forall a \in A_S \cup A_T, \quad \forall t \in T, \quad \forall k \in K, \tag{4}$$

$$(1 - x_{atk})M \geq \sum_{j=t+1}^{t+d_a-1} x_{ajk}, \qquad \forall a \in A_S \cup A_T, \quad \forall t \in T, \quad \forall k \in K, \tag{5}$$

$$p_{at} = \sum_{k \in K} x_{atk}, \qquad \forall a \in A_S, \quad \forall t \in T, \tag{6}$$

$$l_{at} \geq l_{a,t-1} + i_{at} - p_{at} M, \qquad \forall a \in A_S, \quad \forall t \in T, \tag{7}$$

$$l_{at} \geq 0, \qquad \forall a \in A_S, \quad \forall t \in T, \tag{8}$$

$$l_{at} < H + (1 - s_{at})M, \qquad \forall a \in A_S, \quad \forall t \in T, \tag{9}$$

$$x_{atk} \in \{0, 1\}, \qquad \forall a \in A_S \cup A_T, \quad \forall t \in T, \quad \forall k \in K, \tag{10}$$

$$s_{at} \in \{0, 1\}, \qquad \forall a \in A_S, \quad \forall t \in T. \tag{11}$$

Equation (1) defines the objective function, while Constraints (2) force every vehicle to traverse an arc connected to a depot at time 0. Returning to the depot, however, is not required. At any time, a vehicle may only start the traversal of a single arc due to Constraints (3). Constraints (4) enforce vehicles to never be idle and to choose a connecting arc. If arc a's traversal started at time t, and therefore, finishes its traversal at $t+d_a$, a new traversal must start on an arc i, right where a ended. Constraints (5) guarantee that no additional traversals take place during this time; with $d_a - 1$ being sufficiently large big-Ms. The service of an arc is determined by Constraints (6), while the levels of snow are calculated using Constraints (7) and (8). Constraints (9) define the safety of the arcs. The largest value needed for the big-Ms in Constraints (7) for a specific arc a and time t is $M = l_{a0} + \sum_{n=0}^{t} i_{an}$, as it enables clearing the largest amount of snow that would be possible on arc a at time t. Similarly, for Constraints (9) with arc a and time t, a big-M of $M > l_{a0} + \sum_{n=0}^{t} i_{an} - - - H$ is sufficient. In addition, when only using a single big-M, the smallest sufficient value for both types of constraints would be $M = \max_{a \in A_S}\{l_{a0} + \sum_{t \in T} i_{at}\}$, which is the largest amount of snow that can accumulate on a single arc if no plowing is performed.

3.2 | Rolling horizon adaptation

If the problem is embedded within a rolling horizon framework, the following conditions are required:

- Vehicles do not necessarily start at the depot, but at the node they stopped previously (e_k).
- Instead of the time frame $\mathcal{T}$, a reduced time frame T_i' is considered. This reduced time frame consists of the updating interval T_i and the look-ahead E_i. All vehicles must end their movement within T_i'.
- Due to the look-ahead component, vehicles may be forced to first finish a previously started movement and, consequently, start later $\left(t_k^s\right)$. For example, a vehicle started with arc a in the previous subproblem during T_i but finished its movement in E_i.

These adaptions can be represented in the MIP by replacing Constraints (2)–(5) and Constraints (10) with the following constraints. All other constraints would replace the set T with T_i', as a reduced time frame T_i' would be used instead of the whole time frame.

$$\sum_{a \in A_S \cup A_T | \sigma_a = e_k} x_{a,t_k^s,k} = 1, \qquad \forall k \in K, \tag{12}$$

$$\sum_{a \in A_S \cup A_T} x_{atk} \leq 1, \qquad \forall t \in T_i' | t \geq t_k^s, \quad \forall k \in K, \tag{13}$$

$$\omega_a x_{atk} = \sum_{j \in A_S \cup A_T} x_{j,t+d_a,k} \sigma_j, \qquad \forall a \in A_S \cup A_T, \quad \forall t \in T_i' | t \geq t_k^s, \quad \forall k \in K, \tag{14}$$

$$(1 - x_{atk}) M \geq \sum_{j=t+1}^{t+d_a-1} x_{ajk}, \qquad \forall a \in A_S \cup A_T, \quad \forall t \in T_i' | t \geq t_k^s, \quad \forall k \in K, \tag{15}$$

$$x_{atk} = 0, \qquad \forall a \in A_S \cup A_T, \quad \forall t \in T_i' | t < t_k^s, \quad \forall k \in K, \tag{16}$$

$$x_{atk} \in \{0, 1\}, \qquad \forall a \in A_S \cup A_T, \quad \forall t \in T_i' | t \geq t_k^s, \quad \forall k \in K. \tag{17}$$

Constraints (12) and (16) handle the vehicle finishing any previous movement and starting after the arc of the previous movement. Constraints (13)–(15) and (17) are adapted to start after the previous movement is done ($\forall t \in T_i' | t \geq t_k^s$). The calculations of the snow level and penalties (cf. Constraints (6)–(9) and (11)) would not be influenced by the previous movement ($\forall t \in T_i'$).

The following stopping criteria might be applied for the rolling horizon optimization: (i) all streets are safe for a sufficient period of time, and (ii) the snowstorm has ended. For the first case, no additional steps are required. Vehicles would return to the depot(s); but due to costs not being considered in the model, the details of the return would not matter. For the second case, however, unsafe arcs must be plowed. As every arc requires at most one plowing operation during this phase, a different problem formulation would be required for the end of the horizon phase. Nevertheless, the rolling horizon framework could be used until all streets are cleared.

3.3 | Splitting of arcs

Some arcs cannot be traversed, as their traversal or service time is larger than T_i', whereas other arcs may be so long that they cannot be combined with any other arc. For example, consider three arcs with service durations of 1.5 times, 0.9 times, and 0.4 times the time frame T_i' (cf. the arcs on the left graph of Figure 1). The arc with a duration of $1.5T_i'$ clearly cannot be serviced

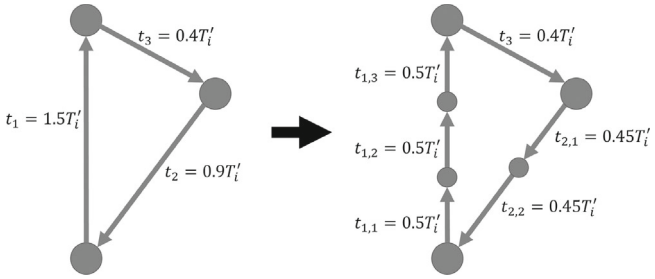


FIGURE 1 Illustration of the splitting of arcs. On the left side, t_1 could not be scheduled at all, while t_2 could not be combined with any other arc. On the right side, all arcs are split into smaller arcs with times of at most $T_i'/2$.

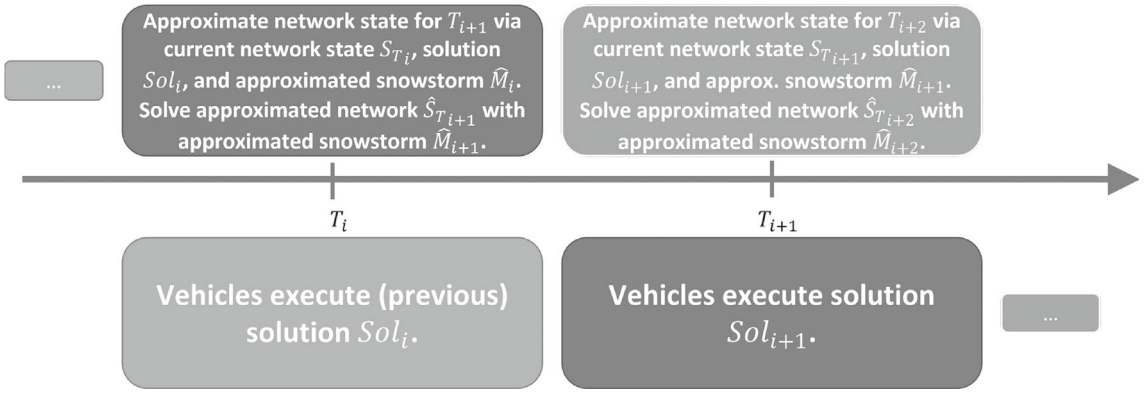


FIGURE 2 Updating process within the rolling horizon framework. The previous solution Sol_i is executed at the start of interval T_i and until the start of T_{i+1}. The next solution is determined with all available information.

within the time frame T_i'. The other two arcs could be serviced, but only individually. To enable the servicing of every arc in this problem and enable more combinations, they are remodeled into several smaller arcs, each within $T_i'/2$. Assuming an arc a has a traversal time t, the number n of required segments can be calculated by $n = \lceil 2t/T_i' \rceil$. The arc is then split into n arcs of equal length. For all arcs except the last one, the only successor is the directly following arc ($Succ(a_i) = \{a_{i+1}\} \forall i < n$). Note that within the updating framework, either all arcs $a_1, \ldots, a_n$ must be serviced consecutively or not serviced at all (e.g., service of a_2, but not a_1, is prohibited). Figure 1 provides an example with one arc of each type. While the arcs representing the arc that previously had a duration of $1.5T_i'$ still could not be serviced within T_i', the arc can now be serviced over two periods (e.g., first $t_{1,1}$ and $t_{1,2}$, and in the next period $t_{1,3}$). Similarly, the service of the arc with a duration $0.9T_i'$ could follow the arc with a duration $0.4T_i'$, by starting with the first segment in the first period and finishing the second segment in the second period.

3.4 | Information handling for the rolling horizon

Figure 2 and Table 2 illustrate the sequencing within the rolling horizon framework. During each updating interval T_i, while the solution Sol_{T_i} for the current period is executed (see Figure 2), solution $Sol_{T_{i+1}}$ for the next period is determined. Available data includes the current state of the system S_{T_i}, the solution Sol_{T_i} for the current period, and the previous snowstorm movements $M_{T_0}, \ldots, M_{T_{i-1}}$. In all, four steps are performed. First, the previous snowstorm movements are used to predict the upcoming snowstorm movements $\widehat{M}_{T_i}$ and $\widehat{M}_{T_{i+1}}$ (see step 1 in Table 2). Next, the current state and the solution for the current period are combined with the predicted snowstorm movements for the current period, resulting in the predicted starting state for the next period $\widehat{S}_{T_{i+1}}$ (see step 2 in Table 2). Based on the predicted starting state for the next period and the predicted snowstorm movements, the solution $Sol_{T_{i+1}}$ for the next time period is determined (see step 3 in Table 2). Finally, time passes to the next period and the real state $S_{T_{i+1}}$ and snowstorm movement M_{T_i} become known (see step 4 in Table 2).

Three criteria are used for predicting the snowstorm: angular speed, acceleration, and intensity change. After every iteration of the rolling horizon, the true movement of the snowstorm during the period is read and saved for the three criteria. Individual entries correspond to 1 s. Furthermore, entries that are more than 1000 s in the past are deleted, as their likelihood of still being correlated to the current diminishes. Based on the remaining entries, the expected values are calculated by moving averages for the three criteria. They are further used for the angular speed, acceleration, and intensity change of the predicted snowstorm.

Table 3 depicts a small example of such a prediction. Given are the measured centers of the snowstorm and snowstorm intensities over the first 4 time units. From them, the speed, angle, acceleration, angular speed, and intensity change are determined.

TABLE 2 Sequences of information becoming available within the rolling horizon framework.

Time	Step	Known	Predicted	Unknown	Step
T_{i-1}	...	...	...	...	...
T_i	1	$S_{T_i}, Sol_{T_i}, M_{T_0}, \ldots, M_{T_{i-1}}$	—	$Sol_{T_{i+1}}, S_{T_{i+1}}, M_{T_i}, M_{T_{i+1}}$	$predictMovement(M_{T_0}, \ldots, M_{T_{i-1}}) \rightarrow \{\widehat{M}_{T_i}\widehat{M}_{T_{i+1}}\}$
T_i	2	$S_{T_i}, Sol_{T_i}, M_{T_0}, \ldots, M_{T_{i-1}}$	$\widehat{M}_{T_i}, \widehat{M}_{T_{i+1}}$	$Sol_{T_{i+1}}, S_{T_{i+1}}$	$predictState(S_{T_i}, Sol_{T_i}, \widehat{M}_{T_i}) \rightarrow \widehat{S}_{T_{i+1}}$
T_i	3	$S_{T_i}, Sol_{T_i}, M_{T_0}, \ldots, M_{T_{i-1}}$	$\widehat{S}_{T_{i+1}}, \widehat{M}_{T_i}, \widehat{M}_{T_{i+1}}$	$Sol_{T_{i+1}}$	$performALNS(\widehat{S}_{T_{i+1}}, \widehat{M}_{T_{i+1}}) \rightarrow Sol_{T_{i+1}}$
T_i	4	$S_{T_i}, Sol_{T_i}, M_{T_0}, \ldots, M_{T_{i-1}}, Sol_{T_{i+1}}$	$\widehat{S}_{T_{i+1}}, \widehat{M}_{T_i}, \widehat{M}_{T_{i+1}}$	—	$timePasses() \rightarrow \{S_{T_{i+1}}, M_{T_i}\}$
T_{i+1}	1	$S_{T_{i+1}}, Sol_{T_{i+1}}, M_{T_0}, \ldots, M_{T_i}$	—	$Sol_{T_{i+2}}, S_{T_{i+2}}, M_{T_{i+1}}, M_{T_{i+2}}$	$predictMovement(M_{T_0}, \ldots, M_{T_i}) \rightarrow \{\widehat{M}_{T_{i+1}}\widehat{M}_{T_{i+2}}\}$
T_{i+1}	...	...	...	...	...

TABLE 3 Example for predicting the snowstorm.

	Measured values				Predicted values	
Time	1	2	3	4	5	6
Snowstorm's center	(0, 0)	(0, 1)	(1, 2)	(2, 3)	(3.5, 3.62)	(5.33, 3.62)
Speed	—	1	1.414	1.414	1.621	1.828
Angle	—	0°	45°	45°	67.5°	90°
Intensity	—	10	8	9	8.5	8
Acceleration	—	—	0.414	0	0.207	0.207
Angular speed	—	—	45°	0°	22.5°	22.5°
Intensity change	—	—	-2	1	-0.5	-0.5

The expected acceleration, angular speed, and intensity change are calculated by moving average (e.g., $(0.414+0)\div 2 = 0.207$ for the acceleration), and are then used to calculate the predicted values via linear equations (e.g., $45° + 22.5° \cdot 1 = 67.5°$). For calculating a snowstorm's center (x, y) based on speed s and angle g, circle equations are used; $(x, y) = (x_{old} + s\sin(g), y_{old} + s\cos(g))$. Arcs within a limited range of the predicted snowstorm centers (e.g., at most one kilometer away) are predicted to accumulate snow according to the intensities.

3.5 | Calculating changes in safety

While the calculation of weighted safety is not difficult, it can take some time should a straightforward approach be used. For every arc and every discretized time unit, a check whether the accumulated snow is too much must be performed. Should the solution be changed by inserting or removing sequences of arcs, any later parts of the solution would change as well due to being plowed later/earlier resulting in more/less snow than before. A straightforward, but slow approach would again be simply making a conditional check for every affected arc.

However, by preprocessing the benefits b_{at} of plowing an arc a at time t these calculations can be sped up significantly. Should multiple plowings of an arc be allowed during time frame T_i', such precalculations would be problematic, as the plowings would influence one another. For example, assume an unsafe arc, for which snow accumulates at a rate that it is unsafe again after 5 minutes. Individually, plowing it at $t = 3$ would keep it safe for 5 minutes, and plowing it at $t = 5$ would keep it safe for 5 minutes. When plowing it at $t = 3$ and at $t = 5$ it would be safe from $t = 3$ to $t = 10$; only 7 minutes. Fortunately, servicing every arc at most once in T_i' is a reasonable assumption given the relatively small time frames. If the arc is unsafe at time t, and enough snow for it to be unsafe again is accumulated at time t', the plowing has yielded a benefit of $b_{at} = w_a(t' - t)$. But if the arc was safe at the time of plowing and instead had become unsafe at t', the new time of it becoming unsafe t'' must be determined. This then yields the benefit of $b_{at} = w_a(t'' - t')$.

Those benefits can be used to efficiently calculate the benefit/detriment of changes to a route. For instance, assume a route with services at $t_1, \ldots, t_n$ on arcs $a_1, \ldots a_n$. The current benefit then is $\sum_{i=1}^{n} b_{a_i,t_i}$. When rerouting and inserting a new serviced arc a^* after arc a_j at time t^*, by replacing the path $\{a_j, b_1, \ldots, b_m, a_{j+1}\}$ with $\{a_j, c_1, \ldots, c_k, a^*, c_{k+1}, \ldots, c_l, a_{j+1}\}$, all following services are postponed by $\delta = d_{a^*} + \sum_{i=1}^{l} d_{c_i} - \sum_{i=1}^{m} d_{b_i}$. This results in the benefit of $\left(\sum_{i=1}^{j} b_{a_i,t_i}\right) + b_{a^*,t^*} + \left(\sum_{i=j+1}^{n} b_{a_i,t_i+\delta}\right)$. Should a consecutive sequence of service be inserted at $t_1^*, \ldots t_m^*$, the benefit would be $\left(\sum_{i=1}^{j} b_{a_i,t_i}\right) + \left(\sum_{i=1}^{m} b_{a_i^*,t_i^*}\right) + \left(\sum_{i=j+1}^{n} b_{a_i,t_i+\delta}\right)$.

While not directly related to these calculations, we also want to point out that early plowing of an arc a may decrease the benefit. For instance, assume a steady snowstorm causing arcs to be unsafe within 20 minutes, and two different plowing schedules. In the first hour-long schedule, arc a is plowed every 30 minutes, once at $t = 0$ and then at $t = 30$. This results in arc a being safe in the intervals $[0, 20]$ and $[30, 50]$; in total 40 minutes. In the second schedule, the second plowing is brought

forward to $t = 10$, resulting in a safe interval $[0, 30]$; in total 30 minutes. This prohibits schemes of optimizing the benefit by simply performing everything as soon as possible.

4 | SOLUTION METHODS

Due to the underlying problem's complexity, we propose to embed ALNS into the rolling horizon framework. ALNS is a renowned metaheuristic that was introduced by Pisinger and Ropke [14]. It has been applied to a multitude of related problems (e.g., [15-17]). To the best of our knowledge, we are the first to apply it to the introduced dynamic stochastic snow plowing problem. Additionally, we also propose problem-specific operators.

4.1 | Basic component: Adaptive large neighborhood search

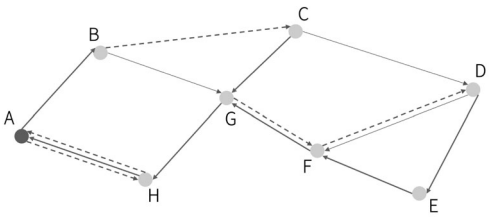
ALNS iteratively selects pairs of destroy and repair operators (d and r), which are applied to the incumbent solution s. The selection is based on scores, which are updated depending on the operator pairs' performance during the last interval of v iterations. Every pair begins each interval with a score of 1, which increases for a good performance (i.e., if an operator pair finds a new best solution s^* or a new incumbent solution s). After v iterations, the old scores are overwritten with the currently accumulated scores. The new solution s' is further improved by a Variable Neighborhood Descent (VND), which strengthens it further. An acceptance test using SA is performed to determine whether the solution is accepted as a new incumbent solution or not. SA includes a *temperature* and *cooling* parameters. As long as the temperature is high, the likelihood of accepting worse solutions is also high $\left(exp\left(\frac{currentSolutionValue - incumbentSolutionValue}{temperature}\right)\right)$. *Cooling*, on the other hand, indicates how much the temperature is lowered in each iteration. The whole process is iterated until it reaches the given stopping criterion. For the pseudocode, see Algorithm 1.

The following standard and problem-specific operators are used.

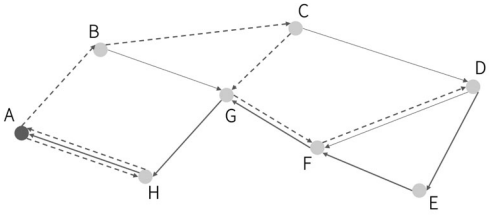
1. *[d_1] Random cluster*: Consecutive sequences of serviced arcs are randomly chosen and only traversed.
2. *[d_2] Random cluster and shorten*: Consecutive sequences of serviced arcs are randomly chosen and only traversed. If beneficial, paths between serviced arcs are replaced with the shortest path.
3. *[d_3] Greedy cluster*: Consecutive sequences of serviced arcs, whose traversal yields the greatest benefit, are only traversed.
4. *[d_4] Isolated*: Serviced arcs that are the most isolated from the previous and the following serviced arcs, are only traversed.
5. *[d_5] Individual saving cluster*: Consecutive sequences of serviced arcs, whose service, without considering other arcs, yields the smallest benefit, are only traversed.
6. *[r_1] Random*: Unserviced arcs are randomly chosen and assigned to a random vehicle. If the vehicle's current route already includes a traversal of the arc, service takes place at that time. Otherwise, *Random path change* is executed.

Algorithm 1. ALNS

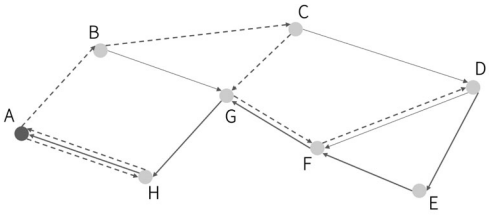
```
{s', s, s*} ←generateInitialSolution()
while stoppingCriterionNotReached do
    {d, r} ←selectDestroyAndRepairOperators(D, R, scores)
    s' ← r(d(s))
    s' ← VND(s')
    {s, s*, scores'} ←acceptanceTestViaSimulatedAnnealing(s', s, s*, scores', d, r)
    if iterationsWithoutImprovementSinceLastKick= w then
        s ←randomDestroyAndRepair(s)
        s ←VND(s)
    end if
    n ← n + 1
    if n = v then
        {scores, scores'} ←updateScores(scores, scores')
        n ← 0
    end if
end while
return s*
```



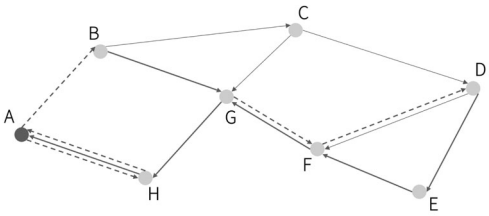
(A) Base route starting at A servicing: HAB, CG, DEFGH. [*Full route AHABCGFDEFGHA*]



(B) Removing a cluster of two consecutive services (AB & CG): HA, DEFGH. (Possible via d_1, d_3, d_5.) [*Full route AHABCGFDEFGHA*]



(C) Shortening the route by exchanging the current path from HA to DE with the shortest path. (Via d_2.) [*Full route AHABCDEFGHA*]



(D) Inserting a service between two consecutive services (HA & DE), which changes the path between them. (Possible via r_2, r_3, r_4.) [*Full route AHABGFDEFGHA*]

FIGURE 3 Illustration of possible changes to a route via the ALNS. The routes start and end at A. Bold lines are traversed and serviced, while dashed lines are only traversed. Thin lines are unused arcs.

7. *[r_2] Random path change*: Unserviced arcs are inserted into random routes at a random position by removing a sequence of pure traversals and reconnecting the route with two shortest paths.
8. *[r_3] Greedy insertion*: Unserviced arcs with the greatest insertion benefits are inserted at their best positions.
9. *[r_4] k-regret*: Unserviced arcs with the highest regret value, when comparing their best and k-best option (where $k \leq 3$), are inserted at their best positions. (e.g., the benefit of best option 100, the benefit of second best option 70, the benefit of third best option 20 $\Rightarrow$ 2-regret of 30, 3-regret of 80.)

Figure 3 illustrates some movements possible with these operators. From the initial solution in Figure 3A the operators d_1, d_3, and d_5 could remove two consecutive services (arcs AB & CG), by changing them to pure traversal (cf. Figure 3B). The route itself would not change through these operators. In contrast to d_1, d_2 could change the route. In Figure 3C, it not only removes arcs AB & CG, but also replaces the path between the services HA & DE with a shorter path (ABCD instead of

ABCGFD). All repair operators (except for r_1, should it not call r_2) can change the route as they insert services regardless of whether the arc was already part of the route. For example, in Figure 3D, where the service BG is added to the route between services HA & DE by replacing the path ABCD with ABGFD.

For all operators except the random ones (d_1, d_2, r_1, r_2), we include noise factors u, which we randomly draw from an interval $U = [-\psi, +\psi]$. The values used to select the arcs are multiplied by $1 + u$, which increases diversification.

In addition to the destroy and repair operators, we use VND operators that only change when an arc is only traversed or traversed as well as serviced. This is why these operators do not impact the routing. Further, due to limited computational effort, VND operators can be applied whenever a new solution is generated. The following operators are proposed.

1. *[v_1] Beneficial traversals to services*: A previously traversed arc is serviced.
2. *[v_2] Single arc traversal-service-switch on same route*: For an arc that was traversed multiple times during a route and also serviced, the service is moved to a different traversal (e.g., given traversals at t_1 and t_2 and service at t_1, the changes will bet that the service is scheduled for t_2 instead of t_1).
3. *[v_3] Single arc traversal-service-switch on different routes*: For an arc that was traversed multiple times and also serviced (on any route), the service is moved to a different traversal.
4. *[v_4] Two arcs traversal-service-switch on same route*: A previously serviced arc is only traversed, while a previously only traversed arc is serviced; both arcs are on the same route.

The operators are applied in the order listed above. Should an operator not find an improvement, the next operator is used (e.g., v_3 after v_2). However, if an improvement is found, the VND continues with v_1. The VND ends once v_4 was applied without any improvement. For the pseudocode, see Algorithm 2.

Figure 4 illustrates some movements possible with these operators. An initial solution is given in Figure 4A. Operator v_1 changes BG to be not only traversed but serviced as well (cf. Figure 4B). Operator v_2 causes HA, which was traversed twice and serviced during the first traversal, to be serviced on the second traversal instead (cf. Figure 4C). Lastly, operator v_4 causes the previously only traversed arc FD to now be serviced, while the inverse holds true for arc FG (cf. Figure 4D).

If no new best solution result is found for w iterations, operators d_1 and d_2 are applied to the incumbent solution several times before the solution is improved using the VND operators (see Algorithm 1, lines 7–10). Whether d_1 or d_2 is used, is determined randomly. The number of times it is applied depends on the iterations since the last best solution was found, and this is also decided randomly (e.g., if it was found recently, it will be applied once; but if the last best is very far in the past, it will be applied 3 to 5 times).

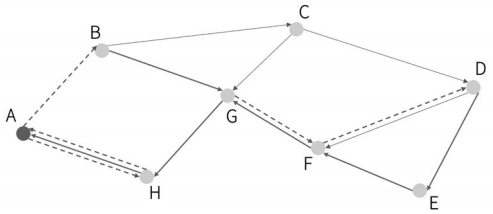
The algorithmic parameters (see Table 4) were set similarly to previous work of Fröhlich et al. [5]. Scores were awarded for finding (i) a new best solution, (ii) a solution better than the current incumbent solution but not a new best solution, and (iii) a solution worse than the current incumbent but sufficiently good to be accepted via SA.

4.2 | Basic component: Rolling horizon framework

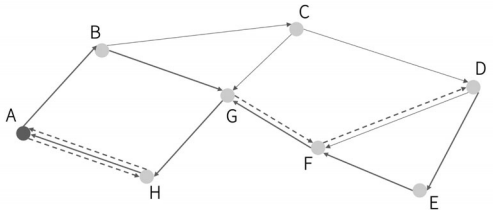
Given a total time horizon $\mathcal{T}$, the rolling horizon solves subproblems, each with a time horizon of T (equaling the updating interval) and an extra horizon E (equaling the look-ahead). The steps are identical to Section 3.4 for all but the first iteration, which requires a slight modification. First, the previously measured snowstorm movements ($M_{-1}, \ldots, M_{i-1}$) are used to predict the next snowstorm movements ($\widehat{M}_i$ and $\widehat{M}_{i+1}$). Then the next predicted state $\widehat{S}_{i+1}$ is created from the current state S_i, the expected snowstorm movement $\widehat{M}_i$ and solution Sol_i for this updating interval. Based on the next predicted state $\widehat{S}_{i+1}$ and snowstorm movement $\widehat{M}_{i+1}$, the next solution Sol_{i+1} is created using ALNS. At the end of the updating interval, the current state is updated,

Algorithm 2. VND

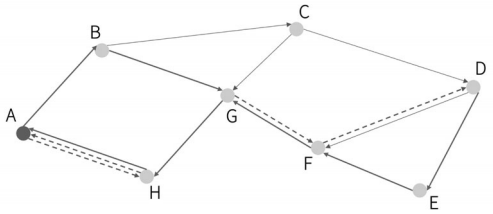
```
1:  i ← 0
2:  while i ≠ 4 do
3:      {s, improvementBoolean} ←performNeighborhood(s, i)
4:      if improvementBoolean then
5:          i ← 0
6:      else
7:          i ← i + 1
8:      end if
9:  end while
10: return s
```



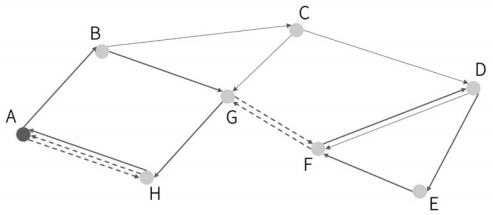
(A) Base route. [*Full route AHABGFDEFGHA*]



(B) Including the service of AB, which was previously just traversed. (Via v_1.) [*Full route AHABGFDEFGHA*]



(C) Servicing HA at the end of the route instead of at the beginning. (Via v_2.) [*Full route AHABGFDEFGHA*]



(D) Including the service of FD, which was previously just traversed, while not servicing FG and instead only traversing it. (Via v_4.) [*Full route AHABGFDEFGHA*]

FIGURE 4 Illustration of possible changes to a route via the VND. The routes start and end at A. Bold lines are traversed and serviced, while dashed lines are only traversed. Thin lines are unused arcs.

TABLE 4 List of algorithmic parameters.

Parameter	Values
Noise ψ	0.05
Scores for ALNS	{10, 3, 1}
Iterations before updating scores ν	40
Iterations without improvement before random destroy-and-repair w	1000
Initial temperature	Initial solution value
Cooling factor SA	0.975
Termination criterion (time in seconds)	3600

Algorithm 3. Rolling horizon framework

$\{\widehat{M}_{-1}, \widehat{M}_0\} \leftarrow$predictInitialSnowstormMovements()
$\widehat{S}_0 \leftarrow$createInitialPredictedState$(S_{-1}, \widehat{M}_{-1})$
$Sol_0 \leftarrow$ALNS$(\widehat{S}_0, \widehat{M}_0)$
$S_0 \leftarrow$updateState(S_{-1}, M_{-1})
$i \leftarrow 0$
while $i + 1 < \mathcal{T}/T$ **do**
  $\{\widehat{M}_i, \widehat{M}_{i+1}\} \leftarrow$updateAndCreateNextPredictedMovements$(M_{-1}, \dots, M_i)$
  $\widehat{S}_{i+1} \leftarrow$createNextPredictedState$(S_i, \widehat{M}_i, Sol_i)$
  $Sol_{i+1} \leftarrow$ALNS$(\widehat{S}_{i+1}, \widehat{M}_{i+1})$
  $S_{i+1} \leftarrow$updateState(S_i, M_i, Sol_i)
end while

TABLE 5 Benchmarking ALNS against the optimal results available in [2].

Instance set	Average ratio	Standard deviation	Worst-case ratio	# of optimal solutions	# of opt. sol. in all runs
Hertz_p0_d_2	98.73%	1.37%	95.39%	11 from 27	7 from 27
Hertz_p0_d_3	98.19%	1.78%	94.18%	8 from 27	7 from 27

by using the actual snowstorm movement M_i, instead of the prediction $\widehat{M}_i$. The deviations for the first iteration are that there might not be any snowstorm data available for predicting the movements and that no solution is available for creating the predicted state and updating the actual state. The pseudocode is provided in Algorithm 3. More details on the generation of predictions are provided in Section 3.4. If all the streets are safe, the vehicles can return to the depot.

5 | COMPUTATIONAL STUDY

Our computational study, which was performed on the Vienna Scientific cluster 4 (3.1 GHz), consists of two parts. First, we compare the proposed ALNS to the optimal results on benchmark instances. Second, we present a newly generated set of real-world-based instances, which we use to derive computational and managerial findings.

5.1 | Results on benchmark instances

To validate the efficacy of the proposed ALNS, we apply it to the team orienteering arc routing problem (TOARP), as this problem class is arc-based and does not require all arcs to be visited. The main difference is its need for vehicles to return to the depot and a time-independent reward for service. However, as the time-independence is similar to an arc starting with snow but not being further affected by the snowstorm, this characteristic is covered by our proposed ALNS. Therefore, only the need to return to the depot is added.

All experiments are conducted with a runtime of 1 h on the *Hertz_p0_d_2* and *Hertz_p0_d_3* instances, which have either two or three vehicles and networks with 96–846 arcs, including 10–121 profitable arcs. We benchmark the proposed ALNS against the optimal results available in Archetti et al. [2], who also had a time limit of 1 h. For each instance, we perform five runs. Table 5 lists the average percentage ratios (i.e., we divide our results by the state-of-the art result, which are optimal solutions in most cases), standard deviations, worst-case ratios, and the number of instances solved to optimality.

We observe an average ratio of 98.46% (i.e., an average percentage gap of less than 1.6%), and a worst-case ratio of about 95%. More detailed results are available in Table A1 in the Appendix. Furthermore, we examine the ratios after 30 s, as this is the smallest updating interval for the rolling horizon framework. Even for the most difficult instances, an average ratio of 94.02% is achieved.

5.2 | Real-world-based instances

To gain computational and managerial insights, we generate a new set of instances based on data from the city of Vienna. In the following sections, we first describe the details of data generation and then provide detailed results and discussions.

TABLE 6 List of parameters.

Parameter	Values
Updating interval	30, 60, 120, 300
Look-ahead	50, 100, 200, 300
Instance	vienna_1, … , vienna_5
Snowstorm movement	snow_1, … , snow_4
Prediction	Normal, perfect

TABLE 7 Information on real-world-based instances.

	Arcs when split to be equal or less				
Instance	∞	300	160	80	40
vienna_1	1778	1826	1925	2194	2763
vienna_2	499	509	521	595	739
vienna_3	1670	1675	1727	1900	2251
vienna_4	1469	1469	1472	1526	1647
vienna_5	1214	1246	1296	1465	1855
vienna_6	3637	3668	3765	4183	5180
vienna_7	2498	2567	2725	3259	4495
vienna_8	3828	3840	3884	4084	4795

As the instances are far too large for the MIP-formulation to deliver results within a reasonable time frame, no comparison to (near) optimal solutions are performed. The instances are made available at https://bda.univie.ac.at/research/data-and-instances/vehicle-routing-problems/.

5.2.1 | Instance generation

We generate eight instances based on the street network of Vienna (denoted as *vienna*-instances); five smaller ones, and three larger ones. For each district, the set of arcs S initially consists of all arcs starting or ending in a district. However, as Vienna has many one-way streets, S would only be partially connected and some arcs or whole sets may be unreachable. Hence, for every arc a in S that could not be reached when using only S, the shortest path P between any other arc in S and a using the whole street network of Vienna is inserted ($S := S \cup P$). We use Cartesian coordinates to calculate the arcs' centers and normalize them to have 0 as the smallest value (e.g., the smallest x-coordinate in S is 2000; therefore, the x-coordinate of 2500 is normalized to 500). Traversal times are set according to the length and speed limits.

Further, snowstorm movements are generated according to different shapes (straight line (*straight*), wavy line (*snake*), full circle (*clock*), and half circle that is backtracked (*half-half*)). Figure A1 in the appendix depicts the movements. Detailed results are not provided as the shape of the snowstorm did not lead to significantly different findings.

5.2.2 | Computational results

For each setting, consisting of updating interval, look-ahead, instance, snowstorm movement, and prediction type, we conduct 8 runs of the proposed rolling horizon framework with a duration of 3600 s (e.g., the problem is split into 120 subproblems, when the updating interval is 30 s, and 12 subproblems, when the updating interval is 300 s). Table 6 lists the applied parameter settings. Table 7 lists the number of arcs for the *vienna*-instances, when performing the splitting according to Section 3.3. However, as all 16 combinations of updating intervals and look-aheads would have been too confusing, we only list four combinations. Instances *vienna_6*—*vienna_8* were used in later tests, to examine whether observations change for larger instances. *Normal* prediction corresponds to the procedure described in Section 3.4, while *perfect* prediction simply reads the input file for the snowstorm and replicates it, and thus has no prediction errors.

Tables 8–11 report the results for the binomial tests that determine whether the results show significant differences based on the applied setting. Detailed results for different movements of snowstorms are provided in Tables A2 and A3 in the Appendix. Tables 8 and 9 compare the influence of the updating interval. Their binomial tests had identical look-aheads (e.g., a comparison of $T = 30$, $E = 50$ with $T = 60$, $E = 50$; but not a comparison of $T = 30$, $E = 50$ with $T = 60$, $E = 100$). The first row denotes the updating interval and the following rows contain the results for different instances. Every result x/y lists the number of snowstorm movements, for which the base setting yields significantly better (x) and significantly worse (y) according to

TABLE 8 Aggregated results for the binomial test comparing updating intervals, when *predicting* snowstorm movements.

	$T = 30$ compared to			$T = 60$ compared to			$T = 120$ compared to			$T = 300$ compared to		
	$T = 60$	$T = 120$	$T = 300$	$T = 30$	$T = 120$	$T = 300$	$T = 30$	$T = 60$	$T = 300$	$T = 30$	$T = 60$	$T = 120$
vienna_1	0/4	0/1	0/4	4/0	0/0	0/3	1/0	0/0	0/1	4/0	3/0	1/0
vienna_2	0/4	0/4	0/4	4/0	0/4	0/3	4/0	4/0	0/2	4/0	3/0	2/0
vienna_3	0/4	0/4	0/4	4/0	0/4	0/4	4/0	4/0	0/4	4/0	4/0	4/0
vienna_4	0/4	0/4	0/4	4/0	0/4	0/4	4/0	4/0	0/0	4/0	4/0	0/0
vienna_5	0/4	0/4	0/4	4/0	0/4	0/3	4/0	4/0	0/0	4/0/	3/0	0/0
Total	**0/20**	**0/17**	**0/20**	**20/0**	**0/16**	**0/17**	**17/0**	**16/0**	**0/7**	**20/0**	**17/0**	**7/0**

Note: The first number indicates the number of times a setting performed significantly better ($\alpha = 5\%$) than the other setting; the second number of times is performed significantly worse.

TABLE 9 Aggregated results for the binomial test comparing updating intervals, when *perfectly predicting* snowstorm movements.

	$T = 30$ compared to			$T = 60$ compared to			$T = 120$ compared to			$T = 300$ compared to		
	$T = 60$	$T = 120$	$T = 300$	$T = 30$	$T = 120$	$T = 300$	$T = 30$	$T = 60$	$T = 300$	$T = 30$	$T = 60$	$T = 120$
vienna_1	0/4	0/4	0/4	4/0	0/4	0/4	4/0	4/0	0/4	4/0	4/0	4/0
vienna_2	0/4	0/4	0/4	4/0	0/4	0/4	4/0	4/0	0/4	4/0	4/0	4/0
vienna_3	0/4	0/4	0/4	4/0	0/4	0/4	4/0	4/0	0/4	4/0	4/0	4/0
vienna_4	0/4	0/4	0/4	4/0	0/3	0/4	4/0	3/0	0/4	4/0	4/0	4/0
vienna_5	0/1	0/4	0/4	1/0	0/4	0/4	4/0	4/0	0/4	4/0	4/0	4/0
Total	**0/17**	**0/20**	**0/20**	**17/0**	**0/19**	**0/20**	**20/0**	**19/0**	**0/20**	**20/0**	**20/0**	**20/0**

Note: The first number indicates the number of times a setting performed significantly better ($\alpha = 5\%$) than the other setting; the second number of times is performed significantly worse.

TABLE 10 Aggregated results for the binomial test comparing look-aheads, when *predicting* snowstorm movements.

	$E = 50$ compared to			$E = 100$ compared to			$E = 200$ compared to			$E = 300$ compared to		
	$E = 100$	$E = 200$	$E = 300$	$E = 50$	$E = 200$	$E = 300$	$E = 50$	$E = 100$	$E = 300$	$E = 50$	$E = 100$	$E = 200$
$T = 30$	0/0	4/0	4/0	0/0	4/0	4/0	0/4	0/4	4/0	0/4	0/4	0/4
$T = 60$	0/1	4/0	4/0	1/0	4/0	4/0	0/4	0/4	4/0	0/4	0/4	0/4
$T = 120$	0/4	0/0	3/0	4/0	0/0	4/0	0/0	0/0	3/0	0/3	0/4	0/3
$T = 300$	0/3	0/3	0/2	3/0	0/4	0/1	3/0	4/0	3/0	2/0	1/0	0/3

Note: The first number indicates the number of times a setting performed significantly better ($\alpha = 5\%$) than the other setting; and the second refers to the number of times a setting performed significantly worse.

binomial tests (e.g., 0/2 in Table 8 for vienna_2 denotes that the updating interval of 120 never performs significantly better, but performs significantly worse than the updating interval of 300 in 2 cases; thus, in the remaining 2 cases, there is no significant difference). Tables 10 and 11 compare the effect of a look-ahead with different fixed updating intervals (30, 60, 120, and 300). For Tables 8 and 9, every binomial test consists of 32 comparisons (4 different look-aheads with 8 runs each). For Tables 10 and 11, every binomial test consists of 40 comparisons (5 different instances with 8 runs each).

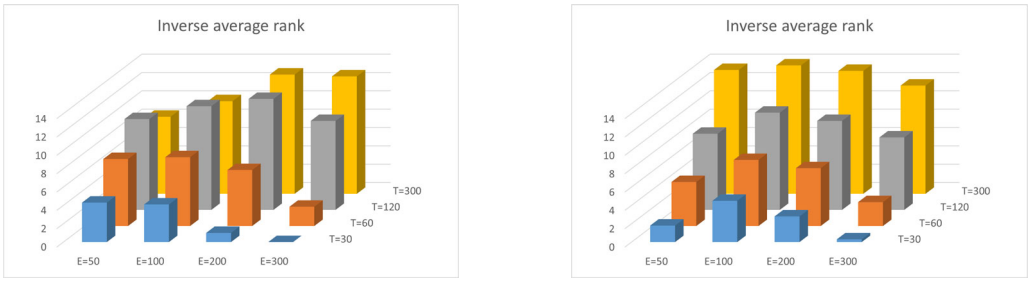
Table 8 shows a clear picture of shorter updating intervals performing worse than the larger ones, when predictions are imperfect. This is an overall trend, but it is strongest when $T = 30$ and $T = 60$ are compared to the larger intervals. Further, $T = 30$ and $T = 60$ are significantly worse in 57 tests out of 60 (95%) and 33 tests out of 40 tests (82.5%), respectively. The effect decreases between $T = 120$ and $T = 300$ with 7 tests out of 20 (35%) being significantly worse. We explain this observation by the rolling horizon framework effectively splitting the problem into smaller subproblems. Another aspect to be considered is that a smaller T results in less computation time for each problem. As Table 9 shows, these findings are similar, when the snowstorm movement is known ahead of time/predicted perfectly (57 out of 60 tests for $T = 30$ and 39 out of 40 tests for $T = 120$). The exception is $T = 300$, compared to $T = 120$, performing even better than previously, as it has significantly better results for all 20 binomial tests.

Table 10 shows a clear trend that the look-ahead should increase with the updating interval, when predictions are imperfect. For small T, the small look-aheads $E = 50$ and $E = 100$ perform best, as they dominate $E = 200$ and $E = 300$ in all tests. This can be explained by the computational effort required for long look-aheads. However, $E = 100$ performs better than $E = 50$

TABLE 11 Aggregated results for the binomial test comparing look-aheads, when *perfectly predicting* snowstorm movements.

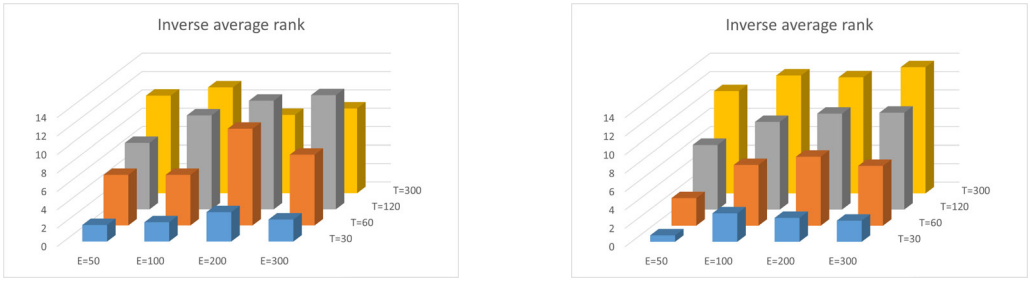
	$E = 50$ compared to			$E = 100$ compared to			$E = 200$ compared to			$E = 300$ compared to		
	$E = 100$	$E = 200$	$E = 300$	$E = 50$	$E = 200$	$E = 300$	$E = 50$	$E = 100$	$E = 300$	$E = 50$	$E = 100$	$E = 200$
$T = 30$	0/4	0/0	4/0	4/0	4/0	4/0	0/0	0/4	4/0	0/4	0/4	0/4
$T = 60$	0/4	0/1	4/0	4/0	2/0	4/0	1/0	0/2	4/0	0/4	0/4	0/4
$T = 120$	0/4	0/0	3/0	4/0	3/0	4/0	0/0	0/3	3/0	0/3	0/4	0/3
$T = 300$	0/1	2/0	3/0	1/0	2/0	3/0	0/2	0/2	0/0	0/3	0/3	0/0

Note: The first number indicates the number of times a setting performed significantly better ($\alpha = 5\%$) than the other setting; and the second refers to the number of times a setting performed significantly worse.



(A) Results for small instances with imperfect prediction. (*Top 3:* 300/200, 300/300, 120/200

(B) Results for small instances with perfect prediction. (*Top 3:* 300/100, 300/50, 300/200



(C) Results for large instances with imperfect prediction. (*Top 3:* 120/300, 120/200, 300/100

(D) Results for large instances with perfect prediction. (*Top 3:* 300/300, 300/100, 300/200

FIGURE 5 Inverse average ranks for different settings of updating interval T and look-ahead E. The best and worst possible values are 15 and 0, respectively.

when T increases to 60 and 120, and $E = 200$ dominates in case of very long updating intervals ($T = 300$). These observations can be explained by the following: (i) a lower increase in problem size—and therefore computational effort—compared to smaller T (e.g., an increase by 71.4% for $T = 300$ when extending $E = 50$ to $E = 300$), which is outweighed by (ii) larger look-aheads enabling less myopic solutions, and (iii) smoother transitions between sub-problems occurring due to more arcs being available as the "connecting arc" (the last arc starting within the updating interval and ending within the look-ahead).

Table 11 shows slightly different results when predictions are perfect. The small look-ahead ($E = 100$) proves best for all updating intervals. However, this effect diminishes (e.g., it is significantly better for $T = 30$ in 12 out of 12 tests, but it is only better in 6 out of 12 tests for $T = 300$).

To validate the results, we increase the available computation time by 900% (e.g., a setting with $T = 30$ and $E = 120$ runs with 300 instead of 30 s). Regarding updating intervals, the observed trends are supported for $T = 30$ and $T = 60$. Hence, myopic behavior leads to poor solutions. However, the trend of $T = 300$ proving better than $T = 120$ (as it performed significantly better in 7 out of 20 settings and never worse, previously) does not hold as there is even an instance, where $T = 120$ performs better. This suggests that once enough computation time is available, prediction errors can be sufficiently large enough to result in smaller updating intervals (in this case $T = 120$) performing better. Regarding look-aheads, the additional computation time did not bring new insights. All observations, particularly the fact that larger look-aheads are more suitable for larger updating intervals when predictions are imperfect, are confirmed.

To further examine whether observations change with problem size, we conduct experiments on large instances with 2498–3828 arcs (before splitting). Note that *vienna_1* to *vienna_5* have 499–1778 arcs before splitting. Once more, larger updating intervals perform significantly better, especially when using perfect information.

For $T = 30$ and $T = 60$, without perfect forecasting of the snowstorm, the look-ahead $E = 200$ performs best. However, when using $T = 300$, a trend toward shorter look-aheads is noticeable. This can be explained by the problem becoming too large when using larger look-aheads, as a similar effect was already observed for the largest instance of $T = 200$ and $T = 300$.

Results are depicted in Figure 5, where we show the performance of individual settings on the smaller and larger instances when using imperfect and perfect predictions. The height of the bars in each subfigure corresponds to the inverse average rank of the respective pair of updating interval and look-ahead. For instance, if a pair yields the best average performance in 18 out of 20 aggregated runs, and the second best performance in the remaining runs, it results in the average rank of $(1 \cdot 18 + 2 \cdot 2)/20 = 1.1$ and, as 16 pairs are compared, an inverse average rank of $16 - 1.1 = 14.9$. For perfect predictions, the updating interval is clearly the most impactful parameter, as in Figure 5B,D, the four best average ranks were achieved by $T = 300$. For imperfect predictions (see Figure 5A,C), the updating interval still seems to be more important. However, a poorly chosen look-ahead can be detrimental. When instances are large, the sum of updating interval and look-ahead should not be too large (see Figure 5C). However, when instances are small, both large updating intervals and look-aheads yield good results. Lastly, we want to point out that these findings depend on the method's relative efficacy. A more effective method might find larger instances to deliver results more similar to smaller instances, and vice versa for a less effective method. Nonetheless, should these findings prove valuable for a decision maker appropriately assessing their method.

6 | CONCLUSION

While a large body of literature on problems related to snow plowing does exist, a gap regarding more efficient and adaptive methods exists, as well. Possible reasons for this may be the technical difficulties in predicting the street conditions. However, as measuring systems and data communication have improved and become more affordable, such adaptive methods have become of interest.

In this study, we examined a novel snow plowing problem based on several challenging characteristics, such as ongoing stochastic snowfall, the necessity for arcs to be plowed repeatedly, and a time-dependent objective function where earlier service could be detrimental. As these characteristics make it difficult to apply exact methods, we tackled the problem using a rolling horizon framework with embedded ALNS. The ALNS was benchmarked against optimal solutions on available TOARP instances.

Our computational study on real-world-based instances examined the effect of updating intervals as well as the effect of look-aheads within the rolling horizon framework. The results suggested that the loss of optimality is low when the horizon is split into larger parts, but that larger problems suffer from imperfect predictions that lead to inefficient moves. The derived results also pointed to the importance of sufficient look-aheads, as well as to the fact that the connection of individual horizons is important. Further, managerial findings included that large look-aheads are more suited for large updating intervals, and vice versa. Additionally, they concluded that small updating intervals should not be paired with very large look-aheads, even when sufficient computational time is available.

Future improvements of the model can be made by adding other common snow plowing considerations such as turn restrictions or heterogeneous fleets. Since the necessity for turn restrictions typically arises from street conditions, they should be assumed to be time-dependent. This would not only make the problem more challenging, but it could also render some insertions or removals infeasible. Another future research direction would be examining different partitioning schemes for the rolling horizon framework.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Georg E. A. Fröhlich https://orcid.org/0000-0002-6653-5820
Karl F. Doerner https://orcid.org/0000-0001-8350-1393

REFERENCES

[1] C. Ahabchane, A. Langevin, and M. Trepanier, *Robust optimization for the hierarchical mixed capacitated general routing problem applied to winter road maintenance*, Comput. Ind. Eng. **158** (2021), 107396.

[2] C. Archetti, M. G. Speranza, Á. Corberán, J. M. Sanchis, and I. Plana, *The team orienteering arc routing problem*, Transp. Sci. **48** (2014), no. 3, 442–457.
[3] R. A. Castro Campos, C. A. Rodriguez Villalobos, and F. J. Zaragoza Marinez, *Plowing with precedence in polynomial time*, Networks **76** (2020), 451–466.
[4] B. Dussault, B. Golden, C. Groer, and E. Wasil, *Plowing with precedence: A variant of the windy postman problem*, Comput. Oper. Res. **40** (2013), 1047–1059.
[5] G. E. A. Fröhlich, K. F. Doerner, and M. Gansterer, *Secure and efficient routing on nodes, edges, and arcs of simple-graphs and of multi-graphs*, Networks **76** (2020), no. 4, 431–450.
[6] L. Hajibabai and Y. Ouyang, *Dynamic snow plow fleet management under uncertain demand and service disruption*, IEEE Trans. Intell. Transp. Syst. **17** (2016), no. 9, 2574–2582.
[7] R. Hajizadeh and K. Holmberg, *A branch-and-dive heuristic for single vehicle snow removal*, Networks **76** (2020), 509–521.
[8] K. Holmberg, *The (over)zealous snow remover problem*, Transp. Sci. **53** (2019), no. 3, 867–881.
[9] 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*, Comput. Oper. Res. **33** (2006), no. 1, 209–238.
[10] 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*, Comput. Oper. Res. **33** (2006), no. 1, 239–262.
[11] 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*, Comput. Oper. Res. **34** (2007), no. 1, 211–257.
[12] 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*, Comput. Oper. Res. **34** (2007), no. 1, 258–294.
[13] N. Perrier, A. Langevin, and C.-A. Amaya, *Vehicle routing for urban snow plowing operations*, Transp. Sci. **42** (2008), no. 1, 44–56.
[14] D. Pisinger and S. Ropke, *A general heuristic for vehicle routing problems*, Comput. Oper. Res. **34** (2007), no. 8, 2403–2435.
[15] O. Quirion-Blais, A. Langevin, and M. Trepanier, *A case study of combined winter road snow plowing and de-icer spreading*, Can. J. Civ. Eng. **44** (2017), no. 12, 1005–1013.
[16] O. Quirion-Blais, A. Langevin, F. Lehuede, O. Peton, and M. Trepanier, *Solving the large-scale min-max k-rural postman problem for snow plowing*, Networks **70** (2017), no. 3, 195–215.
[17] M. A. Salazar-Aguilar, A. Langevin, and G. Laporte, *Synchronized arc routing for snow plowing operations*, Comput. Oper. Res. **39** (2012), 1432–1440.
[18] M. Scott, *Montreal snow clearing costs went up by $10 million in 2018*, Montreal Gaz. **53** (2019), no. 3 https://montrealgazette.com/news/local-news/snow-clearing-costs-up-by-10-million.
[19] S. Xu and T. J. Kwon, *Optimizing snowplow routes using all-new perspectives: Road users and winter road maintenance operators*, Can. J. Civ. Eng. **48** (2021), 959–968.

How to cite this article: G. E. A. Fröhlich, M. Gansterer, and K. F. Doerner, *A rolling horizon framework for the time-dependent multi-visit dynamic safe street snow plowing problem*, Networks. (2023), 1–20. https://doi.org/10.1002/net.22189

APPENDIX A

Table A1 lists the $\left(\frac{OurResult}{LiteratureResult}\right)$ ratios of individual runs for *Hertz_p0_d2* and *Hertz_p0_d3* instances, which include 2 and 3 vehicles, respectively. The networks have 96–846 arcs including 10–121 profitable arcs.

Figure A1 depicts the 4 types of snowstorm movements. We tried more realistic movements and also more severe ones, to test whether any effect become be visible. Movements *straight* and *snake* seem to be the most natural ones. For *straight*, the wind would not turn at all, while for *snake*, the wind would gradually change its direction and then gradually change its direction back into the original direction. The movement *half-half* might seem natural, when looking at the figure, but after it has moved in a half-circle once, it perfectly backtracks the half-circle. It, therefore, assumes that the wind gradually turns right, then makes a 180-degree turn, and finally gradually turns left. Similarly atypical, *clock* would be described by a wind that is permanently turning into the same direction.

Tables A2 and A3 list the results for binomial tests, when having imperfect predictions. In Table A2, the influence of the updating interval on different snowstorm movements and instances is compared. The first row denotes the comparison of updating intervals. The following rows contain similar results for different instances. Every four rows correspond to one instance (vienna_1, … , vienna_5). The superscript $^+$ denotes that the base setting has performed significantly better, for example, for the *clock*-movement, the $T = 60$ setting performs better than $T = 30$. Table A3 compares the effect of a look-ahead with different fixed updating intervals.

Table A2 clearly shows smaller updating intervals performing worse than larger updating intervals. The results are consistent for *vienna_2*-*vienna_5* and also do not significantly deviate for specific snowstorm movements. For *vienna_1*, which is the largest instance of the five, updating interval $T = 120$ delivers mixed results, which were better on average, but not generally dominating.

TABLE A1 Results for benchmark tests on TOARP instances using the proposed ALNS.

Instance	Run 1	Run 2	Run 3	Run 4	Run 5	Instance	Run 1	Run 2	Run 3	Run 4	Run 5
hertzd10_d2	1	1	1	1	1	hertzd10_d3	1	1	1	1	1
hertzd11_d2	1	1	1	1	1	hertzd11_d3	1	1	1	1	1
hertzd12_d2	1	1	1	1	1	hertzd12_d3	0.9762	0.9762	0.9762	0.9762	0.9762
hertzd13_d2	1	1	1	1	1	hertzd13_d3	1	1	1	1	1
hertzd14_d2	1	1	1	1	1	hertzd14_d3	1	1	1	1	1
hertzd15_d2	0.9758	1	0.9653	0.9811	0.9811	hertzd15_d3	1	1	1	1	1
hertzd16_d2	0.9811	0.9811	0.9811	0.9811	0.9811	hertzd16_d3	0.9957	0.9957	0.9957	0.9957	0.9957
hertzd17_d2	1	1	1	1	0.9951	hertzd17_d3	0.9552	0.9552	0.9552	0.9552	0.9552
hertzd18_d2	0.9907	0.9907	0.9907	0.9907	0.9907	hertzd18_d3	0.9844	0.9932	0.9932	0.9932	0.9932
hertzd19_d2	1	0.9785	1	0.9785	1	hertzd19_d3	0.9886	0.9886	0.9943	0.9943	0.9943
hertzd20_d2	0.9795	0.9795	0.9966	0.9966	0.9795	hertzd20_d3	0.9953	0.9953	0.9953	1	1
hertzd21_d2	1	1	1	1	1	hertzd21_d3	0.9514	0.9514	0.9514	0.9514	0.9514
hertzd22_d2	0.9728	0.9728	0.9728	0.9747	0.9728	hertzd22_d3	0.986	0.9686	0.9674	0.9942	0.9686
hertzd23_d2	0.9981	0.9981	0.9981	0.9981	0.9907	hertzd23_d3	1	1	1	1	1
hertzd24_d2	0.9983	0.9983	0.9983	0.994	0.9983	hertzd24_d3	0.9537	0.966	0.9528	0.9528	0.9537
hertzd25_d2	1	0.981	0.9744	0.9744	0.9905	hertzd25_d3	0.9898	0.9898	0.9898	0.9898	0.9898
hertzd26_d2	0.9969	0.9939	0.9977	0.9939	0.9969	hertzd26_d3	1	1	1	1	1
hertzd27_d2	0.98	0.9736	0.9703	0.9736	0.9714	hertzd27_d3	0.9726	0.975	0.9726	0.9726	0.975
hertzd28_d2	0.9762	0.9955	0.9955	0.9955	0.9762	hertzd28_d3	0.9713	0.9713	0.9809	0.974	0.9809
hertzd29_d2	0.9981	0.9981	0.9893	0.9981	0.9845	hertzd29_d3	0.945	0.9881	0.9916	0.9904	0.9916
hertzd30_d2	0.9732	0.9844	0.9844	0.9732	0.9732	hertzd30_d3	0.9565	0.9565	0.9565	0.9565	0.9552
hertzd31_d2	0.9606	0.9606	0.9606	0.9619	0.9606	hertzd31_d3	0.9926	0.989	0.9845	0.9948	0.9897
hertzd32_d2	1	1	1	1	1	hertzd32_d3	0.9859	0.9792	0.9933	0.9963	0.9908
hertzd33_d2	0.9762	0.9612	0.9721	0.9721	0.9721	hertzd33_d3	0.9775	0.9628	0.9675	0.9775	0.9628
hertzd34_d2	0.9539	0.9544	0.9544	0.9539	0.9539	hertzd34_d3	0.9676	0.9878	0.9692	0.9708	0.9718
hertzd35_d2	0.9737	0.9761	0.9972	0.9681	0.9756	hertzd35_d3	0.9596	0.9444	0.9633	0.9523	0.9418
hertzd36_d2	0.9805	0.9805	0.9921	0.9782	0.994	hertzd36_d3	0.956	0.9513	0.9658	0.9902	0.9814

Note: We provide ratios found within 1 h run times.

Table A3 shows not only that smaller look-aheads are desirable for smaller updating intervals, but also the independence of these results from the snowstorm movement. For the two smallest updating intervals, small look-aheads ($E = 50$ and $E = 100$) are desirable for all snowstorm movements, as all entries show that they perform significantly better. For $T = 120$, the look-ahead of $E = 100$ still performs significantly better than the other setting, independent from the snowstorm movement. Similar effects can be observed for $T = 300$ and $E = 200$.

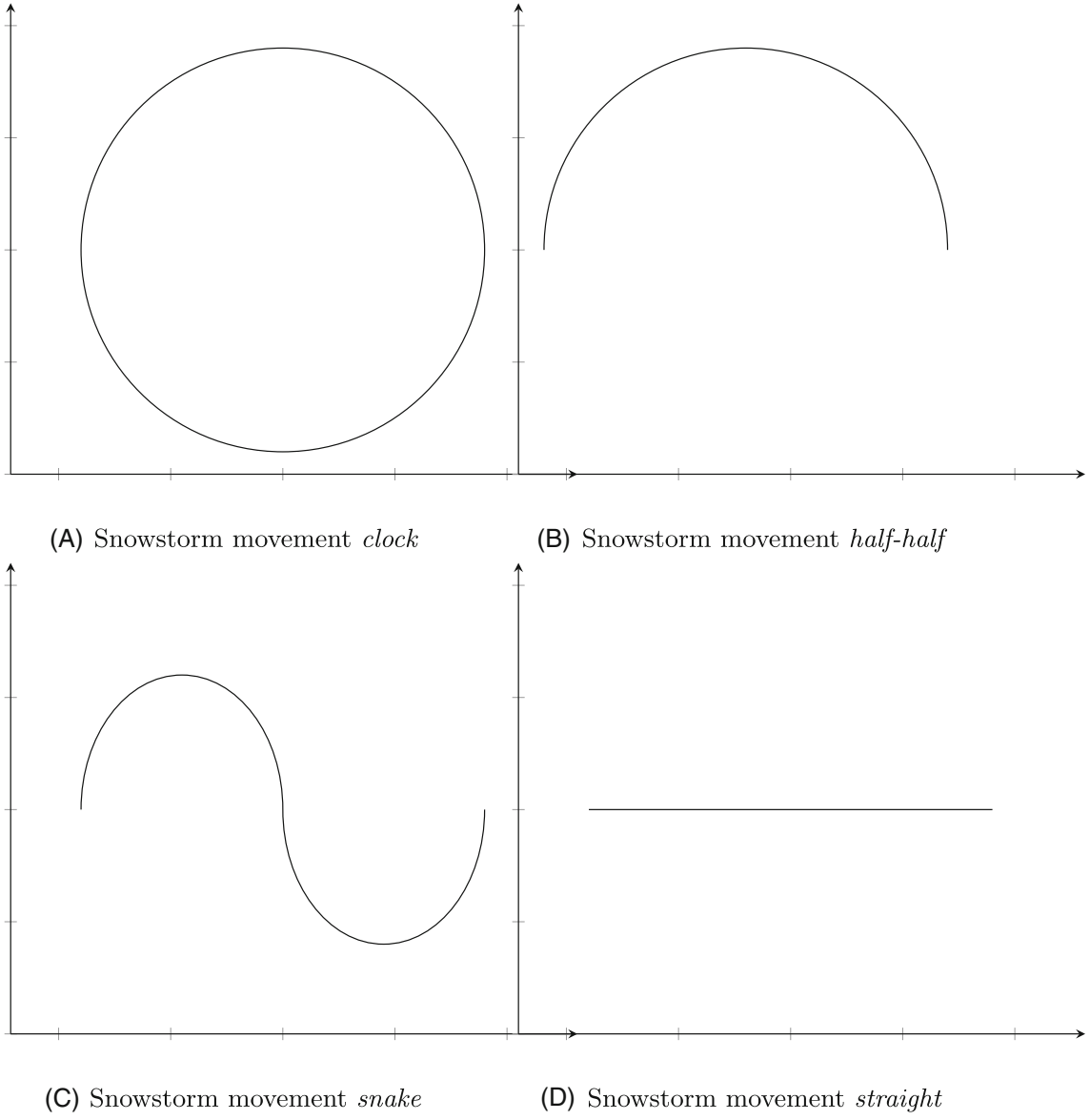


FIGURE A1 Illustration of the four different snowstorm movements used.

TABLE A2 Results for the binomial test comparing updating intervals when predicting the snowstorm movement (in %).

$T = 30$	$T = 60$	$T = 120$	$T = 300$	$T = 60$	$T = 30$	$T = 120$	$T = 300$	$T = 120$	$T = 30$	$T = 60$	$T = 300$	$T = 300$	$T = 30$	$T = 60$	$T = 120$
Clock	**0.02**$^{-}$	50	**0**$^{-}$	Clock	**0.02**$^{+}$	27.06	**0**$^{-}$	Clock	50	27.06	**0.03**$^{-}$	Clock	**0**$^{+}$	**0**$^{+}$	**0.03**$^{+}$
HH	**0**$^{-}$	25.17	**0**$^{-}$	HH	**0**$^{+}$	33.88	**0**$^{-}$	HH	25.17	33.88	30.36	HH	**0**$^{+}$	**0**$^{+}$	30.36
Snake	**0.02**$^{-}$	*4.81*$^{-}$	**0.21**$^{-}$	Snake	**0.02**$^{+}$	33.88	32.38	Snake	*4.81*$^{+}$	33.88	11.33	Snake	**0.21**$^{+}$	32.38	11.33
Straight	**0.81**$^{-}$	14.31	**0.12**$^{-}$	Straight	**0.81**$^{+}$	7.58	**0.01**$^{-}$	Straight	14.31	7.58	19.38	Straight	**0.12**$^{+}$	**0.01**$^{+}$	19.38
Clock	**0**$^{-}$	**0**$^{-}$	**0.03**$^{-}$	Clock	**0**$^{+}$	**0**$^{-}$	11.48	Clock	**0**$^{+}$	**0**$^{+}$	57.22	Clock	**0.03**$^{+}$	11.48	57.22
HH	**0**$^{-}$	**0**$^{-}$	**0.04**$^{-}$	HH	**0**$^{+}$	**0**$^{-}$	*1.73*$^{-}$	HH	**0**$^{+}$	**0**$^{+}$	20.24	HH	**0.04**$^{+}$	*1.73*$^{+}$	20.24
Snake	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	Snake	**0**$^{+}$	**0**$^{-}$	**0**$^{-}$	Snake	**0**$^{+}$	**0**$^{+}$	*3.84*$^{-}$	Snake	**0**$^{+}$	**0**$^{+}$	*3.84*$^{+}$
Straight	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	Straight	**0**$^{+}$	**0**$^{-}$	**0**$^{-}$	Straight	**0**$^{+}$	**0**$^{+}$	**0**$^{-}$	Straight	**0**$^{+}$	**0**$^{+}$	**0**$^{+}$
Clock	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	Clock	**0**$^{+}$	**0.01**$^{-}$	**0**$^{-}$	Clock	**0**$^{+}$	**0.01**$^{+}$	**0.17**$^{-}$	Clock	**0**$^{+}$	**0**$^{+}$	**0.17**$^{+}$
HH	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	HH	**0**$^{+}$	**0**$^{-}$	**0**$^{-}$	HH	**0**$^{+}$	**0**$^{+}$	**0.03**$^{-}$	HH	**0**$^{+}$	**0**$^{+}$	**0.03**$^{+}$
Snake	**0.01**$^{-}$	**0**$^{-}$	**0**$^{-}$	Snake	**0.01**$^{+}$	**0.11**$^{-}$	**0.01**$^{-}$	Snake	**0**$^{+}$	**0.11**$^{+}$	*2.51*$^{-}$	Snake	**0**$^{+}$	**0.01**$^{+}$	*2.51*$^{+}$
Straight	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	Straight	**0**$^{+}$	*1*$^{-}$	**0**$^{-}$	Straight	**0**$^{+}$	*1*$^{+}$	*1*$^{-}$	Straight	**0**$^{+}$	**0**$^{+}$	*1*$^{+}$
Clock	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	Clock	**0**$^{+}$	**0**$^{-}$	*1*$^{-}$	Clock	**0**$^{+}$	**0**$^{+}$	18.85	Clock	**0**$^{+}$	*1*$^{+}$	18.85
HH	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	HH	**0**$^{+}$	**0**$^{-}$	**0.11**$^{-}$	HH	**0**$^{+}$	**0**$^{+}$	43	HH	**0**$^{+}$	**0.11**$^{+}$	43
Snake	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	Snake	**0**$^{+}$	**0**$^{-}$	*2.51*$^{-}$	Snake	**0**$^{+}$	**0**$^{+}$	10.77	Snake	**0**$^{+}$	*2.51*$^{+}$	10.77
Straight	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	Straight	**0**$^{+}$	**0**$^{-}$	**0.11**$^{-}$	Straight	**0**$^{+}$	**0**$^{+}$	43	Straight	**0**$^{+}$	**0.11**$^{+}$	43
Clock	**0.02**$^{-}$	**0**$^{-}$	**0.02**$^{-}$	Clock	**0.02**$^{+}$	**0**$^{-}$	7.48	Clock	**0**$^{+}$	**0**$^{+}$	50	Clock	**0.02**$^{+}$	7.48	50
HH	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$	HH	**0**$^{+}$	**0**$^{-}$	*1*$^{-}$	HH	**0**$^{+}$	**0**$^{+}$	50	HH	**0**$^{+}$	*1*$^{+}$	50
Snake	**0**$^{-}$	**0**$^{-}$	**0.33**$^{-}$	Snake	**0**$^{+}$	**0**$^{-}$	*1.13*$^{-}$	Snake	**0**$^{+}$	**0**$^{+}$	27.06	Snake	**0.33**$^{+}$	*1.13*$^{+}$	27.06
Straight	**0**$^{-}$	**0**$^{-}$	**0.33**$^{-}$	Straight	**0**$^{+}$	**0**$^{-}$	**0.01**$^{-}$	Straight	**0**$^{+}$	**0**$^{+}$	41.94	Straight	**0.33**$^{+}$	**0.01**$^{+}$	41.94

Note: Bold numbers indicate strong significance according to the binomial test, while italic numbers indicate weak significance. Superscript $^{+}$ denotes that the setting of the block performs better than the setting of the column, while $^{-}$ denotes the inverse.

TABLE A3 Results for the binomial test comparing look-aheads with real time horizons fixed at 30, 60, 120, and 300 when predicting the snowstorm movement (in %).

$E = 50$	$E = 100$	$E = 200$	$E = 300$	$E = 100$	$E = 50$	$E = 200$	$E = 300$	$E = 200$	$E = 50$	$E = 100$	$E = 300$	$E = 300$	$E = 50$	$E = 100$	$E = 200$
Clock	36.01	**0**$^{+}$	**0**$^{+}$	Clock	36.01	**0**$^{+}$	**0**$^{+}$	Clock	**0**$^{-}$	**0**$^{-}$	**0**$^{+}$	Clock	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
HH	6.07	**0**$^{+}$	**0**$^{+}$	HH	6.07	**0**$^{+}$	**0**$^{+}$	HH	**0**$^{-}$	**0**$^{-}$	**0**$^{+}$	HH	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
Snake	36.01	**0**$^{+}$	**0**$^{+}$	Snake	36.01	**0**$^{+}$	**0**$^{+}$	Snake	**0**$^{-}$	**0**$^{-}$	**0.03**$^{+}$	Snake	**0**$^{-}$	**0**$^{-}$	**0.03**$^{-}$
Straight	43.4	**0**$^{+}$	**0**$^{+}$	Straight	43.4	**0**$^{+}$	**0**$^{+}$	Straight	**0**$^{-}$	**0**$^{-}$	**0**$^{+}$	Straight	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
Clock	8.14	**0.07**$^{+}$	**0**$^{+}$	Clock	8.14	**0**$^{+}$	**0**$^{+}$	Clock	**0.07**$^{-}$	**0**$^{-}$	**0**$^{+}$	Clock	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
HH	56.27	**0.32**$^{+}$	**0**$^{+}$	HH	56.27	**0.01**$^{+}$	**0**$^{+}$	HH	**0.32**$^{-}$	**0.01**$^{-}$	**0**$^{+}$	HH	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
Snake	*1.19*$^{-}$	**0.05**$^{+}$	**0**$^{+}$	Snake	*1.19*$^{+}$	**0.01**$^{+}$	**0**$^{+}$	Snake	**0.05**$^{-}$	**0.01**$^{-}$	**0**$^{+}$	Snake	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
Straight	56.27	**0.17**$^{+}$	**0**$^{+}$	Straight	56.27	**0.01**$^{+}$	**0**$^{+}$	Straight	**0.17**$^{-}$	**0.01**$^{-}$	**0**$^{+}$	Straight	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
Clock	*2.66*$^{-}$	9.98	**0.04**$^{+}$	Clock	*2.66*$^{+}$	31.79	**0.01**$^{+}$	Clock	9.98	31.79	**0.35**$^{+}$	Clock	**0.04**$^{-}$	**0.01**$^{-}$	**0.35**$^{-}$
HH	**0.69**$^{-}$	12.79	23.66	HH	**0.69**$^{+}$	31.79	**0.01**$^{+}$	HH	12.79	31.79	18.85	HH	23.66	**0.01**$^{-}$	18.85
Snake	**0.01**$^{-}$	9.98	**0**$^{+}$	Snake	**0.01**$^{+}$	50	**0**$^{+}$	Snake	9.98	50	**0**$^{+}$	Snake	**0**$^{-}$	**0**$^{-}$	**0**$^{-}$
Straight	*1.92*$^{-}$	31.79	*1*$^{+}$	Straight	*1.92*$^{+}$	21.48	**0**$^{+}$	Straight	31.79	21.48	*1*$^{+}$	Straight	*1*$^{-}$	**0**$^{-}$	*1*$^{-}$
Clock	*1.21*$^{-}$	**0**$^{-}$	*4.48*$^{-}$	Clock	*1.21*$^{+}$	**0.01**$^{-}$	5.51	Clock	**0**$^{+}$	**0.01**$^{+}$	**0.25**$^{+}$	Clock	*4.48*$^{+}$	5.51	**0.25**$^{-}$
HH	15.37	*1*$^{-}$	*1*$^{-}$	HH	15.37	**0**$^{-}$	**0.01**$^{-}$	HH	*1*$^{+}$	**0**$^{+}$	56.27	HH	*1*$^{+}$	**0.01**$^{+}$	56.27
Snake	*3.84*$^{-}$	9.46	14.31	Snake	*3.84*$^{+}$	*1.13*$^{-}$	7.58	Snake	9.46	*1.13*$^{+}$	*4.94*$^{+}$	Snake	14.31	7.58	*4.94*$^{-}$
Straight	**0.03**$^{-}$	**0**$^{-}$	33.18	Straight	**0.03**$^{+}$	*1.13*$^{-}$	27.06	Straight	**0**$^{+}$	*1.13*$^{+}$	*2.35*$^{+}$	Straight	33.18	27.06	*2.35*$^{-}$

Note: Bold numbers indicate strong significance according to the binomial test, while italic numbers indicate weak significance. Superscript $^{+}$ denotes that the setting of the block performs better than the setting of the column, while $^{-}$ denotes the inverse.