Logistics Research

135

Solving the stochastic dynamic contagious disease testing problem

David Wolfinger
University of Vienna, Vienna, Austria
Margaretha Gansterer
University of Klagenfurt, Klagenfurt, Austria, and
Karl F. Doerner
University of Vienna, Vienna, Austria

Received 14 May 2025
Revised 7 July 2025
25 July 2025
Accepted 25 July 2025

Abstract

Purpose – In early 2020, the coronavirus disease 2019 (COVID-19) pandemic reached global proportions within only a few weeks. A key strategy to contain and control pandemics is to isolate infected people, which requires making test results available quickly, in order to be effective. The contagious disease testing problem (CDTP) arises precisely in this context of testing potential cases (of infection). In this paper, we address the stochastic and dynamic version of the CDTP, where only some suspected cases are known in advance, while others arrive randomly throughout the course of the day. For each newly arriving suspected case, it must be decided whether to accept or reject it. The specimens of accepted suspected cases must be collected on the same day; either by assigning the case to a time slot in a test-center or by visiting the patient with a mobile test-team. Rejected suspected cases must be serviced on the next day. The task in this problem is to decide how many mobile test-teams to use, how many test-centers to open and where, which suspected cases to visit with a mobile test-team and which to assign to a test-center, and designing the vehicle routes for the mobile test-teams. The objective is to identify a dynamic assignment-and-routing policy that minimizes the number of open test-centers and used vehicles, services all early-known suspected cases, and maximizes the expected number of serviced late-known suspected cases.

Design/methodology/approach – We introduce this new problem, which we call the stochastic dynamic contagious disease testing problem (SDCDTP), and formulate it as a Markov decision process. We propose a solution method based on value function approximation for solving the SDCDTP, and illustrate its performance with an extensive computational study.

Findings – We solve and analyze real-world-based problem instances from which we draw insights regarding the solution quality produced by our approach.

Originality/value – We find that a (significant) increase in the number of covered late-known suspected cases can be achieved by not minimizing the number of used vehicles and test-centers, albeit for the price of equally significant increases in the amount of used infrastructure.

Keywords Vehicle routing, Stochastic dynamic optimization, Approximate value iteration, Large neighborhood search, COVID-19

Paper type Research article

1. Introduction

Epidemics and pandemics of infectious diseases are constant companions of humanity. In its handbook for epidemics, the World Health Organization (WHO) cautions that there is a steady risk of a new epidemic emerging. That is, according to it, the question is not *if*, but *when* the next epidemic (or even pandemic) will emerge. What is more, due to the ever-increasing interconnection of all parts of the globe, epidemics can spread faster and further than ever before (WHO, 2018). The outbreak of coronavirus disease 2019 (COVID-19) bears witness to

This research was funded in whole by the Austrian Science Fund (FWF, Grant number P-34151-N).

Logistics Research
Vol. 18 No. 1/2, 2025
pp. 135-160
Emerald Publishing Limited
e-ISSN: 1865-0368
p-ISSN: 1865-035X
DOI 10.1108/LORE-05-2025-0061

the harsh truthfulness of this assessment. In only a few weeks, COVID-19 spread all over the globe, infecting hundreds of thousands of people, putting virtually the entire world in a collective state of emergency.

A key strategy to effectively contain and control a pandemic is to isolate infected people. More formally, this strategy is known as the *find, test, trace, isolate and support (FTTIS)* strategy. The idea being that anyone with symptoms shall be tested, and if positive, they shall be isolated. It is obvious that test results must be made available quickly in order for the FTTIS strategy to be effective (Santini, 2021). In other words, it is vital to achieve a short turnaround time between a suspected case becoming known and specimen collection.

Motivated by the COVID-19 pandemic, Wolfinger *et al.* (2023b) introduced a novel logistics problem that arises precisely in this context of testing potential cases (of infection). They call this new problem the contagious disease testing problem (CDTP). In the CDTP, specimens can be collected in two ways. Either by means of a mobile test-team, which visits the potentially infected person at their home and retrieves the specimen there, or by means of a stationary test-team in a (walk-/drive-in) test-center, to which the potentially infected person travels on their own. In the CDTP, all potential cases are known in advance, and the objective is to assign each case to either a mobile test-team or a test-center, such that all suspected cases are tested and the total cost of opening test-centers and routing mobile test-teams is minimized. For a detailed discussion of the CDTP, we refer the interested reader to the article of Wolfinger *et al.* (2023b).

The most closely related problem to the CDTP is the location-or-routing problem (LORP). In the LORP, like in the CDTP, the aim is to cover each customer either by a vehicle route or by an open facility (Grabenschweiger *et al.*, 2022). The objective is likewise to minimize the total cost of routing vehicles and opening facilities. A major difference to the CDTP is that in the LORP, it is assumed that opening a facility automatically covers every customer located within its coverage range, whereas in the CDTP, opening a facility merely grants the possibility of being covered by it. Additionally, while in the CDTP facilities and vehicle depots are separate from each other, in the LORP facilities also serve as vehicle depots. Lastly, vehicles in the LORP are capacitated and customers have a positive demand, while in the CDTP, each customer supplies exactly one specimen, which are so small in size that vehicle capacities can be neglected.

In the article at hand, we address the stochastic and dynamic version of the CDTP, called the stochastic dynamic contagious disease testing problem (SDCDTP). In the SDCDTP, only some suspected cases are known in advance, while others arrive randomly throughout the course of the day. As such, it represents more closely the structure of the real-world problem, compared to the CDTP. We assume that stochastic information on the dynamically occurring suspected cases is available (for example, through historical data). For all suspected cases that are known in advance, the corresponding specimens must be collected until the end of the day; either by assigning the case to a time slot in a test-center or by visiting the patient with a mobile test-team. For each newly arriving suspected case, it must be decided whether to accept or reject it. If a suspected case is accepted, the corresponding specimen must likewise be collected until the end of the day. Rejected suspected cases become members of the set of cases which are known in advance of the next day. In accordance with Wolfinger *et al.* (2023b), we assume that the test-centers are already built, and we only decide which test-centers to open (i.e. staff has to be assigned) on a given day and which ones to keep closed. This assumption is grounded in a real-world testing strategy employed in Vienna, Austria. Furthermore, we assume that the mobile test-teams are vehicles from public health services (such as ambulances) which are withdrawn from their usual services in order to collect specimens. Thus, there is an incentive to minimize the number of used vehicles, as every vehicle not used for collecting specimens is available for regular (emergency) services. Due to the high degree of dynamism and stochasticity in the underlying real-world problem, it is reasonable to take the resource decisions (test-centers and fleet of mobile test-teams) each day anew. The considered planning horizon in the SDCDTP is therefore one day.

The SDCDTP is a generalization of the stochastic dynamic vehicle routing problem (SDVRP) as it combines the routing decisions with location and assignment decisions. The core idea for solving an SDVRP is utilizing the available stochastic information to dynamically adjust the vehicle routes in anticipation of future events. The goal in stochastic dynamic optimization, in general, is to identify a (preferably optimal) policy. The objective in the SDCDTP is thus to identify a dynamic assignment-and-routing policy that minimizes the number of open test-centers and used vehicles, services all early-known suspected cases, and maximizes the expected number of serviced late-known suspected cases (using the decided upon open test-centers and vehicle fleet).

For that purpose, we model the SDCDTP as a Markov decision process (MDP). Due to the model's vast state space, it is, however, computationally intractable to solve it to optimality. Instead, we use approximate value iteration (AVI), an approximate dynamic programming (ADP) method that belongs to the category of value function approximations (VFAs). Powell (2019) identifies two fundamental strategies for creating policies: policy search (searching within a family of functions to find one that works best) and lookahead approximations (approximating the impact of a decision now on the future). VFA belongs to the latter category (lookahead approximations) and aims at learning the value of a decision via repeated simulations and training (Ulmer *et al*., 2020). In VFA, as the name implies, the expected future rewards in each state (called *values*) are merely approximated rather than calculated exactly. In AVI, this approximation is done by means of simulation, using forward dynamic programming. The advantage of using AVI is that the majority of the computational effort – approximating the values – takes place offline, i.e. before actually solving a problem instance. This allows an efficient computation of decisions when solving a problem instance (in the so-called online execution phase), where the *learned*, respectively approximated, values can simply be used to select good decisions. We use the large neighborhood search (LNS) proposed in Wolfinger *et al*. (2023b) to compute the feasibility and immediate reward of a decision in each state – both during the training phase and in the execution phase. Furthermore, because storing an individual value for each possible state is not feasible, an aggregation and partitioning of the state space is required. For that purpose, we use the state space representation proposed in Ulmer *et al*. (2018a) and adapt it to our problem setting: multiple vehicles, facilities (i.e, test-centers) and time slots. This representation consists of an aggregation of the state space to a three-dimensional vector space combined with a three-dimensional dynamic lookup table (DLT). We then approximate the values for every entry of the DLT instead of the values for specific states.

The contribution of our article is three-fold. First, we introduce the SDCDTP and model it as an MDP. Second, we develop a solution method based on VFA for solving the SDCDTP and present an extensive computational study that illustrates its performance. Our proposed solution method interleaves AVI with LNS, a novel and promising approach. To the best of our knowledge, this is the first study in stochastic dynamic vehicle routing that uses a powerful, albeit computationally expensive, solution method, such as LNS, for computing the decisions and rewards in each state, instead of a simple, but computationally inexpensive, method, such as cheapest insertion (CI), for example. The computational results confirm the superiority of this approach. Furthermore, the results show that our method is comparable with the state-of-the-art offline solution methods for the SDVRP. Third, we solve and analyze real-world-based problem instances for the SDCDTP, from which we draw and present insights regarding the solution quality produced by our approach. We find that a (significant) increase in the number of covered late-known suspected cases can be achieved by not minimizing the number of used vehicles and test-centers, albeit for the price of equally significant increases in the amount of used infrastructure.

The remainder of this article is structured as follows. In Section 2, we discuss the related literature. In Section 3, we give a formal description of the SDCDTP and present its mathematical formulation (MDP). Section 4 is devoted to the solution method. The computational study is presented in Section 5. Concluding remarks are reported in Section 6.

2. Related literature

In this section, we review the related literature. We first discuss the most important studies related to the CDTP, and then focus on works that address SDVRPs with stochastic service requests. For a detailed discussion of studies related to the CDTP, we refer the reader to the article of Wolfinger *et al.* (2023b). We conclude our literature review by summarizing how our study closes an open research gap.

2.1 Problems related to the CDTP

The LORP was introduced by Arslan (2021). They propose a set covering formulation for the problem and a branch-and-price algorithm to solve it. Haghi *et al.* (2023) introduce a generalized version of the LORP that considers distance-decaying coverage for the facilities, relaxes the assumption that every customer should be covered and allows partial coverage by vehicles as well as multiple vehicle visits to the same customer locations. They call this new problem the location-or-routing problem with partial and decaying coverage and present two mixed-integer linear programming formulations for it. Both Arslan (2021) and Haghi *et al.* (2023) address the deterministic and static versions of the respective problems.

The CDTP shares similarities with other well-known optimization problems, such as the location-routing problem (LRP) and the covering location problem (CLP). In the LRP, customers cannot be covered by facilities; instead, all customers must be serviced by a vehicle departing from an open facility (Srivastava, 1993; Schneider and Drexl, 2017). The CLP, on the other hand, does not make use of vehicles, meaning that customers must be covered solely by opening facilities. Both problems are thoroughly studied in the scientific literature.

In the context of the COVID-19 pandemic, Shahnejat-Bushehri *et al.* (2022) present a case study of a medical laboratory that offers an in-home COVID-19 PCR test service. The service consists of nurses employed by the laboratory visiting patients at their home to collect specimens and delivering those specimens back to the laboratory for analysis. Shahnejat-Bushehri *et al.* (2022) model the problem as a vehicle routing problem with time windows and workload balancing. The aim is to both balance the nurses' workload and minimizes total travelling costs; whereby the number of visited patients by a nurse is considered as the workload. Ghasemi *et al.* (2025). present a bi-objective and multi-period mathematical model for mobile tester route plans and testing resource allocation. In contrast to our work, both of these studies neither consider test-centers, nor address stochastic and/or dynamic aspects of the problem.

Another related study motivated by the COVID-19 pandemic is the work of Martínez-Reyes *et al.* (2020), who address the problem of locating distribution centers for personal protective equipment and planning associated vehicle routes to supply intensive care units with it. They investigate this problem for the city of Bogotá, Colombia, and formulate it as a capacitated LRP with stochastic demands (capacity restrictions are present for both facilities and vehicles).

2.2 SDVRPs and their solution approaches

The SDCDTP is a variant of the SDVRP, which deals with the construction of vehicle routes under uncertainty, where only limited and stochastic information is available in advance and new information is dynamically revealed throughout the planning horizon. We can observe two essential differences between the SDCDTP and classical SDVRPs. First, while in the standard SDVRP, the only available resource to cover demand is the vehicles, the SDCDTP additionally considers facilities – the test-centers – to cover demand. As a consequence, the SDCDTP also considers location and assignment decisions, in addition to the routing decisions. Second, in classical SDVRPs, the amount of available resources is typically assumed to be fixed, whereas in the SDCDTP, it is flexible. Moreover, in the SDCDTP, there is an incentive to minimize resource usage, which is why it aims at minimizing the number of used vehicles and test-centers.

Regarding anticipatory solution approaches for SDVRPs, they can be categorized into *online* and *offline* approaches, depending on when the (majority of the) computational effort for decision making takes place: either during the execution phase or during an offline learning phase beforehand. In the following, we start with discussing online solution approaches, followed by offline ones. Finally, we also provide an overview of hybrid approaches.

2.3 Online solution approaches

Online approaches often use sampling (of future events) in order to evaluate potential decisions. Typically, several times during plan execution (for example, whenever a dynamic event occurs), future service requests are sampled from some known spatial-temporal distribution. Then, those requests, together with the known information, are used to compute and evaluate different solutions and decisions. While this allows a detailed consideration of future events, it also requires a large amount of computational effort during plan execution. Bent and Van Hentenryck (2004) develop a sample-scenario-based planning approach that maintains a set of solutions (routing plans) generated by constructing solutions for different scenarios, which include known and sampled future requests. At each decision, they use a *consensus-function* to choose a distinguished routing plan, which is then used to decide the vehicle movements. Hvattum *et al*. (2006) also use sampling in combination with a *consensus-function* (applied to a set of routes constructed for the sampled service requests) to build routing plans. They address a real-world case study with their proposed hedging heuristic. In Hvattum *et al*. (2007), the authors extend the problem of Hvattum *et al*. (2006) with stochastic demands for known customers and propose a new sampling-based solution approach based on the previously developed heuristic. Ghiani *et al*. (2009) use short-term sampling to estimate the expected cost-to-go for a dynamic pickup and delivery problem.

Another strategy used in online approaches is to exploit the stochastic information to devise and implement waiting strategies. These approaches use the probabilistic information about potential future requests to decide, in addition to the routing sequence, when, where and for how long vehicles should wait. Examples include the works of Mitrović-Minić and Laporte (2004), Ichoua *et al*. (2006), Thomas (2007) and Ghiani *et al*. (2012). The advantage of such approaches is that they require less computational effort than sampling-based approaches, while potentially achieving similar results (Ghiani *et al*., 2012).

2.4 Offline solution approaches

Offline anticipatory approaches, such as ours, move the computational effort to a learning phase, during which they identify a decision policy. This policy can then be used during the execution phase to evaluate decisions with only minimal computational effort, allowing for fast and efficient decision-making during plan execution. The downside of offline approaches is that they typically suffer from (one, possibly all three) curses of dimensionality: exponential growth of the state space, the decision space and/or the outcome space of the random variable. Thus, to be able to compute and store the values of the (post-decision) states, aggregated representations of the state-, decision- and outcome-space are required, which inhibits a detailed consideration of future events. Ulmer *et al*. (2015) propose a cost-benefit heuristic that uses the cost-benefit ratio of inserting (subsets of) the newly arrived service requests to decide whether to confirm or reject them. The idea is to avoid confirming requests that are too far out-of-the-way of the current route. Ulmer *et al*. (2018a) develop temporal-only VFAs for real-world-sized single-vehicle routing problems. They develop an AVI algorithm, for which they present an efficient state space representation (aggregation and partitioning) and show that their approach is capable of identifying high-quality dynamic routing policies. They propose to aggregate the state space to the two features *point of time* and *free time budget* (of the vehicle). Additionally, they propose a DLT to dynamically partition the aggregated state space. We use their state space representation but implement an AVI algorithm that is more closely related to the one presented in (Powell, 2011, pp. 139–141) (with respect to the updating mechanism of

the approximated values). In order to account for several vehicles as well as test-centers, we use the *free time budget* twice – once for vehicles and once for test-centers – and adapt it to represent the *average* instead of the absolute values. Thus, we aggregate the state space to a three-dimensional vector space. To store the values, we implement the proposed DLT, and, likewise, lift it from two dimensions to three dimensions. Soeffker *et al.* (2019) propose an adaptive state space partitioning for dynamic decision processes. Instead of using a (dynamic) lookup-table to partition the aggregated state space, they identify representative post-decision states and subsequently assign each newly observed state to one of the representative states, effectively clustering and thereby partitioning the state space. To account for a potential change of the relevant areas of the state space, the set of representative states may change throughout the approximation process. Ulmer *et al.* (2018b) adapt the approach presented in Ulmer *et al.* (2018a) to address the multi-period version of the original problem.

2.5 Hybrid (offline–online) solution approaches

Hybrid approaches, as the name suggests, combine offline approaches with online components. Ulmer *et al.* (2019) extend the approach from Ulmer *et al.* (2018a) with an online rollout algorithm that introduces spatial anticipation into the method. Given the current state, rollout algorithms simulate potential future realizations of the MDP by sampling transitions and deriving decisions with the help of a given base policy. Ultimately, rollout algorithms enable the anticipation of details (for example, spatial considerations) that are absent in the base policy. Ulmer *et al.* (2019) show that their hybrid procedure, which now incorporates temporal and spatial anticipation, yields high-quality and computationally tractable routing policies. Ulmer (2020) also combines offline VFA with online rollout algorithms, but they limit the horizon of the rollout algorithm and estimate the remaining horizon with a VFA value. In a recent study, Zhang *et al.* (2023) propose a hybrid approach that approximates the expected reward-to-go by using knapsack-based linear models. By combining decision rules of an optimal policy with these approximations, they derive good online scheduling policies. Additionally, they show how to use these approximations to design good initial route plans.

While hybrid approaches work well for the SDVRP (in terms of solution quality), they also generate a substantial computational effort in each state during the execution phase, which has a negative impact on the time it takes to make a decision. We therefore developed a pure offline approach – given the inherent urgency in the SDCDTP – in order to allow for quick decision-making in each state.

2.6 Summary and research gap

Concluding the literature review, we wish to remark that to the best of our knowledge, there are no studies which combine vehicle routing, facility location and assignment decisions in a stochastic dynamic context. Our work is a first step towards closing this research gap. In particular, we introduce the SDCDTP, which is the stochastic dynamic version of the CDTP introduced in Wolfinger *et al.* (2023b). In contrast to this earlier work, we assume that not all potential cases are known in advance. Hence, our study further contributes to existing literature by showing how to handle the problem-inherent complexities in a highly dynamic setting.

3. Problem statement and mathematical formulation

In this section, we formally describe the SDCDTP. In the SDCDTP, collection of a specimen from a suspected case of infection can be done by either assigning the case to a time slot in a test-center or by visiting it with a mobile test-team. Let H denote the set of test-centers and F be the vehicle fleet (representing the mobile test-teams). Each test-center is associated with a coverage range χ, representing a limit on the duration someone is expected to travel to a test

center. Any person located further away from the closest open test-center than χ units of time must be visited by a mobile test-team. In addition, some people must be visited by a mobile test-team regardless of this limit; for example, when they have no access to a private means of transportation or are too sick to take the journey. Let $\Gamma_p \in \{0, 1\}$ denote if potential case p can be serviced in a test-center ($\Gamma_p = 0$) or must be visited by a mobile test-team ($\Gamma_p = 1$). Each test-center $h \in H$ houses a certain number r_h of test-stations and offers ζ time slots of equal duration. The number of suspected cases which can be scheduled per test-station and time slot is the same for all test-centers and is denoted with u. For a successful public health response, it is of utmost importance to achieve a short turnaround time between a potential case occurring and specimen collection. For that purpose, we consider a time limit ψ which restricts the allowed amount of time between the potential case becoming known and the collection of the specimen. Accordingly, this limit – which we call the *time-to-test* limit – imposes a time window on each suspected case within which the specimen must be collected.

In the SDCDTP, some suspected cases are known in advance, represented by set P^{early}, while others arrive randomly during the course of the day. The specimens of all cases $p \in P^{\text{early}}$ must be collected until the end of the day. For each newly arriving suspected case, on the other hand, we must decide whether to accept or reject it. If a suspected case is accepted, the corresponding specimen must be collected until the end of the day, as well. We assume that no further testing resources exist, other than the ones considered in the problem. Rejected suspected cases therefore become members of the set P^{early} on the next day. It is assumed that stochastic information regarding the spatial-temporal distribution of potential cases is available. The objective in the SDCDTP is to identify a dynamic assignment-and-routing policy that minimizes the number of open test-centers and used vehicles, services all early-known suspected cases in set P^{early}, and maximizes the expected number of serviced late-known suspected cases (using the decided upon open test-centers and vehicle fleet). We acknowledge that under other circumstances, the assumption of dynamically opening and closing facilities would be unrealistic. However, our work is based on the real-world testing operations in Vienna, Austria. In a relatively early phase of the pandemic, a set of testing facilities was employed such that depending on the number of infections, these could be dynamically activated or kept closed, where opening a facility just meant to assign staff there. Staff availability was no issue as personnel was dynamically reassigned for the sake of public health.

In the following, we present the mathematical formulation for the SDCDTP.

We formulate the SDCDTP as an *MDP*. In this framework, we assume that there is a set of states $\mathcal{S}$ (called the state space), where we have to make a decision $x_k \in X(S_k)$ in state $S_k \in \mathcal{S}$ at epoch k, which generates a contribution $C(S_k, x_k)$. The outcome of each decision x_k in state S_k is known with certainty and defined as the post-decision state S_k^x. Given a post-decision state S_k^x and some exogenous random information W_k, the system transitions into a new state S_{k+1}. The task is to identify a policy for making decisions which maximizes the expected contributions over time.

Figure 1 illustrates exemplary states, decisions, post-decision states and a state transition. In state S_0 ($t_0 = 0$) a set of seven suspected cases must be served. The available resources are two test-centers, T_1 and T_2, as well as two vehicles located at depot 0. The applied decision x_0 consists of several components. First, it selects the set of test-centers to open and the set of vehicles to use. Note, decision x_0 is the only decision which includes this selection, as the decided-upon resources stay fixed until the end of the horizon. Decision x_0 opens test-center T_1 and uses both vehicles located at the depot 0. Furthermore, decision x_0 sets suspected Cases 1 and 3 as the next cases to visit for both vehicles, respectively. Lastly, decision x_0 assigns Cases 5 and 6 to the first time slot in test-center T_1 (which starts at time point 0 and ends at time point 4), and Case 7 to the third time slot in T_1. The resulting post-decision state S_0^x is depicted in the second picture in Figure 1. The arrows show the planned vehicle routes, whereby only the first leg is fixed (indicated by the solid arrow). The realization of exogenous information W_1

Figure 1. Exemplary (post-decision) states, decisions and state transition

transitions the system into the next state S_1, which is depicted in the third picture in Figure 1. In state S_1 at $t_1 = 1$, seven new suspected cases arrived (Cases 8–14), the first vehicle is currently positioned at Case 3, while the second vehicle is still in transit towards Case 1. The applied decision x_1 accepts Cases 8, 10, 12 and 14 and rejects Cases 9, 11 and 13. Suspected Case 12 is set as the next case to visit for the first vehicle. Suspected Case 8 is assigned to the third time slot in T_1, while Case 14 is assigned to the second time slot in T_1. The planned tour for the second vehicle is adjusted to include suspected Case 10. The resulting post-decision state S_1^x is depicted in the most right picture in Figure 1.

We now describe the MDP formulation for the SDCDTP in more detail.

3.1 Decision epochs

Decision epochs represent points in time at which decisions are made. In the SDCDTP, a new decision epoch k begins when a new suspected case of infection occurs – i.e. when it becomes known – or when a vehicle arrives at a location. We assume a finite horizon with time limit T in which decisions are made at epochs 0, 1, …, K, where K is a random variable. The time at which decision epoch k begins is denoted with t_k, where $t_k \in [0, T]$.

3.2 Exogenous information

We observe exogenous information in the form of new suspected cases starting at time 0. Let W_k be the (possibly empty) set of suspected cases which occurred at or prior to time t_k but after time t_{k-1}, i.e. between decision points $k-1$ and k.

3.3 State variable

Let S_k be the state of the underlying dynamic system at decision epoch k. The state of a single vehicle $f \in F$ at epoch k can be described by the tuple $a_{kf} = (a_{kf}^{loc}, a_{kf}^{time})$, where a_{kf}^{loc} is the vehicle's current location (respectively, destination location, if it is in transit) and a_{kf}^{time} is the time of arrival at a_{kf}^{loc}. The state of all vehicles at epoch k is then given by the tuple $A_k = (a_{k1}, \ldots, a_{k|F|})$.

The state of a single test-center $h \in H$ at epoch k can be described by the tuple $b_{kh} = (b_{kh}^{open}, b_{kh1}^{cap}, \ldots, b_{kh\zeta}^{cap})$, where b_{kh}^{open} indicates if the test-center is open $(b_{kh}^{open} = 1)$ or closed $(b_{kh}^{open} = 0)$ and $b_{kh1}^{cap}, \ldots, b_{kh\zeta}^{cap}$ indicate the residual capacity in the corresponding time slots 1, …, ζ, with $b_{khj}^{cap} \leq r_h \cdot u, j = 1, \ldots, \zeta$. The state of all test-centers at epoch k is then given by the tuple $B_k = (b_{k1}, \ldots, b_{k|H|})$.

The state of a suspected case p can be described by the tuple $e_p = (e_p^{loc}, e_p^{occur})$, where e_p^{loc} is the location of the case and e_p^{occur} its time of occurrence. We use the sets P_k and W_k to model the state of the demand at epoch k in the underlying dynamic system, where P_k is the set of *accepted*, not yet serviced potential cases.

The state S_k of the entire dynamic system at decision epoch k can now be fully described by $S_k = (A_k, B_k, P_k, W_k)$. In the initial state $S_0 = (A_0, B_0, P^{\text{early}}, \varnothing)$ all vehicles are located at their start depot $\mathfrak{o}_f^+$ (i.e. $a_{0f} = (\mathfrak{o}_f^+, 0), \forall f \in F$), all test-centers are closed (i.e. $b_{0h} = (0, r_h \cdot u, \ldots, r_h \cdot u), \forall h \in H$), the suspected cases in P^{early} have yet to be served, and the set of late-known suspected cases W_k is empty. In the final decision epoch K, the system occupies a terminal state S_K in the set $\{(A_K, B_K, \varnothing, \varnothing) \mid a_{Kf} = (\mathfrak{o}_f^-, t_K), t_K \in [0, T], \forall f \in F, b_{Kh} = (b_{Kh}^{open}, b_{Kh1}^{cap}, \ldots, b_{Kh\zeta}^{cap}), b_{Kh}^{open} \in \{0, 1\}, b_{Khm}^{cap} \in \{0, \ldots, r_h \cdot u\}, \forall m \in \{1, \ldots, \zeta\}, \forall h \in H\}$, where all vehicles have returned to their end depot $\mathfrak{o}_f^-$ until time T, some test-centers may be open and suspected cases assigned to their time slots, all early-known suspected cases and all accepted late-known suspected cases have been serviced, and we assume no new potential cases become known in the last decision epoch, i.e. W_K is empty.

3.4 Decision variable

Decisions for the SDCDTP contain an acceptance, an assignment and a movement component. More specifically, a decision $x_k \in X(S_k)$ at epoch k selects the subset of newly arrived potential cases $P_k^{acc} \subseteq W_k$ to accept, assigns a subset $P_k^H \subseteq P_k^{acc}$ of the accepted cases to time slots in open test-centers and selects for each used vehicle currently located at a suspected case or its start depot the next location to visit. Let $P_k^F = P_k^{acc} \backslash P_k^H$ be the set of accepted new potential cases to be visited with a mobile test-team. Decision x_0 at $t_0 = 0$ is special to the extent that it additionally selects the set of test-centers $\tilde{H} \subseteq H$ to open and the set of vehicles $\tilde{F} \subseteq F$ to use. Decision x_0 is the only decision throughout the entire process in which this selection takes place. In other words, the decision which test-centers to open and which vehicles to use is made once at the beginning of the time horizon and then kept fixed until the end of the time horizon.

We denote a decision as the tuple $x_k = (\mathcal{F}, \mathcal{H}, g, q, l)$. $\mathcal{F}$ is a $|F|$-tuple with binary entries indicating for each vehicle f if it can be used for visiting suspected cases ($\mathcal{F}_f = 1$) or not ($\mathcal{F}_f = 0$). $\mathcal{H}$ is a $|H|$-tuple with binary entries indicating for each test-center h if it is open ($\mathcal{H}_h = 1$) or closed ($\mathcal{H}_h = 0$) g is a $|W_k|$-tuple with binary entries indicating acceptance ($g_p = 1$) or rejection ($g_p = 0$) for each suspected case $p \in W_k$. q is a $|P_k^H|$-tuple with pairs of the form (q_p^H, q_p^{slot}) as entries, assigning each suspected case $p \in P_k^H$ to a time-slot (q_p^{slot}) in a test-center (q_p^H). Finally, l is a n-tuple indicating for each available vehicle f its next location to visit l_f. A vehicle f is available if $\mathcal{F}_f = 1$, it is currently not in transit and has not yet returned to the depot. Let $F_k^+ = \{f \in \tilde{F} |\ a_{kf}^{loc} \neq \mathfrak{o}_f^- \wedge a_{kf}^{time} \leq t_k\}$ be the set of available vehicles at epoch k, where $\tilde{F} = \{f \in F |\ \mathcal{F}_f = 1\}$ is the set of vehicles decided to be used by $\mathcal{F}$. Let d_{ij} denote the travel time between two locations i and j.

The set of feasible decisions (also called decision space) in state S_k at decision epoch k is

$$X(S_k) = \{(\mathcal{F}, \mathcal{H}, g, q, l) : \mathcal{F} \begin{cases} \in \{0,1\}^{|F|} & \text{iff } k = 0 \\ = () & \text{iff } k > 0, \end{cases} \tag{1}$$

$$\mathcal{H} \begin{cases} \in \{0,1\}^{|H|} & \text{iff } k = 0 \\ = () & \text{iff } k > 0, \end{cases} \tag{2}$$

$$g \in \{0,1\}^{|W_k|}, \tag{3}$$

$$p \in P_k^H \oplus p \in P_k^F \quad \forall p \in P_k^{acc}, \tag{4}$$

$$p \in P_k^F \quad \forall p \in \left\{p' \in P_k^{acc} |\ \Gamma_{p'} = 1\right\}, \tag{5}$$

$$q \in \left\{\tilde{H} \times \{1, \ldots, \zeta\}\right\}^{|P_k^H|}, \tag{6}$$

$$\left|\left\{p \in P_k^H |\ q_p^H = h \wedge q_p^{slot} = j\right\}\right| \leq b_{khj}^{cap} \quad \forall h \in \tilde{H}, j = 1, \ldots, \zeta, \tag{7}$$

$$d_{e_p^{loc}, loc\left(q_p^H\right)} \leq \chi \quad \forall p \in P_k^H, \tag{8}$$

$$time\left(q_p^{slot}\right) \leq e_p^{occur} + \psi \quad \forall p \in P_k^H, \tag{9}$$

$$e_p^{occur} + d_{e_p^{loc}, loc\left(q_p^H\right)} < time\left(q_p^{slot} + 1\right) \quad \forall p \in P_k^H, \tag{10}$$

$$l_f \in P_k \cup P_k^F \cup \left\{a_{kf}^{loc}\right\} \cup \left\{\mathfrak{o}_f^-\right\} \quad \forall f \in F_k^+, \tag{11}$$

$$l_f \neq \mathfrak{o}_f^- \text{ if } P_k \cup P_k^F \neq \emptyset \wedge \forall f' \in \widetilde{F} \backslash \{f\} : a_{kf'}^{loc} = \mathfrak{o}_{f'}^- \vee l_{f'} = \mathfrak{o}_{f'}^-, \ \exists \Theta_k : \text{feasible}\}. \tag{12}$$

Condition (1) requires to decide for each vehicle whether it is used or not. Likewise, Condition (2) requires for each test-center to be either opened or closed. Both conditions require that these decisions are made at the beginning of the time horizon (i.e. $k = 0$) and that they stay fixed during the entire process. Therefore, tuples in $k > 0$ are empty for both the movements of vehicles and the opening decisions of facilities, which we denote as $S_k = (\,)$. Condition (3) requires each suspected case in W_k to be either accepted or rejected. Condition (4) requires each accepted suspected case to either be assigned to a test-center or be visited with a mobile test-team, where $\oplus$ denotes the XOR-operator, $P_k^{acc} = \{p \in W_k |\ g_{W_k^{-1}(p)} = 1\}$ is the set of suspected cases accepted by g, and $W_k^{-1}(p)$ returns the index of potential case p in W_k. Condition (5) ensures that all accepted cases which must be visited by a mobile test-team, are visited by a vehicle. Per Condition (6) each accepted suspected case in P_k^H must be assigned to a time slot in an open test-center, where $\tilde{H} = \{h \in H |\ \mathcal{H}_h = 1\}$ is the set of test-centers opened by $\mathcal{H}$. Conditions (7)–(10) ensure a feasible assignment of suspected cases to test-centers. Specifically, they make sure that the capacity of the time slots is not exceeded (Condition (7)), and that each assigned case is located within χ units of time from its assigned test-center (Condition (8), where function $loc(\cdot)$ returns the location of a test-center), scheduled within its time window (Condition (9), where function $time(\cdot)$ returns the start time of a time slot) and able to reach the assigned time slot in time (Condition (10)). Condition (11) restricts the next location for each available vehicle f to belong to the set $P_k \cup P_k^F \cup \{a_{kf}^{loc}\} \cup \{\mathfrak{o}_f^-\}$. Setting $l_f = a_{kf}^{loc}$ indicates that the vehicle idles at its current location for one base unit of time $\bar{t}$. Condition (12) forbids vehicle f to return to its depot if there are still unvisited accepted suspected cases and all remaining vehicles either have already returned to their depot or are assigned to travel there next. Let Θ denote a route plan, representing a set of routes. Condition (12), then, requires that at least one feasible route plan Θ_k exists. A route plan is feasible if it contains a feasible route for each vehicle $f \in \tilde{F}$ and all outstanding suspected cases $p \in P_k \cup P_k^F$ are covered by it (exactly once). A route $(\nu_1, \ldots, \nu_m)$ for a vehicle f is feasible if it starts at the vehicles current location $(\nu_1 = a_{kf}^{loc})$, returns to the vehicle's depot not later than T $(\nu_m = \mathfrak{o}_f^-)$, and all time windows of locations $\nu_2, \ldots, \nu_m$ are respected. For each vehicle $f \in F_k^+$, additionally, the next location it travels to (ν_2) must be l_f.

In an MDP, decisions are made by using a policy π from the set of decision policies Π. A policy $\pi = (X_0^\pi, \ldots, X_K^\pi)$ is a sequence of decision rules X_k^π that assign a feasible decision $x_k = X_k^\pi(S_k) \in X(S_k)$ to each state S_k. The solution to an MDP is a policy $\pi \in \Pi$.

3.5 Transition function

The transition function describes how, as a result of the decisions and random information, the system evolves from one state to another. It is typically denoted with

$$S_{k+1} = S^M(S_k, x_k, W_{k+1}),$$

and gives the equations from the pre-decision state S_k to the pre-decision state S_{k+1} (Powell, 2019). The transition can, however, also be split into two parts: (1) from the pre-decision state S_k to the so-called post-decision state S_k^x and (2) from the post-decision state S_k^x to the next pre-decision state S_{k+1}. The post-decision state S_k^x is the state at epoch k immediately after a decision has been taken, but before any new information becomes known. Consequently, the transition to the post-decision state is deterministic. We denote the corresponding transition functions with

$$\begin{aligned} S_k^x &= S^{M,x}(S_k, x_k) \\ S_{k+1} &= S^{M,W}\left(S_k^x, W_{k+1}\right). \end{aligned}$$

The states, decisions and random information, thus, evolve as

$$\left(S_0, x_0, S_0^x, W_1, S_1, x_1, S_1^x, W_2, \ldots, S_{T-1}, x_{T-1}, S_{T-1}^x, W_T, S_T\right).$$

Taking decision x_k when in state S_k transitions the system to post-decision state $S_k^x = S^{M,x}(S_k, x_k) = (A_k^x, B_k^x, P_k^x)$, where the vehicle state tuple A_k^x, the test-center state tuple B_k^x, and the set of outstanding suspected cases P_k^x are updated to reflect decision x_k. $A_k^x = (a_{k1}^x, \ldots, a_{k|F|}^x)$ is updated by setting for each vehicle $f \in \tilde{F}$ the tuple

$$a_{kf}^x = \begin{cases} \left(l_f, t_k + d_{a_{kf}^{loc}, l_f}\right) & \text{iff } f \in F_k^+ \wedge l_f \neq a_{kf}^{loc} \text{ (13a)} \\ \left(l_f, t_k + \bar{t}\right) & \text{iff } f \in F_k^+ \wedge l_f = a_{kf}^{loc} \text{ (13b)} \\ a_{kf} & \text{iff } f \notin F_k^+. \text{ (13c)} \end{cases} \tag{13}$$

Equations (13a) and (13b) state that for all available vehicles f, their current location (respectively destination) is set to their next location to visit (l_f), and the arrival time at this location is updated accordingly. Equation (13c) states that for all other (currently not available) vehicles, no changes take place and the current values are adopted as is. The test-center state tuple $B_k^x = (b_{k1}^x, \ldots, b_{k|H|}^x)$ is updated by setting for each test-center $h \in \tilde{H}$ the elements of the corresponding tuple $b_{kh}^x = (b_{kh}^{open,x}, b_{kh1}^{cap,x}, \ldots, b_{kh\zeta}^{cap,x})$ to

$$b_{kh}^{open,x} = \begin{cases} \mathcal{H}_h & \text{iff } k = 0 \text{ (14a)} \\ b_{kh}^{open} & \text{iff } k > 0 \text{ (14b)} \end{cases} \tag{14}$$

$$b_{khj}^{cap,x} = b_{khj}^{cap} - \sum_{p \in P_k^H} \Phi\left(q_p, h, j\right) \quad j = 1, \ldots, \zeta, \tag{15}$$

where the function $\Phi(q_p, h, j)$ returns 1 if $q_p = (h, j)$, i.e. if decision x_k assigns newly arrived suspected case $p \in P_k^H$ to time slot j in test-center h and 0 otherwise. Equation (14a) states that in the first epoch ($k = 0$) the opening status of each test-centers is set according to decision x_k, while Equation (14b) states that in all subsequent epochs ($k > 0$), the current values are adopted as is (indicating that no changes to the opening status of any test-center take place). Equation (15) states that the residual capacity of each test-center time slot is updated by reducing it by the number of newly assigned suspected cases. The set of outstanding suspected cases $P_k^x = P_k \cup P_k^F \setminus \{p \in P_k^F \mid \exists f \in \tilde{F}: p = l_f\}$ is updated to include all newly accepted suspected cases which are decided to be visited with a mobile test-team but not yet assigned to a vehicle.

The next decision epoch $k+1$ begins either when a new suspected case occurs (possibly several) or when a vehicle arrives at its next destination ν. The process then transitions to pre-decision state $S_{k+1} = (A_{k+1}, B_{k+1}, P_{k+1}, W_{k+1})$, where

$$
\begin{aligned}
A_{k+1} &= A_k^x, \\
B_{k+1} &= B_k^x, \\
P_{k+1} &= \begin{cases} P_k^x & \text{iff a new suspected case occurred or a vehicle arrived at its depot} \\ P_k^x \backslash \{\nu\} & \text{iff a vehicle arrived at a suspected case} \end{cases}
\end{aligned}
$$

and a new set W_{k+1} of new suspected cases may be observed.

3.6 Objective function

Taking decision x_k in state S_k generates a contribution

$$
C_k(S_k, x_k) = \begin{cases} |P_k^{acc}| - \rho_1 \cdot |\tilde{F}| - \rho_2 \cdot |\tilde{H}| & \text{iff } k = 0 \\ |P_k^{acc}| - \rho_3 \cdot \dfrac{T}{t_k} \cdot |W_k \backslash P_k^{acc}| & \text{iff } k > 0. \end{cases}
$$

The contribution generated by the very first decision (x_0) consists of the number of *accepted early suspected cases* (first term) minus the number of *used vehicles* (second term) and *opened test-centers* (third term). The contribution generated by all subsequent decisions consists of the number of *accepted newly arrived suspected cases* (first term) minus the number of *rejected newly arrived suspected cases* (second term). We use the weights (ρ_1, ρ_2 and ρ_3) in order to scale the respective terms with respect to the number of accepted suspected cases. Additionally, we weight the number of rejected cases with factor T/t_k, making rejections earlier in the time horizon more expensive than rejections later in the time horizon. We do this to discourage a situation where, on the next day, a large number of the suspected cases in P^{early} have only very little time left until the end of their time window, which would significantly complicate the initial route generation (and may even lead to infeasibility).

Our objective is to maximize the expected total contribution starting from an initial state S_0

$$
\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{k=0}^{K} C_k\left(S_k, X_k^{\pi}(S_k)\right) \middle| S_0 \right]. \tag{16}
$$

More precisely, we want to identify an optimal decision policy $\pi^* \in \Pi$ that maximizes the expected total contribution, given by

$$
\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}\left[\sum_{k=0}^{K} C_k\left(S_k, X_k^{\pi}(S_k)\right) \middle| S_0 \right]. \tag{17}
$$

4. Solution approach

In this section, we present our solution approach for the SDCDTP. We first describe AVI and elaborate on the implemented state space aggregation and partitioning. Then, we discuss the LNS algorithm used to compute the routing, location and assignment decisions. Finally, we present the complete AVI algorithm and outline the overall process of solving a problem instance.

4.1 Approximate value iteration

Let $V_k(S_k)$ be the value of being in state S_k. Then, the stochastic optimization problem in Equation (16) can be solved by computing *Bellman's equation*. This computation can, in principle, be done by means of backward dynamic programming (Powell, 2011), where the value $V_k(S_k)$ is computed from $V_{k+1}(S_{k+1})$ by stepping backward in time. However, this would require to compute $V_k(S_k)$ for each state S_k in the state space $\mathcal{S}$, which is typically vast, making it computationally intractable (see the *curse of dimensionality*, e.g. in Powell, 2011).

A common way to overcome this intractability is to approximate the values of the states; a strategy known in the literature as *VFA* (Powell, 2019). In this article, we will use *AVI*, an ADP method belonging to the category of VFAs (Powell, 2011). The idea of AVI is to approximate the value function by means of simulation. Here, decisions are made by stepping forward in time, in an iterative fashion. It is therefore also sometimes described as "forward approximate dynamic programming" (Powell, 2019).

Let Ω denote the overall set of stochastic problem realizations, with a single problem realization $\omega \in \Omega$ defined as $\omega = \{P^{\text{early}}, W_1, \ldots, W_K\}$. AVI simulates a set of sampled problem realizations $\overline{\Omega} \subset \Omega$.

To counteract a pure exploitation policy and to avoid local optima, decisions can be selected randomly in some states in order to enforce *exploration* of the state space. For that purpose, we implement the so-called epsilon-greedy exploration strategy (Powell, 2011), where we define an exploration rate $\epsilon^i(S_k)$ that expresses the probability with which we explore. It is computed as

$$\epsilon^i(S_k) = \frac{\gamma}{\mathcal{N}^i(S_k)},$$

where $0 \leq \gamma \leq 1$ and $\mathcal{N}^i(S_k)$ is the number of times we have observed state S_k until iteration i. In case we explore, we choose a decision x_k with probability $1/|X(S_k)|$. Note that the probability of exploration decreases with an increasing number of observations. At the end of the simulation, AVI returns the final values $\overline{V}^{x,|\overline{\Omega}|}$. These values constitute a policy that can then be applied in the execution phase. As a consequence, decisions in the execution phase can be computed very efficiently, since no further approximations need to take place.

4.2 State space aggregation

Due to the high dimensionality of our state space, we need to aggregate it (to a set of state features) in order to be able to store the values for the vast number of post-decision states. For that purpose, we use the aggregated state space representation proposed in Ulmer *et al*. (2018a) and adapt it to our problem setting. They address a single vehicle problem and propose to aggregate the state space to the two features *point of time* t_k and *free time budget* (of the vehicle) β_k; also called *slack*. t_k indicates how much time has already passed and how much time is still remaining until the end of the time horizon. β_k refers to how much of the remaining time is still free and can be used for servicing new requests. In order to account for several vehicles and test-centers, we instead use the *average free time budget* over all vehicles β_k^F and the *average free time budget* over all test-centers β_k^H in our implementation. Thus, we aggregate the state space to a three-dimensional vector space: one dimension for point of time t_k, one for average free time budget of vehicles β_k^F, and a third for average free time budget of test-centers β_k^H. AVI, then, approximates the values for these vectors instead of the values for specific post-decision states. That is, it estimates the value of being at time t_k with an average free time budget of vehicles β_k^F and an average free time budget of test-centers β_k^H.

4.3 State space partitioning

If we assume the time to be discretized, the vector space can be represented with a three-dimensional lookup table with dimensions t, $\beta \in \{1, \ldots, T\}$. Thus, even with state space

aggregation, the number of individual values (entries in the lookup table) is potentially still very high. Since a large number of observations is required to adequately approximate a value, it is typically not beneficial to approximate the value for each vector $(t_k, \beta_k^F, \beta_k^H)$ individually. That is, it would require a large number of simulation runs and/or lead to low-quality approximations. Therefore, to further alleviate the computational intractability, as well as to improve the quality of the approximation process, we additionally apply a state space partitioning. The idea is to partition the aggregated state space by grouping several vectors $(t_k, \beta_k^F, \beta_k^H)$ to a set, which is then represented by a single lookup table entry. A naive approach would be to use a lookup table with equidistant interval lengths of the parameters. This approach suffers from the trade-off between approximation reliability and state space representativeness. On the one hand, entries must be observed often to provide an efficient and reliable approximation. This is best achieved with a low number of entries, i.e. a coarse-grained lookup table. On the other hand, for the approximation to be effective, the partitioning must enable an accurate representation of the state space. That is, it must be able to distinguish between heterogeneous states. This is best achieved with a large number of entries, i.e. a fine-grained lookup table. To meet this trade-off, we implement the DLT proposed in Ulmer *et al.* (2018a), which adapts the partitioning according to the approximation process, where We represent the DLT as an oct-tree in our implementation.

4.4 Large neighborhood search for computing the decisions

For each potential decision x_k in a state S_k, we use an LNS algorithm to determine the corresponding routing, assignment and facility location decisions. In other words, we use an LNS to generate the set of feasible decisions $X(S_k)$ in each state S_k. Specifically, we use the LNS proposed in Wolfinger *et al.* (2023b), which was designed to solve the CDTP. For the sake of brevity, we only outline its main characteristics here and refer the interested reader to the original article for a detailed description of the method.

The proposed LNS uses six destruction heuristics and three repair heuristics. Each destruction heuristic removes a certain number of suspected cases from the solution. The repair heuristics iteratively reinsert the removed potential cases into the solution until either all cases are covered or none can be inserted any more. Some of the used heuristics are well-known general-purpose heuristics from the literature, while others are specifically designed to address either the vehicle routing aspect, the facility location aspect, or the assignment aspect. The selection of the destruction heuristic and the repair heuristic in each iteration is done at random, according to a uniform probability distribution. To exploit promising solutions even further, a local search is performed on all new solutions that are within a certain threshold of the current best found solution. As an acceptance criterion threshold-acceptance is used, which allows occasional deterioration of solutions throughout the search, in order to better escape local optima. The stopping criterion is run time.

In order to be applicable to our problem setting, some minor modifications of the LNS algorithm were required. First, any functionality that can change the used infrastructure – opening/closing of vehicle routes and test-centers – is only permitted to run in the first epoch ($k = 0$) and switched off in all subsequent epochs ($k > 0$). Second, all destruction heuristics were modified such that suspected cases which were scheduled in a test-center by a previous decision (in a previous epoch) cannot be removed anymore. The rationale for this being that once a person has been instructed to visit a certain test-center at a certain time, changing this scheduling would result in great inconvenience for them. Third, Wolfinger *et al.* (2023b) explicitly consider the scheduling of collected specimens in clinical laboratories. Since we do not consider this problem aspect in this work, we switched off all functionality regarding laboratories in the LNS. Lastly, we extended the stopping criterion such that the algorithm also terminates after a certain number of iterations without improvement. Computing the set of feasible decisions $X(S_k)$ means that the LNS is typically called several times in each epoch. To be precise, the number of calls to the LNS is equal to the cardinality of the power-set of W_k^i;

since each subset of W_k^i constitutes a potential set P_k^{acc} to accept. Using the number of iterations without improvement as a termination criterion allows us to better distribute the computational effort to where it is needed, which in turn helps us to improve the overall performance of the AVI algorithm. We distinguish between the first epoch ($k = 0$) and all subsequent epochs ($k > 0$) with regard to the stopping criteria, given that the number of suspected cases in epoch 0 is significantly larger than in any other epoch, and additionally the decision which test-centers to open and which vehicles to use must be made as well.

We kept all parameter settings of the LNS the same as proposed in Wolfinger *et al.* (2023b), with only a few exceptions. We list these exceptions in the computational study section when we discuss the parameter values of the AVI algorithm (Section 5.2).

4.5 Approximate value iteration algorithm

Let $\mathcal{E}^i$ denote the DLT in simulation run i, η denote an entry in the DLT, $N(\eta)$ denote the number of observations of entry η, $\sigma(\eta)$ denote the standard deviation of the value of η (used to measure the degree of heterogeneousness of values within an entry), and $\overline{N}$ and $\overline{\sigma}$ be the averages over all entries in the lookup table. The input to the algorithm are the initial DLT $\mathcal{E}^0$, the initial values $\overline{V}^{x,0}$, the step size α, and the set of sampled problem realizations $\overline{\Omega}$. In the initialization phase, i as well as N and σ for each entry $\eta \in \mathcal{E}^0$ are set to zero. Afterwards, the algorithm iteratively simulates the problem realization. For every problem realization ω^i, it simulates states, decisions based on the current value estimates, and stochastic transitions to the next states. Additionally, it stores observed entries and newly obtained value estimates. At the beginning, the set of observed states $\mathcal{P}$ and the set of new value estimates $\widehat{\mathcal{V}}$ are both empty. For each $k > 0$, the state $S_k = (S_{k-1}^x, W_k^i)$ is the combination of the previous post-decision state S_{k-1}^x and the respective part of the problem realization $W_k^i \subset \omega^i$. In a given state S_k, the algorithm first computes the set of feasible decision $X(S_k)$ using the LNS metaheuristic, and then calculates the exploration rate $\epsilon^i(S_k)$ to decide whether exploration should be performed or not. Accordingly, it selects either a random decision $x_k^i \in_R X(S_k)$, where $\in_R$ denotes the random selection, or the decision x_k^i, where the corresponding entry η_k^x maximizes *Bellman's equation*. Either way, a new value estimate $\widehat{v}_k^i$ is obtained. Then, the algorithm saves the observed entry and newly obtained value estimate and uses the selected decision to transition to the according post-decision state. The simulation run for problem realization ω^i stops when $S_k = S_K$. After each simulation run i, the value estimates $\overline{V}^{x,i}$, and $N(\eta)$ and $\sigma(\eta)$ for each observed entry $\eta \in \mathcal{P}$ are updated. The function UpdateSigma$(\sigma(\eta), \widehat{v})$ takes the standard deviation $\sigma(\eta_{k-1}^x)$ of entry η_{k-1}^x and updates it with the new value estimate $\widehat{v}_k^i$. Finally, the observed entries are analyzed and, if beneficial, split into eight new entries, and the lookup table updated accordingly. At the end, the algorithm returns the value estimates $\overline{V}^{x,m}$ after m simulation runs, together with the corresponding DLT $\mathcal{E}^m$.

4.5.1 Update lookup table. To decide whether to separate a DLT entry, we use the condition proposed by Ulmer *et al.* (2018a). They propose to split a cell if

$$\frac{\sigma(\eta_k^x) N(\eta_k^x)}{\overline{\sigma}\overline{N}} \geq \tau,$$

where the input parameter τ represents the separation frequency (small values of τ lead to a fast separation of the table and ultimately a large number of DLT cells, while large values for τ lead to a slow separation and a small final number of cells). Additionally, in order to prevent premature separation of entries, we require a minimum number of observations before a cell may be separated. That is, the equation $N(\eta_k^x) \geq \phi$ must hold as well, where ϕ is an input parameter.

When an entry is separated, we create eight new cells by dividing all three intervals in half. The value estimate, the number of observations and the standard deviation are transferred to

these new cells, and the original cell is replaced. Like in Ulmer *et al.* (2018a), due to a lack of further knowledge regarding the distribution of the values, the standard deviation and the number of observations are equally divided among the new cells, and the value estimate remains as is.

The overall process to solve a problem instance – the complete solution approach, so to say – consists of two phases. In the first phase, the training/simulation phase, a policy is identified by using the above-described AVI algorithm to train the DLT. Then, in the second phase, the execution phase, when we actually solve a problem instance, the trained DLT is used to estimate the value of each potential decision.

5. Computational study

In this section, we present the computational experiments performed to assess the performance of our solution approach and summarize insights obtained from analysing real-world-based problem instances. The experiments can be categorized into two groups: simulation runs and test runs. Simulation runs are used to train the lookup tables. Test runs, which use the trained lookup tables, are used to estimate the expected rewards achieved by our approach, which we in turn use to analyze our approach and draw insights from. All algorithms were implemented in C++, and each experiment was executed on a single thread.

In the following, we first describe the problem instances that we used for the computational experiments (Section 5.1). We then discuss the algorithm parameter values in Section 5.2 and evaluate its performance in Section 5.3. Finally, in Section 5.4, we present some algorithmic and managerial insights.

5.1 Test instances

We evaluate the performance of our approach on the test instances presented in Ulmer *et al.* (2018a). The proposed instances are designed for a single vehicle problem and vary with respect to the service region size, the *degree-of-dynamism (dod)*, and the geographical clustering of service requests. Two sizes of a square service area are considered – 15 kilometers ($\mathcal{A}_{15}$) and 20 kilometers ($\mathcal{A}_{20}$) side length – with the depot always located in the center. Time is discretized and the end of the time horizon T is set to 360 min in all instances. The number and location of service requests are treated as independent random variables. For all instances, the expected number of overall service requests n is equal to 100. The expected number of early requests n_0 depends on the $dod \in \{0.5, 0.75\}$ and is computed as $n_0 = (1 - dod) \cdot n$. The number of late service requests in the interval $[1, T]$ follows a Poisson process with λ requests per $T - 1$ min, where $\lambda = dod \cdot n$. Hence, the number of requests in the interval $(t_k, t_{k+1}]$ is Poisson-distributed with rate $\lambda \cdot (t_{k+1} - t_k)/(T - 1)$. The number of early service requests is Poisson-distributed as well, with rate n_0.

Three different geographical distributions of service requests are considered: uniformly distributed across the service region ($\mathcal{F}_U$), grouped in two clusters ($\mathcal{F}_{2C}$) and grouped in three clusters ($\mathcal{F}_{3C}$). Within the clusters, the service requests are normally distributed around the cluster centers. For the large service area ($\mathcal{A}_{20}$), the spatial parameters are as follows. For $\mathcal{F}_{2C}$, the cluster centers are located at coordinates (5,5) and (15,15), and the service requests are equally distributed to each cluster. For $\mathcal{F}_{3C}$, the cluster centers are located at coordinates (5,5), (5,15) and (15,10), whereby 50% of the requests are assigned to cluster two and 25% to each other cluster. All standard deviations are set to 2 kilometers. For the smaller service region ($\mathcal{A}_{15}$), all spatial parameters are multiplied with the factor 0.75. The travel time between two locations is calculated by dividing the Euclidean distance by a constant speed (25 km/h) and rounding up to the next full minute. The base unit of time $\bar{t}$ is set to one minute.

Overall, all combinations of service area size, *dod* and spatial distribution yield a total of twelve problem instances. Like Ulmer *et al.* (2018a), we generated 10,000 realizations of early- and late-service requests for each problem instance, which we then used for the test runs. Henceforth, we will call the above-described instances SDVRP instances.

In addition to the SDVRP instances, we generated problem instances specifically designed for the SDCDTP. They are based on real-world data and aim at replicating the public health strategy for clarifying suspected COVID-19 cases in the city of Vienna, Austria. The implemented strategy in Vienna was to operate a small number of large test-centers together with a large vehicle fleet located at a single depot. Thus, the considered service area in our SDCDTP instances is the territory of Vienna. We use the real-world locations of the test-centers (three in total) and the vehicle depot. For the geographical distribution of the suspected cases, we draw from the real-world distribution of the population in Vienna. We set the end of the time horizon T to 720 min (12 h shifts) and consider two different values for the overall number of expected suspected cases: $n \in \{250, 500\}$. The number of suspected cases and their times of occurrence for a realization are generated by a Poisson process, in the same way as in the SDVRP instances. We consider two values for the *dod* (0.5 and 0.75) and calculate the travel time between two locations in the same way as described for the SDVRP instances. The base unit of time $\bar{t}$ is set to 10 min. In all problem instances, the coverage range of each test-center is set to 60 min, and the length of a single test-center time slot is set to 90 min. Furthermore, we set the time-to-test limit to 24 h and assume that 30% of the suspected cases must be clarified by a mobile test-team (i.e. $\Gamma_p = 1$). For the SDCDTP, all combinations of expected number of suspected cases and *dod* yield a total of four problem instances. We again generated 10,000 realizations of early- and late-known suspected cases for each problem instance and used them for the test runs.

All realizations for all problem instances are publicly available at Wolfinger *et al*. (2023a).

5.2 Algorithm parameter values

In the following, we describe the considered parameter values for the DLT, the LNS and the AVI. For several of the parameters, we use different values depending on which instance set we address, given the substantial structural differences between the SDVRP instances and the SDCDTP instances. Additionally, when solving the SDVRP instances, we represent the DLT as a quad-tree (instead of an oct-tree), since only the two features *point of time* and *free time budget* are necessary for the state space aggregation. While this has no noteworthy influence on the behavior of our approach, it yields a substantial performance increase in terms of run time and memory consumption.

Regarding the parameters for the DLT, we set the initial interval length to 16 min for the SDVRP instances and to 64 min for the SDCDTP instances. Furthermore, for the SDVRP instances, we test two different values for the separation frequency τ, 1.0 and 1.5, and two different values for the minimum number of observations before a cell may be separated ϕ, 100 and 200. For the SDCDTP instances, we use $\tau = 1.0$ (the better value for the SDVRP instances) and test three different values for ϕ: 75, 100 and 150.

For the LNS, we set the run time limit to 120 s and 5 s for the first epoch and all subsequent epochs, respectively, for all instances. The maximum number of iterations without an improvement was set to 250 iterations and 50 iterations for the first epoch and all subsequent epochs, respectively, for all instances. All other parameter values for the LNS are kept the same as presented in Wolfinger *et al*. (2023b).

We initialize the value estimates for the AVI with $\overline{V}^{x,0} = 0$, and test three values for the exploration parameter γ: 0.5, 0.75 and 0.99. Like Ulmer *et al*. (2018a), we run $|\overline{\Omega}| = 1$ million simulation runs for each lookup table for the SDVRP instances. For the SDCDTP instances, we run $|\overline{\Omega}| = 500000$ simulation runs for each lookup table. At the end of each simulation run, we update the value estimates to the moving average, i.e. $\alpha = 1/N(\eta)$. Regarding the objective function weights, ρ_1 can be seen as opportunity costs, since every vehicle used for clarifying suspected cases is not available for regular emergency services. It can, however, also be seen as a required (or desired) minimum gain – in the form of additionally covered suspected cases – for using an additional vehicle. Following the second approach, we set ρ_1 to 20 and 25 for the SDCDTP problem instances with 250 and 500 expected suspected cases, respectively. These

values are taken from the computational study of Wolfinger *et al.* (2023b) and (roughly) correspond to the average number of visited suspected cases by a vehicle in their problem instances. Like ρ_1, ρ_2 represents a required (or desired) minimum gain for using an additional test-center. For this reason, we make ρ_2 dependent on the size of the test-center and set it to 25% of the respective test-center's capacity. The rationale being that a test-center should only be opened if we expect to use at least one-quarter of its capacity. Finally, we set ρ_3 to 0.3. For the SDVRP instances, all three weights – ρ_1, ρ_2 and ρ_3 – are of course set to 0.

For every parameter value combination and problem instance, we run 10,000 test runs to determine the best combination for both instance sets. The corresponding policies are then used for the evaluation.

5.3 Evaluating the performance of the AVI algorithm

In order to evaluate the performance of our approach, we compare our results on the SDVRP instances with the results obtained by Ulmer *et al.* (2018a). They use a simple CI heuristic as the routing algorithm to compute the decisions in each epoch, which we do likewise.

Table 1 presents for both approaches the percentage of served late-known service requests, averaged over 10,000 test runs. On the clustered instances, our approach performs slightly better than the approach from Ulmer *et al.* (2018a), with improvements between ca. 0.5% and 4.7%. On the uniform instances, on the other hand, it performs slightly worse, with deteriorations between ca. 1.6% and 6.3%. The deterioration of more than 15% for the large uniformly distributed instances with a *dod* of 50% constitutes an exception to this. The reason for this large gap is that for roughly 25% of these instances, our approach does not find a feasible start solution, i.e. it cannot cover all early-known customer requests. Whenever this happens, the algorithm simply aborts, which results in zero served late requests. Nevertheless, overall, our approach still achieves good results, with average gaps between −4.8% and 1.63%, depending on the instance size and *dod*.

Table 1. SDVRP – comparison with Ulmer *et al.* (2018a) denoted as UMK. Served late-known service requests (in %) and percentage gap between results

Service area	Distribution	UMK	AVI-CI	Gap
$dod = 0.5$				
$\mathcal{A}_{15}$	$\mathcal{F}_U$	58.4	57.2	−2.05
	$\mathcal{F}_{2C}$	81.4	82.9	1.88
	$\mathcal{F}_{3C}$	73.9	77.3	4.65
	Average			1.49
$\mathcal{A}_{20}$	$\mathcal{F}_U$	24.1	20.3	−15.76
	$\mathcal{F}_{2C}$	66.5	67.0	0.77
	$\mathcal{F}_{3C}$	59.4	59.7	0.48
	Average			−4.84
$dod = 0.75$				
$\mathcal{A}_{15}$	$\mathcal{F}_U$	59.5	58.5	−1.62
	$\mathcal{F}_{2C}$	77.4	79.4	2.61
	$\mathcal{F}_{3C}$	72.0	74.8	3.89
	Average			1.63
$\mathcal{A}_{20}$	$\mathcal{F}_U$	45.3	42.5	−6.25
	$\mathcal{F}_{2C}$	65.6	66.2	0.90
	$\mathcal{F}_{3C}$	59.6	60.5	1.52
	Average			−1.28

Note(s): In both approaches, the CI was used in all epochs to compute decisions

In the next set of experiments, we evaluate the impact on the overall solution quality of using LNS as the routing algorithm, instead of CI. For that purpose, we solve each SDVRP instance three more times, varying which algorithm is used to compute the decisions in epoch 0 and all subsequent epochs, respectively: CI/LNS, LNS/CI and LNS/LNS. Table 2 presents for each of these three variants the percentage of served late requests as well as the gap compared to using CI in all epochs (i.e. compared to the results presented in Table 1). The results show that for all problem instances, using LNS leads to an improvement of the solution quality. The improvements range from less than 1% to up to more than 90%, depending on the variant, the service area size, the request clustering and the degree of dynamism. Furthermore, comparing the results with the results of Ulmer *et al*. (2018a), it can be seen that all three variants produce a better solution quality. In general, using LNS in all epochs (LNS/LNS) yields the overall best results (as can be expected), while the CI/LNS variant outperforms the LNS/CI variant on all but one problem instance ($\mathcal{A}_{20}|\mathcal{F}_U|dod = 0.5$). The latter circumstance provides an indication that sophisticated routing during plan execution may be more valuable than a high-quality initial route plan.

From the above results, we can conclude that the quality of the solutions produced by our approach is comparable with that of the current state-of-the-art offline solution method for stochastic dynamic vehicle routing. Thus, applying our approach to a new, but related problem is justified.

5.4 Managerial insights

We start our analyses by summarizing the principal characteristics of the solutions in Table 3. The results show that, on average, roughly 74–85% of the late-known suspected cases were covered, depending on the number of expected cases and the *dod*. To service those cases, 1.0–2.3 vehicles and 1.5–2.0 test-centers were used, resulting in an average T2TD of less than 2 hours. As can be seen, the number of used vehicles and test-centers is well below the number of available vehicles and test-centers. Likewise, the T2TD is significantly lower than the maximum allowed.

Table 2. SDVRP – served late-known requests (in %) and percentage gap compared to using CI in all epochs

Service area	Distribution	CI/LNS Requests	Gap	LNS/CI Requests	Gap	LNS/LNS Requests	Gap
$dod = 0.5$							
$\mathcal{A}_{15}$	$\mathcal{F}_U$	63.2	10.53	61.8	8.12	64.0	11.86
	$\mathcal{F}_{2C}$	84.3	1.61	83.6	0.87	84.4	1.79
	$\mathcal{F}_{3C}$	79.6	2.89	79.1	2.24	79.9	3.32
	Average		5.01		3.74		5.66
$\mathcal{A}_{20}$	$\mathcal{F}_U$	29.9	47.51	37.3	83.63	38.9	91.69
	$\mathcal{F}_{2C}$	70.3	4.97	69.2	3.29	70.7	5.46
	$\mathcal{F}_{3C}$	63.6	6.64	62.4	4.56	63.8	6.92
	Average		19.71		30.49		34.69
$dod = 0.75$							
$\mathcal{A}_{15}$	$\mathcal{F}_U$	62.5	6.79	61.2	4.63	63.0	7.55
	$\mathcal{F}_{2C}$	80.7	1.65	79.7	0.40	80.7	1.60
	$\mathcal{F}_{3C}$	76.1	1.73	75.4	0.80	76.2	1.91
	Average		3.39		1.94		3.69
$\mathcal{A}_{20}$	$\mathcal{F}_U$	46.6	9.84	45.7	7.61	47.1	10.85
	$\mathcal{F}_{2C}$	68.0	2.80	67.3	1.73	68.7	3.81
	$\mathcal{F}_{3C}$	62.8	3.82	61.1	1.04	63.0	4.18
	Average		5.49		3.46		6.28

Table 3. SDCDTP – served late-known suspected cases (in %), used infrastructure and time-to-test duration (T2TD, in hours)

Size	Cases	Vehicles (29)	Test-centers (3)	T2TD
dod = 0.5				
250	81.3	1.0	1.8	1.6
500	85.2	2.3	2.0	1.7
Average	83.3	1.7	1.9	1.6
dod = 0.75				
250	85.2	1.0	1.5	1.2
500	74.4	1.0	1.7	1.1
Average	79.8	1.0	1.6	1.2

Note(s): CI was used in all epochs to compute the decisions. Furthermore, the numbers in brackets next to *Vehicles* and *Test-Centers* represent the number of available vehicles and test-centers

When evaluating (again) the impact on the overall solution quality of using the LNS instead of CI as the algorithm for computing the decisions, we observe that using LNS in all epochs *except* the first (CI/LNS) leads to an improvement of the solution quality by roughly 0.4–4.2%, depending on the number of expected cases and the *dod*. Contrary to the SDVRP instances, and perhaps a bit unexpected, using the LNS in the first epoch (LNS/CI and LNS/LNS) significantly reduces the solution quality, with deteriorations of up to 77%. The reason for this is two-fold. First, the LNS always finds a feasible start-solution that does not use any test-centers but rather services all early-known suspected cases with (only a few) vehicles. Since this is the maximum reward solution, and because no exploration takes place in the execution phase, this solution is always chosen in the first epoch. As a consequence, only (very) little free time budget for servicing late-known suspected cases is available throughout the rest of the day, which ultimately results in a large number of rejections. Second, the AVI algorithm was not able to train the DLT such that it would off-set this negative effect.

One way to mitigate the above-described negative effect of the LNS is to require that at least one test-center must be open in order for a solution to be feasible. Considering the real-world application, this is a reasonable requirement. Table 4 presents the results when this requirement is taken into account. As can be seen, the two variants using LNS in the first epoch (LNS/CI and LNS/LNS) perform significantly better now. Indeed, for *dod* = 0.75 they even outperform the CI/LNS variant with respect to the average percentage of served late-known customers. For the larger instances, when *dod* = 0.5, we can, however, observe that one open test-center is still not enough to compete with two open test-centers, which is indicated by the roughly 10% negative gap. Either way, for all variants, the average amount of used infrastructure is well below the available amount of infrastructure, and likewise, the average T2TD is significantly less than the *time-to-test* limit.

In the next set of experiments, we take a closer look at the trade-off between minimizing the amount of used infrastructure and serving more suspected cases. For that purpose, we solve each SDVRP instance once more and explicitly select, in the first epoch, the decision that opens all test-centers and uses all vehicles. This comparison is given in Table 5. The results show that when we do not minimize the amount of used infrastructure, roughly 97% of the late-known suspected cases can be served on average, across all instances. Furthermore, we can see that still (significantly) less vehicles and test-centers are used than there are available. This may seem counterintuitive at first, and justifiably, the question arises why these unused vehicles and/or test-centers are not used to serve the remaining 3% of still uncovered suspected cases. However, these cases occur so late that they simply cannot be visited by a vehicle or travel to a test-center before the end of the time horizon. Depending on the *dod* and instance size, the above-mentioned 97% coverage rate amounts to an average improvement of

Table 4. SDCDTP – served late-known suspected cases (in %), percentage gap compared to using CI in all epochs, used infrastructure, and time-to-test duration (T2TD, in hours) when it is required that at least one test-center must be open in order for a solution to be feasible

Size	Cases			Gap			Vehicles (29)			Test-centers (3)			T2TD		
	CI/LNS	LNS/CI	LNS/LNS	CI/LNS	LNS/CI	LNS/LNS	CI/LNS	LNS/CI	LNS/LNS	CI/LNS	LNS/CI	LNS/LNS	CI/LNS	LNS/CI	LNS/LNS
dod = 0.5															
250	83.0	85.5	83.5	2.02	5.18	2.72	1.1	2.6	2.5	1.8	1.0	1.0	1.6	3.5	3.5
500	88.8	76.6	76.4	4.20	−10.13	−10.39	2.4	4.1	4.1	2.0	1.0	1.0	1.6	3.7	3.8
Average	85.9	81.1	80.0	3.11	−2.48	−3.83	1.8	3.4	3.3	1.9	1.0	1.0	1.6	3.6	3.6
dod = 0.75															
250	85.5	85.0	86.3	0.39	−0.19	1.30	1.0	1.8	1.8	1.5	1.0	1.0	1.2	2.3	2.1
500	75.3	77.9	76.4	1.30	4.73	2.77	1.1	2.7	2.5	1.7	1.0	1.0	1.1	2.4	2.4
Average	80.4	81.4	81.3	0.84	2.27	2.03	1.1	2.2	2.1	1.6	1.0	1.0	1.2	2.4	2.3

Table 5. SDCDTP – served late-known suspected cases (in %) and number of used vehicles and test-centers when the amount of used infrastructure is minimized (columns *Min*), respectively *not* minimized (columns *All*)

Size	Cases			Vehicles (29)			Test-centers (3)		
	All	Min	Gap	All	Min	Gap	All	Min	Gap
dod = 0.5									
250	96.7	83.0	16.5	2.6	1.1	1.5	2.1	1.8	0.3
500	96.4	88.8	8.6	5.6	2.4	3.1	2.2	2.0	0.2
Average	96.6	85.9	12.6	4.1	1.8	2.3	2.1	1.9	0.3
dod = 0.75									
250	96.5	85.5	12.9	3.6	1.0	2.6	2.1	1.5	0.6
500	96.4	75.3	28.0	7.7	1.1	6.6	2.2	1.7	0.4
Average	96.5	80.4	20.4	5.7	1.1	4.6	2.1	1.6	0.5

Note(s): Additionally, the percentage gap (cases) and the absolute gap (vehicles and test-centers) between both results are shown

approximately 9–28% compared to when the amount of used infrastructure *is* minimized. The price of this improvement is, of course, an increase in the number of used vehicles and test-centers. On average, between 1.5 and 6.6 additional vehicles and 0.2–0.6 additional test-centers are used to achieve this improvement. This corresponds to an average increase of roughly 130–600% and 9–43% of the number of used vehicles and test-centers, respectively, depending on the *dod* and number of expected suspected cases. We can conclude that not minimizing the amount of used infrastructure can be very beneficial in terms of the amount of served suspected cases, but is bought for a (high) price in terms of a (potentially) significant increase in the number of used vehicles and/or test-centers.

We end this chapter by presenting additional solution characteristics that provide further insights into the solution structure. Table 6 shows the average percentage of suspected cases that is tested by a mobile test-team, respectively, in a test-center. Additionally, it shows the average length of a vehicle route (in kilometers), the average number of stops (i.e. number of serviced suspected cases) in a vehicle route, and the average idle time of a vehicle (in hours). Lastly, the table depicts the average utilization of open test-centers. The results show that almost all of the suspected cases that can be tested in a test-center are tested in a test-center. Recall that 30% of the suspected cases must be served by a mobile test-team. Looking at the vehicle routes, we can see that, on average, a vehicle travels 90–130 kilometers and serves around 50 suspected cases, which amounts to roughly 4 cases per hour. Furthermore, we observe that the mobile test-teams are (for the most part) fully utilized, which is indicated by the average idle time of close to 0.0 h. The degree of utilization of the test-centers, on the other

Table 6. SDCDTP – further solution characteristics

Size	Distribution (%)		Vehicle routes			Test-centers
	Mobile	Test-Ctr	Length (km)	# Stops	Idle time (hrs.)	Util. (%)
dod = 0.5						
250	23.9	76.1	109.4	51.8	0.3	6.6
500	26.3	73.7	90.2	52.2	1.0	10.9
Average	25.1	74.9	99.8	52.0	0.6	8.7
dod = 0.75						
250	22.0	78.0	126.4	48.7	0.0	8.0
500	14.8	85.2	102.2	55.0	0.1	13.2
Average	18.4	81.6	114.3	51.9	0.1	10.6

hand, is rather low; with utilization rates between 6.6% and 13.2%, on average. However, this can be expected, given the comparatively large capacity provided by them.

6. Conclusion

In this article, we introduced the SDCDTP, a novel logistics problem that arises in the context of the public health response to epidemics and pandemics of contagious diseases. The task in the SDCDTP is to simultaneously plan vehicle routes for mobile test-teams and to decide the location of test-centers, so that potentially infected people can be tested for the disease, either by a mobile test-team or in an open test-center. In the SDCDTP, only some suspected cases are known in advance, while others arrive randomly throughout the course of the day. For each newly arriving suspected case, we must decide whether to accept or reject it. The objective is to identify a dynamic assignment-and-routing policy that minimizes the number of open test-centers and used vehicles, services all early-known suspected cases, and maximizes the expected number of serviced late-known suspected cases.

We presented a mathematical formulation for the SDCDTP (modeled as an MDP) and proposed a solution approach based on value function approximation for solving the problem. Our proposed solution method interleaves AVI with LNS. We validated the performance of our solution approach through extensive computational experiments. It outperforms the current offline state-of-the-art solver for the SDVRP on the benchmark instances proposed in Ulmer *et al*. (2018a), with improvements of up to more than 90%, depending on the service area size, the request clustering and the degree of dynamism. Furthermore, using real-world-based SDCDTP problem instances, we showed that our algorithm also obtains high-quality solutions for the SDCDTP.

We analyzed and drew insights from the solutions of the real-world-based SDCDTP problem instances. We found that minimizing the number of used vehicles and test-centers can indeed have a (significant) negative impact on the number of covered late-known suspected cases. However, to increase the coverage rate, an equally significant increase in the amount of used infrastructure would be required.

For the SDCDTP problem instances, using LNS in the first decision epoch leads to a significant decrease in the solution quality. The reason for this was that the AVI algorithm was not able to adequately train the dynamic lookup table such that it could off-set the solving power of the LNS. In other words, the required learning effect for the first epoch was not achieved. An interesting direction for future research would therefore be to improve the learning capabilities of our approach. This may be achieved by significantly increasing the number of simulation runs in the training phase through parallelization (although we believe this to be unlikely), or by a more drastic change like using a different state space representation. Another interesting direction for future research would be to additionally consider "no-shows" – patients who do not show up at the test-center or are not at home when the mobile test-team arrives.

References

Arslan, O. (2021), "The location-or-routing problem", *Transportation Research Part B: Methodological*, Vol. 147, pp. 1-21, doi: 10.1016/j.trb.2021.02.010.

Bent, R.W. and Van Hentenryck, P. (2004), "Scenario-based planning for partially dynamic vehicle routing with stochastic customers", *Operations Research*, Vol. 52 No. 6, pp. 977-987, doi: 10.1287/opre.1040.0124.

Ghasemi, P., Ehmke, J.F. and Bicher, M. (2025), "Managing equitable contagious disease testing: a mathematical model for resource optimization", *Omega*, Vol. 135, 103305, doi: 10.1016/j.omega.2025.103305.

Ghiani, G., Manni, E., Quaranta, A. and Triki, C. (2009), "Anticipatory algorithms for same-day courier dispatching", *Transportation Research Part E: Logistics and Transportation Review*, Vol. 45 No. 1, pp. 96-106, doi: 10.1016/j.tre.2008.08.003.

Ghiani, G., Manni, E. and Thomas, B.W. (2012), "A comparison of anticipatory algorithms for the dynamic and stochastic traveling salesman problem", *Transportation Science*, Vol. 46 No. 3, pp. 374-387, doi: 10.1287/trsc.1110.0374.

Grabenschweiger, J., Doerner, K.F. and Hartl, R.F. (2022), "The multi-period location routing problem with locker boxes", *Logistics Research*, Vol. 15, p. 8.

Haghi, M., Arslan, O. and Laporte, G. (2023), "A location-or-routing problem with partial and decaying coverage", *Computers and Operations Research*, Vol. 149, 106041, doi: 10.1016/j.cor.2022.106041.

Hvattum, L.M., Løkketangen, A. and Laporte, G. (2006), "Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic", *Transportation Science*, Vol. 40 No. 4, pp. 421-438, doi: 10.1287/trsc.1060.0166.

Hvattum, L.M., Løkketangen, A. and Laporte, G. (2007), "A branch-and-regret heuristic for stochastic and dynamic vehicle routing problems", *Networks: An International Journal*, Vol. 49 No. 4, pp. 330-340, doi: 10.1002/net.20182.

Ichoua, S., Gendreau, M. and Potvin, J.Y. (2006), "Exploiting knowledge about future demands for real-time vehicle dispatching", *Transportation Science*, Vol. 40 No. 2, pp. 211-225, doi: 10.1287/trsc.1050.0114.

Martínez-Reyes, A., Quintero-Araújo, C.L. and Solano-Charris, E.L. (2020), "A decision support tool for the location routing problem during the covid-19 outbreak in Colombia", *International Conference of Production Research–Americas*, Springer, pp. 33-46.

Mitrović-Minić, S. and Laporte, G. (2004), "Waiting strategies for the dynamic pickup and delivery problem with time windows", *Transportation Research Part B: Methodological*, Vol. 38 No. 7, pp. 635-655, doi: 10.1016/j.trb.2003.09.002.

Powell, W.B. (2011), *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, 2nd ed., John Wiley & Sons, Hoboken, NJ.

Powell, W.B. (2019), "A unified framework for stochastic optimization", *European Journal of Operational Research*, Vol. 275 No. 3, pp. 795-821, doi: 10.1016/j.ejor.2018.07.014.

Santini, A. (2021), "Optimising the assignment of swabs and reagent for pcr testing during a viral epidemic", *Omega*, Vol. 102, 102341, doi: 10.1016/j.omega.2020.102341.

Schneider, M. and Drexl, M. (2017), "A survey of the standard location-routing problem", *Annals of Operations Research*, Vol. 259 Nos 1-2, pp. 389-414, doi: 10.1007/s10479-017-2509-0.

Shahnejat-Bushehri, S., Kermani, A., Arslan, O., Cordeau, J.F. and Jans, R. (2022), "A vehicle routing problem with time windows and workload balancing for covid-19 testers: a case study", *IFAC-PapersOnLine*, Vol. 55 No. 10, pp. 2920-2925, doi: 10.1016/j.ifacol.2022.10.175.

Soeffker, N., Ulmer, M.W. and Mattfeld, D.C. (2019), "Adaptive state space partitioning for dynamic decision processes", *Business and Information Systems Engineering*, Vol. 61 No. 3, pp. 261-275, doi: 10.1007/s12599-019-00582-7.

Srivastava, R. (1993), "Alternate solution procedures for the location-routing problem", *Omega*, Vol. 21 No. 4, pp. 497-506, doi: 10.1016/0305-0483(93)90082-v.

Thomas, B.W. (2007), "Waiting strategies for anticipating service requests from known customer locations", *Transportation Science*, Vol. 41 No. 3, pp. 319-331, doi: 10.1287/trsc.1060.0183.

Ulmer, M.W. (2020), "Horizontal combinations of online and offline approximate dynamic programming for stochastic dynamic vehicle routing", *Central European Journal of Operations Research*, Vol. 28 No. 1, pp. 279-308, doi: 10.1007/s10100-018-0588-x.

Ulmer, M.W., Brinkmann, J. and Mattfeld, D.C. (2015), "Anticipatory planning for courier, express and parcel services", in *Logistics Management*, Springer, pp. 313-324.

Ulmer, M.W., Goodson, J.C., Mattfeld, D.C. and Hennig, M. (2019), "Offline–online approximate dynamic programming for dynamic vehicle routing with stochastic requests", *Transportation Science*, Vol. 53 No. 1, pp. 185-202, doi: 10.1287/trsc.2017.0767.

Ulmer, M.W., Goodson, J.C., Mattfeld, D.C. and Thomas, B.W. (2020), "On modeling stochastic dynamic vehicle routing problems", *EURO Journal on Transportation and Logistics*, Vol. 9 No. 2, 100008, doi: 10.1016/j.ejtl.2020.100008.

Ulmer, M.W., Mattfeld, D.C. and Köster, F. (2018a), "Budgeting time for dynamic vehicle routing with stochastic customer requests", *Transportation Science*, Vol. 52 No. 1, pp. 20-37, doi: 10.1287/trsc.2016.0719.

Ulmer, M.W., Soeffker, N. and Mattfeld, D.C. (2018b), "Value function approximation for dynamic multi-period vehicle routing", *European Journal of Operational Research*, Vol. 269 No. 3, pp. 883-899, doi: 10.1016/j.ejor.2018.02.038.

WHO (2018), *Managing Epidemics: Key Facts about Major Deadly Diseases*, World Health Organization, Geneva, Licence: CC BY-NC-SA 3.0 IGO.

Wolfinger, D., Gansterer, M. and Doerner, K.F. (2023a), "sdcdtp", mendeley data, v1, doi: 10.17632/gryttgptsg.1.

Wolfinger, D., Gansterer, M., Doerner, K.F. and Popper, N. (2023b), "A large neighbourhood search metaheuristic for the contagious disease testing problem", *European Journal of Operational Research*, Vol. 304 No. 1, pp. 169-182, doi: 10.1016/j.ejor.2021.10.028.

Zhang, J., Luo, K., Florio, A.M. and Van Woensel, T. (2023), "Solving large-scale dynamic vehicle routing problems with stochastic requests", *European Journal of Operational Research*, Vol. 306 No. 2, pp. 596-614, doi: 10.1016/j.ejor.2022.07.015.

Corresponding author
Margaretha Gansterer can be contacted at: margaretha.gansterer@aau.at