European Journal of Operational Research xxx (xxxx) xxx

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Production, Manufacturing, Transportation and Logistics

A learning-based granular variable neighborhood search for a multi-period election logistics problem with time-dependent profits

Masoud Shahmanzari [a], Renata Mansini [b,*]

[a] Brunel Business School, Brunel University London, Kingston Ln, Uxbridge, UB8 3PH, London, UK
[b] University of Brescia, Department of Information Engineering, Via Branze 38, 25123, Brescia, Italy

ARTICLE INFO

Keywords:
Election logistics
Multi-period routing
Multiple roaming salesman problem
Time-dependent profits
Adaptive variable neighborhood search

ABSTRACT

Planning the election campaign for leaders of a political party is a complex problem. The party representatives, running mates, and campaign managers have to design an efficient routing and scheduling plan to visit multiple locations while respecting time and budget constraints. Given the limited time of election campaigns in most countries, every minute should be used effectively, and there is very little room for error. In this paper, we formalize this problem as the multiple Roaming Salesman Problem (mRSP), a new variant of the recently introduced Roaming Salesman Problem (RSP), where a predefined number of political representatives visit a set of cities during a planning horizon to maximize collected rewards, subject to budget and time constraints. Cities can be visited more than once and associated rewards are time-dependent (increasing over time) according to the day of the visit and the recency of previous visits. We develop a compact Mixed Integer Linear Programming (MILP) formulation complemented with effective valid inequalities. Since commercial solvers can obtain optimal solutions only for small-sized instances, we develop a Learning-based Granular Variable Neighborhood Search and demonstrate its capability of providing high-quality solutions in short CPU times on real-world instances. The adaptive nature of our algorithm refers to its ability to dynamically adjust the neighborhood structure based on the progress of the search. Our algorithm generates the best-known results for many instances.

1. Introduction

Designing an efficient itinerary with an optimal schedule of visits and minimum routing costs is crucial for many real-world routing and scheduling problems, including the large gamut of orienteering problem variants (Gunawan, Lau, & Vansteenwegen, 2016), the tourist trip planning (da Silva, Morabito, & Pureza, 2018; Schilde, Doerner, Hartl, & Kiechle, 2009; Vansteenwegen, Souffriau, Berghe, & Oudheusden, 2009a; Wang, Golden, & Wasil, 2008), the election logistics (Shahmanzari, Aksen, & Salhi, 2020), and the nurse routing (Cinar, Salman, & Bozkaya, 2021; Gobbi, Manerba, Mansini, & Zanotti, 2023; Manerba & Mansini, 2016) . While operational constraints may vary, discussed problems all share similar characteristics: one or more campaigners seek to visit a set of locations during a limited planning horizon. Each site is associated with a reward (measured in terms of interest/convenience to visit it) and some cities can remain not served due to the time limit for each period. The primary challenge among discussed problems is to determine the schedule of visits over a finite time horizon of several days. In election logistics, for instance, most countries (e.g., France, Canada, Australia, Japan, Singapore, and Turkey) limit the length of the political election campaign periods to less than 40 days.[1] In US general elections, campaign intervals range from a few days to three weeks. According to political experts, the scheduling and routing of the state primaries seem inefficient as the planning of the campaign is not operationally beneficial to the party leader (Saltzman & Bradford, 2022). To be effective, an election campaign must reach as many people as possible by visiting many cities while minimizing costs. This increases the likelihood of obtaining financial support and votes by actively engaging a broader audience. Thus, within a limited budget and time, party leaders should plan an efficient campaign schedule that grants them enough time to visit important cities and minimizes their traveling costs. Upon necessity of press conferences and speeches, party representatives may decide to visit some cities more than once, whereas complying with time and budget constraints may prevent them to serve all the cities.

In this paper, we introduce the multiple Roaming Salesman Problem (mRSP), a new variant of the recently introduced Roaming Salesman

* Corresponding author.
E-mail addresses: Masoud.Shahmanzari@brunel.ac.uk (M. Shahmanzari), renata.mansini@unibs.it (R. Mansini).

[1] https://www.europarl.europa.eu/at-your-service/en/be-heard/elections

https://doi.org/10.1016/j.ejor.2024.06.009
Received 14 April 2023; Accepted 7 June 2024
Available online 11 June 2024

Problem (RSP), where a predefined number of political representatives visit a set of cities during a planning horizon to maximize collected rewards, subject to budget and time constraints. In contrast to the discussed problems, where all daily tours should be either closed or open tours, in the election logistics problem, the daily tours do not have a predefined structure so that on the same day it is possible for different travelers to have open and closed tours. While the former tours happen when starting and ending locations are not the same, the latter tours refer to the periods where starting and ending cities coincide. Motivated by real-world requirements, we also allow for no tour periods as well, indicating that the campaigner does not leave the selected dwelling during that period. Additionally, unlike recent extensions of VRP and TSP models, we do not predetermine the set of depot cities for each period, but leaving them as decision variables for each political representative in our model. Therefore, at the beginning of each period, every city is a potential depot. Visiting a city provides a positive reward to the campaign. During election seasons, it is extremely crucial for party leaders and representatives to schedule their speeches and political meetings in important cities near the election day. For this reason, the reward associated with cities is time-dependent and increasing as the campaign approaches its conclusion. This reward mechanism is informed by a comprehensive examination of political science literature and insightful consultations with political experts in Turkey. While there is no universal agreement among political experts on the precise nature of the reward mechanism, there is a common consensus that as the campaign progresses towards its final stages, the importance of political meetings and speeches escalates. Existing political science literature highlights that people tend to forget a significant portion of campaign information over time (Box-Steffensmeier & Kimball, 1999; Lodge, Steenbergen, & Brau, 1995). Recent studies in fundraising indicate that as the campaign nears completion, individuals are more inclined to contribute (Morvinski, Lupoli, & Amir, 2022; Van Aelst, Aalberg, et al., 2012). In response to these dynamics, political parties implement last-minute initiatives to mobilize voters and secure their support. The intensity of campaign events increases, with candidates strategically visiting key battleground states and organizing large rallies to energize their base and attract media attention (Trent & Friedenberg, 2008).

Moreover, a city may accommodate more than one activity throughout the campaign (debates, public events, media interviews, or rallies). This implies that the same city may be visited more than once by the same candidate or by different ones. However, for each political candidate, there is an explicit limit on the number of activities that can be realized per day. This is equivalent to assuming that a demand is specified for each site and a capacity bound is associated with each traveler. Revisiting a city, although still yielding a positive reward, results in a reward lower than that one might obtain on a first visit. The magnitude of this reward is contingent on the time elapsed since the previous visit, with a more consistent reduction in reward when less time has passed since the last visit.

Fig. 1 illustrates a feasible mRSP solution with all tour types for an instance with 3 candidates (blue, green, and black) and 14 cities in two consecutive days of the planning horizon. Note that mRSP is a selective routing problem. While some nodes (e.g., o) are not visited, some are visited but not included in the reward collection (n in day $t-1$ and i in day t). The solution of the first day consists of an open tour route for the green candidate ($a \to j \to k \to n$), with the reward collection only in j and k, a closed tour route for the blue representative ($a \to c \to f \to h \to a$), with reward collection in a, c, f, h, and a no tour route for the last one (the candidate remains in the city i with a reward collection). On the other hand, the routes of the second day include two open tours and one closed tour route. Note that while node h is visited by the blue candidate on the first day, the black candidate makes a repeated visit there on the second one. In this example, revisiting is permitted on subsequent days. Also, the same candidate can be assigned different tour types during the campaign. For instance, the blue candidate executes one closed tour and one open tour on the two days, respectively.

The complexity of operational requirements puts an immense strain on campaign planners. Two critical dimensions must be considered: (*i*) the timing and frequency of visits to each city, and (*ii*) the routing plan for each candidate on each day of the campaign period. From a scheduling perspective, the first dimension is embedded into the objective function of the problem by maximizing the collected time-dependent rewards from the visited cities. This objective is particularly important in election logistics given that the success of political parties relies on holding meetings in the most crucial cities. According to political experts, in the 2016 US presidential elections, the loss of the unsuccessful party candidate could be attributed to several factors, one of which was not having enough time to visit Michigan state close to the end of the election campaign.[2] The second dimension deals with determining optimal routes for campaigners such that all traveling costs are minimized. Experts estimate the total spending for the 2020 US presidential election exceeded $14.4 billion, more than doubling the cost of the record-breaking election in 2016, where a notable part of this money went for traveling and accommodation costs (Arbay, Pasha, & Widodo, 2021). Before each election, many party leaders design a routing and scheduling plan to effectively manage the election campaign. During the 2020 presidential election in the US, it is estimated that each trip taken by the winning party leader has a cost of $55,000, roughly 70% of which was spent on routing and accommodation expenses. Therefore, the magnitude of involved costs in this field implies that even a few percent of improvement results in saving millions of dollars for stakeholders. Without an optimal itinerary plan in this multi-billion dollar business, engaged costs can skyrocket rapidly.

The mRSP has promising applications in various real-world routing and scheduling problems. Below, we list a few potential applications in other contexts than election logistics.

- *Humanitarian logistics*: mRSP can address logistic challenges faced by search-and-rescue teams that are dispatched from a campaign center to serve disaster-affected areas (DAAs) during post-disaster relief operations. Rewards associated with DAAs are necessarily time-dependent. For instance, in the event of an earthquake, the probability of saving lives significantly decreases within the crucial 72-hour golden time window.
- *Tourism and travel planning*: In the tourism sector, tour operators offer multi-city tours for different groups of tourists where there is no fixed hotel for overnight stays and daily tours can consist of open, closed, and no tour routes. Moreover, the collected rewards from points of interest are time-dependent, e.g., it is more efficient to visit Louvre Museum during weekdays as it is less crowded. mRSP can be utilized to design itineraries that maximize the enjoyment of tourists while staying within budgetary and time limitations.
- *Courier and delivery services*: In sales and marketing operations, mRSP can assist in planning the routes of representatives who need to visit multiple clients by forming various tour types for each of them within a specific budget and time constraints. The problem can optimize the sequence of customer visits where rewards are time-dependent as postponing a visit increases the associated penalty.
- *Healthcare logistics*: In healthcare logistics, mRSP can be applied to optimize the routing of healthcare professionals visiting multiple patients or clinics. For instance, mRSP can effectively tackle a variant of the nurse routing problem involving multiple travelers, wherein the rewards for patient visits dynamically change

[2] https://www.politico.com/story/2016/12/michigan-hillary-clinton-trump-232547

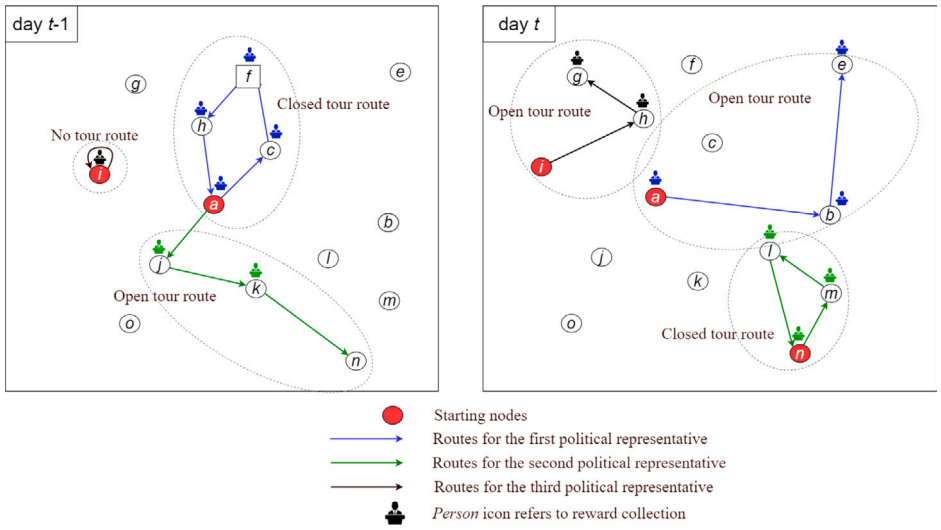


Fig. 1. Example of mRSP's solution including all tour types: instance with 3 candidates over 2 days.

over time, e.g., the urgency of visiting certain patients increases throughout the planning horizon. The flexible features of mRSP can readily be applied to such scenarios to maximize the collected reward from these time-sensitive visits.

A close examination of previous election campaign plans in countries such as the US and Turkey reveals that current optimization models in the literature (team orienteering problem, multi-period traveling salesman problem, etc.) are not capable of reflecting the real-world operational requirements. In this research, we present a new variant of RSP to capture all practical assumptions. The contributions of this work are multi-fold:

1. We introduce a new variant of a multi-period multi-vehicle selective problem incorporating realistic routing and scheduling assumptions, including (*i*) time-dependent rewards associated with locations to visit, (*ii*) presence of multiple campaigners, (*iii*) the possibility of visiting the same location more than once possibly by different campaigners (multi-visit) over the time horizon (multi-period) within specified constraints, (*iv*) the inclusion of three tour types (closed, open, and no tour routes), (*v*) flexibility of campaigners in returning to any city (e.g., the absence of a central depot), and (*vi*) the requirement for all campaigners of returning to the campaign main center after a specific amount of time. To the best of our knowledge, none of the previous research on multi-period orienteering problem variants has jointly considered all these characteristics together.
2. Time-dependent rewards have a non-additive nature when considering multi-visits of the same city over time. This aspect along with the presence of multiple representatives jointly bounded when visiting the same city more than once over the planning horizon, strongly differentiates this problem from the single-vehicle special case. Moreover, time dependency has been included in the objective function as a reward multiplicative term that changes according to the time. This formulation is well-suited to our problem, where reward function linearly increases over time, but it can be easily adapted and generalized to accommodate different functions based on the specific application context, whether rewards can be increasing or decreasing either linearly or non-linearly.
3. We investigate the application of the mRSP in political campaigns by incorporating realistic routing and scheduling assumptions. However, the problem is highly relevant to other real-world routing problems and can be applied to various practical contexts.
4. We develop a compact MILP formulation for the problem along with some valid inequalities. The formulation addresses constraints associated with modeling open, close, and no tours concurrently on the same day. It is worth noting that mRSP is not a mere extension of the single-vehicle case; rather, it intricately intertwines tour-type choices with time-dependent rewards and temporal constraints for the vehicles. For example, the problem allows multiple visits to the same city but it incorporates restrictions that prevent a campaigner from revisiting the same city before a certain number of days has elapsed, whereas different limitations apply to other representatives. Furthermore, this multi-vehicle formulation serves as an initial stride towards tackling more interesting and real-world variants, potentially involving dynamic information or data uncertainty. In this context, assessing the challenge of solving this static variant and emphasizing its complexity related to scalability such as the number of cities, representatives, and especially the time horizon's duration, represents an important insight for future research.
5. To address real-sized instances, we introduce a hyper-heuristic controlling a sequence of Variable Neighborhood Search (VNS) methods based on a range of local search operators, some of which incorporate problem-specific knowledge. Each VNS visits a sequence of nested neighborhoods parametrized by their size (the neighborhood radius). These operators serve as the core components and make the method work as an adaptive granular Variable Neighborhood Search. This approach encompasses an initial learning phase that seeks to identify the most promising neighborhood radius values from among all the sequence options. The subsequent phase selects neighborhoods in a non-systematic manner, favoring VNSs that have shown more promise. A shaking phase, exploiting collected information, is called when further improvements for the currently selected VNS become challenging.
6. We benchmark the performance of the developed solution method with a state-of-the-art matheuristic duly adapted to the problem and the commercial solver Gurobi when solving the model.
7. We perform extensive computational experiments on real-world instances to attain managerial insights about crucial aspects of election logistics. These insights revolve around the effects of time-dependent rewards, the convenience of accommodating diverse tour structures beyond classical closed tours, and the significance of multiple visits. A sensitivity analysis is also

achieved on some parameters such as the maximum number of visits per day and the maximum daily time enabling an in-depth exploration of the trade-off between visiting many nearby cities (which require less travel time but may have lower rewards) versus a few geographically distant but highly profitable cities.

The paper is structured as follows. We present a review of related literature in Section 2. We formally describe mRSP in Section 3. The developed MILP formulation and the valid inequalities are presented in Section 4. Section 5 and Section 6 describe the proposed algorithms and provide the computational results, respectively. We discuss the additional analysis performed to derive managerial insights in Section 7. Finally, we conclude our paper in Section 8. The online supplementary material includes additional details about our numerical results and a description of a state-of-the-art solution method, which is evaluated against our hyper-heuristic approach.

2. Literature review

The Multiple Roaming Salesman Problem (mRSP) is an NP-Hard problem. It expands upon the Roaming Salesman Problem (RSP) (Shahmanzari et al., 2020) by removing the assumption of a single vehicle (campaigner) and RSP is well-known to be NP-Hard being a generalization of the Orienteering Problem. In this review of the current state of the art, we examine problems closely related to the RSP. Over the past few decades, the Traveling Salesman Problem (TSP) has undergone various modifications to better align with practical routing scenarios (Feillet, Dejax, & Gendreau, 2005). One notable variant of the TSP is the Orienteering Problem (OP), which falls under the category of selective routing problems that include profits associated with locations to visit (Tsiligirides, 1984). The OP is a selective routing problem since the traveler holds a list of potential sites each one with an associated reward (profit), but it may not be possible nor desirable to visit all of them. In practical applications, the traveler is constrained by a maximum tour duration. The OP, inspired by a hunting game, maximizes the overall collected reward by determining the order of visits to a subset of sites.

In the routing literature, a wide range of variants of the OP have been investigated. These include problems that incorporate time windows (Labadie, Mansini, Melechovskỳ, & Calvo, 2012), where visits are restricted by specific time intervals (Vansteenwegen, Souffriau, Berghe, & Van Oudheusden, 2009b), problems that allow for split delivery options, in which a customer can be serviced multiple times with each visit satisfying a portion of their demand Wang, Golden, and Gulczynski (2014), arc routing orienteering problems (Riera-Ledesma & Salazar-González, 2017), generalizations that include multiple routes or vehicles (Ruiz-Meza, Brito, & Montoya-Torres, 2021; Shahmanzari & Aksen, 2021; Shahmanzari, Aksen, & Salhi, 2022; Tarantilis, Stavropoulou, & Repoussis, 2013) also with multi visits to the same node and precedence constraints (Hanafi, Mansini, & Zanotti, 2020), and variants that explore multi-period planning horizons (Kotiloglu, Lappas, Pelechrinis, & Repoussis, 2017). For a comprehensive survey of recent OP variants, readers are referred to Gunawan et al. (2016). When in addition to a length bound the OP also includes node demand and a capacity bound the generalization is called Capacitated Orienteering Problem (COP) (interested readers are referred to Bock and Sanità (2015)). A relevant application of the COP in freight is known as the Capacitated Team Orienteering Problem (CTOP) where there are multiple vehicles such that a maximum capacity is imposed on each vehicle and a weight is assigned to each node (location) (Tarantilis et al., 2013). The CTOP involves closed tours of a homogeneous fleet of vehicles with capacity and maximum tour duration constraints. The objective of the CTOP is to design a route for each vehicle that maximizes the total collected profit such that the time and capacity constraints are not violated.

The OP variants traditionally assume a fixed travel time and cost between sites (nodes). However, in the context of intra-city logistics, route planners must take into account congestion-related issues when designing routes for each vehicle. When the travel time between two nodes is contingent upon the time of traversal, such as traffic during peak hours, the time-dependent Orienteering Problem (OP) arises (Riera-Ledesma & Salazar-González, 2017; Verbeeck, Sörensen, Aghezzaf, & Vansteenwegen, 2014). The time dependency of the OP variant is not limited to travel times alone. Khodadadian, Divsalar, Verbeeck, Gunawan, and Vansteenwegen (2022) examine the time-dependent OP with time-dependent rewards, in which the reward collected at each node increases linearly with the service time. This variant of the OP is more realistic as it is applicable to many real-world contexts, such as reverse logistics, where a longer service time results in a higher reward. In Barrena, Canca, Coelho, and Laporte (2022), the authors analyze a generalization of the selective TSP where each site has a time window, a service time, and a profit that increases over time, and the traveler can wait along the route if beneficial to collect higher profits.

It is worth noting that our study, which primarily focuses on intercity logistics applications, would not be significantly impacted by time-dependent travel times. Thus, our main focus would be on the time dependency of rewards that change according to the day of the visit. It is also noteworthy that our study, differently from multi-vehicle OP variants, allows multiple visits meaning that each node can be visited more than once. A famous variant of the TOP is the Team Orienteering Problem with Time Windows (TOPTW) where visits are constrained by specific time intervals (Vansteenwegen et al., 2009b). Kotiloglu et al. (2017) address an interesting multi-period application of the OP in tourist trip planning by developing a "filter-first, tour-second" approach to devise tour planning for tourists that consider their preferences. In a more realistic variant of the TOP, rewards associated with each location are time-dependent. Moreover, the travel time of an arc alters with respect to the time that arc is traversed (Riera-Ledesma & Salazar-González, 2017). This problem is known as the Time-Dependent Team Orienteering Problem (TDTOP). In some variants of the TOP, customers are not assigned to the nodes of a network. Instead, the vehicles should traverse some arcs to serve those customers (Riera-Ledesma & Salazar-González, 2017). This problem is known as the Team Orienteering Arc Routing Problem (TOARP). The majority of the existing literature has primarily focused on applications of the Orienteering Problem (OP) and its variants in vehicle routing (Aksen & Shahmanzari, 2017; Juan, Freixes, Panadero, Serrat, & Estrada-Moreno, 2020; Panadero, Juan, Bayliss, & Currie, 2020; Souffriau, Vansteenwegen, Vanden Berghe, & Van Oudheusden, 2013) and tourist trip design problems (Dang, Guibadj, & Moukrim, 2013; Gavalas, Konstantopoulos, Mastakas, Pantziou, & Vathis, 2015). However, there have been relatively few studies that have examined the implementation of the OP in other areas, such as designing logistics for political campaigns and marketing strategies (Shahmanzari et al., 2020). The proposed Multi-Roaming Salesman Problem (mRSP) can be viewed as a new variant of the multi-period Capacitated Orienteering Problem (COP) that accounts for time-dependent rewards, multiple travelers, and various tour types. Although the problem is analyzed in an election context, it can be easily applied to other domains as in the organization of social campaigns (against gender violence, and in favor of human rights just to name a few) or surveillance routing and scheduling activities in military and civilian application.

3. Problem description

The Multi-Roaming Salesman Problem (mRSP) is defined over a directed graph $G = (V, A \cup A_0)$. The node set $V = N \cup \{0\}$ comprises the set $N = \{1, \dots, n\}$ of $n-1$ cities plus the campaign center indexed as 1, and a node 0 representing a fictitious node introduced for modeling purposes. The arc set A encompasses connections between nodes within N, whereas set $A_0 = \{(i,0) | i \in N\} \cup \{(0,i) | i \in N\}$ consists of back and forth arcs connecting the fictitious node 0 with each node $i \in N$. Let $M = \{1, \dots, \bar{m}\}$ be the set of representatives (election candidates or

Table 1
Sets.

$M = \{1, \dots, \bar{m}\}$	Set of political representatives
$N = \{1, \dots, n\}$	Set of cities possibly holding activities and campaign center (node 1)
$V = N \cup \{0\}$	Set of cities N plus fictitious node 0
$A = \{(i,j) \mid i,j \in N (i \neq j)\}$	Set of arcs connecting nodes in N
$A_0 = \{(i,0) \mid i \in N\} \cup \{(0,i) \mid i \in N\}$	Set of arcs connecting nodes $i \in N$ with the fictitious node
$T = \{1, \dots, t_{\max}\}$	Set of $t_{\max}$ days comprising the planning horizon duration

campaigners) that have to plan their visits to the set of cities over a finite time horizon consisting of $t_{\max}$ days. We denote the corresponding set of days within the planning campaign as $T = \{1, \dots, t_{\max}\}$.

Each city $i \in N$ is associated with a time-dependent reward $b_{it}, i \in N, t \in T$, that increases as days go by and the end of the electoral campaign approaches. Reward b_{it} depends on a nonnegative base reward b_i related to the city i (and the number of the corresponding potential voters) and by a time-dependent increasing function δ_t, where $t \in T$ is the day of the activity. A city offers a reward only when a representative holds an activity there. Each city can host at most one activity per day. However, the same city can arrange multiple events on different days, contributing to the reward of one or more representatives. To prevent visiting the same city too frequently by the same representative, if a representative m visits a city on a day t, that city cannot be visited by the same candidate within the next ξ days. This constraint does not apply to other candidates, different from m. However, visiting the same city multiple times consecutively may not be advantageous in general. Thus, subsequent visits, while still resulting in a positive reward, apply a penalty function α_s that reduces the reward collected at time t according to the number of days elapsed since the previous visit (s) and is more significant when the previous visit is closer in time. Moreover, a gap of a minimum number of r ($r < \xi$) consecutive days has to be respected between two visits to the same city independently of the representative.

A time duration a_i is required by each activity held in the city $i \in N$ and a cost h_i has to be paid for overnight accommodation. A representative can remain overnight in a city without accomplishing any activity. Every arc $(i,j) \in A$ is associated with a travel cost $c_{ij} \in \mathbb{R}^+$ and a travel time $d_{ij} \in \mathbb{R}^+$. Traveling times (and costs) are assumed to satisfy triangular inequality, thus only cities holding activities will be visited by a political candidate in a tour but for those representing starting and ending nodes. Each representative is restricted to a maximum number, p, of visits per day. Also, the duration of each daily tour must not exceed the designated daily maximum tour threshold q. Political candidates can end their daily tour in any nodes in set N. However, it must be guaranteed that the route of today originates where the route of yesterday terminates. Moreover, every traveler must visit the campaign center every t_{away} days. Given that political representatives in real life usually return to central offices frequently, this assumption makes mRSP more realistic.

The problem looks for a subset of cities to visit and the definition of a route (open or closed) complying with the defined constraints for each representative in each day of the planning horizon while maximizing the difference between global collected rewards (possibly from multiple visits) and traveling costs. To make these two components of objective function compatible, we multiply traveling costs with a normalization coefficient β. No tour routes are modeled as a back and forth tour from the current location to the fictitious node indexed by 0, wherein all associated travel time, travel costs, and rewards are set to zero. In Tables 1–2 we summarize the main notation used, reporting the main sets and parameters of the problem.

In mRSP, all representatives are allowed to include open and closed tours on different days of the planning horizon. We present a toy-size mRSP instance in Fig. 2 to illustrate how the combination of closed and open daily tours improves a solution with only closed tours. The instance consists of 5 nodes and the campaign center. We consider the tours over a 3-day time horizon of a single representative. Travel time and travel costs between each pair of nodes are identical in the two

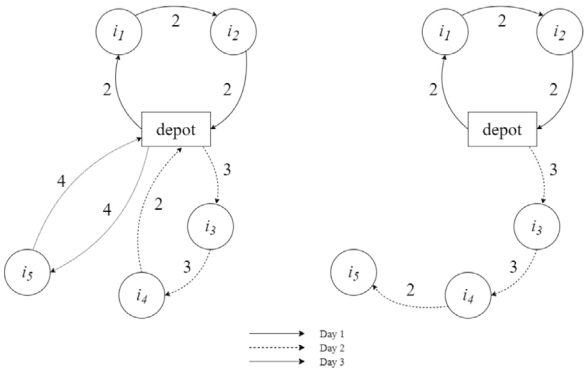


Fig. 2. (a) Optimal solution with only closed daily tours, spanning 3 days; (b) mRSP solution, spanning 2 days.

solutions and the maximum daily tour duration is 8 hours. According to traditional TSP models where starting node and ending node should coincide (Fig. 2.a), the traveling time in the optimal solution is 22, spanning 3 days of the planning horizon and visiting all cities. Compared with the previous solution, the mRSP solution (Fig. 2.b) provides improved efficiency where all nodes are visited, the total travel time is 14, and one day is saved.

We now highlight the main differences between mRSP and RSP:

(*i*) mRSP involves m homogeneous travelers (political representatives) instead of one.
(*ii*) The reward mechanism in the objective function of RSP is decreasing over time. In mRSP, however, the rewards associated with cities increase as we get closer to the end of the planning horizon (election day). Such a reward scheme reflects real-life assumptions in many contexts.
(*iii*) The RSP postulates that the campaign planner has access to an unlimited budget. However, in practical applications, budget limitations are a crucial consideration in the planning process and must be taken into account when making routing and scheduling decisions. To account for this, mRSP also incorporates budget constraints for overnight stays at terminal nodes.
(*iv*) Multiple visits to the same city are regulated not only for individual representatives but also collectively for all representatives. This is because a repeated activity cannot occur in the same city until r consecutive days have passed since the previous occurrence, irrespective of the individual involved.

4. The mathematical formulation

In this section, we present a compact mathematical formulation for the mRSP. In Tables 3–4, we list and categorize decision variables into two distinct groups: the routing variables and the reward ones. Routing variables deal with determining the set of nodes to visit, the order of visits, and the terminal nodes in any period. Reward variables determine the amount of reward collected from single or multiple visits to nodes.

Based on the two groups of variables described above, the objective function of mRSP consists of two components (total reward and

Table 2
Parameters.

c_{ij}	Traveling cost from node $i \in V$ to node $j \in V$ $(i \neq j)$; $c_{ij} = 0$ for $(i,j) \in A_0$
d_{ij}	Traveling time from node $i \in V$ to node $j \in V$ $(i \neq j)$; $d_{ij} = 0$ for $(i,j) \in A_0$
b_i	Base non-negative reward associated with city $i \in N$
b_{it}	Reward associated with city $i \in N$ when visited at time $t \in T$
a_i	Non-negative activity duration associated with city $i \in N$
h_i	Cost of a night accommodation in city $i \in N$
ϕ	Total daily cost of overnight stays for each representative
p	Maximum number of daily activities allowed for each representative
q	Maximum daily tour duration
t_{away}	Maximum number of consecutive days a representative can be away from campaign center
β	Reward normalization coefficient to make traveling costs and daily rewards compatible
ξ	Minimum number of days during which a representative is restricted from making repeat visits to the same city
r	Minimum gap in terms of consecutive days between any two activities organized in the same city by any representative, where $r < \xi$

Table 3
Routing decision variables.

X_{mijt}	Binary variable taking value 1 if arc $(i,j) \in A \cup A_0$ is traversed by traveler $m \in M$ on day $t \in T$; 0 otherwise.
L_{mit}	Binary variable taking value 1 if traveler $m \in M$ does not enter but only leaves city $i \in N$ on day $t \in T$; 0 otherwise.
E_{mit}	Binary variable taking value 1 if traveler $m \in M$ does not leave, but only enters city $i \in N$ on day $t \in T$; 0 otherwise.
S_{mit}	Binary variable taking value 1 if traveler $m \in M$ stays overnight (sleeps) in city $i \in N$ at the end of period $t \in T$; 0 otherwise.
U_{mit}	Continuous nonnegative variable determining for traveler $m \in M$ the order of visit of city $i \in V$ on day $t \in T$ (Modified Miller–Tucker–Zemlin subtour elimination constraints).

Table 4
Reward collection decision variables.

Z_{mit}	Binary variable taking value 1 if traveler $m \in M$ collects reward in city $i \in N$ on day $t \in T$; 0 otherwise.
F_{mit}	Binary variable taking value 1 if the first activity of representative $m \in M$ in city $i \in N$ is performed on day $t \in T$; 0 otherwise.
R_{mits}	Binary variable taking value 1 if city $i \in N$ accommodates two consecutive activities by traveler $m \in M$ on days $t \in T$ and $t - s$ $(\xi \leq s \leq t-1)$; 0 otherwise.

traveling costs) accordingly:

$$\max \sum_{m\in M}\sum_{i\in N}\sum_{t\in T} b_{it}F_{mit} + \sum_{m\in M}\sum_{i\in N}\sum_{t\in T}\sum_{\xi\leq s\leq t-1}\alpha_s b_{it}R_{mits} - \beta\sum_{m\in M}\sum_{(i,j)\in A}\sum_{t\in T} c_{ij}X_{mijt} \quad (1)$$

The reward collection part consists of two terms:

(i) $\sum_{m\in M}\sum_{i\in N}\sum_{t\in T} b_{it}F_{mit}$ expresses the collection of time-dependent rewards in first visits with $b_{it} = \delta_t b_i$ being a reward function depending on the base city reward b_i and an increasing function $\delta_t = \frac{t+t_{\max}}{t_{\max}}$, where t is the day of the visit.

(ii) $\sum_{m\in M}\sum_{i\in N}\sum_{t\in T}\sum_{\xi\leq s\leq t-1}\alpha_s b_{it}R_{mits}$ represents the collection of time-dependent rewards in repeated visits. For each repeated visit, the time-dependent reward b_{it} undergoes a penalty function $\alpha_s = \frac{s}{\alpha t_{\max}}$ where s represents the number of days elapsed since the last visit and coefficient $\alpha > 1$. The lower the value of s, the lower the value of the adjustment function α_s, and the higher the penalty on the time-dependent reward b_{it}. The inclusion of this penalty component aims to discourage repeated visits to the same city within a short time interval by the same representative.

The routing part is the last term in (1). The coefficient β is applied to balance the traveling costs with collected rewards. The objective function seeks to maximize the total benefit defined as the difference between the collected rewards and the incurred routing costs for all candidates.

The mRSP can be formulated subject to the following sets of constraints:

(a) *Traditional Selective TSP constraints (Laporte & Martello, 1990):*

$$\sum_{j\in V} X_{mijt} \leq 1 \qquad m \in M, i \in V,\ t \in T \quad (2)$$

$$\sum_{j\in V} X_{mjit} \leq 1 \qquad m \in M, i \in V,\ t \in T \quad (3)$$

These inequalities limit the number of incoming and outgoing arcs for every representative in each node and each time period.

(b) *Constraints on general reward collection:*

$$1 \leq \sum_{i\in N} Z_{mit} \leq p \qquad m \in M, t \in T \quad (4)$$

$$\sum_{m\in M} Z_{mit} \leq 1 \qquad i \in N,\ t \in T \quad (5)$$

$$\sum_{k=t+1}^{t+\xi} Z_{mik} \leq 1 - Z_{mit} \qquad m \in M, i \in N,\ 1 \leq t \leq t_{\max} - \xi + 1 \quad (6)$$

Constraints (4) impose both a lower and an upper bound on the total number of meetings held for each representative in each day t. Inequalities (5) ensure every node can be visited by at most one political candidate on any given day t. According to constraints (6), if a representative visits a node on the day t, that node cannot be visited by the same representative in the upcoming ξ days.

(c) *Constraints on reward collection for the first time*

$$F_{mit} \leq Z_{mit} \qquad m \in M, i \in N,\ t \in T \quad (7)$$

$$F_{mit} \leq 1 - Z_{miu} \qquad m \in M, i \in N,\ t \in T,\ 1 \leq u \leq t-1 \quad (8)$$

$$\sum_{t\in T} F_{mit} \le 1 \qquad m \in M, i \in N \tag{9}$$

$$Z_{mit} \le \sum_{j\in N} X_{mijt} + E_{mit} \qquad m \in M, i \in N,\ t \in T \tag{10}$$

$$Z_{mit} \le \sum_{j\in N} X_{mjit} + L_{mit} \qquad m \in M, i \in N,\ t \in T \tag{11}$$

Constraints (7) couple variables F and Z. Inequalities (8) ensure that if the first meeting in city i was held by representative m on the day t, then there cannot be a meeting on an earlier day u, $u \le t-1$. Constraints (9) guarantee that the first meeting for each city and each representative can be held at most once during the planning horizon. The set of constraints (10) and (11) ensure that in order for a city i to hold a meeting for representative m, it should be visited that day.

(d) *Constraints on repeated reward collection*

$$R_{mits} \le Z_{mit} \qquad m \in M, i \in N,\ t \in T,\ \xi \le s \le t-1 \tag{12}$$

$$R_{mits} \le Z_{mi(t-s)} \qquad m \in M, i \in N,\ t \in T,\ \xi \le s \le t-1 \tag{13}$$

$$\sum_{k=t-s+1}^{t-1} Z_{mik} \le s(1-R_{mits}) \qquad m \in M, i \in N, \xi+1 \le t \le t_{max},\ \xi \le s \le t-1 \tag{14}$$

$$R_{mius} \le 1 - F_{mit} \qquad m \in M, i \in N, \quad t \in T\backslash\{1\}, t+1 \le u \le t_{\max}, u-t+1 \le s \le u-1 \tag{15}$$

$$\sum_{m\in M}\sum_{l=t}^{t+r-1} Z_{mil} \le 1 \qquad i \in N,\ 1 \le t \le t_{\max}-r+1 \tag{16}$$

The disaggregated inequalities (12) and (13) establish the logical relation between decision variables R_{mits} and Z_{mit}. Constraints (14) assure that if a city host two meetings on days t and $(t-s)$, then all corresponding Z_{mik} variables in the interval $[t-s+1,\ t-1]$ should be zero. Constraints (15) couple binary decision variables R and F. Note that the latter constraints are valid inequalities not necessary to the formulation but added from scratch to strengthen it. Since the model does not allow for repeated visits to a given city each day, a representative may visit that city in the upcoming days although the collected reward for this visit is penalized with a lower reward in the objective function. To avoid such a situation, we introduce constraints (16) to impose that at least r days should pass between two activities in the same city by any representatives, where $r < \xi$. In our case $r = 2$.

(e) *Constraints on maximum daily tour duration*

$$\sum_{i\in N} a_i Z_{mit} + \sum_{(i,j)\in A} d_{ij} X_{mijt} \le q \qquad m \in M, t \in T \tag{17}$$

Constraint (17) requires that for each day $t \in T$ and representative $m \in M$, the sum of travel time and activity time must not exceed q.

(f) *Constraints on closed tour routes*

$$\sum_{j\in N} X_{mijt} - \sum_{j\in N} X_{mjit} = L_{mit} - E_{mit} \qquad m \in M, i \in N,\ t \in T \tag{18}$$

$$L_{mit} + E_{mit} \le 1 \qquad m \in M, i \in N,\ t \in T \tag{19}$$

$$\sum_{i\in N} (L_{mit} + E_{mit}) \le 2 \qquad m \in M, t \in T \tag{20}$$

$$S_{mi(t-1)} \le L_{mit} + S_{mit} \qquad m \in M, i \in N,\ t \in T\backslash\{1\} \tag{21}$$

$$S_{mi(t-1)} + E_{mit} \ge S_{mit} \qquad m \in M, i \in N,\ t \in T\backslash\{1\} \tag{22}$$

Equalities (18) couple variables $X, L,$ and E. Inequalities (19) establish that for each representative m, city i and day t, variables L_{mit} and E_{mit} are mutually exclusive. Constraints (20) guarantee that a representative m on day t can select either an open or a closed tour. Constraints (21)–(22) determine the depot node for a representative involved in a closed tour.

(g) *Constraints on no tour routes*

$$S_{m0t} = 0 \qquad m \in M, t \in T \tag{23}$$

$$S_{mit} \ge X_{mi0t} \qquad m \in M, i \in N,\ t \in T \tag{24}$$

$$S_{mi(t-1)} \ge X_{mi0t} \qquad m \in M, i \in N,\ t \in T\backslash\{1\} \tag{25}$$

$$X_{mi0t} = X_{m0it} \qquad m \in M, i \in N,\ t \in T \tag{26}$$

Equalities (23) ensure that fictitious city 0 will never be selected as a depot node. Constraints (24)–(25) prevent inclusion of the fictitious node in daily tours for each representative. Constraints (26) model a no tour route imposing that if representative m goes from city i to the fictitious city, they must return from there the same day.

(h) *Constraints on open tour routes*

$$L_{mit} \le S_{mi(t-1)} \qquad m \in M, i \in N,\ t \in T\backslash\{1\} \tag{27}$$

$$E_{mit} \le S_{mit} \qquad m \in M, i \in N,\ t \in T \tag{28}$$

$$S_{mit} \le \sum_{j\in V} X_{mij(t+1)} \qquad m \in M, i \in N,\ 1 \le t \le t_{\max}-1 \tag{29}$$

$$\sum_{i\in N} S_{mit} = 1 \qquad m \in M, t \in T \tag{30}$$

Constraints (27)–(28) determine the open tours for each representative. Inequalities (29) ensure that representative m departs the depot city on the next day. Constraints (30) assure that each representative stays overnight only in one city.

(i) *Constraints forcing each representative to return campaign base at least every t_{away} days*

$$\sum_{k=t}^{t+t_{away}-1} S_{m1k} \ge 1 \qquad m \in M, 1 \le t \le t_{\max}-t_{away}+1 \tag{31}$$

(j) *Modified Miller–Tucker–Zemlin constraints for subtour elimination*

$$(p+1)(S_{mj(t-1)} + 1 - X_{mijt}) + U_{mjt} \ge U_{mit} + 1 \qquad m \in M, (i,j) \in A,\ t \in T\backslash\{1\} \tag{32}$$

$$U_{mit} \le p+1 \qquad m \in M, i \in V, t \in T \tag{33}$$

$$U_{mit} \le \sum_{j\in N}\sum_{k\in N} X_{mjkt} + 1 \qquad m \in M, i \in V,\ t \in T \tag{34}$$

$$U_{mit} \ge S_{mi(t-1)} \qquad m \in M, i \in N \quad t \in T\backslash\{1\} \tag{35}$$

$$(p+1)(1 - S_{mi(t-1)}) + 1 \ge U_{mit} \qquad m \in M, i \in N \quad t \in T\backslash\{1\} \tag{36}$$

Inequalities (32)–(36) are sub-tour elimination constraints derived from the well-known Miller–Tucker–Zemlin (MTZ) constraints, where variable U_{mit} indicates the rank order in which node i is visited on the day t by representative m. Inequalities (32) represent MTZ conditions. Since the final depot is not known in advance, the variable $S_{mj(t-1)}$ is introduced multiplied by a large constant value implying that if the tour has node j as the final depot (i.e. $S_{mj(t-1)} = 1$), the corresponding constraint is not binding. Inequalities (33)–(34) impose that each U_{mit} variable is bounded in value by the minimum between the maximum number of visits p plus 1 and the number of edges in the tour plus 1. The presence of +1 is related to the fact that the variable U_{mit} associated with the starting node of the tour (i.e. when $S_{mi(t-1)} = 1$) is forced to 1 by constraints (35)–(36).

(k) *Decision variable definition*

$$L_{mit}, E_{mit}, S_{mit}, Z_{mit}, F_{mit} \in \{0,1\} \qquad m \in M, i \in N, t \in T \tag{37}$$

$$X_{mijt} \in \{0,1\} \qquad m \in M, (i,j) \in A \cup A_0, t \in T \tag{38}$$

$$R_{mits} \in \{0,1\} \qquad m \in M, i \in N, t \in T,\ \xi \le s \le t-1 \tag{39}$$

$$U_{mit} \ge 0 \qquad m \in M, i \in V, t \in T \tag{40}$$

4.1. Budget constraints for overnight stays

To further increase the real-life applicability of mRSP, we have also assumed that the representatives have a limited total budget ϕ for daily

overnight accommodations:

$$\sum_{m\in M}\sum_{i\in N} h_i S_{mit} \leq \phi \quad t \in T \tag{41}$$

where h_i is the cost of one night stay [3] in the city $i \in N$.

4.2. Valid inequalities and bounds enforcement

We can tighten model (1)–(40) by introducing valid inequalities and enforcing the bounds on binary decision variables Z, R, L, and E. Since the coefficients of variables R_{mits} in the objective function are all positive, we can increase the model's solution speed by setting the following lower bound for these variables:

$$R_{mits} \geq Z_{mi(t-s)} + Z_{mit} - \sum_{k=t-s+1}^{t-1} Z_{mik} - 1 \quad m \in M, i \in N,\ t \in T,\ \xi \leq s \leq t-1 \tag{42}$$

Variables L_{mit} and E_{mit} take value one if and only if an open tour occurs on a given day. Thus, for closed tour type their values can be forced to zero:

$$L_{mit} \leq 2 - S_{mi(t-1)} - S_{mit} \quad m \in M, i \in N,\ t \in T\backslash\{1\} \tag{43}$$

$$E_{mit} \leq 2 - S_{mi(t-1)} - S_{mit} \quad m \in M, i \in N,\ t \in T\backslash\{1\} \tag{44}$$

Finally, the following constraints enforce the handling of closed tours with only two edges:

$$X_{mijt} + X_{mjit} \leq 1 + S_{mit} + S_{mjt} \quad m \in M,\ i, j \in N,\ (i \neq j),\ t \in T \tag{45}$$

5. Solution methods

Being an extension of the team orienteering problem, the mRSP also belongs to the class of NP-Hard optimization problems, indicating that obtaining an optimal solution in a reasonable CPU time is not guaranteed. We develop a learning-based hyper-heuristic for the mRSP, based on the popular Variable Neighborhood Search (VNS) introduced in Mladenović and Hansen (1997). Instead of targeting a specific problem domain, hyper-heuristics aim to develop a general method able to control low-level heuristics. Hyper-heuristics recently became popular in tackling a wide range of routing problems (see, for instance, Ahmed, Mumford, and Kheiri (2019) and Drake, Kheiri, Özcan, and Burke (2020)). VNS is a metaheuristic based on a finite set of neighborhood structures systematically visited during the search. The method starts with the first (typically small) neighborhood, randomly perturbs the current solution (shaking phase), and locally searches around this perturbation. If no solution is found better than the current one, the search is moved to the next (larger) neighborhood, and the procedure is repeated. Otherwise, the solution becomes the new incumbent and the search is restarted with the initial neighborhood. The algorithm continues to iterate through different neighborhoods until it reaches a predefined stopping criterion. One of the key benefits of VNS is that it can escape local optima without staying trapped in a narrow region of the solution space.

Given the complexity of the mRSP that stems from the number of periods, cities, and travelers, VNS in its basic form falls short of finding high-quality solutions in a reasonable amount of CPU time. Accordingly, we introduce a new hyper-heuristic, named Adaptive Granular Variable Neighborhood Search (AGVNS), that extends VNS by exploiting a simple learning feature, granular moves, and a delayed shaking phase. Let us consider a sequence of neighborhood structures $\mathcal{N}_k$ for k ranging from 1 to k_{max}. In turn, each structure $\mathcal{N}_k$ consists of a sequence of nested neighborhoods $\{N_{r_1^k}, \ldots, N_{r_{max}^k}\}$ defined by a parameter r^k (the radius) ranging from a minimum r_1^k to a maximum r_{max}^k value and controlling the size of each neighborhood move. AGVNS is a hyper-heuristic designed to explore the search space of the k_{max} low-level VNS heuristics, each one constructed over a different neighborhood structure $\mathcal{N}_k$. AGVNS primary focus is on pinpointing the most promising neighborhood structures (*learning phase*) so to avoid their pure sequential application. In each iteration of the subsequent *main phase*, AGVNS pseudo-randomly selects a structure $\mathcal{N}_k$ with a probability proportional to the score yielded by that particular neighborhood structure during the learning phase. It then systematically explores the different nested neighborhoods of that structure. If the search stagnates, meaning that a locally optimal solution is reached, AGVNS responds by choosing a fresh k value and thus moving to a different neighborhood structure. This action, in turn, leads to the identification of a new appropriate low-level VNS heuristic that possibly allows diversifying the search to another area of the solution space. AGVNS enhances VNS by rendering it more versatile. The method is also easily adaptable to other problems including various multi-traveler multi-period routing problems. In the following subsections, we describe the initial solution generation algorithm, the learning and main steps of AGVNS, local search, shaking, neighborhood operators, and granular moves.

5.1. Initial solution generation

Rather than using simple insertion heuristics to build an initial feasible solution, we develop a constructive algorithm that generates good quality routes for each representative by including a promising sequence of cities. The pseudo-code is given in Algorithm 1.

Assuming that the representative m begins her travel in the city i_0^m and considering the time constraints, let ζ_t^m be the set of all cities that the representative m can feasibly visit on the day t. A city j is included in ζ_t^m if visiting j does not violate constraints (6), (16), (31), and $d_{i_0^m j} + a_j < q$. The net profit of a city $j \in N$ at time t for traveler m (Π_{jt}^m) is calculated based on the (*i*) updated reward of the city j on the day t, (*ii*) travel time from city i_0^m to city j ($d_{i_0^m j}$), (*iii*) activity time at city j (a_j), and (*iv*) maximum required time to carry out the next activity ($\arg\max_{k\in\zeta_t^m}\{d_{i_0^m k} + a_k\}$). For each node $j \in \zeta_t^m$, we first calculate a balanced sum of the normalized required cost and time to carry out an activity at node j, i.e., $\frac{d_{i_0^m j}+a_j}{\arg\max_{k\in\zeta_t^m}\{d_{i_0^m k}+a_k\}}$; then, the net profit of node j at time t for traveler m is computed as follows::

$$\Pi_{jt}^m = \frac{\frac{(t+t_{\max})b_j}{t_{\max}}}{\frac{d_{i_0^m j}+a_j}{\arg\max_{k\in\zeta_t^m}\{d_{i_0^m k}+a_k\}}} \tag{46}$$

Once the net profit of the nodes in ζ_t^m is calculated, in Line 8, we sort the set of cities in the increasing order of their net profit (N_{sorted}^m). Then, for each day t, starting from an empty route for traveler m ($Route_t^m$), we assign cities one by one (Line 18 in Algorithm 1), while ensuring that the chain feasibility is held and cities with high rewards are evenly distributed among travelers. In every $Route_t^m$, i_0^m and i_{depot}^m denote the first and last (depot) node of the daily route, respectively. The algorithm assigns the cities to days in T in such a way that cities with higher rewards are visited near the end of the campaign. For those days where the traveler is required to return to the campaign base, we remove the last node(s) in $Route_t^m$ and add it to N_{sorted}^m. Next, the campaign center is added to the end of that day (Lines 24 and 25 in Algorithm 1). While appending a city to $Route_t^m$, the feasibility of the insertion is checked. In case of obtaining a feasible solution, the insertion is performed.

[3] In our implementation, the value of h_i is determined according to the average cost of accommodation in the top 20 hotels for city i identified with Expedia.com by sorting all the hotels in descending order based on the 'Star Rating', with the 'City Center' option checked.

Algorithm 1: Pseudo-code of the initial feasible solution generation algorithm

Input: an mRSP instance
Output: a feasible mRSP solution ▷ $|M| \times t_{\max}$ routes (chains of nodes)
Parameters: $t_{\max}, q$ ▷ number of days and maximum tour duration
$Route^m = \{\}$ ▷ set of $t_{\max}$ routes for traveler m
for $t = 1, \ldots, t_{max}$ **do**
  for $m = 1, \ldots, |M|$ **do**
    compute ζ_t^m
    $N^m_{sorted} \leftarrow$ sorted list of ζ_t^m
    $n_1^m \leftarrow$ the first element of N^m_{sorted}
    $Route_t^m = \{\}$ ▷ the route of traveler m in day t
    if $t = 1$ **then**
      $i_0^m \leftarrow \{1\}$
    else:
      $i_0^m \leftarrow i^m_{depot}$ ▷ denoting the terminal node of the previous day
    end if
    $Route_t^m \leftarrow \{i_0^m\}$ ▷ last day's terminal node is today's depot
    while $time(Route_t^m) < q$ **do**
      append node n_1^m to $Route_t^m$ if it is feasible wrt to constraints (4), (6), (16), (17), and (41).
      remove n_1^m from N^m_{sorted}
      $n_1^m \leftarrow$ the first element of N^m_{sorted}
    end while
    $i^m_{depot} \leftarrow$ the last element of $Route_t^m$
    if $t \,\%\, t_{away} = 0$ **then** ▷ satisfying constraint (31)
      remove the last node(s) in $Route_t^m$ and add it to N^m_{sorted}
      append node 1 to $Route_t^m$
      $i^m_{depot} \leftarrow \{1\}$
    end if
    add $Route_t^m$ to $Route^m$
  end for
end for

5.2. General structure of AGVNS

AGVNS, as shown in Algorithm 2, consists of two steps: a learning step and a main one. The objective of the learning step is to acquire relevant information about the different neighborhood structures $\mathcal{N}_k = \{N_{r_1^k}, \ldots, N_{r^k_{max}}\}$, $k = 1, \ldots, k_{max}$, by learning the effectiveness of their nested operators. During the initial search phase, AGVNS monitors the frequency of solution improvements attributed to each neighborhood structure $\mathcal{N}_k$, $k = 1, \ldots, k_{max}$, and records this information in the set $\Omega = \{\Omega_1, \ldots, \Omega_{k_{\max}}\}$. Subsequently, the second phase uses this information to strategically choose more effective operators in each iteration. The fundamental distinction between AGVNS and a traditional VNS lies in the former being a hyper-heuristic controlling when and how different VNSs have to be used.

Learning step: At the beginning of the learning step (Lines 4 and 5 in Algorithm 2), we first define the set of neighborhood structures $\mathcal{N}_k$ for $k = 1, \ldots, k_{max}$ and the corresponding nested neighborhoods $\{N_{r_1^k}, \ldots, N_{r^k_{max}}\}$ controlled by a radius r^k ranging from r_1^k to r^k_{max}. Then we initialize the value of the score Ω_k for each neighborhood operator in the set Ω to a common value ω. This base value is set to 20 based on preliminary experiments. AGVNS starts with setting the initial feasible solution S_0 obtained by Algorithm 1 as the global best (Line 6 in Algorithm 2). While the number of iterations without improvement ($iter$) is less than parameter $iterMaxl$, the main body of the learning step is repeated iteratively (Lines 9–27 in Algorithm 2). In this step, if the local search procedure corresponding to one of the nested neighborhoods associated with structure $\mathcal{N}_k$ finds a solution S'' better than the incumbent solution S^*, the score of the neighborhood operator (Ω_k) is updated accordingly (Line 16 in Algorithm 2) by adding the current score to the weighted net gain of neighborhood operator k as given in Eq. (47). Here, φ is the normalizing coefficient we set equal to 1/100. Other score-updating techniques for adaptive VNS

Algorithm 2: Pseudo-code of AGVNS

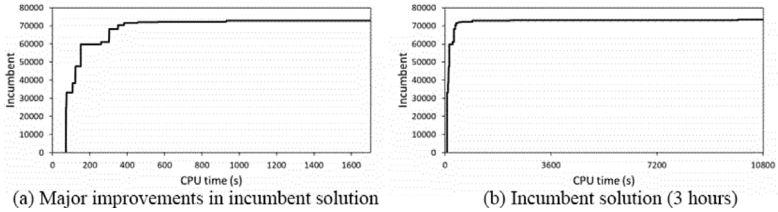
1: **Input:** an initial feasible solution S_0 obtained by Algorithm 1
2: **Output:** a feasible mRSP solution S^*
3: **Parameters:** k_{max}, r^k_{max}, $iterMaxl$, $iterMax$, ω, φ
4: Define neighborhood structures $\mathcal{N}_k = \{N_{r_1^k}, ..., N_{r^k_{max}}\}$ with $k = 1, ..., k_{max}$
5: Initialize to ω all the scores of the set $\Omega = \{\Omega_1, \Omega_2, ..., \Omega_{kmax}\}$
6: $S^* \leftarrow S_0$
7: $k \leftarrow 1$
8: $iter \leftarrow 0$
9: **while** $iter \leq iterMaxl$ **do** ▷ learning step starts
10: **for** $k = 1, \ldots, k_{max}$ **do**
11: $r^k \leftarrow r_1^k$
12: **while** $r^k \leq r^k_{max}$ **do** ▷ VNS based on $\mathcal{N}_k$ neighborhood starts
13: $S' \leftarrow Shaking(S^*, r^k)$
14: $S'' \leftarrow LocalSearch(S', r^k)$
15: **if** $Z(S'') > Z(S^*)$ **then**
16: $\Omega_k = \Omega_k + \varphi\omega(Z(S'') - Z(S^*))$ ▷ updating scores of operators
17: $S^* \leftarrow S''$
18: $Z(S^*) \leftarrow Z(S'')$
19: $r^k \leftarrow r_1^k$
20: $iter \leftarrow 0$
21: **else:**
22: $iter+ = 1$
23: update r^k ▷ move to an operator with a larger radius
24: **end if**
25: **end while**
26: **end for**
27: **end while** ▷ learning step finishes
28: $iter \leftarrow 0$
29: $S \leftarrow S^*$
30: **while** $iter \leq iterMax$ **do** ▷ main step starts
31: $k \leftarrow$ Pseudo-RandomSelection(Ω)
32: $r^k \leftarrow r_1^k$
33: **while** $r^k \leq r^k_{max}$ **do** ▷ VNS on selected $\mathcal{N}_k$ structure starts
34: $S' \leftarrow LocalSearch(S, r_k)$
35: **if** $Z(S') > Z(S^*)$ **then**
36: $S^* \leftarrow S'$
37: $Z(S^*) \leftarrow Z(S')$
38: $iter \leftarrow 0$
39: $r^k \leftarrow r_1^k$
40: **else:**
41: $iter+ = 1$
42: update r_k ▷ move to an operator with a larger radius
43: **end if**
44: **end while**
45: $S \leftarrow Shaking(S^*)$
46: **end while** ▷ main step finishes

are introduced in the literature (e.g., Sze, Salhi, and Wassan (2017)). However, they are usually computationally more expensive than our method.

$$\Omega_k = \Omega_k + \varphi\omega(Z(S'') - Z(S^*)) \tag{47}$$

When analyzing each neighborhood structure $\mathcal{N}_k$, the method works as a basic VNS with the nested neighborhoods parametrized by their size r^k (radius). Thus, if the objective of local minimum S'' is greater than the incumbent best solution S^*, the learning step of AGVNS accepts the new global best ($S^* \leftarrow S'$) and the search reverts to the initial nested neighborhood r_1^k (Line 19); otherwise the search will explore the subsequent nested neighborhood (line 23). Once all neighborhood structures have been explored (Lines 10-26 in Algorithm 2), process continues until counter $iter$ reaches the value $iterMaxl$.

Main step: Hyper-heuristic AGVNS differs from a basic VNS in two significant ways: (*i*) rather than exploring neighborhood operators sequentially, it selects them according to Ω_k scores obtained in the previous step, and for each of them runs a VNS making use of nested Local Search Operators (but for a few cases where VNS reduces to



(a) Major improvements in incumbent solution (b) Incumbent solution (3 hours)

Fig. 3. Improvement of incumbent solutions: GUROBI applied to instance *m2c20t10*.

a simple local search) (*ii*) the shaking approach (not depending on the current neighborhood structure) is performed only when the local search fails to improve the incumbent solution. The algorithm adjusts its search strategy by re-applying the local search method around the incumbent solution to intensify the search, explore that area more thoroughly, and potentially find an even better solution. At the beginning of each iteration of the main step, AGVNS selects an operator using the cumulative probability. In particular, a pseudo-random selection is applied on the set Ω where neighborhoods with higher scores have higher chances of being selected (Line 31 in Algorithm 2). The probability of a neighborhood structure k to be chosen is shown in expression (48).

$$p_k = \frac{\Omega_k}{\sum_{k=1}^{k_{max}} \Omega_k} \tag{48}$$

The algorithm iterates the local search operators within the selected neighborhood structure k as long as it improves the incumbent solution. Once it fails to improve the solution, a *Shaking Procedure* is applied (Line 45 in Algorithm 2), and the search restarts with another pseudo-random selection of neighborhood operators (Line 31 in Algorithm 2).

Local Search Operators (LSO): The algorithm uses several different neighborhood structures, each with a different level of exploration (radius values) representing the number of reachable available moves and their size. Inside a local search algorithm, the current solution S is improved by exploring the neighborhood using the best improvement approach. We describe the LSOs hereafter:

1. **Neighborhood structure** $\mathcal{N}_1$: *Weighted Swap*. This is a simple neighborhood with $r^1_{max} = 1$. Since a high-quality solution of mRSP tends to hold activities in big cities in the last days of the campaign, due to the increasing reward function, this operator randomly selects a city with $b_j > \frac{\sum_{i\in N} b_i}{|N|}$ from the period $\{1, \ldots, \lfloor \frac{t_{\max}}{2} \rfloor\}$ in the current solution and replaces it with a randomly selected city with $b_j \leq \frac{\sum_{i\in N} b_i}{|N|}$ in period $\{\lfloor \frac{t_{\max}}{2} \rfloor + 1, \ldots, t_{\max}\}$.
2. **Neighborhood structure** $\mathcal{N}_2$: *Chain Removal-Separate Insertion*. A number, ranging from $r^2_1 = 1$ to $r^2_{max} = 3$ (with step 1), of consecutive cities, are removed from randomly selected routes of a randomly selected representative. The cities are then split and inserted into the cheapest positions of other routes separately.
3. **Neighborhood structure** $\mathcal{N}_3$: *Shuffle*. A number ranging from $r^3_1 = 2$ to $r^3_{max} = 4$ (with step 1) of non-consecutively located cities are chosen randomly and their visit orders are shuffled randomly.
4. **Neighborhood structure** $\mathcal{N}_4$: *2-1 Swap*. A fixed number equal to $r^4_{max} = 2$ cities are randomly removed from a route and reinserted in two different routes.
5. **Neighborhood structure** $\mathcal{N}_5$: *1-Add*. A single non-visited city ($r^5_{max} = 1$) is inserted on the last possible day of a randomly selected representative to possibly improve the objective function value.
6. **Neighborhood structure** $\mathcal{N}_6$: *Swap Unvisited*. A number equal to $r^6 \in \{1, 2, 3, 4\}$ of cities is randomly selected and removed from the route of randomly selected travelers. An equivalent number of previously unvisited cities are then inserted into the cheapest positions in the routes.
7. **Neighborhood structure** $\mathcal{N}_7$: *Block-Exchange*. A chain of cities from the donor route will be replaced by a different chain of cities from the receiver route. The number of cities in each change will be randomly chosen between $r^7_1 = 2$ to $r^7_{max} = 4$.

Shaking: AGVNS differs from traditional VNS in that the shaking process is only employed when the local search fails to improve the current best solution. This strategy allows for a more intensified search around the best solution found so far. Our shaking procedure uses a destroy–repair framework, where randomly selected travelers and randomly selected cities are removed from the current solution, with those whose $b_j \leq \frac{\sum_{i\in N} b_i}{|N|}$ have a higher chance of being selected. The removal process is done on randomly selected days. Next, randomly selected cities from the pool of unvisited cities are reinserted back into the solution while respecting feasibility criteria.

Granular neighborhoods: AGVNS consists of large neighborhoods which makes it computationally expensive to explore all of them. To address this issue, we suggest a neighborhood reduction scheme based on granular moves used throughout the main step of the algorithm. This mechanism significantly speeds up the search process by only focusing on promising moves and eliminating unpromising ones. Since when exchanging cities in different routes, it is crucial to evaluate the exchange's impact on the cost and time, the reduction strategy works as follows: for each city $i \in N$, we first determine a set of potential neighboring cities that could be inserted next to i. We then restrict the search to those cities only. For this purpose, we build a binary 3-dimensional matrix $\mathbb{M}$, where $\mathbb{M}[i_1, i_2, i_3] = 1$ if $min\{d_{i_1,i_2}, d_{i_1,i_3}\} \leq \gamma$ (city i_1 can be inserted between i_2 and i_3), and 0 otherwise. Parameter d_{i_1,i_2} shows the symmetric travel time between cities i_1 and i_2, and γ is the granularity threshold set to 300 min. The use of the $\mathbb{M}$ greatly enhances the efficiency of the neighborhood search by eliminating unpromising moves.

6. Computational study

In this section, we discuss the computational results of the AGVNS algorithm on instances obtained from the real world in Turkey. We compare the solutions of AGVNS against the commercial MILP solver GUROBI and a variant of the algorithm FDOR adapted from the literature. We carried out the experiments on an MSI GP66 laptop with 11th Gen Intel(R) Core(TM) i7 (11800H model), 2.30 GHz, 16 GB RAM and running a 64-bit Windows 10 Pro. We implemented all algorithms in Python 3.8. The commercial solver GUROBI 9.5 is employed to solve the MILP model with a time limit of 3 hours. Apart from enabling multithreading, the rest of GUROBI's parameter settings are used as default. All CPU times are reported in seconds.

6.1. Dataset

This paper is the first to introduce the mRSP. We test the performance of the proposed model and developed algorithms by modifying

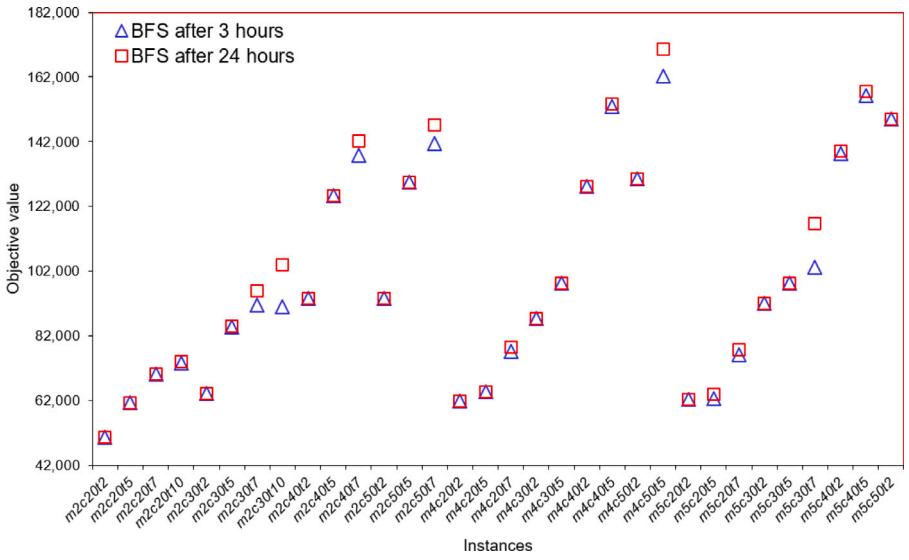


Fig. 4. Comparison of lower bounds obtained by GUROBI in 3 hours and 24 hours of CPU time.

a subset of benchmark instances developed for the RSP (Shahmanzari et al., 2020) with a maximum of 5 travelers, 50 cities, and 10 days. These instances are generated based on real-world city locations extracted from Google Maps and travel times calculated based on the minimum of terrestrial roads and available flight times (or a combination of both). The selection of cities within each mRSP instance is based on a subset of cities derived from the largest RSP instance. In contrast to the original RSP, which involves a single traveler, mRSP involves the inclusion of multiple travelers. The naming convention of instances provides full information about them. For example, instance *m2c50t7* denotes that it has 2 representatives (*m2*), 50 cities (*c50*), and seven days (*t7*). The total number of instances is 48. The maximum number of meetings per day (p) and the maximum tour duration (q) for each representative are limited to 4 and 14 hours, respectively. The original RSP instances divide all cities into three categories according to their importance concerning the political party: big cities, medium cities, and small cities. In our implementation, we allow up to 3 activities in big cities, 2 activities in medium cities, and only one activity in small cities during the campaign period. We set the maximum number of days out of the campaign center (t_{away}) and the minimum number of days during which a representative is restricted from making repeat visits to the same city (ξ) to 5 and 3 days, respectively. For sensitivity analysis purposes, we also generated multiple instances where parameters indicating the number of meetings in big cities and medium cities, t_{away}, and ξ (being employed in constraints (6)) vary. All developed instances are publicly available at (https://shahmanzar.ir/mRSP.html)

6.2. Calculation of rewards

We computed the basic reward b_i for each city $i \in N$ as $b_i = \theta_i(\lambda + \frac{Pop_i}{min.Pop.})$ where λ is a fixed reward, Pop_i is the population of the city and $min.Pop.$ is the minimum population out of all cities. Parameter θ_i is the significance score of city i, with values ranging from 1 to 4. To determine it, we simulated the election process in Turkey based on the recorded vote counts from the 2018 election. Our simulation successfully replicated the seat distribution for all 85 electoral regions in Turkey, indicating the accuracy of our methodology. Following this, we assessed each city's political standing by adjusting the number of votes for the political party in that region by both decreasing and increasing it by 15%. θ_i is set to 1 if ±15% variation in the number of the votes of the political party does not change the number of the political party's seats. If a 15% decrease or increase in the number of the votes of a political party impacts the number of the political party's seats negatively or positively, then θ_i is set to 2 and 3, respectively. Finally, when both a 15% increase and a 15% decrease in the number of votes lead to a corresponding increase and decrease in the number of party seats, respectively, we set the value of θ_i to 4.

6.3. AGVNS parameter settings

In the learning phase of AGVNS, φ is a normalizing coefficient used to determine the amount of updated score of the neighborhood structures. Parameters $iterMaxl$ and $iterMax$ denote the stopping criteria each expressed as the number of iterations without improvement in the learning phase and in the main one, respectively. We have conducted extensive preliminary experiments involving the manipulation of various combinations of these three parameters on 12 pilot test instances consisting of 2 and 3 representatives, 30 and 40 cities, and 5, 7, and 10 days. The goal was to identify the robust parameter configurations that would result in the best trade-off between computational time and solution quality for AGVNS. The tested values for all three parameters are as follows: $iterMaxl = \{100, 200, 300, 400\}$, $iterMax = \{500, 700, 1000, 1200, 1500\}$, and $\varphi = \{0.01, 0.1, 0.5\}$. To determine the best value for any given parameter, the other parameters were temporarily set to their initial values ($iterMaxl = 100$, $iterMax = 700$, and $\varphi = 0.01$) or the values that had previously been identified as best. We performed ten runs for each instance. We finally select $iterMaxl = 200$, $iterMax = 1000$, and $\varphi = 0.1$.

6.4. Computational results using the commercial solver

To evaluate the effectiveness of the developed mathematical model, we first present the results obtained by GUROBI for all mRSP instances. The performance of the commercial solver on all instances with 3-hour runs is reported in the left part of Table 6. Boldface figures indicate proven optimality attained by the solver. Column LB refers to the lower bound (best feasible solution). Column Opt Gap% hosts the relative optimality gap as reported by GUROBI. Finally, t(s) provides the CPU time in seconds.

Fig. 3 illustrates the progression of the incumbent solution for one instance. Since the results for other instances exhibit similar patterns, we make them available in Table E1 in the E-Companion. The image on the right depicts the values of the incumbent solution obtained after three hours of CPU time. The figure on the left serves to provide a more detailed examination of the data in the right figure by magnifying the regions of the chart that exhibit significant improvement. The results

imply that the incumbent solution of the commercial solver barely improves after three hours. We also let GUROBI run for 24 h and report the result in Fig. 4. The solver fails to find even a feasible solution for 16 instances after running for one day. Apart from 3 instances (*m2c30t7*, *m2c30t10*, and *m5c30t7*), GUROBI was not able to improve the lower bound of the problem significantly. More specifically, letting the commercial solver run for 21 hours more results in a 1.84% average improvement on the 48 instances. Additionally, among 32 instances for which GUROBI was not able to obtain an optimal solution in 3 h, 7 instances (*m2c20t7*, *m2c40t5*, *m2c50t5*, *m4c20t5*, *m4c30t2*, *m4c30t5*, *m4c50t2*) did not improve at all. However, we observe two significant improvements in *m2c30t10* and *m5c30t7* by 15.5% and 13.2%, respectively.

Finally, we evaluate the impact of valid inequalities (see Section 4.2) on tightening mRSP formulation. To this aim, we conduct a separate computational experiment where all valid inequalities are turned off (VIs off). In Table 5, we summarize the comparison of the latter analysis with the case where all valid inequalities are included from scratch without separation. Figures in boldface signify optimality obtained by the solver. The results indicate that valid inequalities are indeed very effective. By including them, the average optimality gap is reduced from 9.28% to 7.48%. In total, the lower bound for 23 instances has improved. We observed drastic improvements in the optimality gap of some instances (e.g., *m2c20t7*, *m2c40t7*, *m4c40t5*, and *m5c40t5* with 5.43%, 7.41%, 8.29%, and 6.06% improvements, respectively). In one instance (*m5c30t7*), GUROBI was not able to achieve a feasible solution after running for 3 hours without valid inequalities but it got it when valid inequalities have been added. The last part of Table 5, named Improvement, provides the percentage improvement in terms of LB and time for all instances. The results show a substantial increase both in terms of solution effectiveness and efficiency.

6.5. Computational results on AGVNS

Table 6 displays the objective values of the 48 instances for both GUROBI and AGVNS along with corresponding gaps and CPU times in seconds. The column header Opt Gap% indicates the relative optimality gaps reported by GUROBI, whereas Gap%, is calculated as $100\times\frac{AGVNS.Obj-LB}{AGVNS.Obj}$. Column BKS Δ% reports the percentage of improvement conducted by AGVNS for the best-known solutions, which could be either the optimal solution or the lower bound obtained by GUROBI within 3 hours of CPU time. The boldface figures under the AGVNS section signify the best-known solution. When calculating the average gaps of AGVNS, we omitted those instances for which no solution was obtained by GUROBI after 3 hours. These results prompt a couple of observations. AGVNS not only successfully obtained the optimal solution in all instances where GUROBI got it, but it also improved upon the best feasible solution reported by GUROBI in 39 instances. In some instances (e.g., *m2c30t10*), the best feasible solution is improved by 14.9% in remarkably shorter CPU time. While the average CPU time for GUROBI is 9375 s, AGVNS finds either the same or better solutions in 115.7 s on average, indicating a 98.76% decrease. Taking those 7 instances with optimal solutions into account, the average runtime of the commercial solver is 1029 s, whereas AGVNS attained the same solutions in 43.9 s on average.

Fig. 5 displays the best feasible solutions of GUROBI and AGVNS to showcase the effectiveness of our method. Due to scale issues, we did not include AGVNS results for two instances *m5c50t7* and *m5c50t10* as they were significantly better than GUROBI results. To ensure a fair comparison, we restricted our analysis to the objective values of instances where GUROBI can produce a feasible or optimal solution. The AGVNS method's superiority is most evident in large-scale instances with 5 representatives due to their complex nature. While GUROBI struggles to generate high-quality solutions in these instances,

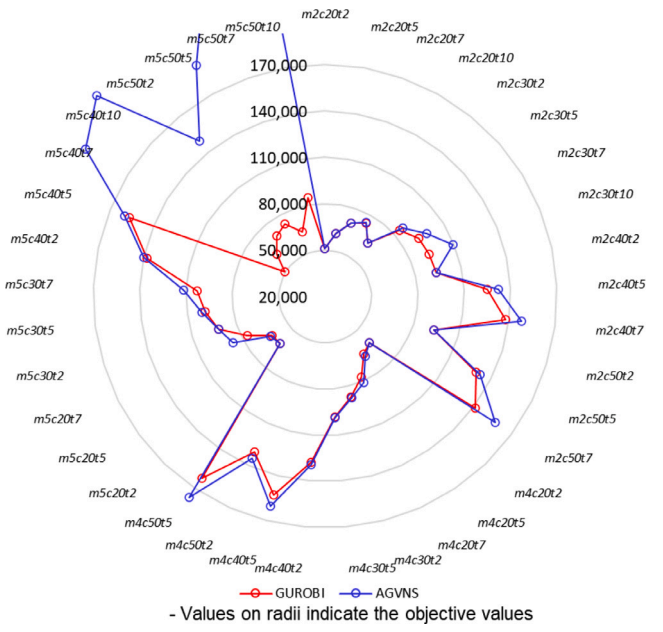


Fig. 5. Solutions value of AGVNS Algorithm and Gurobi solutions on mRSP instances.

AGVNS consistently produces significantly superior solutions, with an average CPU time of 548 s.

Fig. 6 depicts the solutions for the instance *m2c50t7* obtained by AGVNS (left) and GUROBI (right) with an optimality gap of 14.9%. The day of each visit, over the one-week planning, is indicated above the meeting sign. GUROBI managed to schedule 37 meetings, while AGVNS successfully scheduled 46 meetings. Moreover, GUROBI's solution included 6 repeat visits, while AGVNS's campaign involved 5 repeat visits. This can be attributed to the fact that AGVNS allocated more time exploring other cities, particularly in the eastern region of the country. Both solutions incorporate closed and open tours, whereas the "no-tour" type was never selected. Typically, these types of routes are present in solutions with a small number of cities and a longer campaign period, such as instance *m2c20t10*. A noteworthy observation regarding the AGVNS solution pertains to the itinerary of the two-party representatives who daily concentrate their efforts on distinct regions of the country. For example, during the initial phase of the campaign, the red representative roams the western region of the country, while the black one explores cities in the eastern region. In contrast, the GUROBI solution exhibits a different pattern, characterized by multiple jumps between cities that are situated at considerable distances from each other.

6.6. Comparison of AGVNS with a state-of-the-art matheuristic

To further evaluate the performance of AGVNS, we also benchmark it with a modified version of the method Finding Daily Optimal routes (FDOR), a matheuristic introduced in Shahmanzari et al. (2020) for a single vehicle problem. FDOR breaks down the original MILP formulation into several subproblems corresponding to the number of campaign days. It involves two main phases: city selection and route generation. In the city selection phase, a set of promising cities are selected pseudo-randomly. In the route generation phase, an integer program is solved to construct the optimal route from the chosen cities. The model for each day is solved based on the selected subset of cities determined in the first phase of the matheuristic. A more detailed explanation of the modified FDOR (MFDOR) along with a brief description of the mathematical model used for each day can be found in E-Companion. Basically, on each day, MFDOR pseudo-randomly

Table 5
Impact of valid inequalities.

Instance	mRSP without VIs			mRSP with VIs			Improvement	
	LB	Gap%	t(s)	LB	Gap%	t(s)	LB Δ%	t Δ%
m2c20t2	**50719.2**	0.00	566	**50719.2**	0.00	425	0.00	33.06
m2c20t5	**61294.4**	0.00	5729	**61294.4**	0.00	4436	0.00	29.15
m2c20t7	68595.7	7.58	10800	70044.3	2.15	10800	2.11	0.00
m2c20t10	73383.1	10.67	10800	74566.4	6.33	10800	1.61	0.00
m2c30t2	**64055.4**	0.00	658	**64055.4**	0.00	545	0.00	20.86
m2c30t5	83767.4	7.84	10800	84626.4	5.67	10800	1.03	0.00
m2c30t7	88267.6	22.5	10800	91401.1	17.10	10800	3.55	0.00
m2c30t10	89297.2	36.51	10800	92837.7	28.8	10800	3.96	0.00
m2c40t2	**93606.7**	0.00	198	**93606.7**	0.00	134	0.00	47.32
m2c40t5	124585.5	8.25	10800	125376.7	6.87	10800	0.64	0.00
m2c40t7	130272.2	21.21	10800	137865.2	13.80	10800	5.83	0.00
m2c40t10	–	–	10800	–	–	10800	–	–
m2c50t2	**93606.7**	0.00	328	**93606.7**	0.00	156	0.00	109.58
m2c50t5	129112.1	7.03	10800	129452.8	6.60	10800	0.26	0.00
m2c50t7	140466.8	16.21	10800	141311.3	14.90	10800	0.60	0.00
m2c50t10	–	–	10800	–	–	10800	–	–
m4c20t2	**61781.4**	0.00	1021	**61781.4**	0.00	548	0.00	86.35
m4c20t5	64646.3	7.22	10800	64723.0	5.47	10800	0.12	0.00
m4c20t7	77050.2	17	10800	77070.4	13.90	10800	0.03	0.00
m4c20t10	–	–	10800	–	–	10800	–	–
m4c30t2	87430.7	3.78	10800	87430.7	3.61	10800	0.00	0.00
m4c30t5	98247.5	6.07	10800	98258.7	5.13	10800	0.01	0.00
m4c30t7	–	–	10800	–	–	10800	–	–
m4c30t10	–	–	10800	–	–	10800	–	–
m4c40t2	126011	6.93	10800	128057.9	5.20	10800	1.62	0.00
m4c40t5	142525.9	15	10800	152928.6	6.71	10800	7.30	0.00
m4c40t7	–	–	10800	–	–	10800	–	–
m4c40t10	–	–	10800	–	–	10800	–	–
m4c50t2	130106.6	6.77	10800	130504.8	6.19	10800	0.31	0.00
m4c50t5	151864.1	17.8	10800	162275.3	10.20	10800	6.86	0.00
m4c50t7	–	–	10800	–	–	10800	–	–
m4c50t10	–	–	10800	–	–	10800	–	–
m5c20t2	**62402.4**	0.00	1064	**62402.4**	0.00	961	0.00	10.75
m5c20t5	61079.1	12.93	10800	62476.8	9.54	10800	2.29	0.00
m5c20t7	76045.2	20.98	10800	76245.9	17.40	10800	0.26	0.00
m5c20t10	–	–	10800	–	–	10800	–	–
m5c30t2	91859.8	1.26	10800	91914.8	1.02	10800	0.06	0.00
m5c30t5	94758	11.35	10800	98177.1	7.47	10800	3.61	0.00
m5c30t7	–	–	10800	103154.6	28.80	10800	–	–
m5c30t10	–	–	10800	–	–	10800	–	–
m5c40t2	138190.4	6.11	10800	138235.8	6.02	10800	0.03	0.00
m5c40t5	148848	13.63	10800	156392.3	7.57	10800	5.07	0.00
m5c40t7	–	–	10800	–	–	10800	–	–
m5c40t10	–	–	10800	–	–	10800	–	–
m5c50t2	146647.28	4.18	10800	148886.68	2.78	10800	1.53	0.00
m5c50t5	–	–	10800	–	–	10800	–	–
m5c50t7	–	–	10800	–	–	10800	–	–
m5c50t10	–	–	10800	–	–	10800	–	–
Average	98403.9	9.28	9424	100268.3	7.48	9375	1.57	10.87

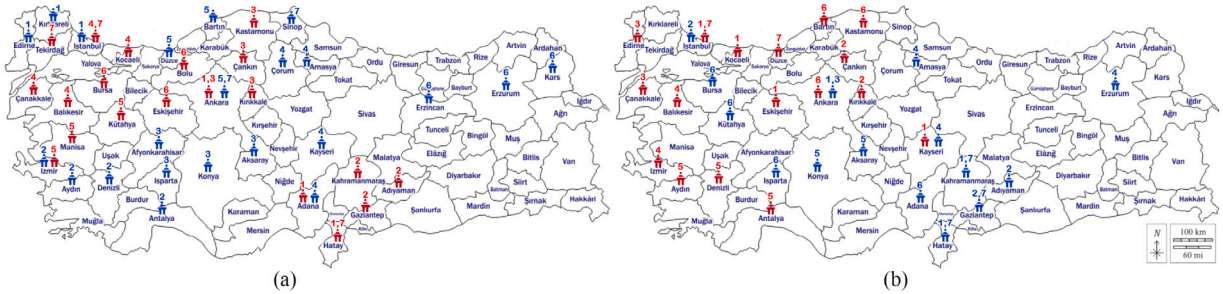


Fig. 6. The solutions generated by AGVNS (a) and GUROBI (b) for the instance *m2c50t7*.

selects a set of promising cities by considering the time-dependent (increasing) nature of their profits. Next, it solves a restricted formulation of the original MILP only including such selected cities. Fig. 7 depicts the comparison of results obtained by the two methods, indicating that AGVNS outperforms MFDOR in terms of solution effectiveness. This divergent performance can be partially explained by the myopic evaluation characterizing MFDOR, where decisions are made locally over each day without considering the time horizon. On the contrary,

Table 6
Comparison of results obtained by AGVNS and GUROBI.

Instance	GUROBI			AGVNS			
	LB	Opt Gap%	t(s)	Obj	Gap%	t(s)	BKS Δ%
m2c20t2	**50719.2**	0	425	**50719.2**	0	11.88	0
m2c20t5	**61294.4**	0	4436	**61294.4**	0	12.98	0
m2c20t7	70044.3	2.15	10800	**70044.3**	0	18.26	0
m2c20t10	74566.4	6.33	10800	**74919**	0.47	25.08	0.5
m2c30t2	**64055.4**	0	545	**64055.4**	0	14.3	0
m2c30t5	84626.4	5.67	10800	**87029.8**	2.84	18.7	2.8
m2c30t7	91401.1	17.1	10800	**97232.5**	6.38	28.38	6
m2c30t10	92837.7	28.8	10800	**109148.1**	17.57	66.66	14.9
m2c40t2	**93606.7**	0	134	**93606.7**	0	16.72	0
m2c40t5	125376.7	6.87	10800	**132523.2**	5.7	33.66	5.4
m2c40t7	137865.2	13.8	10800	**148246.5**	7.53	45.76	7
m2c40t10	–	–	10800	**155793.6**	–	78.76	–
m2c50t2	**93606.7**	0	156	**93606.7**	0	24.64	0
m2c50t5	129452.8	6.6	10800	**132494.9**	2.35	28.6	2.3
m2c50t7	141311.3	14.9	10800	**157251.2**	11.28	54.12	10.1
m2c50t10	–	–	10800	**186290.7**	–	89.54	–
m4c20t2	**61781.4**	0	548	**61781.4**	0	38.06	0
m4c20t5	64723	5.47	10800	**66845.9**	3.28	53.24	3.2
m4c20t7	77070.4	13.9	10800	**81401.7**	5.62	82.28	5.3
m4c20t10	–	–	10800	**81940.1**	–	117.7	–
m4c30t2	87430.7	3.61	10800	**88025.3**	0.68	34.32	0.7
m4c30t5	98258.7	5.13	10800	**98759.8**	0.51	54.34	0.5
m4c30t7	–	–	10800	**107495**	–	79.42	–
m4c30t10	–	–	10800	**139527.3**	–	186.56	–
m4c40t2	128057.9	5.2	10800	**129133.6**	0.84	48.4	0.8
m4c40t5	152928.6	6.71	10800	**160253.9**	4.79	87.56	4.6
m4c40t7	–	–	10800	**173161**	–	196.68	–
m4c40t10	–	–	10800	**191206.6**	–	228.36	–
m4c50t2	130504.8	6.19	10800	**135085.5**	3.51	93.72	3.4
m4c50t5	162275.3	10.2	10800	**176782.7**	8.94	140.14	8.2
m4c50t7	–	–	10800	**185448.2**	–	221.98	–
m4c50t10	–	–	10800	**225238.1**	–	340.34	–
m5c20t2	**62402.4**	0	961	**62402.4**	0	186.34	0
m5c20t5	62476.8	9.54	10800	**63707.6**	1.97	303.38	1.9
m5c20t7	76245.9	17.4	10800	**86691.6**	13.7	348.7	12
m5c20t10	–	–	10800	**89332**	–	623.7	–
m5c30t2	91914.8	1.02	10800	**91914.8**	0	207.24	0
m5c30t5	98177.1	7.47	10800	**100258.5**	2.12	244.64	2.1
m5c30t7	103154.6	28.8	10800	**111747.4**	8.33	476.74	7.7
m5c30t10	–	–	10800	**129111.4**	–	1045.22	–
m5c40t2	138235.8	6.02	10800	**139977.5**	1.26	247.72	1.2
m5c40t5	156392.3	7.57	10800	**160098.7**	2.37	455.18	2.3
m5c40t7	–	–	10800	**201990.4**	–	944.46	–
m5c40t10	–	–	10800	**217072.4**	–	1044.72	–
m5c50t2	148886.68	2.78	10800	**149139.8**	0.17	374.44	0.2
m5c50t5	–	–	10800	**191170.5**	–	490.6	–
m5c50t7	–	–	10800	**251618.5**	–	962.94	–
m5c50t10	–	–	10800	**257170.2**	–	1463	–
Average	100268.3	7.48	9375	127494.7	3.50	115.7	3.30

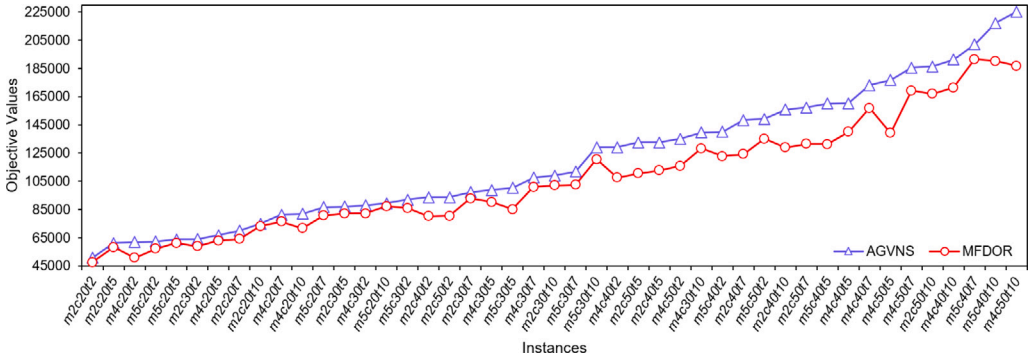


Fig. 7. Comparison of the results obtained by AGVNS and MFDOR.

AGVNS incorporates the features of the reward mechanism for the whole planning horizon.

7. Managerial insights and real-life implications

The main features of the studied multi-period multi-vehicle problem are the time dependency of rewards, the presence of different tour types, and the link among representatives when visiting the same city multiple times over the planning horizon. The latter characteristic also strongly differentiates this problem from the single-vehicle special case. In this section, we provide rules of thumb for decision-maker managers involved in similar problems. In particular, we revolve around (*i*) the type of solution algorithms to use; (*ii*) the advantages of allowing different types of tours against the traditional use of only closed tours; (*iii*) the impact of time-dependent reward on the number and frequency of visited cities. In the following analysis, we included only the 7 instances where GUROBI was able to obtain an optimal solution (*m2c20t2*, *m2c20t5*, *m2c30t2*, *m2c40t2*, *m2c50t2*, *m4c20t2*, and *m5c20t2*).

7.1. Relevance of look-ahead heuristics

One key managerial insight, that comes directly from comparing AGVNS and MFDOR results, is that traditional routing algorithms for multi-period problems based on a myopic rolling horizon mechanism with no lookahead inclusion of future information are not suitable for addressing problems similar to mRSP. Routing algorithms with a rolling horizon mechanism as MFDOR (i.e., Faugère, Klibi, White III, and Montreuil (2022)) are typically designed to address problems in which the rewards, costs, or service times associated with nodes remain constant over time. These algorithms plan routes in a myopic manner based on the current state of the problem and available information often missing the potential benefits of optimizing routes over a longer planning horizon. In mRSP, however, we introduce a level of complexity that algorithms such as MFDOR may struggle to handle effectively given that rewards for visiting cities change over time and also depend on previous visits by other representatives. Some cities may become more attractive as the campaign progresses, while others may lose their appeal. Such algorithms are not inherently equipped to adapt to such dynamic landscapes. In contrast, problems similar to mRSP benefit from methods based on a global perspective that considers the entire campaign duration as AGVNS that can identify strategic patterns, such as the optimal schedule of city visits over the planning horizon, which may not be evident within the narrow scope of a rolling horizon. Additionally, most of the time, the rolling horizon algorithms may lead to inefficient resource utilization, as they may fail to exploit opportunities for consolidating visits to cities with increasing rewards over time. This suboptimal resource allocation can result in the generation of inferior solutions. Time-dependent reward dynamics require a more adaptive approach that can quickly adjust routes based on evolving circumstances. This insight is particularly important in the context of election campaign planning, where maximizing the impact of meetings is critical. A performing algorithm like AGVNS must account for the dynamic changes in rewards based on the day of the visit and the recency of previous visits.

To empirically demonstrate AGVNS's superiority over MFDOR in problems with time-dependent reward mechanisms, we conduct a comparative analysis on instances where optimal solutions are available. The analysis involves comparing both methods by applying a 10-second stopping rule and examining the obtained results. For MFDOR solutions, we calculate the objective function at the point where the solution was obtained. The results are depicted in Fig. 8. We observe that AGVNS consistently generates higher-quality solutions when compared to MFDOR. Our findings suggest that AGVNS exhibits a better ability to approximate the final optimal solution within a limited time frame when contrasted with MFDOR.

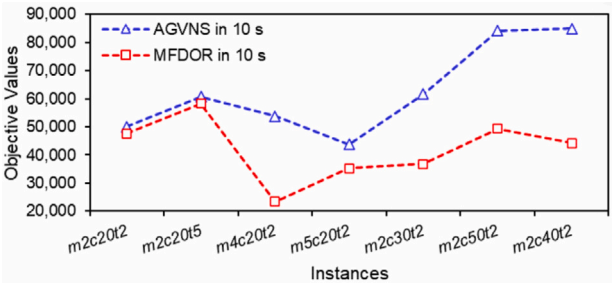


Fig. 8. Comparison of AGVNS and MFDOR under a 10-second stopping rule.

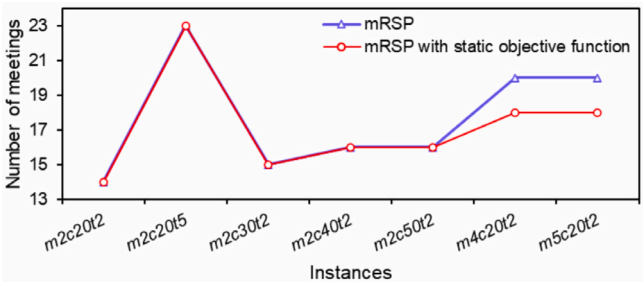


Fig. 9. Value of time-dependent objective function.

7.2. The value of time-dependent rewards

In this section, we compare the solutions conveyed by mRSP with those obtained by eliminating the time dependency from the original model (Static mRSP). The objective function of Static mRSP is formulated by setting $\delta_t = 1$, and thus $b_{it} = b_i, \forall t \in T$, while penalty function α_s, where s is the number of days elapsed since the last visit, remains unchanged.

Due to this fundamental adjustment, the objective function values of the two problems cannot be directly compared. Consequently, our analysis focuses solely on the changes in the number of meetings during the campaign. The results are illustrated in Fig. 9. We observe that the value of including a time-dependent objective function is higher when the number of political representatives increases (compare results for instances *m4c20t2* and *m5c20t2*). This is because, under a static reward mechanism, political representatives have minimal incentive to strategically plan the future to gain more rewards. Our model offers campaign planners a valuable tool for incorporating more realistic assumptions into their decision-making processes. This is supported by comparing the itinerary (cities visited) obtained from the two formulations in the two instances *m4c20t2* and *m5c20t2*, where the resulting number of meetings is different. In the case of *m4c20t2*, we observe that the original model holds two additional meetings in the cities of Artvin and Balikesir. These cities hold significant strategic importance for the selected political party, as evidenced by the recent presidential election results in Turkey. Artvin, for instance, saw the selected party receiving 46.59% of total votes, while the competing party obtained 47.68%, marking a marginal difference of 1.249 votes. This highlights Artvin's criticality for the success of the selected party. Interestingly, our original mRSP scheduled a meeting in Artvin on the second day, whereas Static mRSP failed to hold a meeting there. Similar circumstances are evident in Balikesir, where the competing party secured 48.67% of the votes while the selected party obtained 45.07%. This outlines the city's crucial political significance and suggests that even a slight increase in the selected party's votes could potentially alter the battleground significantly. This analysis signifies the value of a time-dependent reward mechanism, which effectively assesses the dynamics of an election and prioritizes cities critical to electoral success.

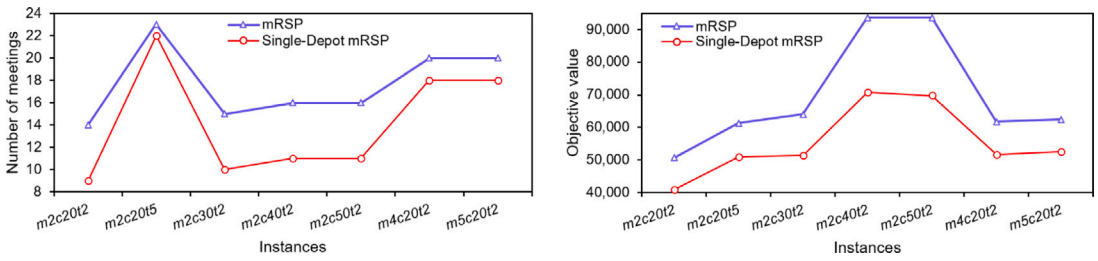


Fig. 10. Value of including closed tour, open tour, and rest days in travel itinerary.

7.3. The value of different tour types

Our problem deals with novel characteristics not typically addressed in routing literature. One of the most important features is the coexistence of closed and open tours within a campaign's daily schedule. To test the value of incorporating three tour types into the itineraries of political representatives, we have developed a model (the Single-Depot mRSP) that mandates all daily tours be closed-tour by indicating that the starting and ending cities for all representatives on every day have to coincide. To implement the Single-Depot mRSP, we set $S_{m,1,t} = 1$ for all $m \in M, t \in T$, where index 1 refers to the campaign center.

Based on analysis results presented in Fig. 10, the highest benefits, both in terms of objective value and the number of realized meetings, are achieved in the mRSP, where all representatives are flexible to create any tour type necessary to cover more cities. In particular, the flexibility to vary tour types results in cost savings by eliminating the need for daily returns to the campaign center. This leads to an increase in the number of meetings by more than 22% on average. When comparing results for instances with an equal number of representatives ($m = 2$) and days ($t = 2$), it becomes evident that the growing trend of the number of meetings is similar for both problems as the number of cities increases. Rewards also show an increase with the number of cities, but the growth is more pronounced for the mRSP compared to the Single-Depot mRSP.

Although this analysis has been conducted on optimal solutions, in several other instances (not solved to optimality), we observed that when a representative cannot visit two highly rewarded cities A and B in a single day due to maximum tour duration constraints, AGVNS forces the traveler to visit city A on the day t and conclude the daily tour in a city nearby city B. This plan allows for a quicker visit to city B the following day, while also granting more time for the traveler to explore additional cities. In other instances, we have observed a traveler beginning the daily tour in city A, exploring neighboring cities, and then returning to city A by the end of the day, forming a closed-tour route. This pattern is often made due to the presence of several highly rewarding cities near city A. On day t+1, the traveler departs from city A without holding any meetings, ventures to other cities, and concludes their tour in a different city, resulting in an open-tour route. Such scenarios underscore the need for flexible planning strategies that adapt to the diverse preferences and schedules of political representatives. The presence of closed tours, open tours, and even no tours in our solutions highlights a managerial challenge and opportunity. Our results indicate that campaign planners may design itineraries to strike the right balance between closed and open tours. Finally, a distinctive feature of mRSP is that not every terminal node is included in the reward collection. In instances with only a few days available, we observe many travelers using their daily remaining time to stay in a city near big cities. The mRSP excels in developing concise yet impactful itineraries. Campaign managers can apply this insight to ensure that every day of the campaign is used effectively. This allows politicians to cover a wider range of cities within tight time constraints.

7.4. Impact of varying values for parameters p and q

Two parameters that significantly influence the scheduling of travelers are the maximum number of daily meetings p and the maximum tour duration q. In our primary experiments, we fixed these values to 4 visits and 14 hours, respectively. To gauge the impact of these parameters, we conducted a sensitivity analysis with varying values for both of them. Regarding the parameter p, we examine two extreme scenarios: one, where only 2 meetings per day were allowed, and the other, where up to 5 meetings per day could be held. The results, as depicted in Fig. 11, reveal that increasing the value of p to 5, while making the model more complex, does not yield any additional value. Conversely, restricting p to 2 leads to a notable decrease in performance, both in terms of the objective value and the number of realized meetings. However, notice that the impact of decreasing p diminishes with a longer planning horizon (cf. m2c20t5), as the longer planning horizon increases flexibility and allows arranging more meetings even with only two representatives. Moreover, as the number of travelers increases, the impact of increasing p diminishes because a higher number of representatives can visit cities that could not be visited with only 2 candidates.

For parameter q, we introduce two additional scenarios in which the maximum tour duration was set to 12 and 16 hours, respectively. The results are displayed in Fig. 12. We note a similar pattern to the analysis for p. Correspondingly, increasing the value of q to 16 hours, though introducing added complexity, does not yield significant additional value. Conversely, in scenarios where q is limited to 12 hours, the model generates inferior solutions characterized by reduced objective values and a reduced number of meetings realized during the campaign. As with the previous analysis, the impact of increasing q diminishes as the number of travelers increases. This observation can be attributed to the real-life travel times used to construct our instances, making it infeasible to visit more cities when the value of p is set at 4. Finally, the impact of decreasing q also diminishes with a longer planning horizon (see *m2c20t5*) as each traveler benefits from a greater time allocation within each period.

7.5. Impact of the constraint limiting the repeated visits

In the original mRSP, we impose a restriction that prevents repeated visits by the same representative to a same city in the upcoming ξ days. This constraint avoids revisiting highly rewarded cities too frequently. To assess the impact of this restriction, we formulated an alternative model in which we excluded Constraints (6). The comparison is made on instance *m2c20t5* where with 5 time periods and $\xi = 2$, the analysis can be made. The results are presented in Fig. 13. Without such constraints, the solution indicates 28 meetings out of which 8 are repeated visits concerning 6 cities. In particular, 2 cities have two repeated visits one for each representative which means unnecessary activities given the horizon of 5 days. Adding these constraints, the meeting number decreases to 23 out of which 4 were repeated visits concerning 4 cities.

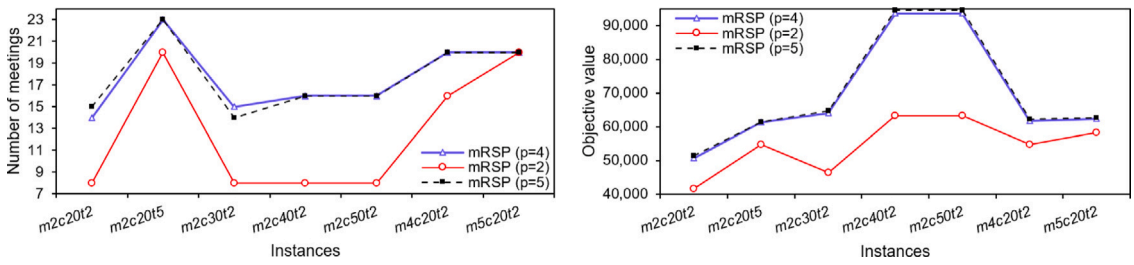


Fig. 11. Impact of varying parameter p values.

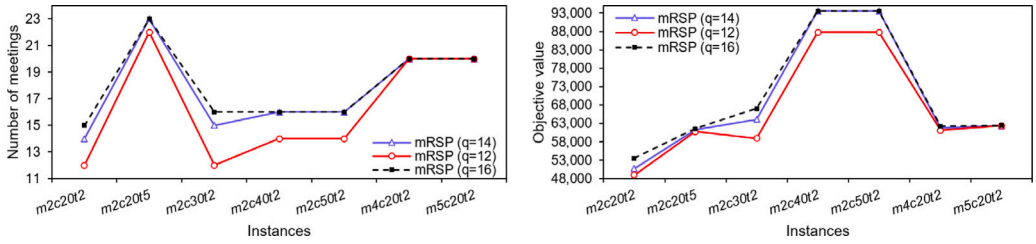


Fig. 12. Impact of varying parameter q values.

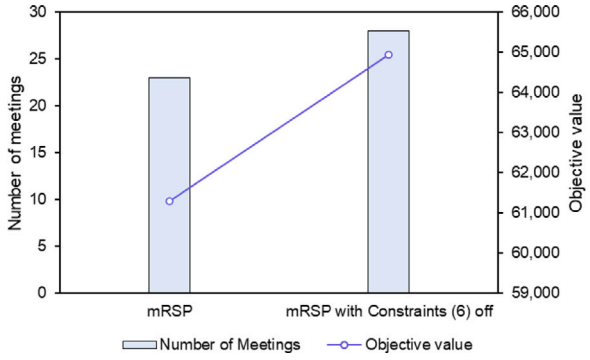


Fig. 13. Impact of constraints (6) on model's performance.

As expected, relaxing Constraints (6) allows the repetition of visits to cities with higher rewards more than once, consequently achieving higher objective function values.

8. Conclusion

In this paper, we study an innovative variant of the roaming salesman problem (RSP), referred to as multiple RSP (mRSP), involving many representatives whose itineraries are linked by budget constraint, and controlled repeated visits to cities. mRSP is a computationally challenging NP-Hard problem characterized by time-dependent rewards and different daily tour types (closed, open, and no tour routes), and that finds application in numerous real-world scenarios.

We introduce 48 new benchmark instances for the problem involving up to 5 representatives, 50 cities, and 10 days. To address the issue of large-sized instances, we develop a hyper-heuristic called adaptive granular variable neighborhood search (AGVNS) that selects low-level neighborhood operators adaptively and thoroughly explores the solution space. Our approach incorporates a learning mechanism and complements it with granular neighborhood operators to facilitate more efficient progress in the search trajectory. Furthermore, to intensify the search, we perturb the current solution only when the local search engine fails to improve it. To evaluate the effectiveness of our AGVNS, we conducted a comprehensive validation analysis. Our computational study demonstrates the potential of AGVNS as an efficient algorithm in time-constrained campaign planning scenarios. AGVNS consistently produced high-quality solutions, often outperforming commercial solvers and a matheuristic adapted from the literature (MFDOR), while achieving this within remarkably short computational times. For campaign planners working under tight schedules and budget constraints, AGVNS can significantly enhance the efficiency of route and meeting plan optimization. Finally, we want to clarify that our paper primarily focuses on the operational aspects of planning election campaigns. While we do acknowledge the potential ethical considerations and concerns regarding the democratic process, we would like to emphasize that our paper does not advocate or endorse any specific political strategies or intentions. Therefore, it is essential to distinguish between the operational aspects of campaign planning, which our paper focuses on, and the broader political considerations, which are beyond the scope of our study.

CRediT authorship contribution statement

Masoud Shahmanzari: Conceptualization, Data curation, Formal analysis, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Renata Mansini:** Conceptualization, Formal analysis, Methodology, Supervision, Writing – original draft, Writing – review & editing.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ejor.2024.06.009.

References

Ahmed, L., Mumford, C., & Kheiri, A. (2019). Solving urban transit route design problem using selection hyper-heuristics. *European Journal of Operational Research, 274*(2), 545–559.

Aksen, D., & Shahmanzari, M. (2017). A periodic traveling politician problem with time-dependent rewards. In *Operations research proceedings 2016: selected papers of the annual international conference of the german operations research society (GOR), Helmut Schmidt University Hamburg, Germany, August 30-September 2, 2016* (pp. 277–283). Springer.

Arbay, E. A., Pasha, J. A., & Widodo, A. S. (2021). The consequences of digitally driven changes in political campaigning for democratic societies: A case study of the 2020 US presidential election. *Journal of Social and Political Sciences, 4*(4).

Barrena, E., Canca, D., Coelho, L. C., & Laporte, G. (2022). Analysis of the selective traveling salesman problem with time-dependent profits. *TOP*, 1–29.

Bock, A., & Sanità, L. (2015). The capacitated orienteering problem. *Discrete Applied Mathematics*, *195*, 31–42.

Box-Steffensmeier, J. M., & Kimball, D. (1999). The timing of voting decisions in presidential campaigns. In *Annual meeting of the midwest political science association, Chicago*.

Cinar, A., Salman, F. S., & Bozkaya, B. (2021). Prioritized single nurse routing and scheduling for home healthcare services. *European Journal of Operational Research*, *289*(3), 867–878.

da Silva, A. A., Morabito, R., & Pureza, V. (2018). Optimization approaches to support the planning and analysis of travel itineraries. *Expert Systems with Applications*, *112*, 321–330.

Dang, D.-C., Guibadj, R. N., & Moukrim, A. (2013). An effective PSO-inspired algorithm for the team orienteering problem. *European Journal of Operational Research*, *229*(2), 332–344.

Drake, J. H., Kheiri, A., Özcan, E., & Burke, E. K. (2020). Recent advances in selection hyper-heuristics. *European Journal of Operational Research*, *285*(2), 405–428.

Faugère, L., Klibi, W., White III, C., & Montreuil, B. (2022). Dynamic pooled capacity deployment for urban parcel logistics. *European Journal of Operational Research*, *303*(2), 650–667.

Feillet, D., Dejax, P., & Gendreau, M. (2005). Traveling salesman problems with profits. *Transportation Science*, *39*(2), 188–205.

Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G., & Vathis, N. (2015). Heuristics for the time dependent team orienteering problem: Application to tourist route planning. *Computers & Operations Research*, *62*, 36–50.

Gobbi, A., Manerba, D., Mansini, R., & Zanotti, R. (2023). Hybridizing adaptive large neighborhood search with kernel search: a new solution approach for the nurse routing problem with incompatible services and minimum demand. *International Transactions in Operational Research*, *30*(1), 8–38.

Gunawan, A., Lau, H. C., & Vansteenwegen, P. (2016). Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, *255*(2), 315–332.

Hanafi, S., Mansini, R., & Zanotti, R. (2020). The multi-visit team orienteering problem with precedence constraints. *European Journal of Operational Research*, *282*(2), 515–529.

Juan, A., Freixes, A., Panadero, J., Serrat, C., & Estrada-Moreno, A. (2020). Routing drones in smart cities: A biased-randomized algorithm for solving the team orienteering problem in real time. *Transportation Research Procedia*, *47*, 243–250.

Khodadadian, M., Divsalar, A., Verbeeck, C., Gunawan, A., & Vansteenwegen, P. (2022). Time dependent orienteering problem with time windows and service time dependent profits. *Computers & Operations Research*, *143*, Article 105794.

Kotiloglu, S., Lappas, T., Pelechrinis, K., & Repoussis, P. (2017). Personalized multi-period tour recommendations. *Tourism Management*, *62*, 76–88.

Labadie, N., Mansini, R., Melechovskỳ, J., & Calvo, R. W. (2012). The team orienteering problem with time windows: An lp-based granular variable neighborhood search. *European Journal of Operational Research*, *220*(1), 15–27.

Laporte, G., & Martello, S. (1990). The selective travelling salesman problem. *Discrete Applied Mathematics*, *26*(2–3), 193–207.

Lodge, M., Steenbergen, M. R., & Brau, S. (1995). The responsive voter: Campaign information and the dynamics of candidate evaluation. *American Political Science Review*, *89*(2), 309–326.

Manerba, D., & Mansini, R. (2016). The nurse routing problem with workload constraints and incompatible services. *IFAC-PapersOnLine*, *49*(12), 1192–1197.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, *24*(11), 1097–1100.

Morvinski, C., Lupoli, M. J., & Amir, O. (2022). Social information decreases giving in late-stage fundraising campaigns. *Plos One*, *17*(12), Article e0278391.

Panadero, J., Juan, A., Bayliss, C., & Currie, C. (2020). Maximizing reward from a team of surveillance drones: A simheuristic approach to the stochastic team orienteering problem. *European Journal of Industrial Engineering*, *14*(4), 485–516.

Riera-Ledesma, J., & Salazar-González, J. J. (2017). Solving the team orienteering arc routing problem with a column generation approach. *European Journal of Operational Research*, *262*(1), 14–27.

Ruiz-Meza, J., Brito, J., & Montoya-Torres, J. R. (2021). A GRASP to solve the multi-constraints multi-modal team orienteering problem with time windows for groups with heterogeneous preferences. *Computers & Industrial Engineering*, *162*, Article 107776.

Saltzman, R. M., & Bradford, R. M. (2022). A data-driven approach to scheduling the US presidential primary elections. *Socio-Economic Planning Sciences*, *79*, Article 101099.

Schilde, M., Doerner, K. F., Hartl, R. F., & Kiechle, G. (2009). Metaheuristics for the bi-objective orienteering problem. *Swarm Intelligence*, *3*, 179–201.

Shahmanzari, M., & Aksen, D. (2021). A multi-start granular skewed variable neighborhood tabu search for the roaming salesman problem. *Applied Soft Computing*, *102*, Article 107024.

Shahmanzari, M., Aksen, D., & Salhi, S. (2020). Formulation and a two-phase matheuristic for the roaming salesman problem: Application to election logistics. *European Journal of Operational Research*, *280*(2), 656–670.

Shahmanzari, M., Aksen, D., & Salhi, S. (2022). Multi-period travelling politician problem: A hybrid metaheuristic solution method. *Journal of the Operational Research Society*, *73*(6), 1325–1346.

Souffriau, W., Vansteenwegen, P., Vanden Berghe, G., & Van Oudheusden, D. (2013). The multiconstraint team orienteering problem with multiple time windows. *Transportation Science*, *47*(1), 53–63.

Sze, J. F., Salhi, S., & Wassan, N. (2017). The cumulative capacitated vehicle routing problem with min-sum and min-max objectives: An effective hybridisation of adaptive variable neighbourhood search and large neighbourhood search. *Transportation Research, Part B (Methodological)*, *101*, 162–184.

Tarantilis, C. D., Stavropoulou, F., & Repoussis, P. P. (2013). The capacitated team orienteering problem: a bi-level filter-and-fan method. *European Journal of Operational Research*, *224*(1), 65–78.

Trent, J. S., & Friedenberg, R. V. (2008). *Political campaign communication: Principles and practices*. Rowman & Littlefield.

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *Journal of the Operational Research Society*, *35*(9), 797–809.

Van Aelst, P., Aalberg, T., et al. (2012). The political information environment during election campaigns. In *How media inform democracy* (pp. 50–63). Routledge.

Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Oudheusden, D. V. (2009a). Metaheuristics for tourist trip planning. In *Metaheuristics in the service industry* (pp. 15–31). Springer.

Vansteenwegen, P., Souffriau, W., Berghe, G. V., & Van Oudheusden, D. (2009b). Iterated local search for the team orienteering problem with time windows. *Computers & Operations Research*, *36*(12), 3281–3290.

Verbeeck, C., Sörensen, K., Aghezzaf, E.-H., & Vansteenwegen, P. (2014). A fast solution method for the time-dependent orienteering problem. *European Journal of Operational Research*, *236*(2), 419–432.

Wang, X., Golden, B., & Gulczynski, D. (2014). A worst-case analysis for the split delivery capacitated team orienteering problem with minimum delivery amounts. *Optimization Letters*, *8*(8), 2349–2356.

Wang, X., Golden, B. L., & Wasil, E. A. (2008). Using a genetic algorithm to solve the generalized orienteering problem. *Vol. 43*, In *The vehicle routing problem: latest advances and new challenges* (pp. 263–274). Springer.