Time-Dependent Stochastic Vehicle Routing Problem with Random Requests: Application to online police patrol management in Brussels

Michael Saint-Guillain[a,*], Célia Paquay[b,c], Sabine Limbourg[c]

[a] Université catholique de Louvain, Belgium
[b] School of Business and Economics, Maastricht University, Maastricht, The Netherlands
[c] HEC-Management School of the University of Liege, Liege, Belgium

---

Abstract

The Static and Stochastic Vehicle Routing Problem with Random Requests (SS-VRP-R) describes realistic operational contexts in which a fleet of vehicles has to serve customer requests appearing dynamically. Based on a probabilistic knowledge about the appearance of requests, the SS-VRP-R seeks *a priori* sequences of vehicle relocations, optimizing the expected responsiveness to the requests. In this paper, an existing computational framework, based on recourse strategies, is adapted to meet the objectives of the SS-VRP-R. The resulting models are applied to a real case study of the management of police units in Brussels. In this context, the expected average response time is minimized. To cope with the reality of the urban context, a time-dependent variant is also studied (TD-SS-VRP-R) in which the travel time between two locations is a function that depends on the departure time at the first location. Experiments confirm the contribution and the adaptability of the recourse strategies to a real-life, complex operational context. Provided an adequate solution method, simulation-based results show the high quality of the *a priori* solutions designed, even when compared to those designed by field experts. Finally, the experiments provide evidence that there is no potential gain in considering time-dependency in such an operational context.

Keywords: Routing, stochastic programming, on-demand transportation, optimization under uncertainty, recourse strategies

---

1. Introduction and background

The Vehicle Routing Problem (VRP) and its variants have become increasingly popular in the academic literature. The VRP consists of determining an optimal set of routes to be carried out by a fleet of vehicles to perform a set of customer requests (Toth and Vigo, 2014) at a minimum distance or some measure more or less functionally related to distance (e.g., travel time or cost). Whereas deterministic VRPs assume that input data are known with certainty, in real-world applications, some input data may be uncertain when computing a solution (Ritzinger et al., 2016).

---

[*] Corresponding author

Email addresses: `michael.saint@uclouvain.be` (Michael Saint-Guillain), `cpaquay@uliege.be` (Célia Paquay), `sabine.limbourg@uliege.be` (Sabine Limbourg)

Following Pillac et al. (2013), VRPs can be classified into four categories, depending on the *solution evolution* and *information quality*. Table 1 summarizes the different VRP categories.

Table 1: The four different VRP categories.

		Information is known with certainty?	
		Yes	No
Decisions can be modified in response to new information revealed after time 0?	No	Static and deterministic	Static and stochastic
	Yes	Dynamic and deterministic	Dynamic and stochastic

In order to highlight the difference from dynamic and stochastic problems, instead of the common appellation of "stochastic VRP" we prefer the full denomination *"static and stochastic VRP"* (SS-VRP). The same taxonomy has been adopted by Psaraftis et al. (2016). Depending on the operational context, we distinguish between two fundamentally different assumptions.

First, if the currently unexecuted part of the solution can be arbitrarily redesigned, then we are facing a *Dynamic and Stochastic VRP*, as described in Pillac et al. (2013): *"dynamic and stochastic problems have part or all of their input unknown and revealed dynamically during the execution of the routes, [...], the vehicle routes can be redefined in an ongoing fashion with the help of technological support"*. In this case, the solution is adjusted by reoptimizing the new current problem while fixing the executed partial routes. Since the new problem is NP-hard, approximation algorithms such as Approximate Dynamic Programming (see, *e.g.*, Maxwell et al. (2010); Schmid (2012)) or (meta)heuristics (see, for example, Ichoua et al. (2006); Bent and Van Hentenryck (2007); Saint-Guillain et al. (2015)) are preferred to exact solution methods.

Second, if the routes can only be adapted by following some minor changes, then we are facing a *Static and Stochastic VRP*. In the SS-VRP, whenever a bit of information is revealed, the current solution is adjusted by applying a *recourse strategy*. Instead of reoptimizing online, a predefined recourse strategy defines the way in which the requests are handled whenever they occur. Based on the probabilistic information, we seek a first-stage (also called *a priori*) solution that minimizes its a priori cost, plus the expected sum of penalties caused by the recourse strategy. For the evaluation function to remain tractable, the recourse strategy must be efficiently computable and hence simple enough to avoid reoptimization. According to Pillac et al. (2013), *"static and stochastic problems are characterized by input partially known as random variables, which realizations are only revealed during the execution of the routes; it is assumed that routes are designed a priori and only minor changes are allowed afterwards"*. For example, in Bertsimas (1992), the customers are known, whereas their demands are revealed online. Two different assumptions are considered, leading to different recourse strategies, as illustrated in Figure 1. In strategy **a**, each demand is revealed when the vehicle arrives at the customer location. If the vehicle reaches its maximal capacity, the first-stage solution is then modified by adding a round trip to the depot. In strategy **b**, each demand is revealed when leaving the

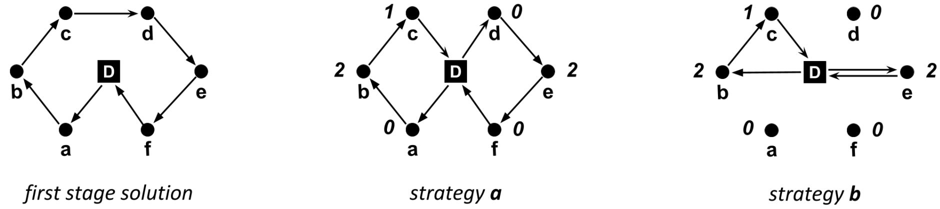


Figure 1: Recourse strategies for the SS-VRP with stochastic customers and demands (Bertsimas, 1992). The vehicle capacity is 3. The first-stage solution states the *a priori* sequence of customer visits. When applying strategy ***a***, the vehicle unloads at the depot after visiting **c**. In strategy ***b***, absent customers (**a**, **d**, **f**) are skipped.

previous customer, allowing to skip customers with null demands.

In the review by Gendreau et al. (2016), the authors argue for new recourse strategies: with the increasing use of information and communications technologies, customer information is likely to be revealed on a high-frequency basis. In this context, the chronological order in which this information is transmitted no longer matches the planned sequences of customers on the vehicle routes. In particular, the authors consider as paradoxical the fact that the existing literature on SS-VRPs with random customers assumes full knowledge of the presence of customers at the beginning of the planning horizon.

In this paper, we introduce the Static and Stochastic VRP with Random Requests (SS-VRP-R) in which requests arriving dynamically must be served as soon as possible. We assume that we have some probabilistic knowledge of the requests; for instance, historical data can be used to determine a probability distribution. Unlike models that exploit time windows, such as the SS-VRP with Time Windows and both Random Customers and Demands (SS-VRPTW-CR) introduced in Saint-Guillain et al. (2017), in the SS-VRP-R, the requests do not have a time window: whenever a request occurs, it is automatically accepted. In this study, the SS-VRP-R is applied to a case study faced by a specific team of mobile police units in Brussels. This team aims at taking action on urgent interventions, such as road traffic accidents, violence, or alarms. Consequently, these units spend their time cruising the city, waiting for intervention requests. We investigate the best relocation policies for each of these mobile units, thereby minimizing the average expected response time.

To tackle this SS-VRP-R, we exploit *a priori* optimization. As described in Toth and Vigo (2014), this approach is based on a two-stage model in which an *a priori* solution is determined during the first stage. This *a priori* solution is then executed in the second stage while uncertainties are revealed and recourse actions are taken according to a chosen strategy. In our case, based on the probabilities of customer request appearance, our approach to tackle the SS-VRP-R consists of seeking an *a priori* solution composed of preventive vehicle routes, minimizing the expected average response time at the end of the day. Such an *a priori* route describes a sequence of strategic vehicle relocations from which nearby requests can be reached rapidly while following a predefined recourse strategy (Gendreau et al., 1996). Such relocation problems arise in several emergency service organizations (Brotcorne et al., 2003; Mitrović-Minić and Laporte, 2004; Simpson and Hancock, 2009). According to Pillac et al. (2013), re-locating idle vehicles to strategic locations is the keystone of emergency vehicle dispatching or redeployment.

Gendreau et al. (2001) develop a dynamic ambulance dispatching and redeployment system that includes a parallel tabu search heuristic to precompute redeployment scenarios. Their simulations are based on a real-life call distribution provided by Urgences Santé. The mathematical model proposed by Haghani and Yang (2007) takes into account service requirements and coverage concerns for future demand by re-locating the on-route vehicles and remaining vehicles among stations. Brotcorne et al. (2003) classify the models and optimization methods related to ambulance services. The authors consider the use of stochastic programming with recourse for the development of dynamic models. This consideration is confirmed by Bélanger et al. (2019) who focus on location, re-location and dispatching decisions, discussing the interactions between them.

The re-location strategy has also been used for other vehicle routing problems. Mitrović-Minić and Laporte (2004) present four waiting strategies for the dynamic pickup and delivery problem with time windows arising in courier companies. For a similar purpose, Ghiani et al. (2009) exploit probabilistic information about future requests while Pureza and Laporte (2008) assume no vehicle repositioning and focuses on scheduling strategies based on waiting and buffering strategies.

In the field of ambulance re-location, the paper by Naoum-Sawaya and Elhedhli (2013) is the closest to our work. The authors present a two-stage stochastic program to reestablish the maximum coverage when some ambulances are responding to emergency calls. The objective is to minimize the number of re-locations while maintaining an adequate service level. In the first stage, their approach determines the location of the ambulances; in the second stage, decisions concern the assignment of emergency calls to ambulances. Their approach differs from ours at two levels. First, they model the vehicle relocations by segmenting the horizon into decision periods. An integer program is solved at each decision period, deciding where the vehicles are relocated. Thus, the vehicle routes and waiting durations are defined implicitly, whereas our model decides those explicitly. Second, the expected quality of service at the second stage is computed based on a limited sample of 50 scenarios, among the 95840 available. In contrary, by reasoning on random variables rather than scenarios, our computational framework computes expected cost in light of all the available stochastic knowledge (*e.g.* all the probabilities carried out by the 95840 scenarios, without explicitly enumerating these).

Commonly, the second stage expected cost is estimated by Monte Carlo sampling whereas our approach is based on exact computation as for Saint-Guillain et al. (2017).

The first contribution of this study is to describe closed-form expressions to compute, in pseudo-polynomial time, the exact expected cost of an *a priori* solution under either fixed or time-dependent travel durations. The second major contribution of this paper is that it comes with an original application to dynamic police patrol management. Indeed, as highlighted in Dewinter et al. (2020)'s review, there is a knowledge gap of solution methods to solve the police patrol routing problem. Thanks to historical data and the field experience provided by the Brussels police department, we empirically show that the SS-VRP-R model can be exploited to enhance the reactivity of mobile units in the Brussels urban area. Finally, our approach considers that the travel durations may vary over the planning horizon. This feature is termed time-dependent in the literature (Gendreau et al., 2015). It has been extensively proven that significant benefits can be achieved if traffic congestion is accounted for when planning the routes and associated schedules (e.g., Ichoua et al. (2003); Potvin et al. (2006); Kok et al. (2012)). In this work, experiments are also conducted while considering the travel duration variability caused by city congestion, eventually leading to surprising results.

Note that whereas the police central receives requests with different levels of urgency, in our case study, we only considered emergency intervention requests. Queuing theory may be needed to determine when to deal with lower priority incidents or to keep a patrol free and available to deal with any urgent incidents that may arise in the near future. However, in our context, each request has the same priority. In Brussels, the frequency of such urgent requests is sufficiently low (7 requests per day for 9 patrols) to assume that a police patrol never deals with more than one open request at a time.

In the next section, we present the formulation of the SS-VRP-R and cover its building components in detail. In Section 3, we depict the particular recourse strategy we propose for the SS-VRP-R and describe how to efficiently compute its expected cost. The case study is introduced in Section 4, in addition to the involved hypotheses. The computational results that are related to the considered case studies are discussed in Section 5, while closing remarks are given in Section 6.

2. Problem definition and methodology

The SS-VRP-R tackled in this article is to serve a set of online requests with a given set of available vehicles while minimizing the expected average response time (or response time). For this purpose, we develop an algorithm to generate a *first-stage solution* based on some probabilistic knowledge about the missing data. Moreover, a *recourse strategy* is provided to adapt the first-stage solution to the requests revealed dynamically. The recourse strategy aims at describing which vehicle located at which waiting location is supposed to serve a request if it reveals to appear. This section presents the SS-VRP-R input data, solutions and objective function for a predefined recourse strategy. A specific recourse strategy is then proposed in Section 3.

2.1. Input data

The SS-VRP-R is defined on a complete and directed graph $G = (V, A)$, where the vertex set $V = \{0\} \cup W \cup C$ is composed of a depot 0, a set of waiting locations W indexed by $w = \{1, ..., |W|\}$, and a set of serviceable customer vertices C, and where $A = \{(i, j) : i, j \in V, i \neq j\}$ is the arc set. A travel duration $d_{i,j}$ is associated with arc $(i, j) \in A$. There is an homogeneous fleet composed of vehicles indexed by $k \in K = \{1, ..., |K|\}$. The planning horizon is discretized as $H = \{1, ..., h\}$, with each moment $t \in H$ representing the beginning of the corresponding period.

A potential request r is described by a location $c_r \in C$, a time $\Gamma_r \in H$, a service duration s_r, and the probability to occur p_r. We assume independence between request probabilities. We denote by R the set of potential requests. A scenario, denoted ξ, is a set of potential requests that appear by the end of the horizon H. We denote the set of scenarios by $\mathcal{S}$, where each scenario $\xi \in \mathcal{S}$ has a probability of occurrence $p(\xi)$.

All of these notations are summarized in Table 2.

Table 2: Notation summary

W	Set of waiting locations, with $W_0 = W \cup \{0\}$
C	Set of serviceable customer locations
$V = \{0\} \cup W \cup C$	Set of vertices composed of the depot (0), W, and C
A	Arc set
$G = (V, A)$	Complete directed graph
$H = \{1, ..., h\}$	Discrete planning horizon
$d_{i,j}$	Travel duration along arc $(i, j) \in A$
K	Set of vehicles
R	Set of potential requests
$r \in R$	Potential request r for customer location c_r at time Γ_r
c_r	Customer location associated with request r
Γ_r	Moment at which the request r appears
s_r	Service duration of request r
p_r	Probability of potential request r
ξ	A scenario is a set of occurring requests during operations
$p(\xi)$	Probability associated with scenario ξ
$\mathcal{S} = \{\xi_i \| i = 1, ..., \|\mathcal{S}\|\}$	Set of scenarios

2.2. *First stage*

The first-stage solution, x, is computed offline before the beginning of the planning horizon. Each waiting location in W is visited at most once in x, over the whole horizon H. The solution consists of a set of $|K|$ plans. A plan x_k is the route assigned to a vehicle k visiting a subset of the waiting locations. A route is an ordered sequence of waiting locations, starting and ending at the depot. An arrival time A_w and a departure time D_w are associated with every waiting location in a route w. These two pieces of information are interdependent considering that if a vehicle visits the vertex w and immediately after w', one has

$$A_{w'} = D_w + d_{w,w'}.$$

Because of this relation, either the arrival time or the departure time is sufficient to describe the route timewise. Therefore, in the following, a route x_k is described as an ordered list of elements (i, A_i), where i is a waiting vertex and A_i is the arrival at the vertex i.

It is to be noted that, whereas for mathematical modelling purposes a waiting location cannot be visited twice, this limitation could be inappropriate in various operational contexts. In practice, this can be circumvented by adding to the model duplicates for the waiting vertices. When using a heuristic solution method, the algorithm may be adapted to allow revisiting waiting locations.

2.3. *Second stage*

Under the predefined recourse strategy $\mathcal{R}$, let

$$\text{response}^{\mathcal{R}}(r, x, \xi)$$

be the deterministic function that returns the response time, that is, the amount of time between the moment Γ_r at which an online request r is revealed and the moment at which a vehicle reaches location of request c_r, when following first-stage solution x within a specific scenario ξ. We are interested in the average time needed to meet all the requests that occurred in ξ, based on the first-stage solution x:

$$\mathcal{Q}(x, \xi) = \frac{1}{|\xi|} \sum_{r \in \xi} \text{response}^{\mathcal{R}}(r, x, \xi) \tag{1}$$

Hence, we define the *expected second-stage value function* as the expectation of the average time needed to meet any request that reveals to appear, over all the possible scenarios described by random variable $\boldsymbol{\xi}$:

$$\begin{aligned} \mathcal{Q}(x) &= \mathrm{E}[\mathcal{Q}(x, \boldsymbol{\xi})] \\ &= \frac{1}{|R|} \sum_{r \in R} \mathrm{E}\big[\, \text{response}^{\mathcal{R}}(r, x, \boldsymbol{\xi}) \mid r \text{ appears} \,\big], \end{aligned} \tag{2}$$

where $\text{response}^{\mathcal{R}}(r, x, \boldsymbol{\xi})$ is a random function of the *a priori* solution x, the recourse strategy $\mathcal{R}$ and the random variables $\boldsymbol{\xi}$ describing the potential requests R. Moreover, we define $\mathrm{P}\{\text{response}^{\mathcal{R}}(r, x, \boldsymbol{\xi}) = \Delta \mid r \text{ appears}\}$ as the probability that r is visited after a response time of Δ time units, conditionally that r appears in ξ. Henceforth, the SS-VRP-R problem can be formulated as the following two-stage stochastic program:

$$\begin{aligned} \text{(SS-VRP-R)} \quad & \underset{x}{\text{Minimize}} \ \ \mathcal{Q}(x) && (3) \\ & \text{s.t.} \quad\ x \text{ is a first-stage solution,} && (4) \end{aligned}$$

where the expected second-stage value function $\mathcal{Q}(x)$ is defined in (2) and implicitly depends on a specific recourse strategy $\mathcal{R}$.

2.4. *Stochastic Integer Programming formulation*

The problem stated by (3)-(4) refers to a nonlinear stochastic integer program with recourse. In order to clarify the problem, a three-index vehicle flow formulation is provided in this section (based on Boland and Savelsbergh (2019)). It uses the following decision variables:

- $y_{ik} = 1$ if and only if the waiting vertex $i \in W$ is visited by vehicle $k \in K$;
- $x_{ijk} = 1$ if and only if the arc $(i, j) \in W^2$ is selected in route $k \in K$;
- A_i is the arrival time at the waiting location $i \in W$ if visited, 0 if not visited in any route.

$$\underset{x}{\text{Minimize}} \quad \mathcal{Q}(x) \tag{5}$$

subject to

$$\sum_{j \in W_0} x_{ijk} = \sum_{j \in W_0} x_{jik} \qquad \forall\, i \in W_0,\ k \in K \tag{6}$$

$$\sum_{j \in W_0} x_{ijk} = y_{ik} \qquad \forall\, i \in W_0,\ k \in K \tag{7}$$

$$\sum_{k \in K} y_{0k} \leq |K| \tag{8}$$

$$\sum_{k \in K} y_{ik} \leq 1 \qquad \forall\, i \in W_0 \tag{9}$$

$$A_i \leq h \sum_{k \in K} y_{ik} \qquad \forall\, i \in W_0, \tag{10}$$

$$(A_i + d_{i,j} - A_j) \leq (1 - \sum_{k \in K} x_{ijk})h \qquad \forall\, i, j \in W_0 \tag{11}$$

$$y_{ik} \in \{0,1\} \qquad \forall\, i \in W_0,\ k \in K \tag{12}$$

$$x_{ijk} \in \{0,1\} \qquad \forall\, i,j \in W_0 :\ i \neq j,\ k \in K \tag{13}$$

$$A_i \in H \qquad \forall\, i \in W_0 \tag{14}$$

Constraints (6) are flow conservation constraints, meaning that on any route, if a vehicle enters a waiting vertex, it has to leave it as well. Constraints (7) ensure that a waiting vertex is visited on a route if an arc entering the waiting vertex is selected on this route. This set of constraints state that a waiting vertex may be visited only once on a route. Constraint (8) limits the number of available vehicles. Constraints (9) ensure that each waiting vertex is visited at most once over all the routes. Constraints (10) and (11) are related to time considerations. Constraints (10) state that if a waiting vertex is not visited on any route, then the arrival time at this waiting vertex is 0. If the waiting vertex is visited, it should be before the end of the planning horizon H. Constraints (11) ensure that if the arc (i, j) is selected on one route, then the arrival time at waiting vertex j should be greater than (*i.e.*, later) or equal to the arrival time at the predecessor waiting vertex i plus the duration $d_{i,j}$ of the arc.

3. Recourse strategy and expected cost computation

Given a first-stage solution x, a recourse strategy states how the requests, which appear dynamically, are handled by the vehicles. In other words, it defines how the second-stage solution is gradually constructed, based on x and depending on these online requests. A more formal description of recourse strategies is provided in Saint-Guillain et al. (2017).

Ideally, whenever a request appears, the right vehicle should be selected to optimize the operational performances. Furthermore, if several requests appear at the same time unit, then the order in which they are handled also plays a critical role. Unfortunately, none of these decisions can be made optimally without reducing to a NP-hard problem. The role of a recourse strategy is then to make such online decisions efficiently computable, usually in linear time. Finally, a proper recourse strategy enables for efficient computation of the expected cost of any first-stage solution under that predefined strategy.

Request preassignment and ordering.

The recourse strategy tailored for our problem is denoted $\mathcal{R}$ in the following. Before the start of the operations and in order to avoid reoptimization, the set R of potential requests is preordered by reveal times and indices (to break further ties). Each potential request $r \in R$ is then preassigned to exactly one planned waiting location visited in the first-stage solution, and therefore one vehicle, based on temporal (arrival and departure times at waiting locations versus Γ_r) and geographical (position of the waiting location and of the request, c_r) considerations. The waiting location associated to a request r is denoted $w(r)$ in what follows.

Vehicle operations.

The recourse strategy $\mathcal{R}$ states the request assignment and ordering beforehand, while no actual requests are known yet. The recourse strategy is then applied during operations (*i.e.*, online) to determine the appropriate vehicle actions. Basically, in $\mathcal{R}$, all the vehicles behave independently. While following its own route plan x_k, a vehicle k becomes available for a period $[A_i, D_i[$ at each waiting location $(i, A_i) \in x_k$ in turn. During that period, the vehicle performs round trips from i to visit its online requests as they appear, based on the predefined assignment and ordering of R, as described in Algorithm 1. The set P refers to the potential requests that have already appeared at current time t, plus those for which we still ignore whether or not they will appear, their presence will be revealed in the future. Figure 2 provides a simplified illustration of the vehicle recourse actions applied to a first-stage solution, at a specific current time t.

Algorithm 1: Operations (TRAVEL, SERVE or WAIT) of a vehicle in $\mathcal{R}$, at current time t. Let i be the waiting vertex the vehicle is currently assigned to, according to first-stage solution x.

```
if $t = D_i$ then TRAVEL from $i$ to $i + 1$;
else
    $P$ ← potential requests $r$ assigned to $i$ (i.e. $w(r) = i$) not yet satisfied, either appeared or
      not revealed ($t < \Gamma_r$);
    if $P = \emptyset$ then TRAVEL to $i + 1$;
    else
        $r^{\text{next}}$ ← smallest element of $P$, according to the request preordering;
        if $t < \Gamma_{r^{next}}$ then WAIT until $t + 1$ ;
        else TRAVEL to $r^{\text{next}}$, SERVE it during $s_{r^{\text{next}}}$ time units, and TRAVEL back to $i$;
```

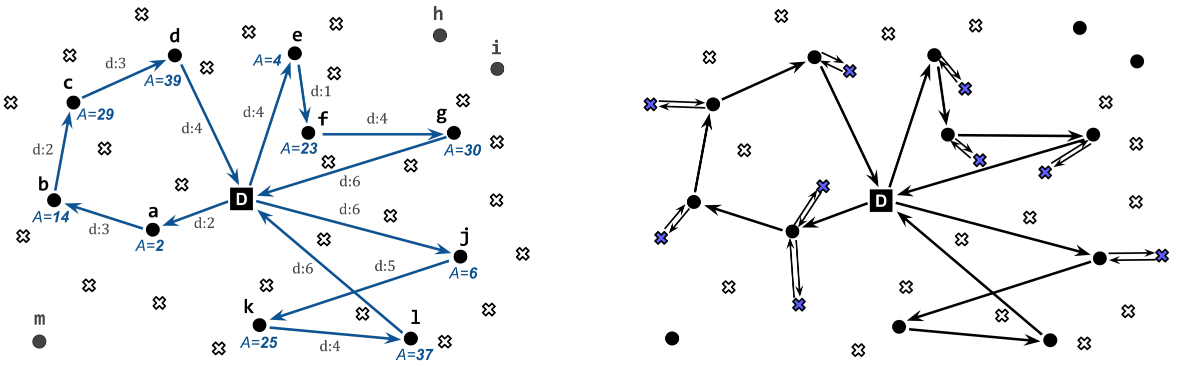


Figure 2: **Left**: a first-stage solution involving three vehicles. Waiting locations are illustrated as circles and customer regions as crosses. Note that waiting locations h, i and m are not part of the plan. **Right**: the $\mathcal{R}$ recourse strategy applied to the first-stage solution. Online requests are visited by performing round trips from their associated waiting locations.

Expected second-stage value function.

In its current form, and provided an implementation of recourse strategy $\mathcal{R}$ allowing the computation of $\text{response}^{\mathcal{R}}(r, x, \xi)$, Eq. (2) can be directly approached by enumerating all the possible scenarios in $\mathcal{S}$:

$$\mathcal{Q}(x) = \sum_{\xi_i \in \mathcal{S}} \frac{p(\xi_i)}{|\xi_i|} \sum_{r \in \xi_i} \text{response}^{\mathcal{R}}(r, x, \xi_i). \tag{15}$$

Naturally, due to the size of $\mathcal{S}$, the support of $\boldsymbol{\xi}$, such an approach is definitely not feasible in practice. Sampling methods, such as the Sample Average Approximation (SAA, Verweij et al. (2003)), allow to approximate $\mathcal{Q}$ on a limited subset of $\mathcal{S}$, but they do not provide any guarantee. So far, most of the approaches proposed in the literature (on stochastic vehicle routing) rely on sampling a limited subset of $\mathcal{S}$. In what follows, we show how to efficiently compute $\mathcal{Q}$, without sampling, by directly exploiting the closed-form expressions provided in Saint-Guillain et al. (2017), for the SS-VRPTW-CR.

3.1. Computation of $\mathcal{Q}$ through closed-form expressions

We compute the probability that at a time $t \in H$, a request r already appeared and the vehicle leaves $w(r)$ to satisfy it. Let us denote this probability $g_1(r, x, t)$:

$$g_1(r, x, t) = \text{P}\{\text{request } r \text{ appeared at time } \Gamma_r \leq t \text{ and } \mathrm{departureTime}(r) = t\}$$

where $\mathrm{departureTime}(r)$ is the time at which the vehicle leaves vertex $w(r)$ in order to serve r. Let $t_r^{\text{min}} = \max\{A_{w(r)}, \; \Gamma_r\}$ be the earliest time for leaving waiting location $w(r)$ to satisfy request r located at vertex c_r. Let also $t_r^{\text{max}} = D_{w(r)} - d_{w(r),c_r} - s_r - d_{c_r,w(r)}$ be the latest time at which a vehicle can handle r from waiting location $w(r)$, with respect to r's planned departure time $D_{w(r)}$ from $w(r)$. Thereby, $\mathcal{R}$ strictly respects the first-stage plan when visiting the waiting vertices. These are guaranteed to be visited according to their arrival (A_i) and departure (D_i) times. The probability that a request r is satisfied is the probability that both r appears and that $\mathrm{departureTime}(r) \leq t_r^{\text{max}}$:

$$\text{P}\{r \text{ satisfiable}\} = \sum_{t=1}^{t_r^{\text{max}}} g_1(r, x, t) = \sum_{t=t_r^{\text{min}}}^{t_r^{\text{max}}} g_1(r, x, t). \tag{16}$$

As $\mathrm{departureTime}(r)$ depends on previous operations on the same waiting location $w(r)$, we calculate $g_1(r,x,t)$ recursively starting from the first potential request r_1 assigned to $w(r)$ and up to the current request r, according to the preordering. The base case, for the first potential request r_1 assigned to $w(r)$, is then

$$g_1(r_1,x,t) = \begin{cases} p_{r_1}, & \text{if} \quad t = t^{\min}_{r_1} \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

Indeed, if r_1 appeared, then the vehicle leaves $w(r)$ at time $t^{\min}_{r_1}$. The general case of a request $r \neq r_1$, with $w(r) = w(r_1)$, depends on the time at which the vehicle has completed the preceding request. Let $f(r,x,t)$ be the probability that at time t, given a first-stage solution x, either the vehicle is back at vertex $w(r)$ after having served r or the vehicle discards r because it was not satisfiable or because r has not been revealed. This moment $t \geq \Gamma_r$ represents the time at which the vehicle becomes available for the next request, right after r (whether it appeared or not), if any further requests are assigned to $w(r)$. For any request r that is not the first request associated with vertex $w(r)$, we define $g_1(r,x,t)$ based on f-probabilities of the direct predecessor of request r denoted r^-:

$$g_1(r,x,t) = \begin{cases} p_r \cdot f(r^-,x,t) & \text{if} \quad t > \Gamma_r \\ p_r \cdot \sum_{t'=A_{w(r)}}^{\Gamma_r} f(r^-,x,t') & \text{if} \quad t = \Gamma_r \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

Indeed, if $t > \Gamma_r$, the vehicle leaves w to serve r as soon as it gets rid of the previous one r^-. In such case, $g_1(r,x,t)$ is the probability that both r has appeared and the vehicle is available for it at time t, that is, finished with request r^- at time t. At time $t = \Gamma_r$, we must consider the possibility that the vehicle was waiting for being able to serve r, but from an earlier time $t' < \Gamma_r$. The overall probability that the vehicle leaves $w(r)$ for request r at time $t = \Gamma_r$ is then p_r multiplied by all the f-probabilities that the vehicle was actually available from a time $t' \in]A_{w(r)}; \Gamma_r[$. At any time before Γ_r, the probability that the vehicle leaves $w(r)$ must obviously be zero.

The f-probabilities of a request r depend on what happened to r. From a time t, there are three possibilities: the request r has been served and thus consumed operational time, r appeared but was not satisfiable, or r did not appear at all:

$$\begin{aligned} f(r,x,t) = \; & \underbrace{g_1(r,x,t-S_r) \cdot \delta(r,t-S_r)}_{r \text{ has been served}} \\ & + \underbrace{g_1(r,x,t) \cdot (1-\delta(r,t))}_{r \text{ was not satisfiable}} \\ & + \underbrace{g_2(r,x,t)}_{r \text{ did not appear}} \end{aligned} \tag{19}$$

where $S_r = d_{w(r),c_r} + s_r + d_{c_r,w(r)}$ is the time needed by the vehicle to serve the request r and the function $\delta(r,t)$ returns 1 if and only if request r is satisfiable from time t and vertex $w(r)$, i.e.,

$\delta(r,t) = 1$ if $t \leq t_r^{\max}$, and $\delta(r,t) = 0$ otherwise. The first term in the summation of the right-hand side of Eq. (19) gives the probability that request r actually appeared and was satisfied. In such a case, $\mathrm{departureTime}(r)$ must be the current time t, minus time s_r needed for serving r and the travel time (both ways). The second and third terms of Eq. (19) add the probability that the vehicle was available at time t but that request r did not consume any operational time. There are only two possible reasons for that: either r actually appeared but was not satisfiable (second term) or r did not appear at all (third term). The probability that r did not appear and is discarded at time t is denoted by $g_2(r,x,t)$ and is computed as follows.

For the first potential request r_1 of waiting vertex $w(r)$, one has

$$g_2(r_1,x,t) = \begin{cases} 1 - p_{r_1} & \text{if} \quad t = t_{r_1}^{\min} \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

In our computational framework, we define the moment at which r is discarded to coincide with the time the vehicle becomes available for other future requests, which is necessarily after it reaches its current waiting location at time $A_{w(r)}$. The general case for any request r assigned to waiting vertex w, happening after r_1, is quite similar to the probability g_1:

$$g_2(r,x,t) = \begin{cases} (1-p_r) \cdot f(r^-,x,t) & \text{if} \quad t > \Gamma_r \\ (1-p_r) \cdot \sum_{t'=A_{w(r)}}^{\Gamma_r} f(r^-,x,t') & \text{if} \quad t = \Gamma_r \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$

where r^- is the direct predecessor of request r at location $w(r)$.

3.2. SS-VRP-R expected response times: constant travel durations

Given a first-stage solution x, function $g_1(r,x,t)$ returns the probability that a vehicle leaves its current location at time t to serve a request r. Function g_1 implicitly depends on a predefined recourse strategy, which we call $\mathcal{R}$ in general. A vehicle must depart from location $w(r)$ at time unit $\Gamma_r + \Delta - d_{v,c_r}$ in order to meet a request r with a response time Δ. If we assume that a request r that appears is immediately accepted, then

$$\mathrm{P}\{\mathrm{response}^{\mathcal{R}}(r,x,\boldsymbol{\xi}) = \Delta \mid r \text{ appears}\} \;=\; \frac{g_1(r,x,\ \Gamma_r + \Delta - d_{w(r),c_r})}{p_r} \tag{22}$$

if and only if x is such that r is always accepted under recourse strategy $\mathcal{R}$. For such a solution x, Eq. (2) then becomes

$$\mathcal{Q}(x) = \frac{1}{|R|} \sum_{r \in R} \sum_{\Delta=0}^{h-\Gamma_r} \Delta \cdot \frac{g_1(r,x,\ \Gamma_r + \Delta - d_{w(r),c_r})}{p_r}. \tag{23}$$

3.3. TD-SS-VRP-R expected response times: time-dependent travel durations

As further discussed in our case study below, time-dependent (TD) travel durations could be an important aspect to take into account while dealing with urban vehicle routing. Indeed, Li and Keskin (2014) highlighted that travel durations should be linked to real-time traffic conditions.

By now, we have described how our equations can be adapted accordingly. A noteworthy feature of our recourse strategy is that it can be easily adapted to a context in which the travel duration along an arc depends on the moment the trip is performed. Let us define

$$\text{dep}^{\text{TD}}_{v,v'}(t^{\text{arr}})$$

as the time a vehicle must depart v in order to reach v' at time t^{arr}. Applying the appropriate modifications to g_1 function, we obtain its time-dependent version g_1^{TD}. One just needs to redefine the mathematical expressions that involve travel duration, namely, $t_r^{\min}$, $t_r^{\max}$ as well as $f(r, x, t)$:

$$t_r^{\text{minTD}} = \max\{A_{w(r)},\ \Gamma_r\}\ ,\ \ t_r^{\text{maxTD}} =\ \text{dep}^{\text{TD}}_{w(r),c_r}\big(\,\text{dep}^{\text{TD}}_{c_r,w(r)}(t^{\text{arr}}) - s_r\,\big)$$

where t_r^{maxTD} is the time at which a vehicle needs to depart from w in order to make a round trip to r while servicing it and returning back to w at time t^{arr}. Finally, we set

$$f^{\text{TD}}(r, x, t) = g_1^{\text{TD}}(r^-, x, S_r^{\text{TD}}(t)) \cdot \delta(r^-,\ S_r^{\text{TD}}(t)) +\ g_1^{\text{TD}}(r^-, x, t) \cdot \big(1 - \delta(r^-,\ t)\big)\ +\ g_2^{\text{TD}}(r^-, x, t).$$

With $t_{r,w}^{\text{minTD}}$ and $t_{r,w}^{\text{maxTD}}$, the time-dependent versions of random functions g_1^{TD} and g_2^{TD} are obtained by replacing the f functions by their time-dependent version f^{TD}. These are the only modifications required to adapt $\mathcal{R}$ to time-dependent travel durations. Eq. (23) then becomes

$$\mathcal{Q}(x) = \frac{1}{|R|} \sum_{r \in R} \sum_{\Delta=0}^{h-\Gamma_r} \Delta \cdot \frac{g_1^{\text{TD}}\big(r, x,\ \text{dep}^{\text{TD}}_{w(r),c_r}(\Gamma_r + \Delta)\big)}{p_r}. \tag{24}$$

In Section 4.2, we explain how the time-dependent departure time function $\text{dep}^{\text{TD}}_{v,v'}$ is computed in practice, within our specific operational context.

3.4. Computational complexity

Given a first-stage solution x, the complexity of computing the expected cost is equivalent to the one of filling up a $|R| \times h$ matrix global along the planned waiting locations in order to store all the $f(r, x, t)$ probabilities. By processing incrementally on each waiting location separately, each matrix cell can be computed in constant time using Eq. (19). In particular, once the probabilities in cells $(r^-, 1), \ldots, (r^-, t)$ are known, the cell (r, t) such that r is not the first of a waiting location can be computed in $\mathcal{O}(1)$ according to Eq. (18) - (21). Given n customer regions and a time horizon of length h, we have at most $|R| = nh$ potential requests. It then requires at most $\mathcal{O}(nh^2)$ constant time operations to compute $\mathcal{Q}(x)$.

Under time-dependency considerations, computing the time-dependent departure times, if not precomputed in advance, may involve at most h operations to each cell, leading to a $\mathcal{O}(nh^3)$ complexity. Precomputing all the time-dependent departure times between any couple of locations (n customer regions and m waiting locations) then leads to $\mathcal{O}\big(nh^2 + (n+m)h^2\big)$.

4. Case study: Brussels police department

Our methodology developed to tackle the SS-VRP-R is applied to the case study faced by a specific team of mobile police units in Brussels in charge of urgent interventions only. We propose the best relocations strategies for every unit to minimize the average expected response time.

In this section, we describe how we derived the problem input data (graph nodes, travel durations, and request probabilities) for the historical data provided by the police department. The historical data are included in a list of recorded events, each corresponding to a call received from citizens (or alarms), or a situation spotted on-the-fly by the police unit (*e.g.*, flagrante delicto). Each event is described by a timestamp and Global Positioning system (GPS) coordinates. It is also described by a type, allowing to distinguish among alarms, armed violence, traffic accidents, *etc.* The recorded period ranges from 2013 to 2017 (inclusive).

Statistics based methods combined with geographic information system and crime mapping techniques allow identifying areas of high crime intensity known as *hot spots*. A crime hot spot is first described as a cluster of addresses (Sherman and Weisburd, 1995). The term is currently frequently defined as being an area of high concentration of crime relative to the distribution of crime across the entire study area (Chainey, 2013). The results of the meta-analysis conducted by Braga et al. (2019) confirm that focusing police efforts at hot spots can be effective in preventing crime.

The problem input data must be realistically inferred from these available records, in a way that best describes the situation while remaining both computationally and practically tractable. In particular, some parts of the data will be ignored while constructing our benchmark. We focus on a particular operational context, namely, what happens in the city from 4 am to 10 am as a first step, and from 4 am to 8 pm as a second step, during regular working days. In what follows, the input data are inferred from historical records ranging from 2013 to 2016. Records of the year 2017 are kept for experimental validation purposes only.

4.1. Graph nodes: customer and waiting vertices

Because our approach requires a finite set of serviceable customer and waiting vertices, the operational area, namely, the city of Brussels, must be appropriately discretized. For practical reasons, such as to limit the size of the time-dependent travel duration matrices, we limit the number of vertices to 150.

To this end, we look at the set of all recorded events and cluster them geographically, using a k-means algorithm, thus with $k = 150$. Figure 3 shows the distribution of these clusters, as well as the distribution of the events as a heat map.

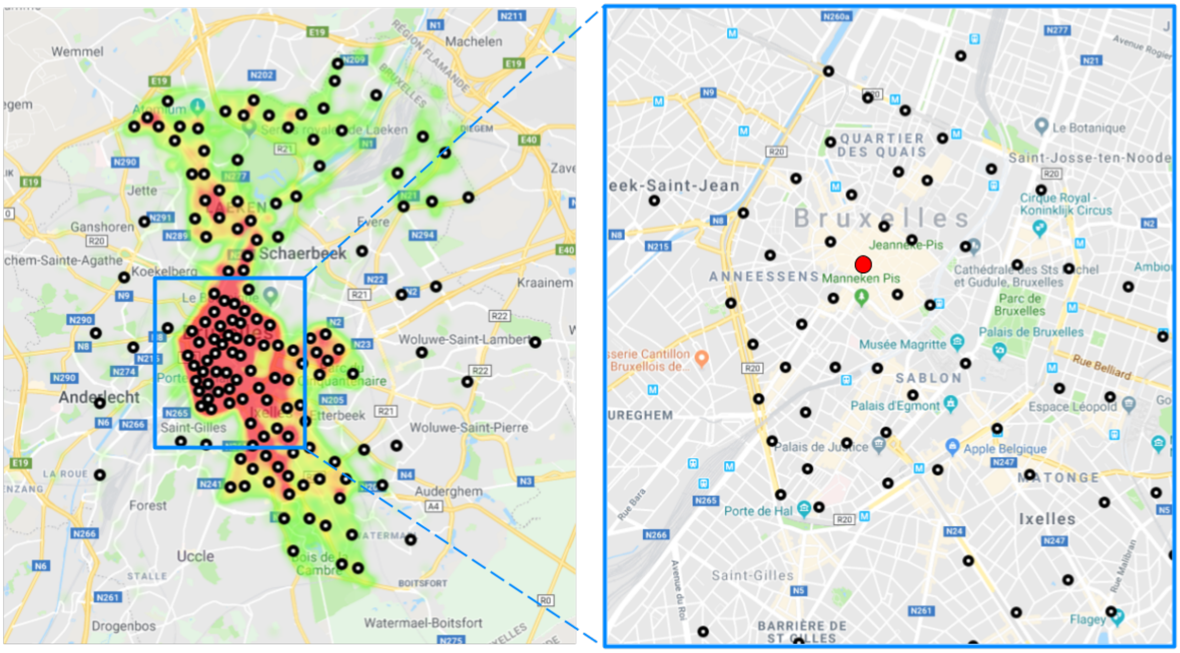


Figure 3: **Left:** heat map of the concentration of events recorded from 2013 to 2017: low (green), mid (yellow), and high (red). The concentration of the events along a vertical axis is due to the particular areas covered by the police department. The 150 clusters, which are the vertices composing V, are represented by circles. **Right:** focus on the city center. The depot vertex is depicted in red, next to the main square.

In what follows, the set of graph vertices V will then be constituted of the locations of these clusters and, consequently, customer and waiting vertices will then be taken among them: $C = V, W \subseteq V$, meaning that they are potential requests on every vertex of V. The configurations of waiting vertices are further described in Section 5. The depot is placed on the area of the Central Commissariat, hence belonging to the vertices $0 \in V$.

4.2. *Time-dependent travel durations*

The time needed to connect two given locations within Brussels is dependent on the time the trip is performed. Figure 4 shows how these travel durations vary for two different journeys illustrated in Figure 5, depending on both the moment and the type of the day: either working day (Tuesday) or day off (Sunday).

These data have been retrieved by requesting Google Maps' API, for every five minutes of a 24-hour day. Interestingly, the travel duration variations during a working day, for the long urban trip between Porte de Hal and the Atomium (Figure 4, upper green curve), appear to be very similar to the variations depicted in Eglese et al. (2006) in the northwest of England.

Based on this analysis, we split the 24 hours into 14 time bins. We use these time bins in order to construct our time-dependent travel-time matrices. Considering Figure 4, we have the following bins: $b_1 =]5:00; 6:00]$, $b_2 =]6:00; 7:00]$, ..., $b_{14} =]21:00; 5:00]$.

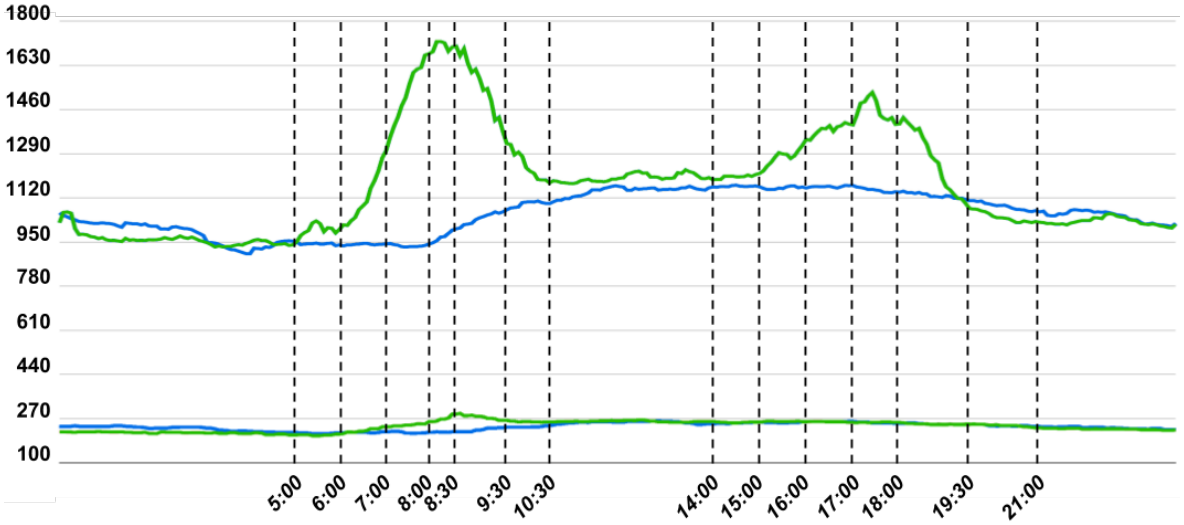


Figure 4: The predicted travel durations (seconds) of two different journeys, depending on the time of day (departure every five minutes) and day of week (green: working day, blue: day off): from Manneken Pis to the Central Station (down) and from Porte de Hal to the Atomium (up). Time bins are separated by dashed lines.

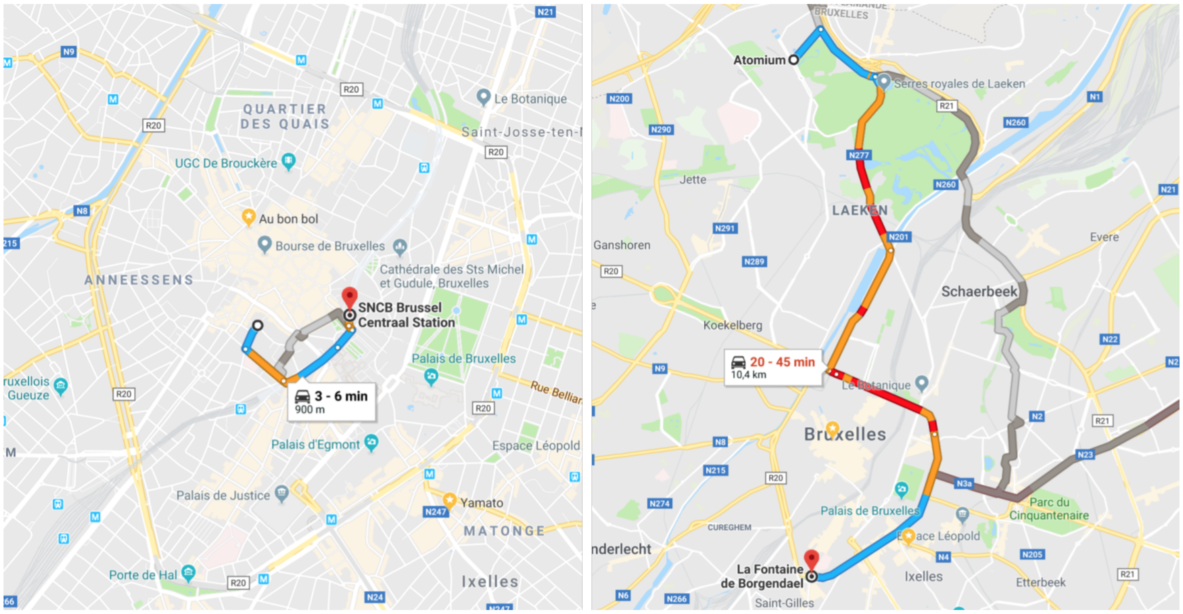


Figure 5: **Left**: a short urban path in Brussels, from Manneken Pis to the Central Station. **Right**: a long urban path, from Porte de Hal to the Atomium.

We hence retrieved, for each arc $(v, v') \in V^2$ of our graph and for each time bin $[t, t'[$, the predicted time needed for traveling from v to v' when leaving v at time t. This is achieved by requesting the Google Maps' API, while setting the departure time to t and departure date to either a working or an off day. The date is taken sufficiently late in the future, so that the prediction best reflects the historical average for this time of the day and day of the week. In practice, considering a working day requires $150^2 \times 14 = 315\ 000$ travel durations. Travel durations during off days are not required, as our benchmark considers regular working days. It should be noted that whereas these travel times come from observations of regular travelers (Google Maps), in practice, police patrol cars traveling with blue lights on are significantly less impacted by congestion. In this study, we do not make the distinction.

For each arc $(v, v') \in V^2$, these time-dependent matrices result in a piecewise-constant function $f_{v,v'} : H \rightarrow \mathbb{R}^+$ returning a travel duration based on the time bin at which the trip is started, with $\forall v, \forall t, f_{v,v}(t) = 0$. However, the leaps from one time bin to another in the step function $f_{v,v'}$ result in unrealistic properties. In particular, as discussed in Ichoua et al. (2003) and Fleischmann et al. (2004), this results in a non-FIFO property.

The FIFO (*First In, First Out*) property, also called the nonpassing property (Ahn and Shin, 1991), states that a vehicle leaving a location v for a location v' at a given time will arrive earlier than any identical vehicle leaving location v for location v' at a later time. In other words, one cannot arrive earlier at the destination by delaying its departure. In order to restore the FIFO property, we compute our FIFO time-dependent departure time function $\text{dep}^{\text{TD}}_{v,v'}$ by using the algorithm described in Eglese et al. (2006). It follows the approach proposed in Ichoua et al. (2003) and has the advantage of being computationally efficient when travel durations are short, on average, compared to the time bins (which is our case). Their procedure works as follows. Provided the travel distance along arc (v, v'), a piecewise constant *time-dependent speed function* is computed based on the original travel duration function $f_{v,v'}$; whenever a trip must be spread across several time bins, the algorithm then makes use of the speed function to determine a total travel duration that preserves the FIFO property. Note that instead of explicitly following the algorithm from Ichoua et al. (2003), we propose a slightly more meaningful dynamic programming version. Let $\mu_{v,v'}(t) = \text{dist}_{v,v'}/f_{v,v'}(t)$ return the travel velocity along arc (v, v') at time t, based on its length $\text{dist}_{v,v'}$. On this basis, we obtain our FIFO time-dependent departure time function $\text{dep}^{\text{TD}}_{v,v'}(t^{\text{arr}})$, discussed in Section 3.3, from the recursive function:

$$\phi_{v,v'}(a,\ d) = \begin{cases} a - \frac{d}{\mu_{v,v'}(a)} & \text{if} \quad a - \frac{d}{\mu_{v,v'}(a)} > \underline{\text{bin}}(a) \\ \phi_{v,v'}\Big(\underline{\text{bin}}(a),\ d - \mu_{v,v'}(a) \cdot \big(a - \underline{\text{bin}}(a)\big)\Big) & \text{otherwise,} \end{cases} \tag{25}$$

returning the correct departure time, when arriving at time a, and traveling along arc (v, v') for a distance d. Here, d measures the distance to travel along a part of the arc (v, v'), with $d \leq dist(v, v')$. The whole idea is that the entire arc to be traveled is split into a number of subparts (of different lengths), each corresponding to a time bin, and each part is traveled at a different speed (also determined by the corresponding time bin). Here, $\underline{\text{bin}}(a)$ is the infimum of the time bin that contains a. In

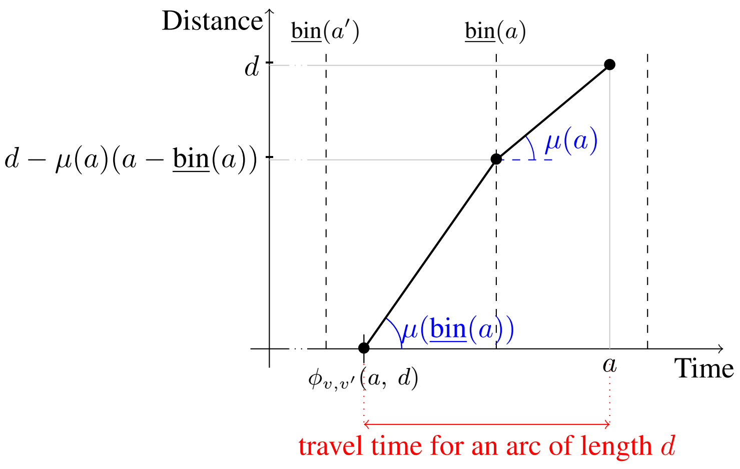

Distance
$\underline{\text{bin}}(a')$
$\underline{\text{bin}}(a)$
d
$d - \mu(a)(a - \underline{\text{bin}}(a))$
$\mu(a)$
$\mu(\underline{\text{bin}}(a))$
$\phi_{v,v'}(a,\ d)$
a
Time
travel time for an arc of length d

Figure 6: Representation of recursive formula (25) if the travel duration of the arc is split over two time bins.

fact, we necessarily have

$$\forall a' < a : \underline{\text{bin}}(a) < a' \ \Rightarrow \ \mu_{v,v'}(a') = \mu_{v,v'}(a),$$

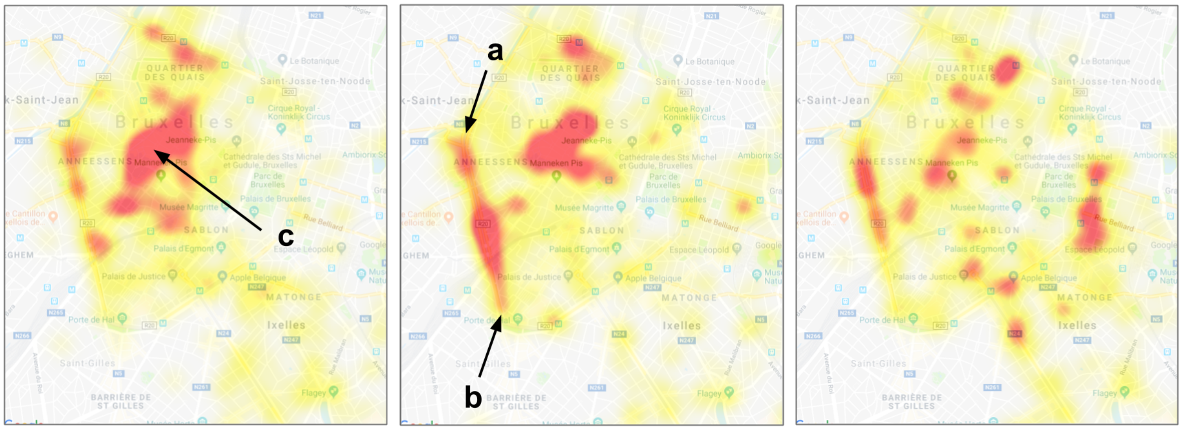
which justifies that if $a' = a - d/\mu_{v,v'}(a)$ falls in the same time bin as a, then a' is the correct departure time. If $a' \leq \underline{\text{bin}}(a)$, then it means that a part of the distance d is traveled during the previous time bin, thus at a different speed. We need to apply the ϕ function on the remaining distance $d - \mu_{v,v'}(a) \cdot \big(a - \underline{\text{bin}}(a)\big)$, now for an arrival time at $\underline{\text{bin}}(a)$. Suppose that $\phi_{v,v'}(a, d)$ falls in the time bin that directly precedes that of a; then, the total distance is thus split into two parts, as illustrated in Figure 6. The departure time, from v in order to reach v' at time t^{arr}, is then finally given by

$$\text{dep}^{\text{TD}}_{v,v'}(t^{\text{arr}}) = \phi_{v,v'}\big(t^{\text{arr}},\ \text{dist}_{v,v'}\big). \tag{26}$$

4.3. Potential requests: exploiting historical data

The history of events that occurred in the city during the observed years (2013 to 2016) allows us, for any period of interest (*e.g.*, Mondays, 7 am to 8 am), to compute the average event activity of a given area. This provides an indication of the likelihood that an event will occur at a corresponding period. For instance, the heat maps in Figure 7 illustrate the observations that can be made on the data set. When looking at events that occurred during every working day morning, we notice a significant evolution from 4 am to 10 am, where traffic incidents, of course, play an important role during the rush hour.

Each historical event occurred at a specific location at a specific date and time. We first partition the period of observations (2013 to 2016) into intervals of interest. We consider only regular working days in our benchmark: Monday to Friday, excluding national holidays. Since they may not be representative of regular working days, the months of July, August and December are also ignored,



4 - 6am **6 - 8am** **8 - 10am**

Figure 7: Evolution of the average event densities during working day mornings around the city center. From 4 am to 6 am, events seem rather concentrated (red) in downtown (**c**). From 6 am to 8 am, a significant part of the events occur along the main traffic lane, between points **a** and **b**. Events then tend to become sparser after the rush hour, from 8 am to 10 am.

as most Belgian citizens take holidays during these periods. After filtering, 5180 events remain in the database, observed over 738 days. We thus have an average of 7 observed events per day.

Similarly, we aggregate events by area of interest. We define these areas as being the clusters described above, computed when discretizing the geography of our operational context. Each event is therefore grouped to the closest vertex $v \in V$ in terms of coordinates. In doing so, the average distance between an event and its associated vertex is approximately 200 meters. The average number of events by cluster is approximately 34.

For a classical operational day, our entire planning horizon thus represents 16 hours (4 am to 8 pm). The horizon is discretized using one time unit per minute, that is, $h = 16 \times 60$. It is further divided into intervals of 30 minutes (or time units), called *time slots*. It is assumed that no more than one request may arise at a given vertex for a given time slot. We hence associate a potential request r to each of vertex $v \in C \subset V$ and time slot $\Gamma \in \{1, 31, 61, \ldots\}$, where Γ is thus the earliest time unit of the corresponding time slot. When considering the entire horizon, this leads to $|R| = 150 \times 16 \times 2 = 4800$ potential requests. We then approximate the request's probability p_r based on historical data, as the average number of times an event appeared in the area belonging to the cluster $c \in V$ and during the interval $[\Gamma, \Gamma + 30 \text{ min}[$ of the day.

5. Experiments and results

Our experiments aim at answering the following questions. Under a two-stage assumption, that is, when excluding online reoptimization, is it possible to identify first-stage solutions that beat a simple, intuitive operational policy? Also, what is the impact of the time-dependent travel durations on the quality of the first-stage decisions?

Based on the stochastic knowledge that we were able to extract from our historical records, the question is to determine whether the SS-VRP-R model is useful at making good *a priori* decisions in our police patrol context. To that extent, first-stage solutions are computed under various experimental conditions (varying the number of vehicles, the set of waiting locations, etc.). The solutions obtained on the basis of the data from 2013 to 2016 are confronted with the observations of 2017. The average behavior of the first-stage solutions is then measured against the events recorded during the year 2017 and compared to the average results obtained with different policies, including some designed by the field experts. According to the expert's knowledge, within our simulations, each intervention is assumed to last for two hours. Although this may vary significantly in practice, there is no available data on these actual durations.

5.1. Simulations: wait-and-serve policy versus recourse strategy

The first baseline policy we consider for assessing SS-VRP-R first-stage solutions is called *wait-and-serve* (w&s) policy. The w&s policy amounts to deciding vehicle actions without taking the stochastic knowledge into account. Under this basic policy, the vehicles will therefore always react to online events by assigning each new request to the closest available vehicle, so that there is no pre-computation nor optimization based on 2013-2016. Thus, simulations here simply consist of starting each 2017 day with all the vehicles at the depot and handling each online event as just described.

Whereas such policy is simple to simulate, in real life, it would already require a minimal communication between the vehicles, which must be able to share their positions and status. Better performances can be achieved by enabling a form of collaboration between the vehicles. However, this implies complicated online decisions belonging to the realm of online reoptimization approaches, such as Saint-Guillain et al. (2015).

On the contrary, when simulating based on a policy consisting of an optimized SS-VRP-R first-stage solution, the vehicles are not even required to share their position or status. Indeed, according to SS-VRP-R recourse strategies, with all the potential requests preassigned to the waiting locations, each vehicle can operate in a totally independent manner. In practice, a SS-VRP-R solution is thus easier to physically implement than the basic w&s policy. All the intelligence lies in the planned sequences of waiting locations.

5.2. Solution method

We use a local search (LS) algorithm similar to the one used for the experiments conducted in Saint-Guillain et al. (2017). Basically, the process starts with an initial solution being a set of empty plans $x_k = \big((0,1),(0,h)\big) \;\; \forall k \in K$ and iteratively tries to improve it using well-known VRP neighborhood operators: relocate, swap, inverted 2-opt, and cross-exchange (see Kindervater and Savelsbergh (1997); Taillard et al. (1997) for detailed description), whereas more specific operators are dedicated to waiting vertices: randomly add/remove a waiting vertex to/from a route, increase/decrease an amount of waiting time at a visited vertex or transfer to another. The variety of operators aims at applying local modifications to the plans by adding waiting points or modifying the waiting durations along the routes. It is coupled with a Simulated Annealing acceptance criterion that finally returns the best solution encountered, while providing adequate diversification based on the objective function $\mathcal{Q}$ described hereafter.

As discussed in Section 3.2, regarding Eq. (16) in practice, because recourse strategy $\mathcal{R}$ enforces the arrival (A_i) and departure (D_i) times at waiting vertices, finding a solution in which a request is always satisfied is not trivial. In such a context, to expected response time of an appeared request, we prefer the terms of *expected response time of an accepted request*. We then implement our LS algorithm while replacing objective function (2) by the more general lexicographic objective function $\mathcal{Q}_{\text{lex}}$, minimizing the expected number of rejected (nonsatisfiable) requests first and their expected response time second:

$$\mathcal{Q}_{\text{lex}}(x) = \Bigg(\sum_{r \in R} \big(p_r - \mathrm{P}\{r \text{ satisfiable}\}\big) \, , \\ \frac{1}{|R|} \sum_{r \in R} \sum_{\Delta=0}^{h-\Gamma_r} \Delta \cdot \frac{g_1^{\text{TD}}\big(r, x, \text{dep}^{\text{TD}}_{w(r),r}(\Gamma_r + \Delta)\big)}{\mathrm{P}\{r \text{ satisfiable}\}} \Bigg), \tag{27}$$

with $\mathrm{P}\{r \text{ satisfiable}\} = \sum_{t=t_r^{\min}}^{t_r^{\max}} g_1^{\text{TD}}(r, x, t)$. In fact, the expected number of rejected requests is the total expected number $\sum_{r \in R} p_r$ of appeared requests minus the expected number of accepted requests $\sum_{r \in R} \mathrm{P}\{r \text{ satisfiable}\}$. Solutions that are SS-VRP-R feasible thus have an objective value (z_1, z_2) with $z_1 = 0$. We see that objective function (27) is more convenient for an LS-based method, as it facilitates transitions between SS-VRP-R feasible solutions, which appear to be very sparse when using strategy $\mathcal{R}$.

5.3. Waiting locations

There is no restriction on the locations where the police units can be relocated. Hence, the set of possible waiting (re)locations W is defined as the set of all 150 vertices describing our urban area: $W = C = V$. However, the solution space, and therefore the computational performance, depends on the number of waiting vertices. That is the reason why three configurations, $|W| \in \{50, 100, 150\}$, are considered in the experiments ($W = V$ when $|W| = 150$). Let W^m denote the set W such that $|W| = m$. The sets of waiting vertices are chosen such that $\forall m < m' : \; W^m \subset W^{m'}$. In this manner, a solution computed using W^m is also a valid solution when using $W^{m'}$. The sets of waiting vertices are depicted in Figure 8.

5.4. Waiting time multiples and time granularity

Our planning horizon represents the 16-hour interval between 4 am and 8 pm. At its original scale, it has a resolution of one time unit per minute, that is, 960 time units.

Both the horizon granularity and the waiting time multiples can be adapted in order to simplify the computations. By considering a coarser-grained horizon, the complexity of computing the objective function $\mathcal{Q}_{\text{lex}}$ is decreased. With a scale of two, for instance, each time unit counts for two minutes, leading to a reduced horizon of 480 time units. Of course, such an approximation comes at the price

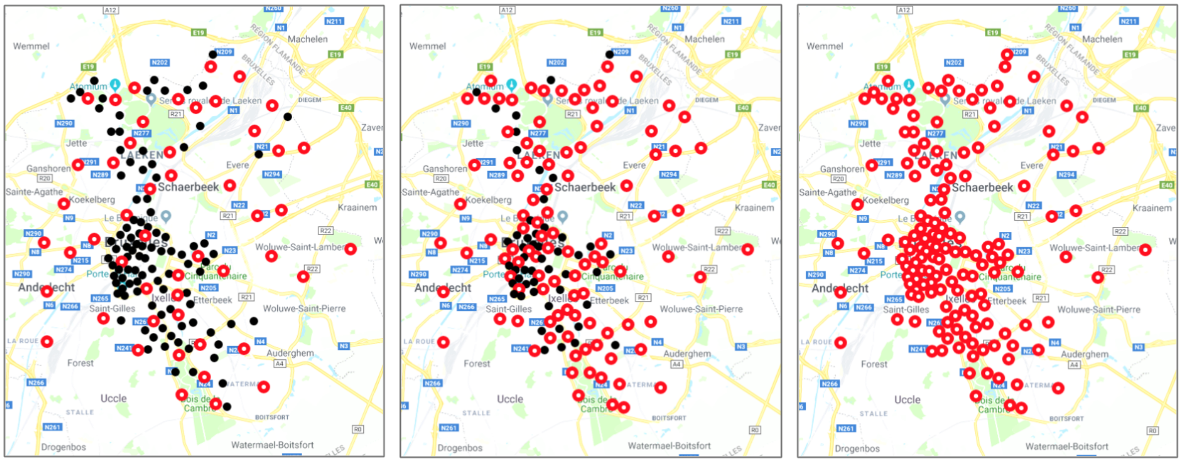


Figure 8: Configurations of waiting locations (red circles). Left: 50. Center: 100. Right: 150. Black: remaining vertices.

of a less accurate resulting expected value. Finally, by restricting the set of possible waiting times that can be assigned to vehicles at waiting locations, we greatly reduce the solution space. Furthermore, assigning waiting times that are multiples of 10 minutes, for instance, is most probably accurate enough in practice.

5.5. *Results*

We now evaluate the average behavior of SS-VRP-R first-stage solutions, optimized using (27) in light of the stochastic knowledge extracted from the observations collected from 2013 to 2016. The evaluation is performed through simulations, by executing the first-stage solutions (or the w&s) on all events recorded during 2017, with each day of observation standing for a specific scenario in the simulation. It is important to note that whereas (27) considers that a potential request may be rejected, during the simulations, a request is never rejected, as defined by the SS-VRP-R, and as it actually is in the real context of our police patrol management case study. Also, although our model assumes that no more than one request may arise at a given vertex for a given time slot, in practice, it could happen. Such multiple requests for the same vertex and time slot are simulated too, although in practice, the police central is more likely to send a different sort of police unit for reinforcement.

In all the experiments, each SS-VRP-R solution is optimised during one hour, on one core of a computational cluster composed of 2.6Ghz cores. This is always repeated ten times, and the average reported in the result tables. The algorithm and the probability computations are developed in C++11 and compiled with GDB using -O3 optimization flag. The Simulated Annealing parameters are: initial temperature 5, minimum temperature 10^{-6}, cooling factor 0.995.

The first experiments analyzed the impact of different experimental factors. After that, we compare the results when the constant or time-dependent travel times are considered during the optimization. Finally, we will get even closer to the reality of the operational context by comparing directly with the operational decisions made by police agents, based on their field experience.

We run all the simulations while taking time-dependency into account. We hence measure how the first-stage solutions behave under realistic time-dependent travel durations.

5.5.1. *Varying waiting time multiples, locations and time granularity.*

During these preliminary experiments, we only consider the subset 4 am to 10 am of the horizon as the problem instance. Each solution is optimized by using time-dependent travel durations in the LS algorithm described in Section 5.2. For each solution, averages (10 runs) are reported in Table 3.

Table 3: Average relative gains (in percentages) compared to the *w&s* policy, which passes the simulations (see Sec. 5.1) with average response times of 11.0 minutes (3 vehicles); 9.8 minutes (4 vehicles) ; 9.7 minutes (6 vehicles). Each cell reports an average over 10 solutions computed with the LS algorithm under the defined conditions: number of vehicles $|K|$, waiting time multiple wm, number of waiting locations $|W|$ and horizon granularity s. Average travel durations (non-time-dependent) are used for optimization. Solutions are evaluated by simulating under the time-dependent travel durations.

		$\lvert W\rvert = 50$			$\lvert W\rvert = 100$			$\lvert W\rvert = 150$		
$\lvert K\rvert$	*wm*	*s = 1*	*s = 2*	*s = 5*	*s = 1*	*s = 2*	*s = 5*	*s = 1*	*s = 2*	*s = 5*
3	10	15.1	21.1	3.5	16.6	18.5	3.1	17.5	17.3	1.3
	30	13.4	20.4	3.0	18.8	18.6	0.7	19.9	19.3	0.9
	60	10.1	17.8	3.0	11.6	15.9	1.6	13.9	17.6	0.1
4	10	25.1	23.4	2.3	27.9	23.1	2.3	27.3	23.5	2.0
	30	24.7	23.4	1.8	27.9	24.4	1.6	26.2	25.2	1.2
	60	21.0	21.4	1.8	23.1	20.7	1.1	24.8	21.3	0.2
6	10	38.4	33.7	1.2	38.8	30.7	3.8	38.5	32.5	2.9
	30	40.1	32.8	0.6	39.7	30.7	3.4	37.0	30.8	2.5
	60	35.4	31.1	0.7	35.0	29.4	2.2	36.1	30.0	1.5

The best results with respect to the number of vehicles are highlighted in Table 3. Provided a fixed number $|K|$ of vehicles, we observe that the gains vary greatly with the experimental conditions (waiting time multiple wm, number of waiting locations $|W|$ and horizon granularity s). With three (*resp.* six) vehicles, the average gains vary from 0.1% (*resp.* 0.6%) to 21.1% (*resp.* 40.1%).

Obviously, using scale 5 does not provide any significant gain, which sounds natural because under scale 5, the horizon is discretized in five-minute time units, which seems clearly not accurate enough when the average response time is less than ten minutes.

5.5.2. *Impact of time-dependent travel durations*

Now, we compute new solutions, which are obtained by taking time-dependency into account during the optimization process. In other words, we now optimize the lexicographic objective function (27), whereas the previous first-stage solutions were obtained while considering its non-time-dependent version. Table 4 reports the relative gains obtained under the same experimental contexts. In particular, the computation time remains set to one hour. We highlight the gains that reveal better than those obtained while optimizing under constant travel durations, that is, improving the gains of Table 3.

Table 4: Average relative gains (in percentages) compared to the *w&s* policy (3 vehicles: 11.0 minutes; 4 vehicles: 9.8; 6 vehicles: 9.7). Solutions are now optimized (and evaluated) while considering time-dependent travel duration matrices. We highlight the cells that are improved compared with Table 3.

		$\lvert W\rvert = 50$			$\lvert W\rvert = 100$			$\lvert W\rvert = 150$		
$\lvert K\rvert$	*wm*	*s = 1*	*s = 2*	*s = 5*	*s = 1*	*s = 2*	*s = 5*	*s = 1*	*s = 2*	*s = 5*
3	10	12.8	20.7	2.8	12.8	19.0	1.4	16.6	19.5	0.5
	30	13.3	19.5	2.3	12.4	20.3	1.2	16.1	19.1	0.4
	60	8.4	17.8	1.8	10.4	18.1	1.5	12.3	17.4	0.6
4	10	20.3	25.1	1.5	24.7	27.0	1.5	23.2	25.0	-0.2
	30	19.5	24.1	1.1	20.7	24.6	1.3	22.2	23.8	-0.4
	60	14.2	21.7	0.5	19.3	22.0	1.3	23.2	22.1	-0.4
6	10	32.1	33.5	0.4	33.8	33.0	2.8	36.4	31.3	1.5
	30	32.9	31.9	0.0	36.3	31.3	2.5	36.1	31.7	1.6
	60	29.7	30.0	-0.7	33.1	29.1	2.3	32.0	30.8	1.0

Empirical evidence highlights two important elements. First, exploiting more accurate, time-dependent travel durations does not permit to improve our results in general. This can be explained by the fact that the improvement in the travel duration accuracy, which is not necessarily very significant in general, does not compensate for the increased computational effort due to the time-dependent functions $f^{\text{TD}}_{v,v'}$ and $\text{dep}^{\text{TD}}_{v,v'}$. Indeed, the number of iterations performed by our LS optimizer is from 2 to 3 times less when dealing with time-dependent travel durations. That also explains the second noteworthy result of Table 4: improvements are only observed when optimizing under scales 2 and 5.

Figure 9 illustrates, for one of the runs of experiment $|K| = 6, wm = 10, s = 2$ in Table 4, how the lexicographic objective function (27) evolves during the local search process. During the experiment, all the intermediate best solutions found by LS were recorded and reinterpreted later on in order to perform costly simulations on it, as well as computing their non-time-dependent expected values. Reasonably good solutions are usually reached during the first 5 minutes. Looking at the expected z_2 values, we notice that the optimization with and without time dependency are rather close and probably tend to get even closer as the quality of the solutions improves. It always remains greater than the values measured through simulations.

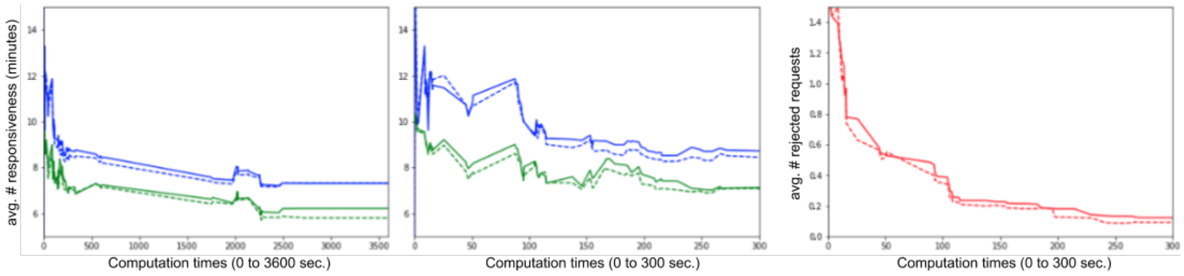


Figure 9: Evolution of the best first-stage solutions found during the local search process when optimizing under time-dependency assumptions. **Left:** temporal evolution of z_2 (blue, expected average responsiveness — in seconds), computed under either time-dependency assumptions (line) following Eq. (24) or not (dashed), *i.e.*, using Eq. (23); evolution of measured average responsiveness during simulations (green — assuming time-dependency), either on 2017 recorded events (line) or on 2013-2016 events (dashed). **Center:** zoom on the first 300 seconds optimization. **Right:** evolution of the expected number of rejected requests z_1 as computed in (27) under time-dependency (line) or not (dashed).

The simulations follow a slightly cleverer recourse strategy than that presented in Section 3: a vehicle is allowed to travel from one served request directly to another one that already appeared without doing a round-trip to its current waiting location. Furthermore, it may also travel to its next planned waiting location directly from a request. Such an improved strategy is trivial to apply using a simulation method. Nevertheless, computing its associated expected cost, while remaining pseudo-polynomial, is, however, computationally much more demanding: $\mathcal{O}(n^2h^3)$ instead of $\mathcal{O}(nh^2)$. The interested reader should refer to Saint-Guillain (2019) in which the corresponding closed-form expressions are derived. In practice, it has been observed that exploiting such a more elaborate recourse strategy, instead of our simplified version, leads to more accurate z_2 expected values. At the same time, it does not compensate for the significant computational effort.

Finally, we also note that the measured average responsiveness always appears better when simulating on 2013-2016 events than 2017, which sounds like a natural overfitting effect, as our model is somehow trained on the 2013-2016 dataset. The evolution of the z_1 objective value depicted on the right of Figure 9 confirms that both the time-dependent and non-time-dependent expected values remain quite similar in practice. Naturally, during the simulations, a request is never rejected, so that the measured average z_1 is always zero.

Provided a significantly higher, or unlimited, computation time would probably yield solutions of better quality when optimized under TD than those obtained with constant travel durations. However, for a given computation time, our experimental results show that considering TD during the optimization process has no impact. The major reasons are therefore that *a)* time-dependency increases the computational effort, thus decreasing the diversification within our LS approach, whereas *b)* the intensification is only slightly increased as the travel durations do not vary that much in general for short trips (as shown in Figure 4). As the SS-VRP-R aims at minimizing the expected average response time, it also indirectly minimizes the expected average length of the trips towards the requests, henceforth minimizing the expected variation of the travel durations, that is, the impact of time-dependency.

5.5.3. Specific and handmade first-stage solutions: exploiting field experience.

Whereas the SS-VRP-R optimization model allows significant gains compared to the baseline w&s strategy, it seems likely that an alternative, more elaborate, baseline strategy can be devised. Here we propose three additional baselines. They all come from the idea that a first-stage solution (a plan) can be constructed based on intuition or field experience.

The first additional baseline is called *geographical first-stage*, or x^{geo} for short. It is based on the intuition that using waiting locations that are both central and well spread over the operational map may constitute a good basis for a first-stage solution. The $|K|$ fixed waiting vertices are thus determined by a k-means clustering algorithm, with $k = |K|$, based on Euclidean distances between vertices in V. Contrary to w&s, each vehicle starts the operations by moving to a predefined waiting location. Requests are then exclusively served from there, meaning that the vehicle resumes its waiting at the predefined location after every request, until the end of the horizon comes. In other words, x^{geo} is the first-stage solution $x = \{x_1, \ldots, x_{|K|}\}$, where each plan is $x_k = \big((0,1), (v_1, A_1), (0, h)\big)$ with v_1 being one of the geographical cluster centroids and $A_1 = d_{0,v_1}$.

The second baseline is the *observational first-stage*, x^{obs} for short. Instead of computing the $|K|$ cluster centroids based on geographical considerations only, observations made from 2013 to 2017 are taken into account. More specifically, the clusters are now computed based on a set of vertices composed of all the recorded events, instead of the 150 vertices of V. Then, each centroid is associated with the closest location in V.

Finally, the third additional baseline directly comes from the stakeholder's experience. Based on the city map and their own field experience, several police officers were asked to compose their own plans. By designating their own strategic action plan, each police agent will actually determine a first-stage solution, denoted x^{field}, without knowing a word about the SS-VRP-R. Practically speaking, the map of Brussels is divided into four operational districts, called Intervention Police Departments (DPIs). Each DPI is under the responsibility of a police agent managing the patrols within the DPI. The number of allocated mobile units varies depending on the period. Over the four DPIs, the total number $|K|$ of mobile units typically ranges from 9 to 17 vehicles.

Results. The results are described in Table 5. To be consistent with the plans designed by the police agents, we consider both $|K| = 9$ and $|K| = 17$ vehicles. We also investigate a 25-vehicle case, corresponding to a hypothetical enhancement of the police vehicle fleet. The operational horizon is either 4 am to 10 am (as in previous experiments) or a broader horizon spanning from 4 am to 8 pm (16 hours). The discrete horizon is still composed of one-minute time units during the computation of (27). As a matter of fact, contrary to x^{geo}, the plan x^{field} designed by our police agents necessarily depends on whether the problem instance horizon ends at 10 am or 8 pm.

Not surprisingly, x^{obs} outperforms both x^{geo} and w&s. It also outperforms x^{field}, but only in the 9-vehicle case. Given 17 mobile units, the field experience is competitive (4 am – 10 am) or even significantly more responsive (4 am – 8 pm) than x^{obs}. Yet, the SS-VRP-R first-stage solutions still outperform all the other approaches. However, unlike the context involving a few vehicles, the number

Table 5: Average daily response times (in minutes) measured while simulating the overall 2017 recorded events. Gains are expressed as percentages relatively to w&s. SS-VRP-R solutions are optimized for 1 hour using parameters based on Table 3. Below are displayed the average gains of best SS-VRP-R solutions over x^{obs} and x^{field}.

	4 am – 10 am (6 hours)				4 am – 8 pm (16 hours)					
	9 patrols		17 patrols		9 patrols		17 patrols		25 patrols	
	respo.	*gain*	respo.	*gain*	respo.	*gain*	respo.	*gain*	respo.	*gain*
w&s	8.82	*n.a.*	8.82	*n.a.*	7.59	*n.a.*	7.59	*n.a.*	7.59	*n.a.*
x^{geo}	5.09	*42.3*	4.84	*45.1*	4.66	*38.5*	4.34	*42.8*	3.24	*57.3*
x^{obs}	4.25	*51.8*	3.59	*59.3*	3.91	*48.4*	3.29	*56.6*	2.71	*64.3*
x^{field}	4.63	*47.5*	3.68	*58.3*	4.22	*44.3*	3.19	*57.9*	*n.a.*	*n.a.*
s=1, wm=10, $\|W\|$=50	4.30	*51.3*	3.48	*60.6*	3.95	*47.9*	3.15	*58.4*	2.89	*61.9*
s=2, wm=10, $\|W\|$=50	4.19	*52.5*	3.64	*58.7*	3.83	*49.5*	3.15	*58.4*	3.06	*59.7*
s=1, wm=30, $\|W\|$=50	4.34	*50.8*	3.78	*57.1*	4.00	*47.3*	3.22	*57.6*	2.91	*61.6*
s=2, wm=30, $\|W\|$=50	4.34	*50.8*	3.53	*60.0*	4.09	*47.3*	3.33	*56.1*	2.92	*61.5*
s=1, wm=30, $\|W\|$=100	4.08	*53.8*	3.30	*62.6*	3.92	*48.3*	3.01	*60.3*	2.53	*66.6*
s=2, wm=30, $\|W\|$=100	4.21	*52.3*	3.32	*62.4*	3.91	*48.4*	2.92	*61.5*	2.57	*66.1*
s=1, wm=60, $\|W\|$=100	4.10	*53.5*	3.24	*63.2*	3.98	*47.5*	3.01	*60.3*	2.56	*66.3*
s=2, wm=60, $\|W\|$=100	4.18	*52.6*	3.28	*62.8*	3.88	*48.8*	2.93	*61.4*	2.58	*65.9*
s=1, wm=10, $\|W\|$=150	4.19	*52.6*	3.20	*63.7*	3.87	*49.0*	2.95	*61.1*	2.44	*67.8*
s=2, wm=10, $\|W\|$=150	4.24	*51.9*	3.35	*62.1*	3.76	*50.5*	2.95	*61.1*	2.56	*66.2*
s=1, wm=120, $\|W\|$=150	4.01	*54.5*	3.12	*64.6*	3.88	*48.8*	2.96	*61.0*	2.47	*67.4*
s=2, wm=120, $\|W\|$=150	4.21	*52.3*	3.23	*63.4*	3.86	*49.2*	2.95	*61.1*	2.56	*66.2*
	obs: +5.6%		*obs: +13.1%*		*obs: +3.8%*		*obs: +11.2%*		*obs: +10.0%*	
	field: +13.4%		*field: +15.2%*		*field: +10.9%*		*field: +8.5%*			

of waiting vertices is revealed to be more critical as the number of vehicles increases. Using $|W| = 50$ here produces solutions of worse quality, which are outperformed by x^{obs} and x^{field}. However, on the 4 am – 8 pm horizon, the best SS-VRP-R solutions obtained with $|W| = 100$ or $|W| = 150$ improve x^{obs} by 3.8% on average with 9 vehicles, 11.2% with 17 vehicles, and 10% with 25 vehicles. We see that the field experience x^{field} can also be improved, by 10.9% (9 vehicles) and 8.5% (17 vehicles). It seems remarkable that while the gain of using a SS-VRP-R optimization framework over x^{obs} increases with the size of the mobile fleet, it tends to decrease when confronted with real field experience.

Figure 10 illustrates first-stage solutions x^{geo}, x^{obs} and x^{field} (10 am to 8 pm), given either 9 or 17 vehicles, but not for 25, as it does not correspond to their operational experience for now.

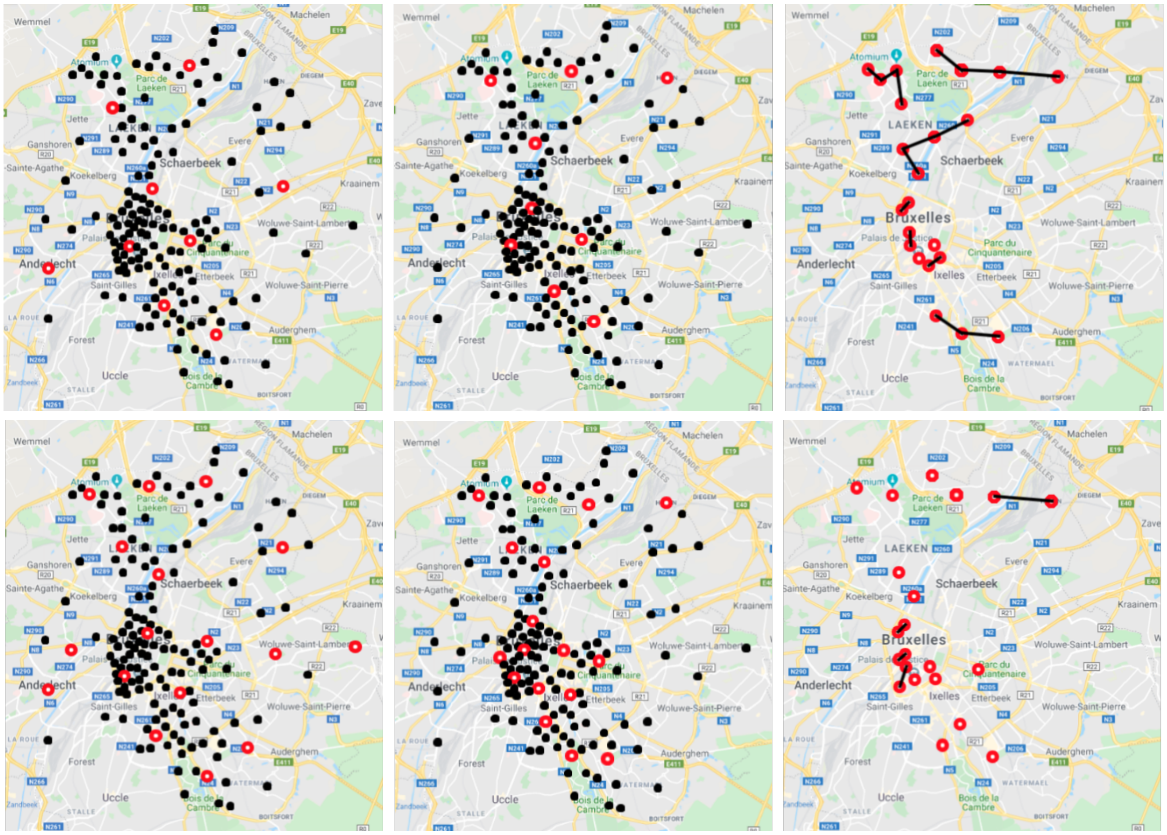


Figure 10: Waiting locations that compose the first-stage solutions x^{geo} (left), x^{obs} (center) and x^{field} (right) for 9 (top) or 17 (down) vehicles. In the case of x^{field}, some routes are sequences of multiple waiting locations, linked here by a bold line.

Finally, following the interest of the stakeholders, we also investigate the 25-vehicle case. Under that context, the average responsiveness of 2.44 minutes can be reached by our first-stage solutions, that is, a 16%—$(2.92 - 2.44)/2.92$—improvement by increasing the vehicle fleet from 17 to 25 units. Since increasing the fleet from 9 to 17 units improved the average responsiveness quite similarly in both the x^{field} case (24%—$(4.22 - 3.19)/4.22$) and SS-VRP-R (22%—$(3.76 - 2.92)/3.76$), we may suggest that an increase in the police actual fleet from 17 to 25 mobile units would similarly yield an improvement in x^{field} of approximately 16%. We hence see that even without the corresponding field experience (the stakeholders are not working with 25 vehicles for now), different operational contexts can be forecasted based on the SS-VRP-R results.

Figure 11 illustrates a selection of first-stage solutions involving 9, 17 or 25 units. Whereas our local search algorithm produces quite concise plans for 9 and 17 vehicles, the first-stage solution for 25 vehicles looks less structured, involving long trips between couples of waiting locations. The solution is hence likely to be far from optimal, meaning that the one-hour computation time should be increased when considering as many as 25 vehicles (although it already suffices to beat x^{obs}).

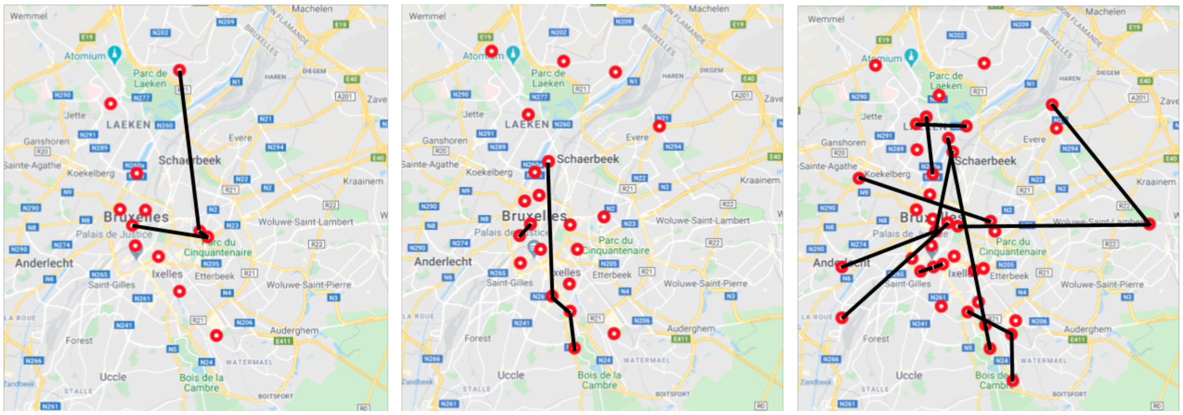


Figure 11: Optimized first-stage solutions with 9 (left), 17 (center) and 25 (right) mobile units, for the 4 am – 8 pm case study.

Variability of the results. The local search nature of our algorithm makes impossible any guarantee on the quality of the first stage solutions that it produces. Hence, we are also interested in its ability to always produce solutions of acceptable quality, in opposition to a method that would yield high-quality solutions only for some runs, and very poor-quality solutions for the remaining ones. Table 6 shows the averages and standard deviations, over the ten runs in each optimization context, of the lexicographical objective values (z_1, z_2) of Eq. (27). The more stable solutions are obtained when considering fewer ($|W|$=50) available waiting locations and a shorter waiting time multiple (wm=10). We also see remarkable increase in z_1 average values and variability when optimizing under scale 2, due to the diminution of the time unit accuracy from 1 to 2 minutes, which plays a significant role when computing the probability to reach a request under the 20-minute deadline.

Overall, this suggests two conclusions on variability. First, the accuracy in the manipulation of time (waiting multiple, time scale) is crucial to produce stable solutions. If the problem is too complex, then decreasing $|W|$ is probably better in terms of stability than working on a coarser-grained time horizon. Second, eventually in real life, and also during our simulations, a request is never rejected. In that context, z_1 indicates the number of requests that could not be satisfied *within a reasonable amount of time*. Whereas Table 5 suggests that $|W| = 150$ produces better average response times, Table 6 suggests that it also produces many more outliers, that is, requests serviced within, for instance, more than 20 minutes (which is unacceptable in our case).

Table 6: Averages and standard deviations for z_1 (expected number of rejected requests) and z_2 (expected average response time of accepted requests) objective values for optimized solutions from Table 5 (10 runs for each combination of s, wm, $|W|$), on the 4 am to 8 pm operational context. The (z_1, z_2) values are evaluated by the SS-VRP-R probabilistic model, and thus differ from those obtained from simulating on 2017 events (in particular, there is no rejected request when simulating).

	4 am – 8 pm (16 hours)											
	9 patrols				17 patrols				25 patrols			
	z_1	*dev*	z_2	*dev*	z_1	*dev*	z_2	*dev*	z_1	*dev*	z_2	*dev*
s=1, wm=10, $\|W\|$=50	0.016	*.0004*	4.99	*.09*	0.011	*.00004*	4.10	*.05*	0.000	*.00000*	3.72	*.01*
s=2, wm=10, $\|W\|$=50	2.269	*.0094*	6.02	*.03*	1.593	*.02228*	5.58	*.07*	0.179	*.01324*	7.30	*.16*
s=1, wm=30, $\|W\|$=50	0.001	*.0005*	5.07	*.15*	0.001	*.00012*	4.12	*.08*	0.011	*.00001*	3.72	*.02*
s=2, wm=30, $\|W\|$=50	0.416	*.0646*	9.42	*.08*	0.204	*.00800*	7.95	*.12*	1.451	*.03113*	5.21	*.07*
s=1, wm=30, $\|W\|$=100	0.009	*.0006*	4.71	*.11*	0.005	*.00003*	3.80	*.08*	0.005	*.00003*	3.27	*.04*
s=2, wm=30, $\|W\|$=100	1.882	*.0268*	6.22	*.10*	0.873	*.04033*	5.76	*.10*	0.527	*.02755*	5.35	*.08*
s=1, wm=60, $\|W\|$=100	0.010	*.0007*	4.83	*.13*	0.005	*.00004*	3.80	*.06*	0.005	*.00002*	3.29	*.05*
s=2, wm=60, $\|W\|$=100	1.901	*.0471*	6.23	*.13*	0.861	*.06264*	5.72	*.11*	0.526	*.02087*	5.41	*.10*
s=1, wm=10, $\|W\|$=150	0.009	*.0007*	4.76	*.12*	0.005	*.00010*	3.74	*.11*	0.005	*.00003*	3.16	*.04*
s=2, wm=10, $\|W\|$=150	1.702	*.0665*	6.27	*.12*	0.784	*.02952*	5.79	*.16*	0.543	*.04251*	5.41	*.13*
s=1, wm=120, $\|W\|$=150	0.010	*.0010*	4.81	*.18*	0.005	*.00011*	3.79	*.15*	0.005	*.00003*	3.21	*.05*
s=2, wm=120, $\|W\|$=150	1.685	*.0714*	6.39	*.17*	0.714	*.03228*	5.72	*.11*	0.423	*.02137*	5.39	*.12*

6. Conclusions

In this paper, we described the SS-VRP-R as a practical modeling framework for the real-world problem of the management of police units in Brussels. Relying on the intervention requests observed from 2013 to 2016, we showed how to minimize the expected average response time. Experiments are conducted while considering 2017's observations as a validation benchmark.

Coupled with a simple recourse strategy, the first-stage solutions obtained under the SS-VRP-R framework reduce the average response times by more than 60% on average compared to a basic wait-and-serve policy, in which the vehicles are never relocated. We also compared those solutions with nontrivial policies based on clustering methods. Compared to a policy that spreads the patrols based on geographical considerations only, SS-VRP-R solutions improve between 19% and 36%. That improvement is due to the preventive nature of the first-stage solutions, which are computed in light of stochastic knowledge. Compared to a policy taking spatial probabilities into account, depending on the operational context, the SS-VRP-R solutions improve from 5 to 11% over 2017. Only the SS-VRP-R model can take advantage of spatio-temporal probabilities to construct a patrolling route (instead of a single relocation point) for each vehicle. Finally, the SS-VRP-R optimized solutions still permit improvement in the average response time from 8 to 15% compared to patrolling routes designed by the field experts (from the police department) themselves.

In order to stick to the reality of the urban context, travel-duration dependency is introduced in the computational models. The obtained results are, however, quite surprising. In fact, experimentations under both constant and time-dependent travel durations show that exploiting time-dependency is not valuable in the context of the current case study. Namely, it appears that the contribution of time-dependency does not compensate for the increased computational cost, leading to first-stage solutions of noticeably less expected quality when computed in light of time-dependent travel duration matrices. Furthermore, while time-dependency is known to have a significant impact whenever time windows are involved, our SS-VRP-R case study does not involve any hard deadlines. Yet, time-dependency remains useful in order to perform more accurate simulations, even though the accuracy difference is also revealed here to not be very significant in a context where the problem at stake aims (indirectly) at minimizing the actual impact of time-dependency. Future contributions might determine to what extent considering TD provides enough improvements to deserve to increase the computation time. More generally, one could be interested in identifying the aspects of the problem that are mainly linked to the impact of TD. For instance, altering the problem by varying the degree of urgency of the requests, or varying the shape of the TD travel duration matrices (urban versus rural) or including deadlines in order to measure how this positively or negatively affects the impact of TD, and eventually suggest insights on what makes an online VRP problem particularly sensitive to TD.

Finally, we empirically measured that a nontrivial policy computed in light of the available stochastic knowledge tends to be outperformed by the field experience of police agents as the complexity of the operational context increases. Thanks to our SS-VRP-R optimization framework, we were also able to predict a possible improvement in police intervention responsiveness of approximately 15% while keeping their current operational scheme (based on field experience only) if their mobile fleet was increased by 50%. Combined with a stochastic optimization technology, such as the proposed SS-VRP-R modeling framework, simulations suggest an average improvement of 20 to 25%. As the final remark from the stakeholders' point of view, it is worth noting that the results we obtained are uniquely expressed in terms of the average response time. Yet, there exist in practice other important Key Performance Indicators to be pursued by the police department. In particular, maximizing the global police unit visibility, that is, the proximity to citizens, is currently considered a critical objective in Belgium. In fact, more than a simple facet of the experienced quality of service, it actually impacts the random events (the intervention requests appearance stochastic process) themselves. In fact, the patrol activity of the police units has a valuable function in preventing crime and providing reassurance to the public. Technically speaking, the actual problem at stake involves an endogenous uncertainty, which is assumed to be exogenous in this study and in the OR literature in general.

Acknowledgments

Computational resources have been provided by the Consortium des Équipements de Calcul Intensif (CÉCI), funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under Grant No. 2.5020.11, and by the Walloon Region (Belgium). We thank Christine Solnon for the useful remarks on the first stage of this study, as well as the anonymous reviewers for their valuable comments and suggestions. Finally, such a realistic case study would not have been possible without the enthusiastic collaboration of the Brussels police department. In particular, we thank Sarah Goesart, Pieter Pijpops, Jurgen Willems, Nancy Pustjens and Philippe Smetcoren.

References

Ahn, B.-H. and J.-Y. Shin (1991). Vehicle-routeing with time windows and time-varying congestion. *Journal of the Operational Research Society 42*(5), 393–400.

Bent, R. W. and P. Van Hentenryck (2007). Waiting and Relocation Strategies in Online Stochastic Vehicle Routing. In *IJCAI*, pp. 1816–1821.

Bertsimas, D. J. (1992). A vehicle routing problem with stochastic demand. *Operations Research 40*(3), 574–585.

Boland, N. L. and M. W. Savelsbergh (2019). Perspectives on integer programming for time-dependent models. *Top 27*(2), 147–173.

Braga, A. A., B. S. Turchan, A. V. Papachristos, and D. M. Hureau (2019). Hot spots policing and crime reduction: an update of an ongoing systematic review and meta-analysis. *Journal of Experimental Criminology 15*(3), 289–311.

Brotcorne, L., G. Laporte, and F. Semet (2003). Ambulance location and relocation models. *European Journal of Operational Research 147*(3), 451–463.

Bélanger, V., A. Ruiz, and P. Soriano (2019). Recent optimization models and trends in location, relocation, and dispatching of emergency medical vehicles. *European Journal of Operational Research 272*(1), 1 – 23.

Chainey, S. P. (2013). Examining the influence of cell size and bandwidth size on kernel density estimation crime hotspot maps for predicting spatial patterns of crime.

Dewinter, M., C. Vandeviver, T. Vander Beken, and F. Witlox (2020, Mar). Analysing the Police Patrol Routing Problem: A Review. *ISPRS Int. J. Geo-Inf. 9*(3), 157.

Eglese, R. W., W. Maden, and A. Slater (2006). A Road Timetable$^{\text{TM}}$ to aid vehicle routing and scheduling. *Computers & Operations Research 33*(12), 3508–3519.

Fleischmann, B., S. Gnutzmann, S. Elke, and E. Sandvoß (2004). Dynamic Vehicle Routing Based on Online Traffic Information. *Transportation Science 38*(4), 420–433.

Gendreau, M., G. Ghiani, and E. Guerriero (2015). Time-dependent routing problems: A review. *Computers & operations research 64*, 189–197.

Gendreau, M., O. Jabali, W. Rei, M. Gendreau, and O. Jabali (2016). Future Research Directions in Stochastic Vehicle Routing. *Transportation science 50*(4), 1163–1173.

Gendreau, M., G. Laporte, and R. Séguin (1996). Stochastic vehicle routing. *European Journal of Operational Research 88*(1), 3–12.

Gendreau, M., G. Laporte, and F. Semet (2001, Nov). A dynamic model and parallel tabu search heuristic for real-time ambulance relocation. *Parallel Comput. 27*(12), 1641–1653.

Ghiani, G., E. Manni, A. Quaranta, and C. Triki (2009, Jan). Anticipatory algorithms for same-day courier dispatching. *Transportation Research Part E: Logistics and Transportation Review 45*(1), 96–106.

Haghani, A. and S. Yang (2007). Real-Time Emergency Response Fleet Deployment: Concepts, Systems, Simulation & Case Studies. *SpringerLink*, 133–162.

Ichoua, S., M. Gendreau, and J.-Y. Potvin (2003). Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research 144*(2), 379–396.

Ichoua, S., M. Gendreau, and J.-Y. Potvin (2006). Exploiting knowledge about future demands for real-time vehicle dispatching. *Transportation Science 40*(2), 211–225.

Kindervater, G. A. P. and M. W. P. Savelsbergh (1997). Vehicle routing: handling edge exchanges. *Local search in combinatorial optimization*, 337–360.

Kok, A. L., E. W. Hans, and J. M. Schutten (2012). Vehicle routing under time-dependent travel times: The impact of congestion avoidance. *Computers & Operations Research 39*(5), 910–918.

Li, S. R. and B. B. Keskin (2014). Bi-criteria dynamic location-routing problem for patrol coverage. *Journal of the Operational Research Society 65*(11), 1711–1725.

Maxwell, M. S., M. Restrepo, S. G. Henderson, and H. Topaloglu (2010). Approximate dynamic programming for ambulance redeployment. *INFORMS Journal on Computing 22*(2), 266–281.

Mitrović-Minić, S. and G. Laporte (2004). Waiting strategies for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B: Methodological 38*(7), 635–655.

Naoum-Sawaya, J. and S. Elhedhli (2013). A stochastic optimization model for real-time ambulance redeployment. *Computers & Operations Research 40*(8), 1972–1978.

Pillac, V., M. Gendreau, C. Guéret, and A. L. Medaglia (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research 225*(1), 1–11.

Potvin, J.-Y., Y. Xu, and I. Benyahia (2006). Vehicle routing and scheduling with dynamic travel times. *Computers & Operations Research 33*(4), 1129–1137.

Psaraftis, H. N., M. Wen, and C. A. Kontovas (2016). Dynamic vehicle routing problems: Three decades and counting. *Networks 67*(1), 3–31.

Pureza, V. and G. Laporte (2008). Waiting and buffering strategies for the dynamic pickup and delivery problem with time windows. *INFOR: Information Systems and Operational Research 46*(3), 165–175.

Ritzinger, U., J. Puchinger, and R. F. Hartl (2016). A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research 54*(1), 215–231.

Saint-Guillain, M. (2019). *Models and algorithms for online stochastic vehicle routing problems*. PhD thesis, Université catholique de Louvain (UCLouvain, Belgium); INSA Lyon (France).

Saint-Guillain, M., Y. Deville, and C. Solnon (2015). A Multistage Stochastic Programming Approach to the Dynamic and Stochastic VRPTW. In *12th International Conference on Integration of AI and OR Techniques in Constraint Programming (CPAIOR 2015)*, pp. 357–374. Springer International Publishing.

Saint-Guillain, M., C. Solnon, and Y. Deville (2017). The Static and Stochastic Vehicle Routing Problem with both random Customers and Reveal Times. In *European Conference on the Applications of Evolutionary Computation*, Volume 2, pp. 110–127.

Schmid, V. (2012). Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming. *European Journal of Operational Research 219*(3), 611–621.

Sherman, L. W. and D. Weisburd (1995). General deterrent effects of police patrol in crime "hot spots": A randomized, controlled trial. *Justice quarterly 12*(4), 625–648.

Simpson, N. C. and P. G. Hancock (2009). Fifty years of operational research and emergency response. *Journal of the Operational Research Society 60*(sup1), S126–S139.

Taillard, É., P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin (1997). A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science 31*(2), 170–186.

Toth, P. and D. Vigo (2014). *Vehicle Routing: Problems, Methods, and Applications*, Volume 18. SIAM.

Verweij, B., S. Ahmed, A. J. Kleywegt, G. Nemhauser, and A. Shapiro (2003). The sample average approximation method applied to stochastic routing problems: a computational study. *Computational Optimization and Applications 24*(2-3), 289–333.