J Heuristics (2016) 22:539–566
DOI 10.1007/s10732-015-9281-6

A hybrid metaheuristic for the vehicle routing problem with stochastic demand and duration constraints

Jorge E. Mendoza · Louis-Martin Rousseau · Juan G. Villegas

Received: 22 November 2013 / Revised: 18 June 2014 / Accepted: 6 January 2015 /
Published online: 31 January 2015

Abstract The vehicle routing problem with stochastic demands (VRPSD) consists in designing optimal routes to serve a set of customers with random demands following known probability distributions. Because of demand uncertainty, a vehicle may arrive at a customer without enough capacity to satisfy its demand and may need to apply a recourse to recover the route's feasibility. Although travel times are assumed to be deterministic, because of eventual recourses the total duration of a route is a random variable. We present two strategies to deal with route-duration constraints in the VRPSD. In the first, the duration constraints are handled as chance constraints, meaning that for each route, the probability of exceeding the maximum duration must be lower than a given threshold. In the second, violations to the duration constraint are penalized in the objective function. To solve the resulting problem, we propose a greedy randomized adaptive search procedure (GRASP) enhanced with heuristic

J. E. Mendoza
LUNAM Université, Université Catholique de l'Ouest, LARIS (EA 7315),
3 Place André Leroy, 49008 Angers, France

J. E. Mendoza (✉)
Université François-Rabelais de Tours, CNRS, LI EA 6300, OC ERL CNRS 6305,
64 Avenue Jean Portalis, 37300 Tours, France
e-mail: jorge.mendoza@univ-tours.fr

L.-M. Rousseau
CIRRELT, École Polytechnique de Montréal, Succ. Centre-ville,
C.P. 6079, Montreal, QC H3C 3A7, Canada
e-mail: louis-martin.rousseau@cirrelt.net

J. G. Villegas
Departamento de Ingeniería Industrial, Facultad de Ingeniería, Universidad de Antioquia,
Calle 70 No. 52 - 21, 050010 Medellín, Colombia
e-mail: juan.villegas@udea.edu.co

concentration (HC). The GRASP component uses a set of randomized route-first, cluster-second heuristics to generate starting solutions and a variable-neighborhood descent procedure for the local search phase. The HC component assembles the final solution from the set of all routes found in the local optima reached by the GRASP. For each strategy, we discuss extensive computational experiments that analyze the impact of route-duration constraints on the VRPSD. In addition, we report state-of-the-art solutions for a established set of benchmarks for the classical VRPSD.

Keywords Distance-constrained vehicle routing problem · Vehicle routing problem with stochastic demands · Two-stage stochastic programming · GRASP · Heuristic concentration

1 Introduction

In the vehicle routing problem with stochastic demands (VRPSD) a set of geographically spread customers demand (or supply) a product that must be delivered (or collected) using a fleet of limited-capacity vehicles located at a central depot. The particular characteristic of the problem is that the exact quantities demanded (supplied) by each customer are only known upon the vehicle's arrival at the customer location (i.e., they are stochastic). It is assumed, however, that each customer's demand follows a known probability distribution. The main impact of stochastic demands is that they introduce uncertainty into the feasibility of the routes; depending on the demand realizations (i.e., the actual values), a vehicle may arrive at a customer without enough capacity to satisfy its demand.

To deal with uncertain demands in the VRPSD, researchers have explored models based on various solution frameworks including chance-constraint programming, stochastic programming with recourse, dynamic programming, Markov decision models, and the multi-scenario approach. Each of these frameworks takes into accounts factors such as instance size, assumptions about available technology (e.g., real-time communication between vehicles and decision-makers), and assumptions about managerial policies (e.g., whether or not routes can be modified during their execution). For a complete discussion of the characteristics of each framework the reader is referred to Secomandi and Margot (2009) and Pillac et al. (2013a).

The most widely studied models in the literature are those based on two-stage stochastic programming (Cordeau et al. 2006). As the name suggests, in this framework the problem is solved in two phases. In the first phase a set of *a priori routes* is planned, and in the second phase the routes are executed. If there is a capacity constraint violation, or *route failure*, a corrective action, known as *recourse*, is taken to recover feasibility. In general, the recourse actions generate an extra cost known only after the second phase. Thus, the objective is to design during the first phase a set of routes that minimizes the sum of the cost of the a priori routes and the expected cost of the recourse actions.

The most traditional recourse action, known as *detour-to-depot*, involves traveling back to the depot to restore the vehicle capacity, returning to the customer to complete the service, and then continuing the route as initially planned (Savelsbergh and Goetschalckx 1995). However, more sophisticated approaches have recently been reported in the literature. These include performing preemptive trips to the depot in an attempt to avoid route failures (Yang et al. 2000; Bianchi et al. 2004), assigning each vehicle a partner to provide back-up in the event of a failure (Ak and Erera 2007), and reassigning the customers of a failing route to the planned route of a vehicle with spare capacity (Novoa et al. 2006). All recourse actions add travel time to the planned routes. Since the exact number of recourses and the extra time they add to each route is not known when the routes are planned, the total duration of a route is itself a random variable. As pointed out by Erera et al. (2010), this may lead to a problem in practice because the routes may be subject to an additional feasibility criterion: duration constraints (DCs).

DCs prevent the duration of a route from exceeding an upper bound. Therefore, they can model a number of industry practices such as shift duration limits and depot opening hours (Erera et al. 2010). Despite their practical relevance, DCs have been studied only rarely in the context of the VRPSD. To the best of our knowledge, the body of work in this domain is limited to about ten references, most of them focusing on approaches based on two-stage stochastic programming. For the sake of brevity, in the remainder of this section we focus on these approaches; however, we refer the reader to the excellent papers by Bent and Hentenryck (2004); Bent and Van Hentenryck (2007) and Goodson et al. (2013, in press) for research based on other frameworks.

Yang et al. (2000) is probably the first reference to DCs in the VRPSD literature. The authors handle these constraints by imposing a limit on the expected duration of the a priori routes. Mendoza et al. (2010, 2011) applied the same strategy in the context of the multi-compartment VRPSD (MC-VRPSD), a problem in which each customer demands several incompatible products that are transported in different vehicle compartments. The main advantage of this *constrained expected duration* approach is its computational convenience. Indeed, since the expected duration of a route is usually computed as part of the objective function, the DC feasibility check requires no additional effort. On the other hand, although this strategy may be adequate for practical situations where DCs are rather soft constraints, it does not provide decision-makers with an explicit mechanism to express their preferences about violations of these constraints.

Tan et al. (2007) and Sörensen and Sevaux (2009) propose an alternative approach, based on penalizing violations of the DCs in the objective function. Tan et al. (2007) use the penalties as part of a cost function called drivers' remuneration that they optimize, along with the total traveled distance and the number of vehicles, using a multi-objective optimization approach. Sörensen and Sevaux (2009) include the penalties directly in the total-duration objective function and use an established mono-objective approach (Sörensen and Sevaux 2006) to solve the problem. In both cases, the authors use Monte Carlo simulation to generate multiple scenarios of the demand realizations that are used to compute the total expected duration of the routes and the penalties for DC violations. Mendoza et al. (2009) propose a different strategy to address DCs in the context of a bi-objective MC-VRPSD: they minimize simultaneously the total expected cost of a set of routes and its coefficient of variation. In their approach, DCs are imposed on planned routes as *chance constraints* ensuring that the probability of completing a route in less than its maximum duration is greater than a given threshold.

To perform the feasibility check of the chance constraints, the authors use Monte Carlo simulation.

From the conceptual point of view, both the penalty and chance-constraint approaches overcome the shortcomings of the constrained expected-duration approach. However, the implementations based on Monte Carlo simulation may be unnecessarily expensive from a computational point of view because one may need to generate a large number of scenarios to achieve statistical significance. Haugland et al. (2007) and Erera et al. (2010) propose approaches for applications in which the DCs are hard constraints. In Haugland et al. (2007) the authors solve a VRPSD with DCs as part of the evaluation of the solution to a districting problem. To check the DC feasibility the authors use an upper bound on the total duration of a route. Erera et al. (2010) propose an algorithm to compute the maximum duration of a route for any realization of the customer demands. They use its result as an input to check the DCs.

In this paper we revisit the penalty and chance-constraint strategies to deal with DCs in the VRPSD. In contrast to previous approaches, we do not use Monte Carlo simulation. We instead explicitly build the probability distribution of the duration of a route. We develop a hybrid metaheuristic that, with minor modifications, is able to solve both versions of the problem. Our method is a greedy randomized adaptive search procedure (GRASP) with heuristic concentration (HC). The GRASP component uses a set of randomized route-first, cluster-second heuristics to generate starting solutions and a variable-neighborhood descent (VND) procedure for the local search phase. The HC component assembles the final solution from the set of all routes found in the local optima reached by the GRASP. We present and discuss the results obtained by our method for both classical VRPSD instances and instances adapted to fit the definition of the VRPSD with DCs. For the latter case, we analyze the advantages and disadvantages of using the penalty and chance-constraint strategies rather than a more classical approach: the constrained expected duration.

The remainder of the paper is organized as follows. Section 2 defines the problem, introduces the relevant notation, and presents our two problem formulations. Section 3 presents our hybrid metaheuristic, and Sect. 4 discusses the computational experiments. Section 5 concludes the paper and outlines future research.

2 Problem formulation

The vehicle routing problem with stochastic demands and DCs (VRPSD-DC) can be defined on a complete and undirected graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{0, \ldots, n\}$ is the vertex set and $\mathcal{E} = \{(v, u) : v, u \in \mathcal{V}, v \neq u\}$ is the edge set. Vertices $v = 1, \ldots, n$ represent the customers and vertex $v = 0$ represents the depot. A weight t_e is associated with edge $e = (v, u) = (u, v) \in \mathcal{E}$, and it represents the travel time between vertices v and u. Each customer v has a random demand $\tilde{\xi}_v$ for a given product. We assume that each customer's demand follows an independent and known probability distribution. The customers are served using an unlimited fleet of homogeneous vehicles with capacity Q and maximum travel time T located at the depot. We assume that the demand realizations $\boldsymbol{\xi}$ are nonnegative and less than the capacity of the vehicle. We also assume that each customer's demand realization is not known until the vehicle arrives at the customer location.

A planned route r is a sequence of vertices $r = (0, v_1, \ldots, v_i, \ldots, v_{n_r}, 0)$, where $v_i \in \mathcal{V} \setminus \{0\}$ and n_r is the number of customers served by the route. Depending on the context, we may refer to route r as an ordered set of edges $r = \{(0, v_1), \ldots, (v_{i-1}, v_i), \ldots, (v_{n_r}, 0)\}$. During the execution of a planned route, if a route failure occurs, that is, the capacity of the vehicle is exceeded, the detour-to-depot recourse is applied to recover the feasibility of the route. We denote by $Pr(v_i)$ the probability of a route failure occurring while serving customer $v_i \in r$. This *failure probability* is given by

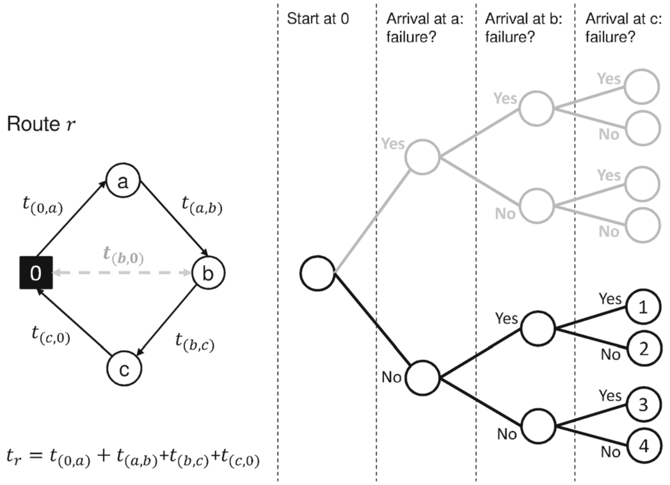
$$Pr(v_i) = \sum_{f=1}^{i} Pr\left(\sum_{j=0}^{i-1} \tilde{\xi}_{v_j} \leq f \cdot Q < \sum_{j=0}^{i} \tilde{\xi}_{v_j}\right) \tag{1}$$

where the probability term represents the probability of the f^{th} failure occurring while serving customer v_i. For the details of the derivation of (1) see Teodorović and Pavković (1992). Note that the number and location of route failures are not known when the routes are planned. Therefore, although all travel times are (assumed to be) deterministic, the total duration of a route $\tilde{T}_r$ is a random variable which realization is only known when the route is completed. On the other hand, the probability distribution of $\tilde{T}_r$ may be computed when the route is planned.

Note that since all the demand realizations are less than the capacity of the vehicle, a route may fail at the most once at each customer but the first one. In other words, the maximum number of failures in a route is $n_r - 1$, and the first failure cannot occur while serving the first customer. Consequently, $\tilde{T}_r$ follows a discrete distribution with 2^{n_r-1} possible outcomes; we refer to each of these outcomes as a *duration profile*. Figure 1 illustrates this concept. Let $\mathcal{P}(r)$ be the set of all possible duration profiles for route r. Let $Pr(p)$ be the probability of observing duration profile $p \in \mathcal{P}(r)$, and let $T_r(p)$ be the total duration of route r if profile p is observed. The leftmost part of the figure depicts a planned route $r = \{(0, a), (a, b), (b, c), (c, 0)\}$ with a total planned duration $t_r = \sum_{(u,v)\in r} t_{(u,v)} = t_{(0,a)} + t_{(a,b)} + t_{(b,c)} + t_{(c,0)}$. The rightmost part of the figure shows a tree in which each leaf node represents one possible duration profile p for route r. For instance, leaf node $p = 1$ represents the duration profile of the route if failures occur while serving customers b and c. In such case, $T_r(p) = t_r + 2t_{(b,0)} + 2t_{(c,0)}$ and $Pr(p) = (1 - Pr(a)) \times Pr(b) \times Pr(c)$, where $Pr(a)$, $Pr(b)$, and $Pr(c)$ denote the probability of having a failure while servicing customers a, b, and c, respectively (see Eq. (1)). It is worth mentioning that since failures cannot occur while servicing customer a, the upper branches of the three (in grey) represent impossible duration outcomes.

2.1 Chance-constraint formulation

In our first formulation we extend the classical two-stage stochastic programming formulation for the VRPSD to include the DCs as chance constraints. The resulting problem involves finding a set $\mathcal{R}$ of planned routes that minimizes



p	$T_r(p)$	$Pr(p)$
1	$t_r + 2t_{(b,0)} + 2t_{(c,0)}$	$(1 - Pr(a)) \times (Pr(b)) \times (Pr(c))$
2	$t_r + 2t_{(b,0)}$	$(1 - Pr(a)) \times (Pr(b)) \times (1 - Pr(c))$
3	$t_r + 2t_{(c,0)}$	$(1 - Pr(a)) \times (1 - Pr(b)) \times (Pr(c))$
4	t_r	$(1 - Pr(a)) \times (1 - Pr(b)) \times (1 - Pr(c))$

Fig. 1 Duration profiles for a given route and their attributes

$$E\left[C_1(\mathcal{R})\right] = \sum_{r \in \mathcal{R}} E\left[\tilde{T}_r\right] \tag{2}$$

s.t.

$$Pr\left(\tilde{T}_r > T\right) \leq \beta \ \ \forall\, r \in \mathcal{R} \tag{3}$$

$$\sum_{i \in r} E\left[\tilde{\xi}_{v_i}\right] \leq Q \qquad \forall\, r \in \mathcal{R} \tag{4}$$

$$r \bigcap r' = \{0\} \qquad \forall\, r, r' \in \mathcal{R}, r \neq r' \tag{5}$$

$$\bigcup_{r \in \mathcal{R}} = \mathcal{V} \tag{6}$$

The objective (2) minimizes the total expected duration of the set of routes $\mathcal{R}$. Constraint (3) ensures that the probability that a route violates the duration limit is lower than a given threshold β. Using the duration profiles of route r as an input, the first term in (3) can be computed as

$$Pr\left(\tilde{T}_r > T\right) = \sum_{p \in \mathcal{P}(r) | T_r(p) > T} Pr(p). \tag{7}$$

Constraint (4) ensures that each planned route is designed so that the total expected load does not exceed the capacity of the vehicle while constraints (5) and (6) guarantee that each customer is included in one and only one planned route.

2.2 Penalty formulation

In our second formulation we follow a completely different approach. To account for the DCs, we extend the classical VRPSD objective to include the expected cost of overtime, i.e., the time that each route travels above the limit T. In this formulation the problem involves finding a set of planned routes $\mathcal{R}$ verifying constraints (4)–(6) and minimizing

$$E\left[C_2(\mathcal{R})\right] = \sum_{r \in \mathcal{R}} E\left[\tilde{T}_r\right] + E\left[\phi\left(\tilde{O}_r\right)\right] \tag{8}$$

where

$$E\left[\phi\left(\tilde{O}_r\right)\right] = \sum_{p \in \mathcal{P}(r) | T_r(p) > T} \phi\left(T_r(p) - T\right) \times Pr(p) \tag{9}$$

is the expected overtime cost. Function $\phi(\cdot)$ models the decision maker's aversion toward overtime. It can take any form depending on the context. For instance, quadratic functions can model situations in which even small violations of the DCs should be discouraged, while piecewise linear functions can be useful when small violations are acceptable but violations beyond a given threshold should be avoided.

In the remainder of the paper, we refer to our chance-constraint and penalty formulations as `CC` and `PF`.

3 GRASP with HC approach

To solve our two formulations for the VRPSD-DC, namely `CC` and `PF`, we developed a GRASP with HC. Algorithm 1 describes the proposed approach. At the kth GRASP iteration (lines 3–14) we greedily construct a starting solution (lines 5–6) and then try to improve it using a local search procedure (line 7). To construct the starting solution, we select a randomized TSP heuristic h from a predefined set $\mathcal{H}$ and use it to build a giant TSP tour tsp^k visiting all the customers (line 5). We then use an adaptation of the s-split procedure for the VRPSD (Mendoza et al. 2011) to optimally partition tsp^k into a set of feasible routes that forms a starting solution s^k (line 6). We next launch a VND procedure from the starting solution s^k (line 7). At the end of iteration k, we update the best solution s^* (line 8) and add the routes of the local optimum (i.e., s^k) to a set Ω (lines 9–11). After K iterations the GRASP stops and we carry out the HC. In this phase, our method solves a set partitioning problem (SPP) over the set of routes Ω (line 15). Note that the specific implementations of split(·) and vnd(·) vary depending on the formulation (i.e., CC or PF) being solved, whereas the implementations of tsp(·), update(·), and setPartitioning(·) are unchanged. In the remainder of this section we present a detailed description of the main algorithmic components of our method.

Algorithm 1 GRASP + HC: General structure

function GRASPHC($\mathcal{H}$,K,mode) ▷ mode={CC, PF}
    $\Omega \leftarrow \emptyset, k \leftarrow 1$
    while $k \leq K$ **do**
        for $h \in \mathcal{H}$ **do**
            $tps^k \leftarrow$tsp(h)
            $s^k \leftarrow$split(tsp^k, mode)
            $s^k \leftarrow$vnd(s^k, mode)
            $s^* \leftarrow$update(s^k, s^*)
            for $r \in s^k$ **do**
                $\Omega \leftarrow \Omega \cup r$
            end for
            $k \leftarrow k + 1$
        end for
    end while
    $\mathcal{R} \leftarrow$setPartitioning(Ω, s^*)
    return $\mathcal{R}$
end function

3.1 Greedy randomized construction

Mendoza and Villegas (2013) observed that using multiple sampling procedures instead of just one, as is traditional, may improve the performance of vehicle routing heuristics that are based on drawing samples from the solution space. Given this observation, we decided to embed in our method four versions of a randomized route-first, cluster-second heuristic.

3.1.1 Routing phase

For the routing phase, our approach uses randomized versions of four TSP constructive heuristics: randomized nearest neighbor (RNN), randomized nearest insertion (RNI), randomized best insertion (RBI), and randomized farthest insertion (RFI). Although the strategies we used to generate the randomized versions of the four heuristics are directly borrowed from Mendoza and Villegas (2013), for the sake of completeness we briefly describe them here.

Let tsp^k be an ordered set representing the TSP tour being built at iteration k, $\mathcal{W}$ the set of vertices visited by tsp^k, and $\mathcal{Z} = \mathcal{V} \setminus \mathcal{W}$ an ordered set of non-routed vertices. For the sake of simplicity, we assume that the sets $\mathcal{W}$ and $\mathcal{Z}$ are updated every time a customer is added to tsp^k. Let us also define three metrics for every customer $v \in \mathcal{Z}$, namely, $t_{min}(v) = \min\{t_{(v,u)} | u \in \mathcal{W}\}$, $t_{max}(v) = \max\{t_{(v,u)} | u \in \mathcal{W}\}$, and

$\Delta_{min}(v) = \min\{t_{(u,v)} + t_{(v,w)} - t_{(u,w)} | (u, w) \in tsp^k\}$. Finally, let l be a random integer in $\{1, \ldots, \min\{L_h, |\mathcal{Z}|\}\}$, where parameter L_h denotes the *randomization factor* of each heuristic. The four sampling heuristics are as follows:

- **RNN:** Set $tsp^k = (0)$ and $u = 0$. At each iteration identify the vertex v that is the lth nearest vertex to u, append v to tsp^k, and set $u = v$. Stop when $|\mathcal{Z}| = 0$ and append 0 to tsp^k to complete the tour.
- **RNI:** Initialize tsp^k as a tour starting at the depot and performing a round trip to a randomly selected customer (henceforth we will refer to this procedure simply as initialize tsp^k). At each iteration sort $\mathcal{Z}$ in non-decreasing order of $t_{min}(v)$. Insert $v = \mathcal{Z}_l$ (i.e., the lth element in the ordered set $\mathcal{Z}$) in the best possible position in the tour tsp^k (i.e., the position generating the smallest increment in the travel time of the tour). Stop when $|\mathcal{Z}| = 0$.
- **RFI:** Initialize tsp^k. At each iteration sort $\mathcal{Z}$ in nondecreasing order of $t_{max}(v)$ and insert $v = \mathcal{Z}_l$ in the best possible position in the tour tsp^k. Stop when $|\mathcal{Z}| = 0$.
- **RBI:** Initialize tsp^k. At each iteration sort $\mathcal{Z}$ in nondecreasing order of $\Delta_{min}(v)$ and insert $v = \mathcal{Z}_l$ in the best possible position in the tour tsp^k. Stop when $|\mathcal{Z}| = 0$.

3.1.2 Clustering phase

To extract a feasible solution s^k from tsp^k, our approach uses an adaptation of the s-split procedure for the VRPSD proposed in Mendoza et al. (2011). S-split builds a directed and acyclic graph $G' = (\mathcal{V}', \mathcal{A})$ composed of the ordered vertex set $\mathcal{V}' = (v_0, v_1, \ldots, v_i, \ldots, v_n)$ and the arc set $\mathcal{A}$. Vertex $v_0 = 0$ is an auxiliary vertex, while vertices $v_1, \ldots, v_n \in tsp^k \setminus \{0\}$. The vertices in $\mathcal{V}'$ are arranged in the order in which they appear in tsp^k. Arc $(v_i, v_{i+n_r}) \in \mathcal{A}$ represents a feasible route $r_{(v_i, v_{i+n_r})}$ with evaluation $\mathrm{e}_{r_{(v_i, v_{i+n_r})}}$ starting and ending at the depot and traversing the sequence of customers from v_{i+1} to v_{i+n_r}. The evaluation of route $r_{(v_i, v_{i+n_r})}$ is the contribution of the route to objective function (2) or (8) depending on the formulation being solved. To retrieve s^t, the procedure finds the set of arcs (i.e., routes) along the shortest path connecting 0 and v_n in G'. Figure 2 illustrates the splitting procedure.

It is worth noting that since G' is directed and acyclic, building the graph and finding the shortest path can be done simultaneously. To accomplish this goal, our method

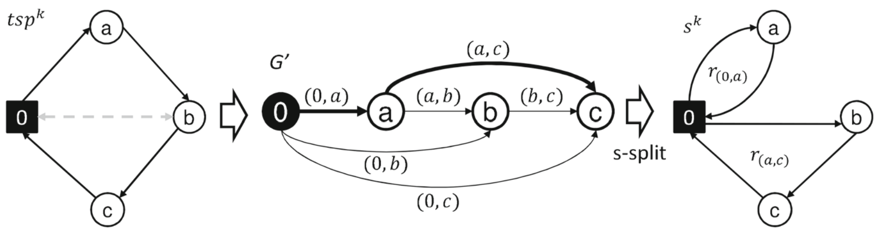


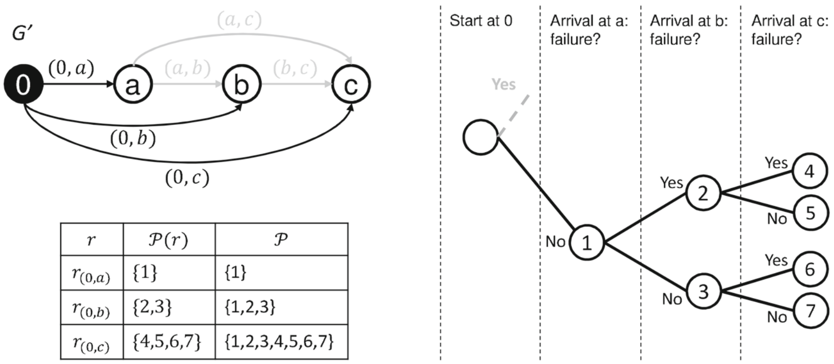
Fig. 2 Splitting procedure: graphical example

uses an algorithm based on the splitting procedure proposed by Prins (2004) for the classical capacitated VRP. Algorithm 2 outlines the procedure. After initializing the shortest path labels (lines 2–5) we enter the outer loop (lines 6–22). Each pass through the outer loop sets the tail of an arc. Then we use the inner loop (lines 9–21) to build all the arcs sharing the same tail node. At the start of each inner-loop iteration, we build a new arc by simply extending the last generated arc. In the next step, we evaluate the weight of the arc and whether or not it should be added to the auxiliary graph. These tasks are accomplished by evaluating the route corresponding to the arc in terms of both its contribution to the objective function e_r and its feasibility f_r (line 12). If the arc is added to the graph, we update the shortest path and predecessor labels (lines 14–17) and move to the next inner-loop iteration; otherwise, we exit the loop. After completing the outer loop we retrieve the solution using the incoming TSP tour and the vector of predecessor labels (for an algorithm that retrieves the solution we refer the reader to Prins 2004).

Algorithm 2 Splitting procedure: Pseudocode

```
1:  function SPLIT(tsp,mode)
2:      c_0 ← 0                                          ▷ c: shortest path labels
3:      for i = 1 to n do
4:          c_i ← ∞
5:      end for
6:      for i = 1 to n do
7:          j ← i + 1
8:          P ← ∅                                        ▷ P : duration profile tree
9:          repeat
10:             r ← r_(i,j)
11:             continue ← false
12:             ⟨e_r, f_r, P⟩ ← evaluate(r, P, mode)     ▷ e_r: evaluation, f_r: feasibility
13:             if f_r = true then
14:                 if c_{i−1} + e_r ≤ c_j then
15:                     c_j ← c_{i−1} + e_r
16:                     p_j ← i − 1                      ▷ p: predecessor labels
17:                 end if
18:                 continue ← true
19:                 j ← j + 1
20:             end if
21:         until j > n or ¬continue
22:     end for
23:     s ← retrieveSolution(tsp, p)
24:     return s
25: end function
```

The route evaluation procedure (line 12) differs slightly depending on the formulation being solved. In both cases, however, the evaluation starts by checking the expected load constraint (which can be checked in constant time). If the route fails the expected load check, the evaluation is truncated to avoid unnecessary computation. In the next step, we compute the duration profiles of the route. To accomplish this task efficiently we maintain a *profile tree* $\mathcal{P}$, storing the duration profiles of all the routes previously evaluated during the current outer-loop iteration. Since the route r



r	$\mathcal{P}(r)$	$\mathcal{P}$
$r_{(0,a)}$	{1}	{1}
$r_{(0,b)}$	{2,3}	{1,2,3}
$r_{(0,c)}$	{4,5,6,7}	{1,2,3,4,5,6,7}

Fig. 3 Route evaluation procedure: Building route-duration profiles

evaluated at a given point is just a one-customer extension of the route evaluated in the previous iteration, its duration profiles $\mathcal{P}(r)$ can be built by adding a new level to $\mathcal{P}$ instead of building a whole new tree for the route. Figure 3 illustrates this operation. As they are built, the duration profiles are used to compute the contribution of the route to the objective function e_r, i.e., Eq. (8) for `PF` and Eq. (2) for `CC`. In the latter case, the duration profiles are also used to check the route's feasibility f_r in terms of the DC (3).

Note that as a result of the expected load constraint the number of customers served by the largest route generated during an inner-loop iteration can be approximated by

$$\bar{n}_r = \left\lceil \frac{nQ}{\sum_{i \in \mathcal{V} \setminus \{0\}} E\left[\xi_i\right]} \right\rceil \tag{10}$$

Note also that with the tree-extension procedure the total number of operations needed to compute the duration profiles of all the routes generated during a single outer-loop iteration is $O(2^{\bar{n}_r})$. Since there are n iterations in the outer loop, our split algorithm runs in $O(n \cdot 2^{\bar{n}_r})$. Although the execution time of the procedure grows exponentially with $\bar{n}_r$, in practical settings the average number of customers per route tends to be rather low (Tricoire 2013).

3.2 Local search procedure

To improve the solutions generated by the constructive phase we use a VND (Hansen et al. 2008) with two neighborhoods: re-locate and 2-opt. From the starting solution our VND runs a pure-descent local search (LS) using re-locate. When re-locate reaches a local optimum, our VND switches to 2-opt and launches from there a new pure-descent LS. Every time an improving solution is found, our VND restarts from that solution using re-locate. Iterations are carried on until 2-opt cannot longer improve the solution. For both neighborhoods we use intra-route and inter-route versions with first-improvement selection.

In general, performing local search in stochastic vehicle routing problems is particularly demanding from a computational point of view, because move evaluations require the computation of complex objective functions and constraints. To overcome these difficulties, authors have proposed various strategies. For instance, Gendreau et al. (1996b) develop quick proxies to evaluate the impact of moves in the objective function of a solution to the VRPSD with stochastic customers. Using the proxies, the authors build an efficient tabu search that needs to compute the actual objective function of the search solution only every few iterations. Goodson et al. (2012) propose a different approach in the context of the classical VRPSD: their approach is a hybrid of simulated annealing and local search. It focuses exclusively on the deterministic part of the objective function (i.e., the total planned duration of the routes) during the first iterations, and it starts considering the stochastic part (i.e., the expected travel time added by recourses) only toward the end of the process. We propose an alternative strategy based on evaluating moves according to a three-step hierarchical procedure.

Let s be the solution of our local search procedure at a given iteration, $f(s)$ the objective function of s, and $t(s) = \sum_{r \in s} t_r$ the total planned duration of the routes in s. Let m be a candidate move and s' the solution that results from applying m to s. Since m is either a re-locate or a 2-opt move, it modifies two routes. Let $r \in s'$ and $r' \in s'$ be the two modified routes. Note that if m is intra-route then $r = r'$. The move evaluation procedure is as follows. In the first step, we check the feasibility of r and r' in terms of the expected load constraint. If either route is infeasible, the move is discarded and the evaluation aborted. In the second step, we apply what we call a *deterministic filter*. The idea is to try to filter moves using only the deterministic part of the objective function of s', thus avoiding the expensive calculation of its actual objective function. The filter consists on testing the condition $t(s') \leq f(s)$. Note that not every solution that passes the test is necessarily an improving solution; however, every solution that fails the test is a necessarily degrading solution. In the third step we complete the evaluation of $f(s')$ and use the result to determine whether or not the move is improving, that is, $f(s') < f(s)$. In the case of `CC` the move undergoes an additional evaluation step in which we check the feasibility of r and r' in terms of the duration constraint. Table 1 summarizes the complexity of each step of the move-evaluation procedure.

Table 1 Move-evaluation procedure: complexity summary

Step	`CC`		`PF`	
	$r = r'$	$r \neq r'$	$r = r$	$r \neq r'$
Check expected load	$O(1)$	$O(1)$	$O(1)$	$O(1)$
Check deterministic filter	$O(1)$	$O(1)$	$O(1)$	$O(1)$
Check improvement	$O(n_r^2)$	$O(n_r^2 + n_{r'}^2)$	$O(2^{n_r})$	$O(2^{n_r} + 2^{n_{r'}})$
Check DCs	$O(2^{n_r})$	$O(2^{n_r} + 2^{n_{r'}})$	N/A	N/A

3.3 Heuristic concentration

The idea behind HC is to try to build a global optimum using parts of the local optima found during a heuristic search procedure. To the best of our knowledge, the term was coined by Rosing and Revelle (1997) in the context of the facility location problem.[1] In the field of vehicle routing, HC has become an important component of metaheuristic-based approaches (see for instance Mendoza and Villegas 2013; Villegas et al. 2013; Pillac et al. 2013b; Subramanian et al. 2013; Contardo et al. 2014).

In the HC phase we use a commercial optimizer to solve an SPP formulation of the VRPSD-DC where the columns correspond to the routes stored in Ω. Since all the routes in Ω satisfy the expected load constraint (and the DC in the case of `CC`), the SPP needs to handle only constraints (5) and (6). The cost $c(r)$ of each column is the evaluation of the associated route depending on the formulation being used (`CC` or `PF`). The resulting SPP is $\min_{\mathcal{R}\subseteq\Omega}\left\{\sum_{r\in\mathcal{R}} c(r) : \bigcup_{r\in\mathcal{R}} = \mathcal{V}; r_i \bigcap r_j = \{0\}\, \forall r_i, r_j \in \mathcal{R}\right\}$. To speed up the HC phase, we use the objective function of the best solution found by the GRASP as an initial upper bound for the SPP.

4 Computational experiments

We implemented our GRASP + HC in Java (jre V.1.7.0_02-b13 64 bit) and used the Gurobi Optimizer (version 5.5.0) to solve the SPP. In the remainder of this section we refer to our method as GRASP + HC(`CC`) or GRASP + HC(`PF`) depending on the formulation used. All the gaps reported in this section are computed as

$$gap = \frac{f(s) - f(s^0)}{f(s^0)} \tag{11}$$

where $f(s^0)$ is the objective function of a reference solution and $f(s)$ is the objective function of the solution being tested. All the experiments were performed on a PC with a Pentium Dual-Core 3.20 GHz and 8 Gb of RAM, running Windows 7 Professional 64 bit.

4.1 Results for standard VRPSD instances

For validation purposes, we first tested our approach on the classical VRPSD. Note that solving the classical VRPSD is equivalent to solving `CC` with $\beta = 1$ (i.e., the DC becomes redundant). However, to avoid expensive verifications of the DC we deactivated it in both the split and move-evaluation procedures. We ran our GRASP + HC on the 40-instance testbed of Christiansen and Lysgaard (2007). These instances range from 16 to 60 customers and assume Poisson-distributed demands. To assess the effectiveness of our method, we compared our results to the best known solutions (BKSs) for the testbed: 38 optimal solutions reported by Gauvin (2012) and 2 heuristic

[1] The mechanism has been given different names, but we believe the term heuristic concentration best encapsulates the spirit of the idea.

Table 2 Summary of results for VRPSD instances

Metric	Method		
	GRASP + HC	MSH	SA
Avg. Gap	0.02 %	0.18 %	0.35 %
Max. Gap	0.19 %	1.16 %	1.89 %
Avg. CV	0.02 %	0.08 %	0.32 %
Avg. Best Gap	0.00 %	0.07 %	0.04 %
NBKS	40/40	27/40	33/40
Max. CPU (s)	102.43	782.77	603.80
Min. CPU (s)	1.69	5.91	9.00
Avg. CPU (s)	36.09	180.78	268.66

Avg. Gap average gap over the 400 runs, *Max. Gap* maximum gap over the 400 runs, *Avg. CV* average coefficient of variation of the objective function over the 40 instances, *Avg. Best Gap* average gap if only the best solution found for each instance is considered, *NBKS* number of best-known solutions matched, *Max. CPU (s)* maximum running time over the 400 runs, *Min. CPU (s)* minimum running time over the 400 runs, *Avg. CPU (s)* average running time over the 400 runs, *MSH* multi-space sampling heuristic of Mendoza and Villegas (2013) in its best-but-slowest configuration, *SA* simulated annealing algorithm of Goodson et al. (2012)

solutions reported by Goodson et al. (2012) and Mendoza and Villegas (2013). For each instance, we executed 10 runs with $K = 500$, $L_{RNN} = 3$, and $L_{RNI} = L_{RBI} = L_{RFI} = 6$. Table 2 summarizes our results (the solutions for each instance are reported in Appendix 1).

The results show that in terms of solution quality our approach outperforms the two state-of-the-art metaheuristics. Our algorithm matched the 40 BKSs for the set, whereas MSH achieves 27/40 and SA achieves 33/40. Moreover, the results for the average and worst-case behavior over multiple runs (i.e., Avg. Gap and Max. Gap) and the coefficient of variation suggest that our method is more stable than MSH and SA (i.e., finds close-to-BKS solutions more often). Although it is difficult to make a precise comparison of the computational performance because of slight differences in the testing environments,[2] the data suggest that our approach also outperforms the two other methods on this measure. In conclusion our GRASP + HC is a valid method for the classical VRPSD, and it can be expected to perform well on the closely related VRPSD-DC.

4.2 Results for VRPSD-DC instances

4.2.1 Instance generation

To the best of our knowledge, there are no publicly available instances for the VRPSD-DC. Therefore, we built a new benchmark set by adding DCs to the VRPSD instances

[2] SA was tested on Intel Xeon X5660 2.8 GHz processors with 12Gb of RAM (running CentOS 5.3), MSH was tested on a PC with an Intel Xeon 2.4 GHz and 12 Gb of RAM (running Windows Server 2008 64 bit).

of Christiansen and Lysgaard (2007). For each instance, we set the maximum duration limit to $T = \left\lceil \max_{r \in \mathcal{R}} E\left[\tilde{T}_r\right]\right\rceil$, where $\mathcal{R}$ is the set of routes in the best solution s^* found for the instance in the experiments reported in Sect. 4.1. Note that by construction s^* is the best known solution for the modified instance, if it is solved using the constrained expected duration formulation as in Yang et al. (2000) and Mendoza et al. (2010, 2011). In the remainder of this section, we refer to this alternative formulation as `ED`. From the adapted instance set, we excluded instance `P-n16-k8` because when solved using `CC` it is infeasible for the most interesting values of β (i.e., $\beta < 0.15$).[3] To allow future comparisons with our results, we made our instances available at www.vrp-rep.org (Mendoza et al. 2014).

4.2.2 *Chance-constraint formulation*

In this section we discuss the results of GRASP + HC(`CC`) for the 39 instances of the adapted set. The main objective of this experiment is to analyze how solutions built using the chance-constraint paradigm compare with those built under the more classical constrained expected duration approach. We first set β to 0.05, a value that we consider plausible from a managerial perspective. Next, we conducted a post-hoc analysis of each of the best-known `ED` solutions. This analysis involves evaluating $Pr(\tilde{T}_r > T)$ for each route r in the solution and finding how many routes become infeasible if the chance constraint is imposed. We then performed 10 GRASP + HC(`CC`) runs with $\beta = 0.05$, $K = 500$, $L_{RNN} = 3$, and $L_{RNI} = L_{RBI} = L_{RFI} = 6$. For the best solution found for each instance we computed the total increase in the objective function ΔOF, with respect to the corresponding best-known `ED` solution using Eq. (11). As expected from the results for the VRPSD instances (Sect. 4.1) our method exhibited stable performance on the adapted set: the minimum, average, and maximum average coefficients of variation among the 39 instances were 0.00, 0.09, and 0.48 %, respectively. Therefore, we feel confident that the conclusions drawn by analyzing the best solutions are valid for the general case. Table 3 presents the results.

Not surprisingly, the `ED` solutions are poor when the chance constraint is added: only 3 out of the 39 solutions remain feasible. The results for the percentages of infeasible routes show that the infeasibilities increase because of multiple failing routes rather than isolated cases. More interestingly, the data also show that the routes in the `ED` solutions tend to have high probabilities of violating the maximum duration limit. Moreover, a close look at the results reveals that the behavior of the routes with respect to this probability is rather unstable. For example, in instance P-n50-k10 the probability of violating the DC ranges from 0.00 to 31.67 % in the 11 routes of the solution. As mentioned earlier, these results are expected, because the `ED` formulation does not provide a mechanism to control the probability of routes violating the DC. Nonetheless, the results of our ad-hoc analysis shed some light on the inconvenience of

[3] In fact, customer 2 violates one of the basic assumptions of the problem since $Pr(\tilde{\xi}_2 > Q) = 0.1573$. Because of the high failure probability and the travel time to the depot, it is impossible to include customer 2 in a route, even the trivial route (0, 2, 0), without violating the DC for $\beta < 0.1573$.

Table 3 Results for the VRPSD-DC instances

Instance	BKS-ED							GRASP + HC(CC)		
	OF	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{I}\rvert$	$\lvert\mathcal{I}\rvert/\lvert\mathcal{R}\rvert$	$Pr\left(\tilde{T}_r > T\right)$			OF	ΔOF	$\lvert\mathcal{R}\rvert$
					Max.	Min.	Avg.			
A-n32-k5	853.60	5	1	20.00	15.94	0.00	3.59	866.77	1.54	5
A-n33-k5	704.20	5	3	60.00	9.34	0.00	4.63	735.00	4.37	6
A-n33-k6	793.90	6	0	0.00	3.87	0.00	1.42	793.90	0.00	6
A-n34-k5	826.80	6	3	50.00	18.18	0.00	7.52	839.01	1.48	6
A-n36-k5	858.70	5	1	20.00	39.43	0.00	7.89	861.74	0.35	5
A-n37-k5	708.30	5	2	40.00	29.47	0.00	7.86	713.99	0.80	5
A-n37-k6	1,030.70	7	1	14.29	18.67	0.00	3.25	1,032.96	0.22	7
A-n38-k5	775.10	6	2	33.33	9.26	0.00	2.57	777.59	0.32	6
A-n39-k5	869.10	6	3	50.00	12.93	0.00	4.83	942.45	8.44	6
A-n39-k6	876.60	6	2	33.33	29.45	0.00	7.80	889.40	1.46	6
A-n44-k6	1,025.40	7	1	14.29	9.26	0.00	2.04	1,032.70	0.71	7
A-n45-k6	1,026.70	7	3	42.86	17.82	0.00	5.57	1,045.71	1.85	7
A-n45-k7	1,264.80	7	4	57.14	28.92	0.00	8.80	1,298.71	2.68	8
A-n46-k7	1,002.20	7	2	28.57	13.17	0.00	3.63	1,007.11	0.49	7
A-n48-k7	1,187.10	7	4	57.14	27.26	0.00	10.15	1,210.79	2.00	7
A-n53-k7	1,124.20	8	1	12.50	9.13	0.00	2.62	1,127.54	0.30	8
A-n54-k7	1,287.00	8	2	25.00	36.04	0.00	6.95	1,309.13	1.72	8
A-n55-k9	1,179.10	10	3	30.00	20.61	0.00	4.39	1,203.92	2.10	10
A-n60-k9	1,529.82	10	3	30.00	21.62	0.00	3.95	1,543.44	0.89	10
E-n22-k4	411.50	4	2	50.00	25.22	0.00	11.16	429.56	4.39	5
E-n33-k4	850.20	4	0	0.00	1.03	0.00	0.33	850.27	0.01	4
E-n51-k5	552.26	6	1	16.67	16.90	0.00	3.51	554.54	0.41	6
P-n19-k2	224.00	3	1	33.33	17.18	0.00	5.73	233.36	4.18	3
P-n20-k2	233.00	2	1	50.00	44.63	0.79	22.71	240.84	3.36	3
P-n21-k2	218.90	2	1	50.00	16.89	0.47	8.68	234.00	6.90	3
P-n22-k2	231.20	2	2	100.00	42.98	16.89	29.93	242.19	4.75	3
P-n22-k8	681.00	9	4	44.44	37.08	0.00	10.16	715.81	5.11	10
P-n23-k8	619.50	9	1	11.11	39.52	0.00	4.98	634.46	2.41	10
P-n40-k5	472.50	5	3	60.00	20.19	0.80	9.68	488.50	3.39	5
P-n45-k5	533.52	5	3	60.00	31.97	0.03	11.05	539.66	1.15	6
P-n50-k10	758.70	11	3	27.27	31.67	0.00	7.90	772.25	1.79	11
P-n50-k7	582.30	7	1	14.29	20.78	0.00	3.09	584.37	0.35	7
P-n50-k8	669.20	9	4	44.44	9.77	0.00	3.91	680.42	1.68	9
P-n51-k10	809.70	11	3	27.27	25.00	0.00	6.59	833.42	2.93	11
P-n55-k10	742.40	10	6	60.00	32.62	0.00	9.25	759.36	2.28	11
P-n55-k15	1,068.00	18	4	22.22	24.40	0.00	3.28	1,086.44	1.73	17
P-n55-k7	588.50	7	0	0.00	3.45	0.00	0.61	588.56	0.01	7

Table 3 continued

Instance	BKS-ED							GRASP + HC(CC)		
	OF	$\|\mathcal{R}\|$	$\|\mathcal{I}\|$	$\|\mathcal{I}\|/\|\mathcal{R}\|$	$Pr\left(\tilde{T}_r > T\right)$			OF	ΔOF	$\|\mathcal{R}\|$
					Max.	Min.	Avg.			
P-n60-k10	803.60	11	4	36.36	15.83	0.00	4.29	811.44	0.98	11
P-n60-k15	1,085.40	16	6	37.50	18.85	0.00	4.90	1,110.72	2.33	17
Max.				100.00	44.63	16.89	29.93		8.44	
Min.				0.00	1.03	0.00	0.33		0.00	
Avg.				34.96	21.70	0.49	6.70		2.10	
SD				20.78	11.22	2.70	5.52		1.93	

OF: objective function; $|\mathcal{R}|$ number of routes in the solution; $|\mathcal{I}|$ number of infeasible routes in the solution, where $\mathcal{I} = \{r \in \mathcal{R} | Pr(\tilde{T}_r > T) > \beta\}$; $|\mathcal{I}|/|\mathcal{R}|$: % of infeasible routes; $Pr(\tilde{T}_r > T)$: probability of violating the duration constraint in %; ΔOF increase in the objective function with respect to the ED solution in %

using the ED formulation in practice. The results of GRASP + HC(CC) suggest that the chance-constraint approach may be better suited for practical situations. Clearly, every route in a CC solution has a probability of violating the DC that is lower than 5.00 %. As the data show, this improvement in the reliability comes with a moderate increase in the total expected travel time of the solutions (2.10 % on average). With the notable exception of instance A-n39-k5, the largest increases in the expected travel time are observed in solutions in which an extra route is needed to achieve reliability (10/39 cases). Note that in practical situations where using an extra route is not possible, the decision-makers can obtain tradeoffs between reliability, the expected travel time, and (indirectly) the number of routes by performing a sensitivity analysis for the value of β.

4.2.3 Penalty formulation

In contrast to CC, the penalty formulation PF does not control the probability of violating the DC but rather the magnitude of the violations. To simulate different profiles of aversion toward overtime, we ran experiments with three different $\phi(\cdot)$ cost functions: linear, piecewise linear, and quadratic. The exact expressions used in our experiments are: $\phi_1(O_r) = 2 \times O_r$; $\phi_2(O_r) = \lambda \times O_r$; and $\phi_3(O_r) = O_r^2$ where

$$\lambda = \begin{cases} 1.5 & \text{if} \quad O_r \leq 0.05 \times T, \\ 3.0 & \text{if} \quad 0.05 \times T < O_r \leq 0.10 \times T, \\ 5.0 & \text{if} \quad O_r > 0.10 \times T. \end{cases} \tag{12}$$

To evaluate how the ED solutions perform in situations where the magnitude of the DC violations is relevant, we computed two metrics for each BKS: the total

Table 4 Results for the VRPSD-DC instances with linear penalty

Instance	ED				GRASP + HC(PF)					
	$E[T(\cdot)]$	$E[O(\cdot)]$	$E[\phi_1(\cdot)]$	$\|\mathcal{R}\|$	$E[T(\cdot)]$	$\Delta E[T(\cdot)]$	$E[O(\cdot)]$	$\Delta E[O(\cdot)]$	$E[\phi_1(\cdot)]$	$\|\mathcal{R}\|$
A-n32-k5	853.60	2.21	4.42	5	853.60	0.00	2.21	0.00	4.42	5
A-n33-k5	704.20	9.06	18.11	5	704.20	0.00	9.06	0.00	18.11	5
A-n33-k6	793.90	4.87	9.73	6	794.15	0.03	4.15	−14.81	8.29	6
A-n34-k5	826.87	25.76	51.53	6	839.01	1.47	4.08	−84.16	8.16	6
A-n36-k5	858.71	13.93	27.87	5	861.74	0.35	0.82	−94.11	1.64	5
A-n37-k5	708.34	12.89	25.77	5	715.37	0.99	0.54	−95.81	1.08	5
A-n37-k6	1,030.73	6.02	12.05	7	1,030.73	0.00	6.02	0.00	12.05	7
A-n38-k5	775.13	4.94	9.89	6	777.59	0.32	1.28	−74.13	2.56	6
A-n39-k5	869.18	18.49	36.97	6	869.18	0.00	18.49	0.00	36.97	6
A-n39-k6	876.60	20.12	40.24	6	895.59	2.17	0.90	−95.55	1.79	6
A-n44-k6	1,025.48	11.71	23.42	7	1,033.69	0.80	5.26	−55.08	10.52	7
A-n45-k6	1,026.73	19.48	38.96	7	1,035.48	0.85	4.39	−77.47	8.78	7
A-n45-k7	1,264.83	42.99	85.98	7	1,308.01	3.41	6.79	−84.20	13.58	8
A-n46-k7	1,002.22	14.26	28.53	7	1,005.31	0.31	6.14	−56.94	12.28	7
A-n48-k7	1,187.14	32.46	64.93	7	1,215.41	2.38	5.71	−82.41	11.42	7
A-n53-k7	1,124.27	10.99	21.99	8	1,128.71	0.40	7.66	−30.28	15.33	8
A-n54-k7	1,287.07	30.95	61.90	8	1,313.50	2.05	6.45	−79.16	12.90	8
A-n55-k9	1,179.11	15.06	30.13	10	1,181.58	0.21	9.67	−35.82	19.33	10
A-n60-k9	1,529.82	36.10	72.19	10	1,543.44	0.89	6.28	−82.61	12.56	10
E-n22-k4	411.57	12.77	25.54	4	419.15	1.84	5.13	−59.84	10.25	5
E-n33-k4	850.27	1.74	3.48	4	850.27	0.00	1.74	0.00	3.48	4
E-n51-k5	552.26	4.11	8.23	6	554.54	0.41	0.39	−90.47	0.78	6
P-n19-k2	224.06	4.05	8.10	3	224.06	0.00	4.05	0.00	8.10	3
P-n20-k2	233.05	6.06	12.12	2	237.06	1.72	2.50	−58.69	5.01	3
P-n21-k2	218.96	3.15	6.30	2	218.96	0.00	3.15	0.00	6.30	2
P-n22-k2	231.26	5.13	10.25	2	231.26	0.00	5.13	0.00	10.25	2
P-n22-k8	681.06	18.51	37.01	9	689.15	1.19	6.99	−62.22	13.98	9
P-n23-k8	619.53	16.51	33.03	9	634.46	2.41	2.16	−86.91	4.32	10
P-n40-k5	472.50	5.63	11.27	5	472.50	0.00	5.63	0.00	11.27	5
P-n45-k5	533.52	6.75	13.50	5	541.65	1.52	0.44	−93.44	0.89	6
P-n50-k10	758.76	15.82	31.63	11	766.17	0.98	3.86	−75.57	7.73	11
P-n50-k7	582.37	4.10	8.19	7	585.05	0.46	0.12	−96.97	0.25	7
P-n50-k8	669.23	8.52	17.04	9	672.22	0.45	2.01	−76.39	4.02	9
P-n51-k10	809.70	15.41	30.81	11	815.52	0.72	6.50	−57.81	13.00	11
P-n55-k10	742.41	8.95	17.91	10	743.90	0.20	4.35	−51.41	8.70	10
P-n55-k15	1,068.05	18.36	36.72	18	1,071.47	0.32	6.62	−63.92	13.25	18
P-n55-k7	588.56	0.92	1.85	7	588.56	0.00	0.92	0.00	1.85	7

Table 4 continued

Instance	ED				GRASP + HC(PF)					
	$E[T(\cdot)]$	$E[O(\cdot)]$	$E[\phi_1(\cdot)]$	$\lvert\mathcal{R}\rvert$	$E[T(\cdot)]$	$\Delta E[T(\cdot)]$	$E[O(\cdot)]$	$\Delta E[O(\cdot)]$	$E[\phi_1(\cdot)]$	$\lvert\mathcal{R}\rvert$
P-n60-k10	803.60	9.91	19.83	11	808.23	0.58	3.52	−64.45	7.05	11
P-n60-k15	1,085.49	22.15	44.29	16	1,100.94	1.42	4.50	−79.68	9.00	17
Max.						3.41		0.00		
Min.						0.00		−96.97		
Avg.						0.79		−52.83		
SD						0.85		35.61		

$E[T(\mathcal{R})]$: total expected duration; $E[O(\mathcal{R})]$: total expected overtime; $E[\phi_1(O(\mathcal{R}))]$: total expected overtime cost; $|\mathcal{R}|$: number of routes; $\Delta E[T(\mathcal{R})]$ (%): relative difference in the expected duration with respect to the ED solution; $\Delta E[O(\mathcal{R})]$ (%): relative difference in the expected overtime with respect to the ED solution

Table 5 Results for the VRPSD-DC instances with piecewise linear penalty

Instance	ED				GRASP + HC(PF)					
	$E[T(\cdot)]$	$E[O(\cdot)]$	$E[\phi_2(\cdot)]$	$\lvert\mathcal{R}\rvert$	$E[T(\cdot)]$	$\Delta E[T(\cdot)]$	$E[O(\cdot)]$	$\Delta E[O(\cdot)]$	$E[\phi_2(\cdot)]$	$\lvert\mathcal{R}\rvert$
A-n32-k5	853.60	2.21	6.12	5	853.60	0.00	2.21	0.00	6.12	5
A-n33-k5	704.20	9.06	45.28	5	727.26	3.28	4.39	−51.48	19.00	5
A-n33-k6	793.90	4.87	24.33	6	803.05	1.15	2.07	−57.51	10.34	7
A-n34-k5	826.87	25.76	128.82	6	839.01	1.47	4.08	−84.16	20.40	6
A-n36-k5	858.71	13.93	67.67	5	861.74	0.35	0.82	−94.11	2.32	5
A-n37-k5	708.34	12.89	56.08	5	715.37	0.99	0.54	−95.81	2.41	5
A-n37-k6	1,030.73	6.02	27.50	7	1, 046.74	1.55	0.73	−87.82	3.65	7
A-n38-k5	775.13	4.94	23.18	6	777.59	0.32	1.28	−74.13	4.83	6
A-n39-k5	869.18	18.49	91.92	6	942.45	8.43	2.14	−88.42	9.93	6
A-n39-k6	876.60	20.12	97.10	6	895.59	2.17	0.90	−95.55	4.39	6
A-n44-k6	1,025.48	11.71	58.54	7	1,033.69	0.80	5.26	−55.08	18.49	7
A-n45-k6	1,026.73	19.48	93.32	7	1,048.58	2.13	1.42	−92.69	6.22	7
A-n45-k7	1,264.83	42.99	212.01	7	1,316.61	4.09	3.45	−91.98	16.57	8
A-n46-k7	1,002.22	14.26	71.31	7	1,005.31	0.31	6.14	−56.94	27.32	7
A-n48-k7	1,187.14	32.46	156.69	7	1,229.31	3.55	2.58	−92.04	12.60	7
A-n53-k7	1,124.27	10.99	54.96	8	1,132.24	0.71	9.57	−12.90	23.48	8
A-n54-k7	1,287.07	30.95	152.95	8	1,323.15	2.80	3.09	−90.03	15.32	8
A-n55-k9	1,179.11	15.06	72.89	10	1,196.01	1.43	6.68	−55.65	27.26	10
A-n60-k9	1,529.82	36.10	178.01	10	1,552.96	1.51	1.80	−95.03	7.34	10
E-n22-k4	411.57	12.77	60.61	4	429.56	4.37	0.81	−93.67	4.02	5
E-n33-k4	850.27	1.74	8.70	4	854.05	0.44	0.46	−73.43	2.29	4
E-n51-k5	552.26	4.11	20.44	6	554.54	0.41	0.39	−90.47	1.59	6
P-n19-k2	224.06	4.05	20.25	3	233.36	4.15	0.19	−95.43	0.93	3
P-n20-k2	233.05	6.06	30.17	2	242.11	3.89	0.22	−96.42	1.00	3

Table 5 continued

Instance	ED				GRASP + HC(PF)					
	$E[T(\cdot)]$	$E[O(\cdot)]$	$E[\phi_2(\cdot)]$	$\lvert\mathcal{R}\rvert$	$E[T(\cdot)]$	$\Delta E[T(\cdot)]$	$E[O(\cdot)]$	$\Delta E[O(\cdot)]$	$E[\phi_2(\cdot)]$	$\lvert\mathcal{R}\rvert$
P-n21-k2	218.96	3.15	15.74	2	218.96	0.00	3.15	0.00	15.74	2
P-n22-k2	231.26	5.13	22.93	2	242.19	4.73	0.99	−80.78	4.24	3
P-n22-k8	681.06	18.51	84.82	9	692.06	1.61	5.57	−69.89	25.36	9
P-n23-k8	619.53	16.51	82.33	9	639.29	3.19	0.27	−98.37	0.96	10
P-n40-k5	472.50	5.63	26.62	5	482.75	2.17	2.73	−51.51	9.40	5
P-n45-k5	533.52	6.75	29.28	5	541.65	1.52	0.44	−93.44	2.00	6
P-n50-k10	758.76	15.82	72.90	11	766.17	0.98	3.86	−75.57	13.37	11
P-n50-k7	582.37	4.10	20.40	7	585.05	0.46	0.12	−96.97	0.26	7
P-n50-k8	669.23	8.52	40.88	9	672.22	0.45	2.01	−76.39	6.89	9
P-n51-k10	809.70	15.41	73.98	11	823.75	1.74	4.14	−73.13	20.49	12
P-n55-k10	742.41	8.95	36.80	10	755.84	1.81	2.01	−77.51	6.94	11
P-n55-k15	1,068.05	18.36	91.80	18	1,083.46	1.44	3.16	−82.78	11.49	17
P-n55-k7	588.56	0.92	4.45	7	591.95	0.57	0.08	−91.03	0.39	7
P-n60-k10	803.60	9.91	48.45	11	812.93	1.16	2.46	−75.22	11.77	11
P-n60-k15	1,085.49	22.15	110.60	16	1,104.47	1.75	4.06	−81.69	14.98	17
Max.						8.43		0.00		
Min.						0.00		−98.37		
Avg.						1.89		−75.51		
SD						1.67		24.83		

$E[T(\mathcal{R})]$: total expected duration; $E[O(\mathcal{R})]$: total expected overtime; $E[\phi_2(O(\mathcal{R}))]$: total expected overtime cost; $|\mathcal{R}|$: number of routes; $\Delta E[T(\mathcal{R})]$ (%): relative difference in the expected duration with respect to the ED solution; $\Delta E[O(\mathcal{R})]$ (%): relative difference in the expected overtime with respect to the ED solution

expected overtime $E[O(\mathcal{R})] = \sum_{r\in\mathcal{R}} E[\tilde{O}_r]$ and the total expected overtime cost $E[\phi(O(\mathcal{R}))] = \sum_{r\in\mathcal{R}} E[\phi(\tilde{O}_r)]$. We then compared the performance with that of the solutions of GRASP + HC(PF) with the linear, piecewise linear, and quadratic penalties. Tables 4, 5, and 6 present the results.

The results show that the ED solutions not only tend to have high probabilities of incurring overtime, as discussed in Sect. 4.2.2, but they also incur excessive overtime. Measured as a proportion of the total expected duration (i.e., $E[O(\mathcal{R})]/E[T(\mathcal{R})]$) the expected overtime accounts on average for 1.65 % of the total expected travel time of the route set and at most 3.4 % (instance A-n45-k7). As expected, the PF solutions are better. Under the softest penalty scheme (linear) the expected overtime of the PF solutions as a proportion of the total duration reduces to 0.64 %, and with the quadratic scheme the figure is 0.07 %. Another way to look at this is through the reductions in the expected overtime reported in the column labeled $\Delta E\,[O(\cdot)]$. For the linear penalty this figure is on average −52.83 %, for the piecewise linear penalty it is −75.51 %, and for the quadratic penalty it is −93.52 %. This improvement in the overtime comes with an increase in the expected duration of the routes. The increases are on average 0.79,

Table 6 Results for the VRPSD-DC instances with quadratic penalty

Instance	ED				GRASP + HC(PF)					
	$E[T(\cdot)]$	$E[O(\cdot)]$	$E[\phi_3(\cdot)]$	$\lvert\mathcal{R}\rvert$	$E[T(\cdot)]$	$\Delta E[T(\cdot)]$	$E[O(\cdot)]$	$\Delta E[O(\cdot)]$	$E[\phi_3(\cdot)]$	$\lvert\mathcal{R}\rvert$
A-n32-k5	853.60	2.21	99.96	5	882.77	3.42	1.30	−40.98	14.33	5
A-n33-k5	704.20	9.06	470.03	5	761.17	8.09	0.09	−99.06	4.84	6
A-n33-k6	793.90	4.87	305.71	6	834.39	5.10	0.39	−91.94	9.89	7
A-n34-k5	826.87	25.76	1,776.39	6	904.10	9.34	0.08	−99.69	4.15	6
A-n36-k5	858.71	13.93	1,849.25	5	872.13	1.56	0.01	−99.93	0.63	5
A-n37-k5	708.34	12.89	706.81	5	721.03	1.79	0.21	−98.36	1.69	5
A-n37-k6	1,030.73	6.02	704.27	7	1,071.56	3.96	0.11	−98.21	6.89	7
A-n38-k5	775.13	4.94	254.13	6	791.32	2.09	0.18	−96.35	5.27	6
A-n39-k5	869.18	18.49	1,408.24	6	969.53	11.55	0.16	−99.12	4.62	6
A-n39-k6	876.60	20.12	1,299.52	6	920.65	5.03	0.39	−98.04	11.78	6
A-n44-k6	1,025.48	11.71	1,079.56	7	1,061.73	3.53	0.49	−95.78	12.06	7
A-n45-k6	1,026.73	19.48	1,469.72	7	1,066.00	3.83	0.00	−99.98	0.48	8
A-n45-k7	1,264.83	42.99	4,023.41	7	1,338.75	5.84	0.77	−98.20	41.24	8
A-n46-k7	1,002.22	14.26	1,122.64	7	1,072.23	6.99	0.25	−98.25	5.78	8
A-n48-k7	1,187.14	32.46	2,356.65	7	1,276.01	7.49	0.52	−98.41	18.27	8
A-n53-k7	1,124.27	10.99	859.81	8	1,174.12	4.43	1.06	−90.31	23.28	8
A-n54-k7	1,287.07	30.95	3,412.74	8	1,369.11	6.37	0.25	−99.20	8.02	8
A-n55-k9	1,179.11	15.06	964.54	10	1,269.32	7.65	2.21	−85.34	22.96	10
A-n60-k9	1,529.82	36.10	4,052.28	10	1,576.97	3.08	0.45	−98.76	8.02	10
E-n22-k4	411.57	12.77	610.51	4	447.90	8.83	0.06	−99.50	0.20	5
E-n33-k4	850.27	1.74	230.74	4	869.74	2.29	0.23	−86.52	19.21	4
E-n51-k5	552.26	4.11	104.47	6	554.54	0.41	0.39	−90.47	4.83	6
P-n19-k2	224.06	4.05	98.15	3	233.36	4.15	0.19	−95.42	7.23	3
P-n20-k2	233.05	6.06	88.11	2	242.11	3.89	0.22	−96.42	5.97	3
P-n21-k2	218.96	3.15	64.26	2	249.02	13.73	0.01	−99.54	0.49	3
P-n22-k2	231.26	5.13	108.16	2	254.11	9.88	0.01	−99.72	0.49	3
P-n22-k8	681.06	18.51	915.52	9	750.56	10.20	1.73	−90.68	35.79	10
P-n23-k8	619.53	16.51	630.02	9	639.29	3.19	0.27	−98.37	6.79	10
P-n40-k5	472.50	5.63	83.16	5	490.48	3.81	0.70	−87.51	7.47	6
P-n45-k5	533.52	6.75	165.61	5	543.79	1.92	0.21	−96.86	2.30	6
P-n50-k10	758.76	15.82	405.58	11	786.41	3.64	1.44	−90.87	10.54	11
P-n50-k7	582.37	4.10	88.87	7	585.05	0.46	0.12	−96.97	1.05	7
P-n50-k8	669.23	8.52	240.41	9	680.65	1.71	1.24	−85.49	9.24	9
P-n51-k10	809.70	15.41	524.44	11	854.42	5.52	0.57	−96.31	6.07	12
P-n55-k10	742.41	8.95	163.11	10	760.93	2.49	1.02	−88.58	9.09	11
P-n55-k15	1,068.05	18.36	659.15	18	1,091.76	2.22	1.54	−91.63	23.57	17
P-n55-k7	588.56	0.92	21.37	7	592.10	0.60	0.11	−87.77	0.92	7

Table 6 continued

Instance	ED				GRASP + HC(PF)					
	$E[T(\cdot)]$	$E[O(\cdot)]$	$E[\phi_3(\cdot)]$	$\|\mathcal{R}\|$	$E[T(\cdot)]$	$\Delta E[T(\cdot)]$	$E[O(\cdot)]$	$\Delta E[O(\cdot)]$	$E[\phi_3(\cdot)]$	$\|\mathcal{R}\|$
P-n60-k10	803.60	9.91	268.30	11	828.32	3.08	1.28	−87.09	11.69	11
P-n60-k15	1,085.49	22.15	721.68	16	1,125.52	3.69	0.97	−95.61	11.75	17
Max.						13.73		−40.98		
Min.						0.41		−99.98		
Avg.						4.79		−93.52		
SD						3.16		9.72		

$E[T(\mathcal{R})]$: total expected duration; $E[O(\mathcal{R})]$: total expected overtime; $E[\phi_3(O(\mathcal{R}))]$: total expected overtime cost; $|\mathcal{R}|$: number of routes; $\Delta E[T(\mathcal{R})]$ (%): relative difference in the expected duration with respect to the ED solution; $\Delta E[O(\mathcal{R})]$ (%): relative difference in the expected overtime with respect to the ED solution

1.89, and 4.79 % for the linear, piecewise linear, and quadratic penalty mechanisms, respectively.

4.2.4 A word about execution times

Table 7 summarizes the computational performance of GRASP + HC(·) for each formulation (detailed results are given in Appendix 2). As expected, dealing with duration-profile computations in CC and PF increases the time needed to solve the problem with respect to the classical VRPSD. The data show similar average running times for ED and CC, but the Max. CPU and Std. Dev. metrics tip the balance toward the former in terms of the computational performance. On the other hand, the CPU times for PF are consistently twice as large as those for ED, independent of the penalty function. There are two reasons for the difference between CC and PF. First, under PF the split procedure in Algorithm 2 tends to perform more inner-loop iterations (lines 6–22), since the expected load constraint is the only condition that can stop arc extensions (line 13). Second, as Table 1 shows, the most computationally expensive part of a move evaluation under CC comes at the last step, which is reached by only a few moves.

Table 7 Execution time summary. CPU: execution time in seconds

Metric	ED	CC	PF		
			Linear	Piecewise	Quadratic
Avg. CPU	36.09	34.73	88.46	88.90	83.45
Min. CPU	1.69	1.71	2.96	2.69	2.64
Max. CPU	102.43	242.00	434.90	450.93	475.19
SD CPU	27.08	46.78	100.15	102.42	102.26

All metrics are computed over 390 runs for each approach

5 Conclusions

We have studied a problem that has received little attention in the literature: the vehicle routing problem with stochastic demands and DCs (VRPSD-DC). We have discussed two different formulations for the problem, namely `CC` and `PF`. In `CC` the DCs are handled as chance constraints, meaning that for each route, the probability of exceeding the maximum duration must be lower than a given threshold. In `PF`, violations to the DC are penalized in the objective function. To solve the problem, we introduce a hybrid metaheuristic (GRASP + HC). In the GRASP phase, our method uses a set of randomized route-first, cluster-second heuristics to generate initial solutions and a VND with two move types for the local search. To accelerate the local search procedure, we use a three-step move-evaluation procedure that allows a quick rejection of unpromising moves. In the HC phase, we use a commercial optimizer to solve an SPP formulation of the problem over the set of routes found in the local optima. In contrast to the few solution approaches previously reported, our method does not use Monte Carlo simulation to verify the chance constraints or to compute the penalties for violations of the maximum duration. These tasks are accomplished by explicitly building the probability distribution of the total duration of the routes. We have discussed in detail the computational implications of our approach.

For validation purposes, we tested our method on a 40-instance standard testbed for the classical VRPSD. Our algorithm matched all 40 BKSs (38 of which are optimal); the two state-of-the-art metaheuristics for the problem cannot match this result. For experiments on the VRPSD-DC, we have proposed a set of 39 instances that we have made publicly available. Our experiments have focused on analyzing how solutions built using the most classical approach in the literature, i.e., enforcing DCs over the expected travel time of the routes (`ED`), differ from those built using the chance-constraint and penalty paradigms. Our results show that under `CC` and `PF` our GRASP + HC provides solutions with a good tradeoff between reliability, measured in terms of violations to the DCs, and increases in the total expected travel time.

Research currently underway includes extensions of our method to solve the VRPSD with a heterogeneous fleet and the VRP with stochastic and correlated demands.

Acknowledgments This research was partially funded by the *Region Pays de la Loire (France)* through project LigéRO, *Universidad de Antioquia (Colombia)* through project CODI MDC11-01-09, and *École Polytechnique de Montréal (Canada)*. The authors would like to thank Charles Gauvin at CIRRELT (Montreal) for providing the optimal solutions for the VRSPD instances used in Sect. 4.1.

Appendix 1: Detailed results for VRPSD instances

See Table 8

Table 8 Results for the Christiansen and Lysgaard (2007) instances

Instance	BKS	GRASP + HC						Mendoza and Villegas- MSH						Goodson et al. - SA					
		Average				Best		Average				Best		Average				Best	
		CPU	OF	CV	Gap	OF	Gap	CPU	OF	CV	Gap	OF	Gap	CPU	OF	CV	Gap	OF	Gap
A-n32-k5	853.60*	14.59	853.60	0.00	0.00	853.60	0.00	83.80	854.13	0.09	0.06	853.60	0.00	199.80	853.60	0.00	0.00	853.60	0.00
A-n33-k5	704.20*	13.15	704.20	0.00	0.00	704.20	0.00	81.30	707.99	0.31	0.54	704.20	0.00	178.20	705.91	0.40	0.24	704.20	0.00
A-n33-k6	793.90*	13.34	793.90	0.00	0.00	793.90	0.00	62.94	794.03	0.02	0.02	793.90	0.00	141.10	793.95	0.01	0.01	793.90	0.00
A-n34-k5	826.87*	14.58	826.87	0.00	0.00	826.87	0.00	84.24	826.87	0.00	0.00	826.87	0.00	236.40	827.26	0.15	0.05	826.87	0.00
A-n36-k5	858.71*	20.26	858.71	0.00	0.00	858.71	0.00	128.27	865.59	0.48	0.80	858.71	0.00	276.10	859.48	0.17	0.09	858.71	0.00
A-n37-k5	708.34*	23.23	708.34	0.00	0.00	708.34	0.00	179.40	709.51	0.00	0.16	709.51	0.16	386.90	709.67	0.59	0.19	708.34	0.00
A-n37-k6	1,030.73*	20.14	1, 030.86	0.04	0.01	1, 030.73	0.00	90.15	1, 030.73	0.00	0.00	1, 030.73	0.00	205.50	1, 031.74	0.31	0.10	1, 030.73	0.00
A-n38-k5	775.13*	20.11	775.13	0.00	0.00	775.13	0.00	118.63	776.33	0.06	0.15	775.13	0.00	313.40	775.25	0.05	0.01	775.14	0.00
A-n39-k5	869.18*	27.89	869.18	0.00	0.00	869.18	0.00	145.48	872.41	0.34	0.37	869.45	0.03	257.60	869.58	0.05	0.05	869.18	0.00
A-n39-k6	876.60*	25.33	876.60	0.00	0.00	876.60	0.00	143.78	876.60	0.00	0.00	876.60	0.00	239.90	883.44	0.83	0.78	876.60	0.00
A-n44-k6	1,025.48*	33.93	1, 025.92	0.09	0.04	1, 025.48	0.00	197.09	1, 025.95	0.05	0.05	1, 025.54	0.01	281.40	1, 029.72	0.59	0.41	1, 025.48	0.00
A-n45-k6	1,026.73*	31.93	1, 026.81	0.01	0.01	1, 026.73	0.00	173.74	1, 026.87	0.00	0.01	1, 026.87	0.01	301.40	1, 027.92	0.13	0.12	1, 026.73	0.00
A-n45-k7	1,264.83*	38.47	1, 267.05	0.22	0.18	1, 264.83	0.00	173.75	1, 268.63	0.10	0.30	1, 266.57	0.14	216.00	1, 288.70	1.33	1.89	1, 264.99	0.01
A-n46-k7	1,002.22*	46.23	1, 002.22	0.00	0.00	1, 002.22	0.00	192.27	1, 002.22	0.00	0.00	1, 002.22	0.00	314.10	1, 003.19	0.11	0.10	1, 002.22	0.00
A-n48-k7	1,187.14*	55.05	1, 187.32	0.05	0.02	1, 187.14	0.00	281.82	1, 200.96	0.12	1.16	1, 198.07	0.92	292.00	1, 188.06	0.16	0.08	1, 187.14	0.00
A-n53-k7	1,124.27*	80.22	1, 124.27	0.00	0.00	1, 124.27	0.00	346.62	1, 128.09	0.12	0.34	1, 125.78	0.13	468.80	1, 129.36	0.36	0.45	1, 124.27	0.00
A-n54-k7	1,287.07*	86.17	1, 287.41	0.02	0.03	1, 287.07	0.00	352.10	1, 291.50	0.18	0.34	1, 287.82	0.06	409.40	1, 292.29	0.69	0.41	1, 287.07	0.00
A-n55-k9	1,179.11*	66.16	1, 179.11	0.00	0.00	1, 179.11	0.00	238.76	1, 180.84	0.19	0.15	1, 179.11	0.00	265.50	1, 184.77	0.48	0.48	1, 179.11	0.00
E-n22-k4	411.57*	4.24	411.57	0.00	0.00	411.57	0.00	22.10	411.57	0.00	0.00	411.57	0.00	104.10	411.57	0.00	0.00	411.57	0.00
E-n33-k4	850.27*	24.66	851.87	0.13	0.19	850.27	0.00	134.35	854.64	0.07	0.51	853.04	0.33	371.70	851.24	0.36	0.11	850.27	0.00
P-n16-k8	512.82*	1.69	512.82	0.00	0.00	512.82	0.00	5.91	512.82	0.00	0.00	512.82	0.00	9.00	512.82	0.00	0.00	512.82	0.00
P-n19-k2	224.06*	3.51	224.06	0.00	0.00	224.06	0.00	40.29	224.06	0.00	0.00	224.06	0.00	234.80	224.06	0.00	0.00	224.06	0.00
P-n20-k2	233.05*	4.76	233.05	0.00	0.00	233.05	0.00	46.08	233.59	0.32	0.23	233.05	0.00	269.60	233.05	0.00	0.00	233.05	0.00

Table 8 continued

Instance	BKS	GRASP + HC						Mendoza and Villegas- MSH						Goodson et al. - SA					
		Average				Best		Average				Best		Average				Best	
		CPU	OF	CV	Gap	OF	Gap	CPU	OF	CV	Gap	OF	Gap	CPU	OF	CV	Gap	OF	Gap
P-n21-k2	218.96*	6.04	218.96	0.00	0.00	218.96	0.00	61.36	218.96	0.00	0.00	218.96	0.00	332.70	218.96	0.00	0.00	218.96	0.00
P-n22-k2	231.26*	7.10	231.26	0.00	0.00	231.26	0.00	81.34	231.26	0.00	0.00	231.26	0.00	352.30	231.26	0.00	0.00	231.26	0.00
P-n22-k8	681.06*	4.72	681.06	0.00	0.00	681.06	0.00	14.19	681.06	0.00	0.00	681.06	0.00	28.60	681.06	0.00	0.00	681.06	0.00
P-n23-k8	619.52*	5.45	619.53	0.00	0.00	619.53	0.00	14.98	619.59	0.01	0.01	619.53	0.00	21.10	619.57	0.02	0.01	619.53	0.00
P-n40-k5	472.50*	26.49	472.50	0.00	0.00	472.50	0.00	155.27	472.50	0.00	0.00	472.50	0.00	367.20	472.50	0.00	0.00	472.50	0.00
P-n45-k5	533.52*	36.25	533.83	0.07	0.06	533.52	0.00	283.84	534.52	0.00	0.19	534.52	0.19	603.80	537.13	0.78	0.68	533.52	0.00
P-n50-k10	758.76*	40.40	758.76	0.00	0.00	758.76	0.00	145.89	758.76	0.00	0.00	758.76	0.00	150.60	764.12	0.51	0.71	760.94	0.29
P-n50-k7	582.37*	44.17	582.37	0.00	0.00	582.37	0.00	248.88	586.36	0.16	0.69	585.05	0.46	343.00	584.95	0.40	0.44	582.37	0.00
P-n50-k8	669.23*	39.70	669.33	0.03	0.02	669.23	0.00	176.39	669.23	0.00	0.00	669.23	0.00	225.30	674.11	0.67	0.73	669.81	0.09
P-n51-k10	809.70*	52.72	809.70	0.00	0.00	809.70	0.00	151.07	810.01	0.02	0.04	809.71	0.00	159.90	816.95	0.52	0.90	812.74	0.38
P-n55-k10	742.41*	56.26	742.41	0.00	0.00	742.41	0.00	230.37	745.24	0.17	0.38	743.04	0.08	208.90	752.36	0.52	1.34	745.70	0.44
P-n55-k15	1,068.05*	72.10	1, 068.05	0.00	0.00	1, 068.05	0.00	174.07	1, 068.05	0.00	0.00	1, 068.05	0.00	117.00	1, 072.17	0.35	0.39	1, 068.05	0.00
P-n55-k7	588.56*	64.54	588.76	0.07	0.03	588.56	0.00	426.56	590.80	0.15	0.38	589.64	0.18	452.80	593.10	0.53	0.77	588.56	0.00
P-n60-k10	803.60*	73.36	803.60	0.00	0.00	803.60	0.00	289.00	804.42	0.11	0.10	803.60	0.00	296.10	810.22	0.31	0.82	804.24	0.08
P-n60-k15	1,085.49*	85.52	1, 085.49	0.00	0.00	1, 085.49	0.00	220.70	1, 085.49	0.00	0.00	1, 085.49	0.00	134.80	1, 098.11	0.63	1.16	1, 087.41	0.18
A-n60-k9	1,529.82	102.43	1, 530.08	0.05	0.02	1, 529.82	0.00	782.77	1, 529.83	0.00	0.00	1, 529.82	0.00	393.70	1, 535.35	0.48	0.36	1, 529.82	0.00
E-n51-k5	552.26	56.75	552.26	0.00	0.00	552.26	0.00	451.47	552.42	0.05	0.03	552.26	0.00	586.00	552.81	0.14	0.10	552.26	0.00
Max.		102.43		0.22	0.19		0.00	782.77		0.48	1.16		0.92	603.80		1.33	1.89		0.44
Min.		1.69		0.00	0.00		0.00	5.91		0.00	0.00		0.00	9.00		0.00	0.00		0.00
Avg.		36.09		0.02	0.02		0.00	180.78		0.08	0.18		0.07	268.66		0.32	0.35		0.04
Std.		27.08		0.04	0.04		0.00	146.41		0.12	0.26		0.17	134.42		0.31	0.44		0.10

BKS best known solution, proven optima are marked with *, *Average* mean value over 10 runs, *Best* best solution found, *CPU* running time in seconds, *OF* objective function value, *CV* Coefficient of variation of the objective function over 10 runs in %, *Gap* gap with respect to the BKS in %

Appendix 2: Detailed CPU times

See Table 9

Table 9 Average running times (in seconds) over ten runs of GRASP + HC for the different VRPSD-DC formulations

Instance	CC	PF		
		Linear	Piecewise	Quadratic
A-n32-k5	11.05	31.31	29.17	21.99
A-n33-k5	7.31	25.77	21.87	19.07
A-n33-k6	10.70	19.41	20.81	19.22
A-n34-k5	8.56	31.40	27.01	18.36
A-n36-k5	33.02	50.21	52.78	52.95
A-n37-k5	17.04	35.70	37.85	30.53
A-n37-k6	26.26	46.72	49.28	51.37
A-n38-k5	9.38	33.95	32.17	26.26
A-n39-k5	29.29	99.62	91.97	87.85
A-n39-k6	18.16	51.02	47.76	37.40
A-n44-k6	43.61	95.73	98.86	98.33
A-n45-k6	26.71	71.64	65.41	60.96
A-n45-k7	68.84	142.19	149.32	158.15
A-n46-k7	52.53	136.61	120.42	113.33
A-n48-k7	63.15	204.34	198.16	154.70
A-n53-k7	90.51	295.03	304.99	272.95
A-n54-k7	176.46	404.58	407.43	407.06
A-n55-k9	55.70	198.37	203.08	166.30
E-n22-k4	2.75	5.02	4.59	4.43
E-n33-k4	10.76	61.38	61.58	40.08
P-n19-k2	1.71	2.96	2.69	2.64
P-n20-k2	2.02	3.32	3.20	3.18
P-n21-k2	2.44	6.95	6.81	4.33
P-n22-k2	2.55	5.79	5.83	4.90
P-n22-k8	4.13	5.36	5.64	7.06
P-n23-k8	4.59	5.01	5.60	5.66
P-n40-k5	6.63	34.67	31.23	22.94
P-n45-k5	10.62	46.28	48.18	35.00
P-n50-k10	18.35	52.66	55.02	57.11
P-n50-k7	35.07	59.36	61.75	58.80
P-n50-k8	18.89	60.40	58.93	52.95
P-n51-k10	38.57	102.66	107.08	100.33
P-n55-k10	28.46	90.43	90.77	88.16
P-n55-k15	38.49	60.02	68.64	83.48
P-n55-k7	37.54	100.26	101.89	87.59
P-n60-k10	36.96	146.07	145.99	133.03

Table 9 continued

Instance	CC	PF		
		Linear	Piecewise	Quadratic
P-n60-k15	48.40	94.63	101.03	117.48
A-n60-k9	242.00	434.90	450.93	475.19
E-n51-k5	15.17	98.34	91.31	73.37
Max.	242.00	434.90	450.93	475.19
Min.	1.71	2.96	2.69	2.64
Avg.	34.73	88.46	88.90	83.45
SD	46.78	100.15	102.42	102.26

References

Ak, A., Erera, A.: A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. Transp. Sci. **41**(2), 222–237 (2007)

Bent, R., Van Hentenryck, P.: Scenario-based planning for partially dynamic vehicle routing with stochastic customers. Oper. Res. **52**(6), 977–987 (2004)

Bent, R., Van Hentenryck, P.: Waiting and relocation strategies in online stochastic vehicle routing. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07), pp. 1816–1821 (2007)

Bianchi, L., Birattari, M., Chiarandini, M., Manfrin, M., Mastrolilli, M., Paquete, L., Rossi-Doria, O., Schiavinotto, T.: Metaheuristics for the vehicle routing problem with stochastic demands. Parallel Problem Solving from Nature—PPSN VIII. Lecture Notes in Computer Science, pp. 450–460. Springer, Berlin (2004)

Christiansen, C., Lysgaard, J.: A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. Oper. Res. Lett. **35**(6), 773–781 (2007)

Contardo, C., Cordeau, J.F., Gendron, B.: A GRASP + ILP-based metaheuristic for the capacitated location-routing problem. J. Heuristics **20**(1), 1–38 (2014)

Cordeau, J.F., Laporte, G., Savelsbergh, M., Vigo, D.: Vehicle routing. In: Barnhart, C., Laporte, G. (eds.) Handbooks in Operations Research and Management Science: Transportation, vol. 14, pp. 367–428. Elsevier, Amsterdam (2006)

Erera, A., Morales, J.C., Savelsbergh, M.: The vehicle routing problem with stochastic demands and duration constraints. Transp. Sci. **44**(4), 474–492 (2010)

Gauvin, C.: Un algorithme de génération de colonnes pour le problème de tournées de véhicules avec demandes stochastiques. Master's thesis, École Polytechnique de Montréal (2012)

Gendreau, M., Laporte, G., Séguin, R.: A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. Oper. Res. **44**(3), 469–477 (1996b)

Goodson, J.C., Ohlmann, J.W., Thomas, B.W.: Cyclic-order neighborhoods with application to the vehicle routing problem with stochastic demand. Eur. J. Oper. Res. **227**(2), 312–323 (2012)

Goodson, J.C., Ohlmann, J.W., Thomas, B.W.: Rollout policies for dynamic solutions to the multivehicle routing problem with stochastic demand and duration limits. Oper. Res. **61**(1), 138–154 (2013)

Goodson, J.C., Thomas, B.W., Ohlmann, J.W.: Restocking-based rollout policies for the vehicle routing problem with stochastic demand and duration limits. To appear in Transportation Science

Hansen, P., Mladenović, N., Moreno-Pérez, J.: Variable neighbourhood search: Methods and applications. 4OR: A Quart. J. Oper. Res. **6**, 319–360 (2008)

Haugland, D., Ho, S., Laporte, G.: Designing delivery districts for the vehicle routing problem with stochastic demands. Eur. J. Oper. Res. **180**(3), 997–1010 (2007)

Mendoza, J.E., Castanier, B., Guéret, C., Medaglia, A.L., Velasco, N.: A simulation-based MOEA for the multi-compartment vehicle routing problem with stochastic demands. In: Proceedings of the VIII Metaheuristics International Conference (MIC). Hamburg, Germany (2009)

Mendoza, J.E., Castanier, B., Guéret, C., Medaglia, A.L., Velasco, N.: A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. Comput. Oper. Res. **37**(11), 1886–1898 (2010)

Mendoza, J.E., Castanier, B., Guéret, C., Medaglia, A.L., Velasco, N.: Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. Transp. Sci. **45**(3), 335–345 (2011)

Mendoza, J.E., Guéret, C., Hoskins, M., Lobit, H., Pillac, V., Vidal, T., Vigo, D.: VRP-REP: the vehicle routing community repository. In: Third Meeting of the EURO Working Group on Vehicle Routing and Logistics Optimization (VeRoLog). Oslo, Norway (2014)

Mendoza, J.E., Villegas, J.G.: A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Optim. Lett. **7**(7), 1503–1516 (2013)

Novoa, C., Berger, R., Linderoth, J., Storer, R.: A set-partitioning-based model for the stochastic vehicle routing problem. Texas State University, Tech. rep. (2006)

Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L.: A review of dynamic vehicle routing problems. Eur. J. Oper. Res. **225**(1), 1–11 (2013a)

Pillac, V., Guret, C., Medaglia, A.L.: A parallel matheuristic for the technician routing and scheduling problem. Optim. Lett. **7**(7), 1525–1535 (2013b)

Prins, C.: A simple and effective evolutionary algorithm for the vehicle routing problem. Comput. Oper. Res. **31**(12), 1985–2002 (2004)

Rosing, K.E., Revelle, C.S.: Heuristic concentration: two stage solution construction. Eur. J. Oper. Res. **17**(96), 75–86 (1997)

Savelsbergh, M., Goetschalckx, M.: A comparison of the efficiency of fixed versus variable vehicle routes. J. Bus. Logist. **16**, 163–187 (1995)

Secomandi, N., Margot, F.: Reoptimization approaches for the vehicle-routing problem with stochastic demands. Oper. Res. **57**(1), 214–230 (2009)

Sörensen, K., Sevaux, M.: MA|PM: memetic algorithms with population management. Comput. Oper. Res. **33**(5), 1214–1225 (2006)

Sörensen, K., Sevaux, M.: A practical approach for robust and flexible vehicle routing using metaheuristics and Monte Carlo sampling. J. Math. Model. Algorithms **8**(4), 387–407 (2009)

Subramanian, A., Uchoa, E., Ochi, L.S.: A hybrid algorithm for a class of vehicle routing problems. Comput. Oper. Res. **40**(10), 2519–2531 (2013)

Tan, K.C., Cheong, C.Y., Goh, C.K.: Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. Eur. J. Oper. Res. **177**(2), 813–839 (2007)

Teodorović, D., Pavković, G.: A simulated annealing technique approach to the vehicle routing problem in the case of stochastic demands. Transp. Plan. Technol. **16**(4), 261–273 (1992)

Tricoire, B.: Optimisation dans les réseaux logistiques: du terrain à la prospective. PhD thesis, Université d'Angers (France) (2013)

Villegas, J.G., Prins, C., Prodhon, C., Medaglia, A.L., Velasco, N.: A matheuristic for the truck and trailer routing problem. Eur. J. Oper. Res. **230**(2), 231–244 (2013)

Yang, W.H., Mathur, K., Ballou, R.: Stochastic vehicle routing with restocking. Transp. Sci. **34**(1), 99–112 (2000)