EURO J Transp Logist (2018) 7:223–251
https://doi.org/10.1007/s13676-016-0101-4

RESEARCH PAPER

# The vehicle routing problem with hard time windows and stochastic service times

F. Errico[1,2] · G. Desaulniers[3,4] · M. Gendreau[4,5] · W. Rei[5,6] · L.-M. Rousseau[4,5]

Received: 11 November 2015 / Accepted: 12 November 2016 / Published online: 30 November 2016

**Abstract** In this paper we consider the vehicle routing problem with hard time windows and stochastic service times (VRPTW-ST); in this variant of the classic VRPTW the service times are random variables. In particular, given a set of vehicle routes, some of the actual service times might not lead to a feasible solution, given the customer time windows. We consider a chance-constrained program to model the VRPTW-ST and provide a new set partitioning formulation that includes a constraint on the minimum success probability of the set of vehicle routes. Under some mild conditions, we develop a method to exactly compute the success probability of the routes. We then solve the VRPTW-ST by a branch-price-and-cut algorithm, where the main challenges are in the solution of the subproblems of the column generation procedure. We adapt the dynamic programming algorithm to account for the probabilistic resource consumption by extending the label dimension and by providing new dominance rules. Extensive computational experiments prove the effectiveness of both the solution method and the stochastic model.

✉ F. Errico
fausto.errico@cirrelt.ca

1 CIRRELT Interuniversity Research Center on Enterprise Networks, Logistics and Transportation, GERAD Group for Research in Decision Analysis, Montreal, Canada

2 Département de génie de la construction, École de Technologie Supérieure de Montréal, Montreal, Canada

3 GERAD Group for Research in Decision Analysis, Montreal, Canada

4 Département de mathématiques et génie industriel, École Polytechnique de Montréal, Montreal, Canada

5 CIRRELT Interuniversity Research Center on Enterprise Networks, Logistics and Transportation, Montreal, Canada

6 Département de management et de technologie, Université du Québec à Montréal, Montreal, Canada

**Keywords** Vehicle routing problem · Service time · Stochastic programming · Chance constraint · Column generation

## 1 Introduction

In this paper, we consider the vehicle routing problem with hard time windows and stochastic service times (VRPTW-ST) which is a variant of the classical vehicle routing problem where customers' service times are random variables and hard constraints are enforced on the time window limits. In this context, the service to a customer starts within the given time window and vehicles are not allowed to arrive after the end of the time window. Vehicles are, however, allowed to arrive before the beginning of a time window. In this case they must postpone the beginning of the service until the time window opens. The central point in this problem is to plan a set of routes before the time required to serve each customer is known, and such that customers' time window constraints are satisfied. Our goal is to develop an exact method for the VRPTW-ST.

The vehicle routing problem (VRP) with deterministic data has been widely studied and applied in many real-world situations. However, usually not all the problem data are available in advance. This has led to an increased interest in probabilistic VRP (for a recent review, see Gendreau et al. 2014). The most common sources of uncertainty considered in the literature are: (1) demand volumes (see for example Bertsimas 1992; Laporte et al. 2002), (2) the presence of customers (see for example Gendreau et al. 1995), and (3) stochastic travel and/or service times (see for example Laporte et al. 1992; Kenyon and Morton 2003; Lei et al. 2012; Taş et al. 2012; Adulyasak and Jaillet 2014). There are three main modeling strategies for stochastic problems: (a) stochastic programs with recourse, where "a priori" solution is provided together with a modification strategy (recourse) to be applied in a second stage, once all the uncertainty is disclosed, (b) chance-constrained programs where a priori plan is provided together with a bound on the probability that the given plan will be feasible once all the uncertainty is disclosed and (c) multi-stage dynamic reoptimization, where the problem is reoptimized as new information becomes available.

The VRPTW-ST has the third type of uncertainty (service times). In most of the related literature, customer time windows are absent or soft, or a maximal route duration is considered. With respect to these studies, the VRPTW-ST is structurally different and more challenging because hard time windows imply more complex relations between the probability distributions of vehicle arrival times at consecutive customers, thus requiring new methodological development.

The problem tackled in this paper corresponds to the situation faced by several companies who provide technicians to repair their equipment in industrial settings. An example of this situation can be found in the paper of Cortés et al. (2014), who describe a collaboration project with Xerox in Santiago, Chile. In that paper, the authors consider the problem of routing the technicians who are responsible for repairing office machines that fail, according to individual Service Level

Agreements, which are specified in terms of a time window for the arrival of the technician.

In Cortés et al. (2014), deterministic service times were considered. However, in practice, some time windows cannot be met due to the uncertainty of the service times. The stochastic version of this problem can be tackled through a two-stage stochastic program with recourse, as we did in a companion paper (Errico et al. 2015) by considering different possible recourse strategies. The problem might also be tackled using robust optimization, as was done by (Souyris et al. 2013).

There are some intangible costs paid by the company in terms of image, for arriving late too often. While a stochastic program with recourse can indirectly decrease the failure probability of the a priori plan by increasing the penalty cost associated with the recourse actions, a chance-constrained model is more suited for a direct control of the failure probability. In this paper, we thus chose to investigate a pure chance-constrained approach to enforce a minimum probability that the planned routes are successful. This kind of approach is typical, for example, of call centre management policies, where the goal is to schedule the agents such that the calls are answered in a reasonable delay (typically 20 s) 80% of the time. The idea of success rate can also be used to enforce that the drivers' route runs smoothly most of the time. By bounding the probability that routes miss a Service Level Agreement, the company makes sure that, in the long run, the drivers do not have to deal too often with unsatisfied customers.

In its chance-constrained version, the VRPTW-ST can be described as follows. Consider a directed graph $G = (V, A)$ where $V = \{0, 1, \ldots, n\}$ is the node set, and $A = \{(i,j) | i,j \in V\}$ is the arc set. Node 0 represents a depot where a fleet of homogeneous vehicles is initially located, and $V_c = \{1, \ldots, n\}$ is the customer set. A time window $[a_i, b_i]$ and a stochastic service time are associated with each customer $i \in V_c$. The service time probability distributions are assumed to be known and mutually independent. A non-negative travel cost $c_{ij}$ and travel time $t_{ij}$ are associated with each arc $(i,j) \in A$. Furthermore, a global "reliability" threshold $0 < \alpha < 1$ is given.

The VRPTW-ST consists of finding a set of vehicle routes (a route plan) such that: (1) The routes start and end at node 0. (2) All the customers are served. (3) The service to a customer must start within the given time window. As previously mentioned, however, vehicles are allowed to arrive before the beginning of a time window. (4) The global probability that the route plan will be feasible with respect to the time windows when realizations of customers' service times become known, is greater than the reliability threshold. (5) The travel distance is minimized. For convenience, we use the expression "success probability" to indicate the probability that a given route will be feasible with respect to time windows when the realization of the customers' service times become known. This allows us to rephrase (4) as: the global success probability of the route plan is greater than the reliability threshold.

The VRPTW-ST finds application in several practical situations such as the dispatch of technicians or repairmen to perform either installation, maintenance or repair operations. On the one hand, customers might specify hard time windows dictated by their specific needs. On the other hand, the service times at customers vary according to different factors that are inherently stochastic: accessibility at the customer's location (e.g., parking conditions), diagnostic of the particular service to perform (e.g., identification of the problems in case of a repair), complexity of the operation to carry out, etc. The proposed chance-constrained model allows the dispatching company to design routes in such a way that the probability of violating a customer time window is bounded by the reliability threshold, which could be determined at the manager's discretion. In the long term, this allows the company to maintain a given service quality to its customers, and be able to quantify the corresponding requirements in terms of the number of needed vehicles/repairmen, route costs, etc.

We provide a set partitioning formulation of the VRPTW-ST that includes a constraint on the minimum total success probability of the route plan. This constraint is linearized by applying the properties of logarithms. It should be noted that the computation of the success probability for a route requires knowledge of the vehicle arrival-time probability distribution at the customer locations. However, the presence of hard time windows generally implies the truncation of these distributions, thus generally preventing the straightforward application of convolution properties when summing the random variables. For these calculations we propose an exact method valid for any discrete service-time distribution with finite support.

Given the success of column-generation-based approaches for the deterministic VRPTW, we decided to implement a branch-price-and-cut algorithm for the VRPTW-ST. With respect to the deterministic VRPTW, the major challenges are in the solution of the subproblems in the column generation algorithm. The resource consumption is probabilistic, and we propose a label-setting algorithm where the label dimension is suitably extended to account for the probabilistic constraint. Furthermore, to reduce the number of labels, we deploy new heuristic and exact dominance rules. We also implement several algorithmic improvements and in particular a tabu search column generator based on the work of Desaulniers et al. (2008).

We perform extensive computational tests. We first build a set of benchmark instances derived from Solomon's database (1987) and consider several service-time distributions and reliability thresholds. We then perform two sets of experiments. The first tests our algorithm over the whole instance set and compares the algorithmic efficiency for different problem types and characteristics. The second analyzes the model behavior in terms of cost and solution quality; it compares several stochastic solutions with deterministic solutions based on the median and worst-case values of the service-time probability distributions. The results show that our method is generally effective and able to solve instances with up to 50 customers exactly. Furthermore, the results underline the advantages of the stochastic model over its deterministic counterparts.

The rest of the paper is organized as follows. In Sect. 2 we provide a review of the related literature. In Sect. 3 we formulate the VRPTW-ST as a set partitioning problem with a probabilistic constraint and show how to compute the success probability of a given route. In Sect. 4 we detail our branch-price-and-cut algorithm. In Sect. 5 we report our computational experiments before concluding in Sect. 6.

## 2 Related literature

In this section, we survey research on the VRP with stochastic service and/or travel times. We also include some research on the traveling salesman problem (TSP) with stochastic times and hard time windows, as well as work considering different sources of uncertainty in combination with hard time constraints. This survey focuses on stochastic programming approaches and excludes dynamic re-optimization approaches.

The work by Laporte et al. (1992) is one of the first addressing travel and service time uncertainty. The authors consider a non-capacitated version of the VRP with no time-window constraints. However, a soft maximal route duration is considered. The authors propose one model based on chance constraints and two models with recourse where the recourse action is the payment of a penality if the maximal route duration is violated. The results obtained by a branch-and-cut algorithm on instances involving a limited number of scenarios are reported. Lambert et al. (1993) present a variant of this problem considering stochastic travel times, customer time windows, and a penalty dependent on the vehicle's load. The authors consider worst-case scenarios to impose the time-window constraints. The problem is solved by adapting the Clarke and Wright heuristic.

Kenyon and Morton (2003) focus on a version of the VRP with stochastic travel and service times and no capacity or time-window constraints. The authors study two versions of the problem: one minimizes the expected completion time, and the other minimizes the probability that the completion time exceeds a given threshold. The authors propose an exact branch-and-cut procedure suitable for problems with small sample spaces, and a combination of Monte Carlo simulation and a branch-and-cut procedure for problems with larger sample spaces.

Wang and Regan (2001) consider the VRP with soft time windows, no capacity constraints, and stochastic travel times. If a customer is visited after its time-window deadline, the service is skipped, but not the visit itself. The goal is to maximize the number of customers serviced and to minimize the routing costs. The authors develop an analytical model and algorithm for the evaluation of the expected load of a route; the computational complexity is $O(2^n)$, where $n$ is the number of customers. A model with a constraint on the minimum expected load is also analyzed. No computational results are provided. Sungur et al. (2010) investigate a non-capacitated version of the VRP with uncertainty in the customer presence and the service times. Soft customer time windows and hard route-duration limits are considered. The authors aim to produce a monthly route plan that can be modified daily according to the actual demand. The objective is to maximize the number of customers served, minimize the total vehicle times, minimize penalties for early or late arrivals at customers, and maximize a measure of route *consistency*. The modeling approach is based on a combination of robust optimization and stochastic programming with recourse. The solution method is a two-phase heuristic.

Li et al. (2010) consider a capacitated vehicle routing problem with soft time windows, soft route-duration limits, and stochastic service and travel times. The authors propose two formulations: the first considers a chance constraint for each time window and route-duration limit, and the second uses a recourse consisting of the payment of penalties for violated time constraints. The authors develop a tabu-search-based heuristic for the solution of both problems; the probability evaluation of the solutions is obtained by Monte Carlo simulation. A similar problem is investigated in Lei et al. (2012). The recourse action is the payment of a penalty when the route-duration limit is violated. The objective is the minimization of the sum of the travel costs, the expected service cost, and the expected recourse cost. The authors provide a closed-form expression of the expected route cost in the case of normally distributed service times, and they develop a generalized variable neighborhood search (GVNS) algorithm for the problem. GVNS is shown to perform better than the variable neighborhood descent and the variable neighborhood search heuristics. A similar problem is considered in Taş et al. (2012), where the objective is the minimization of the combination of customer inconvenience, i.e., the expected earliness and lateness at customers, and operational costs, i.e., the expected driver overtime, vehicle costs, and travel costs. The authors propose a three-phase tabu-search-based algorithm. For a related problem, a branch-and-price solution approach is developed in Taş et al. (2014).

More recently, Adulyasak and Jaillet (2014) focus on the VRP with capacity constraints, customer deadlines and stochastic travel times. They consider two modeling approaches: a stochastic program assuming exact knowledge of the probability distributions and minimizing the sum of the probabilities of deadline violations, and a robust approach, where probability distributions are partially unknown, minimizing a suitably defined lateness index. The authors develop branch-and-cut based algorithms and perform extensive computational analyses.

A robust optimization approach for the VRP with travel and demand uncertainty and customer deadlines is presented in Lee et al. (2012). The authors develop a method based on column generation. A comparison of the robust and deterministic solutions is carried out by performing a Monte Carlo simulation. The results show that the robust solutions are more expensive but also more reliable.

The TSP with hard time windows and stochastic travel and service times is considered in Jula et al. (2006). In their setting, the aim is to find the least-cost tour such that for each customer the success tour probability is greater than a given value. The authors develop a heuristic algorithm based on dynamic programming and on an estimation of the mean and variance of the arrival-time distribution at the customers. Similarly, assuming normally distributed time events and negligible late arrival times at the customers, Chang et al. (2009) provide an approximation of the mean and variance of the arrival time at the customers and propose a dynamic programming algorithm.

Campbell and Thomas (2008) investigate the TSP with customer deadlines and uncertainty about the customer presence. The authors study three different approaches: the first has as a recourse strategy the payment of a penalty in the event of late arrival, the second also skips the corresponding customer, and the third imposes a probabilistic constraint. The paper focuses on the modeling aspects and the calculation of the expected costs. A simple heuristic approach is proposed and experiments comparing the deterministic and stochastic solutions are performed.

With respect to the reviewed literature, our approach has several novel features. The VRPTW-ST generalizes the VRP with a combination of stochastic service times and hard time windows. This setting is much more challenging than the corresponding soft case. Given a route, the arrival-time probability distributions at the customers have to be suitably truncated because of the time windows, and this prevents the straightforward application of convolution, as well as stochastic dominance properties (Wellman et al. 1995), when summing and comparing the random variables. Jula et al. (2006) and Chang et al. (2009) consider a similar setting but focus on the TSP case. Furthermore, they propose heuristic approaches. In this paper, we propose an exact procedure for the calculation of the arrival-time probability distributions at the customers, we prove that stochastic dominance holds for the VRPTW-ST, and we develop an exact solution framework based on column generation. In Errico et al. (2015), some of the material presented in this paper is further developed to address the VRPTW-ST as a stochastic program with recourse. Two recourse strategies are proposed and the resulting problems are solved by branch-price-and-cut algorithms. Finally, a computational comparison between recourses is performed.

## 3 The VRPTW-ST

In Sect. 3.1 we formally introduce the chance-constrained formulation for the VRPTW-ST. In Sect. 3.2 we show how to compute the success probability of a given route.

### 3.1 Set partitioning formulation with a probabilistic constraint

Consider a route $r$ defined as a sequence of nodes $r = (v_0, v_1, \ldots, v_q, v_{q+1})$ where $v_1, \ldots, v_q \in V_c$ and $v_0$ and $v_{q+1}$ represent the depot 0, and let $\mathcal{R}$ be the set of all possible routes. Let $a_{ir}$ be a parameter with value 1 if route $r$ visits customer $i$ and 0 otherwise. The cost associated with a route $r$ is $c_r = \sum_{i=0}^{q} c_{v_i,v_{i+1}}$. Given binary variables $x_r$ with value 1 if route $r \in \mathcal{R}$ is chosen and 0 otherwise, the VRPTW-ST can be formulated as follows:

$$\min \sum_{r \in \mathcal{R}} c_r x_r \tag{1}$$

$$s.t. \sum_{r \in \mathcal{R}} a_{ir} x_r = 1 \quad \forall i \in V_c \tag{2}$$

$$\Pr\{\text{All chosen routes are successful}\} \geq \alpha \tag{3}$$

$$x_r \in \{0, 1\} \quad \forall r \in \mathcal{R}, \tag{4}$$

where the objective function (1) minimizes the total travel cost. Constraints (2) ensure that all customers are visited exactly once. Constraint (3) is the probabilistic constraint; it ensures that the success probability of the overall plan is greater than the reliability threshold. Constraints (4) ensure that the solution vector is binary. The assumption that the service times are mutually independent implies the property stated in the following proposition.

**Proposition 3.1** Let $\mathcal{R}'$ denote a set of routes inducing a proper partition of the customer set $V_c$. Given any two routes $r_1, r_2 \in \mathcal{R}'$, the success probability of $r_1$ is independent of the success probability of $r_2$.

Proposition 3.1 allows us to rewrite constraint (3):

$$\prod_{r \in \mathcal{R}: x_r = 1} \Pr\{\text{Router is successful}\} \geq \alpha. \tag{5}$$

By the properties of logs we have

$$\sum_{r \in \mathcal{R}: x_r = 1} \ln(\Pr\{\text{Router is successful}\}) \geq \ln(\alpha), \tag{6}$$

and extending the summation to all $\mathcal{R}$ we have

$$\sum_{r \in \mathcal{R}} x_r \ln(\Pr\{\text{Router is successful}\}) \geq \ln(\alpha). \tag{7}$$

Substituting

$$\beta_r := -\ln(\Pr\{\text{Router is successful}\}) \tag{8}$$

and $\beta := -ln(\alpha)$, we can write (3) as

$$\sum_{r \in \mathcal{R}} \beta_r x_r \leq \beta. \tag{9}$$

## 3.2 Computing the route success probability

In this section, assuming a discrete customer service-time probability distribution with finite support, we show that the route success probability is strictly linked to the vehicle arrival-time probability distribution at the customers, and we compute it recursively.

Consider a route $r = (v_0, \ldots, v_q, v_{q+1})$ where $v_0$ and $v_{q+1}$ represent the depot 0, and let $r_{v_i} = (v_0, \ldots, v_i)$, $1 \leq i \leq q$, be the subroutes of $r$ starting at the origin and ending at node $v_i$. Let $t_{v_i}$ and $s_{v_i}$ be random variables representing the vehicle arrival time and service time, respectively, at client $v_i$, and let $d_{v_i}$ be the probability mass function of the service time duration at client $v_i$. Functions $d_{v_i}$ are assumed to have discrete, finite supports.

Recalling that the depot does not have time-window restrictions, we can write:

$$\Pr\{r \text{ is successful}\} = \Pr\left(\bigwedge_{i=1}^{q}\{\mathrm{r}_{\mathrm{v_i}} \text{ is successful}\}\right) = \Pr\left(\bigwedge_{i=1}^{q}\{\mathrm{t}_{\mathrm{v_i}} \leq \mathrm{b}_{\mathrm{v_i}}\}\right). \tag{10}$$

Equivalently,

$$\Pr\{r \text{ is successful}\} = \Pr(\{\mathrm{t}_{\mathrm{v_q}} \leq \mathrm{b}_{\mathrm{v_q}}\} \wedge \{\mathrm{r}_{\mathrm{v_{q-1}}} \text{ is successful}\}) \tag{11}$$

Let $m_{v_i}(z)$ be the probability mass function associated with the event $t_{v_i} = z$ and the event that $r_{v_{i-1}}$ is successful, i.e.,

$$m_{v_i}(z) := \Pr(\{t_{v_i} = z\} \wedge \{r_{v_{i-1}} \text{is successful}\}). \tag{12}$$

Then we can write

$$\Pr\{r \text{ is successful}\} = \sum_{z \leq b_{v_q}} m_{v_q}(z). \tag{13}$$

Equation (13) shows that the computation of the success probability of a given route requires the computation of the mass probability function of the vehicle arrival time at the last visited customer. We now show how to do this recursively.

To correctly account for the time-window restrictions, and considering that the vehicle is never allowed to arrive later than $b_{v_i}$, we introduce random variables $\bar{t}_{v_i}$ with values in the interval $[a_{v_i}, b_{v_i}]$, defined as

$$\bar{t}_{v_i} = \begin{cases} a_{v_i} & \text{if } \mathrm{t}_{\mathrm{v_i}} < \mathrm{a}_{\mathrm{v_i}} \\ t_{v_i} & \text{if } \mathrm{a}_{\mathrm{v_i}} \leq \mathrm{t}_{\mathrm{v_i}} \leq \mathrm{b}_{\mathrm{v_i}}, \end{cases} \tag{14}$$

and the corresponding probability mass function $\bar{m}_{v_i}(z)$

$$\bar{m}_{v_i}(z) = \begin{cases} 0 & \text{if } \mathrm{z} < \mathrm{a}_{\mathrm{v_i}} \\ \sum_{l \leq a_{v_i}} m_{v_i}(l) & \text{if } \mathrm{z} = \mathrm{a}_{\mathrm{v_i}} \\ m_{v_i}(z) & \text{if } \mathrm{a}_{\mathrm{v_i}} < \mathrm{z} \leq \mathrm{b}_{\mathrm{v_i}} \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Note that:

$$t_{v_i} = \bar{t}_{v_{i-1}} + s_{v_{i-1}} + t_{v_{i-1},v_i}. \tag{16}$$

We now substitute the above definitions into (12) and use the independence of $t_i$ and $s_i$. This allows us to write distribution $m_{v_i}$ in terms of $\bar{m}_{v_{i-1}}$ and $d_{v_{i-1}}$:

$$m_{v_i}(z) = \Pr(\{\bar{t}_{v_{i-1}} + s_{v_{i-1}} + t_{v_{i-1},v_i} = z\} \wedge \{r_{v_{i-1}} \text{is successful}\}) \tag{17}$$

$$= \sum_{k \in \mathbb{N}} d_{v_{i-1}}(k) \bar{m}_{v_{i-1}}(z - t_{v_{i-1},v_i} - k). \tag{18}$$

Substituting the above expression into (13), we can write:

$$\Pr\{r \text{ is successful}\} = \sum_{z \le b_{v_q}} \sum_{k \in \mathbb{N}} d_{v_{q-1}}(k) \bar{m}_{v_{q-1}}(z - t_{v_{q-1},v_q} - k), \tag{19}$$

Considering

$$\bar{m}_{v_1}(z) = \begin{cases} 1 & \text{if } \mathrm{z} = \max\{\mathrm{a}_{\mathrm{v}_1}, \mathrm{t}_{0,\mathrm{v}_1}\}, \\ 0 & \text{otherwise}, \end{cases} \tag{20}$$

expression (19) can be recursively applied to obtain the route success probability.

Observe that, for ease of presentation, we sometimes allow summations to range in $\mathbb{N}$ (e.g., in (18), (19) and other expressions that can be found in the rest of the paper). In practical computations, the summation indexes are restricted to values compatible with the support of the functions involved.

As a final observation, we note that our development is general and independent of the given discrete probability distribution.

### 3.3 Limitations and extensions

The target application of model (1), (2), (9) and (4) is the dispatch of technicians or repairmen to perform maintenance or repair operations. In this context, the vehicle capacity is not a primary concern. For this reason, in the rest of the paper we do not account for vehicle capacity constraints. As it will be pointed out in Sect. 4.1.2, however, our methodology can be trivially extended to enforce the capacity restrictions.

Similarly, we focus on stochastic service times (repair times are unknown), while travel times are assumed to be deterministic. We observe that our methodology can be easily generalized to account for stochastic travel times, as far as the same assumptions made for service times hold (discrete random variables with finite support and independent). In fact, the main difference is in relation (16), were $t_{v_{i-1},v_i}$ would be a random variable, thus the probability distribution of $t_{v_i}$ can be obtained by applying one more convolution step.

Finally, our approach can be generalized to the case of repairman-specific service time probability distributions by considering one set of variables for each repairman in the model (1), (2), (9) and (4). This modification does not change the structure of the problem, and the proposed method solution still applies. However, instead of solving one subproblem at each iteration of the column generation algorithm (see Sect. 4.1.2), several subproblems need to be solved, one for each repairman.

## 4 Branch-price-and-cut algorithm for the VRPTW-ST

Branch-price-and-cut is an algorithm based on implicit enumeration. Lower bounds at each node of the search tree are computed by applying column generation to the linear relaxation of the original problem, which corresponds, for the VRPTW-ST, to (1), (2), and (9), as well as the nonnegativity requirements on the $x_r$ variables. The lower bounds are then tightened by the dynamic generation of valid inequalities.

In Sect. 4.1 we provide the details of our column generation procedure, and in Sect. 4.2 we propose several accelerating strategies. The cutting planes used in our algorithm are described in Sect. 4.3. In Sect. 4.4 we describe the branching strategies.

## 4.1 Column generation

At a generic node of the enumeration scheme we need to solve the linear program (1), (2), (9), with the nonnegativity requirements on the $x_r$ variables, and augmented by the applicable branching decisions and cutting planes. This problem has an extremely large number of variables and cannot be explicitly solved in this form, except for very small instances. Instead, the column generation algorithm (for details, see Lübbecke and Desrosiers 2005) iteratively solves a restricted master problem (RMP) that considers only a subset of the original variables (routes). When a solution $\bar{x}$ of the current RMP has been obtained by applying the primal simplex algorithm, a subproblem verifies if the solution is optimal. If it is not, new routes are added to the RMP and the process iterates.

The subproblem verifies the optimality of the RMP solution by looking for routes with negative reduced costs. If we associate dual multipliers $\gamma_i \geq 0$ and $\delta \geq 0$ with constraints (2) and (9) respectively, the reduced cost $\bar{c}_r$ of a given route $r \in \bar{\mathcal{R}}$ can be expressed as

$$\bar{c}_r := c_r - \sum_{i \in V_c} a_{ir}\gamma_i + \beta_r \delta. \tag{21}$$

Hence, the subproblem minimizes expression (21) over the set $\mathcal{R}$ of feasible routes. Note that possible incompatibilities among routes (e.g., total success probability less than the threshold, customers visited more than once) are handled in the RMP directly.

The resulting subproblem belongs to the family of elementary shortest path problems with resource constraints (ESPPRC) and is usually solved by dynamic programming. In contrast to previous work, the probabilistic constraint (9) implies that the consumption of one of the resources (the time) is probabilistic. This leads to questions about label choices, extension functions, and the dominance rule adopted in the dynamic programming algorithm. These issues are addressed in the next sections.

### *4.1.1 Classic shortest path problems with resource constraints*

The ESPPRC (see Irnich and Desaulniers 2005) is generally solved by dynamic programming in the form of labeling algorithms. These methods have been considerably improved by several authors (see Feillet et al. 2004; Righini and Salani 2008, for example). The main concepts of the labeling algorithm are as follows. A feasible partial route starting from the depot 0 and ending at any node $i \in V_c$ is implicitly represented by a label associated with node $i$. A label $E = (C, T, L, V^1, \ldots, V^n)$ typically has one component $C$ representing the reduced cost of the partial path and several resource components. A resource is a quantity that accumulates along a route, and at each node its value is restricted within a so-called *resource window*. For the VRPTW, there is one resource $T$ for the time, one resource $L$ for the current load, and $n$ additional resources $V^i$ for each $i \in V_c$, accounting for the path elementarity. In particular, as proposed by Feillet et al. (2004), $V^i$ takes the value 1 if either customer $i$ has already been visited or it cannot be visited because of other resource restrictions.

Starting from an initial label associated with the depot 0, the algorithm extends labels along a route using *extension functions*. For example, consider the extension from node $i$ to $j$, where $i, j \in V_c$. Let $C_i$ and $T_i$ be the cost and time components of label $E_i$ associated with $i$, and let $[a_j, b_j]$ be the time-window restriction at node $j$. Extending $E_i$ along arc $(i, j)$ produces a new label $E_j$ with cost component $C_j = g_{ij}^C(C_i)$ and time component $T_j = g_{ij}^T(T_i)$, where $g_{ij}^C(C)$ and $g_{ij}^T(T)$ are resource extension functions defined as $g_{ij}^C(C) = C + \bar{c}_{ij}$ and $g_{ij}^T(T) = \max\{a_j, T + t_{ij} + \tilde{t}_i^s\}$, where $\tilde{t}_i^s$ denotes the deterministic service time at customer $i$. A new label is feasible and not discarded by the algorithm only if all the resource windows are satisfied.

From the computational-efficiency point of view, the concept of *dominance* is extremely important because it allows us to limit the number of labels by discarding non-useful paths. Consider two partial routes $r^1$ and $r^2$ ending at the same node $i$ and represented by labels $E_i^1$ and $E_i^2$, respectively. Generally speaking, label $E_i^2$ is said to be dominated by $E_i^1$ if

1) Any feasible extension $e$ of $r^2$ ending at a generic node $j$ is also feasible for $r^1$;
2) For any such extension $e$, the inequality $C_j^1 \leq C_j^2$ holds, where $C_j^l$ is the reduced cost of the route obtained by extending route $r^l$, $l = 1, 2$.

In the VRPTW, because of the properties of the extension functions, it can be proved that a sufficient condition for dominance is that each component in $E_i^1$ is not larger than the corresponding component in $E_i^2$.

### *4.1.2 Shortest path problems with probabilistic resource consumption*

The above label components, extension function, and dominance rules cannot be applied to the VRPTW-ST, but the main algorithmic structure can still be used.

*Label choice* The first difference is that in the VRPTW-ST we disregard capacity constraints. We, therefore, omit the load component in our labels. As mentioned in Sect. 3.3, had we wanted to enforce capacity constraints, we would have kept this component. The second main difference is that in the VRPTW-ST the customer service times, and consequently the vehicle arrival times at customers, are random variables. Furthermore, constraint (9) is a probabilistic constraint. To handle this constraint we could consider the route success probability as a resource. It would then be possible to replace the label component $T$ of the VRPTW by a new label component $P := \Pr\{r \text{ is successful}\}$ representing the total success probability of the route. However, according to the recursion in Sect. 3.2, the computation of the success probability at a node of a route requires the probability mass function of the arrival time at its predecessor. In other words, when we extend a path with arc $(i,j) \in A$, it is not possible to express $P_j$ as a function $g^P_{ij}(P_i)$ depending on $P_i$ only, as it was for the label component $T$ in the VRPTW case. A similar issue arises for the extension of the reduced cost, given its dependence on $\beta_r$, which in turn depends on the success probability. Hence, to completely represent a path by a label in the context of the VRPTW-ST, we need more components. We, therefore, represent a feasible partial path from node 0 to $i \in V_c$ by a label with the following $2+n+b_i-a_i$ components:

- One component $C_i$ represents the reduced cost.
- $n$ components $V_i^1, \ldots, V_i^n$ represent visited or unreachable nodes.
- $b_i - a_i + 1$ components $\bar{M}_i(a_i), \ldots, \bar{M}_i(b_i)$, one for each $z \in \mathbb{N}$ and $\mathrm{z} \in [\mathrm{a_i}, \mathrm{b_i}]$, represent the arrival distribution, where we define

$$\bar{M}_i(t) := \sum_{l \le t} \bar{m}_i(l), \tag{22}$$

  and $\bar{m}_i(l)$ was defined in (15). Note that, given the effects of the time windows described in (15), $\bar{M}_i(t)$ turns out to be the restriction to values in $t \in [a_i, b_i]$ of the cumulative probability distribution of the arrival time at customer $i$. This fact will be useful when proving dominance properties in Lemma 4.2.

To avoid a redundant label representation, we do not include $P_i$ among the label components. By the definition of $P_i$ and Eqs. (13) and (22) we can write

$$P_i = \Pr\{r_i \,\text{is successful}\} = \mathrm{M_i(b_i)}, \tag{23}$$

where $r_i$ denotes a generic partial route ending in node $i$. Furthermore, the number of label components at each node is node-dependent and varies according to the customer's time-window width. A feasible label $E_i = (C_i, V^1, \ldots, V^n, \bar{M}_i(a_i), \ldots, \bar{M}_i(b_i))$ must satisfy the following constraints: $V_i^n \in [0,1]$ for all $n \in V_c$; $\bar{M}_i(z_1) \le \bar{M}_i(z_2)$ for all $z_1 \le z_2$, $z_1, z_2 \in [a_i, b_i]$; and $\bar{M}_i(b_i) \ge \alpha$, where the last inequality is implied by (9). The reduced-cost component $C_i$ is not restricted by any resource window.

*Extension functions* Given a node $i \in V_c$, we now show how the associated feasible label $E_i = (C_i, V_i^1, \ldots, V_i^n, \bar{M}_i(a_i), \ldots, \bar{M}_i(b_i))$ can be extended along an arc $(i,j) \in A$. Let us start by considering the label components $V^1, \ldots, V^n$. Their extension is straightforward and follows the approach of Feillet et al. (2004). In particular $V_j^j = 1$, and $V_j^l = V_i^l$ for all $l \in V_c$. Furthermore, for any $l \in V_c$ with $V_j^l = 0$ such that the earliest possible arrival time from $j$ causes infeasibility with the time windows of $l$, $V_j^l$ is set to 1 (that is, $j$ is not considered unreachable).

For the label components of type $\bar{M}(\cdot)$, the extension of a partial route on arc $(i,j) \in A$ requires the creation of $b_j - a_j + 1$ label components $\bar{M}_j(z_j)$, for each $z_j \in [a_j, b_j]$. Each of these components can be expressed as a function of the $b_i -$

$a_i + 1$ labels $\bar{M}_i(z_i)$, for each $z_i \in [a_i, b_i]$. In fact, observe that by combining definition (22) with expression (18), we can write

$$\begin{aligned}\bar{M}_j(z_j) &= \sum_{k\in\mathbb{N}} d_i(k) \sum_{z\le z_j} \bar{m}_i(z - t_{ij} - k) \\ &= \sum_{k\in\mathbb{N}} d_i(k)\bar{M}_i(z_j - t_{ij} - k)\end{aligned} \tag{24}$$

for all $z_j \in [a_j, b_j]$.

For the extension of the reduced cost, observe that its expression (21) depends on $\beta_r$, which depends in turn [see definition (8)] on the total route success probability. Hence, to give the expression for the reduced-cost extension function, we first state and prove the following lemma.

**Lemma 4.1** *Consider any feasible route* $r = (v_0, \ldots, v_q, v_{q+1})$ *where* $v_0$ *and* $v_{q+1}$ *represent the depot 0 and its subroutes* $r_{v_i} = (v_0, \ldots, v_i)$, $1 \le i \le q$, *starting at the origin and ending at node* $v_i$. *Then, parameter* $\beta_r$ *in definition (21) can be expressed as*

$$\beta_r = \sum_{i=1}^{q} p_{v_{i-1},v_i} \tag{25}$$

*where* $p_{v_{i-1},v_i} := -\ln(\bar{M}_{v_i}(b_{v_i})/\bar{M}_{v_{i-1}}(b_{v_{i-1}}))\ \forall i \ge 2$, *and* $p_{v_0,v_1} = 0$.

*Proof* Since no time window is associated with $v_{q+1}$, and conditioning on the probability that $r_{v_{q-1}}$ is successful, we can write:

$$\begin{aligned}\Pr\{r \text{ is successful}\} &= \Pr\{r_{v_q} \text{is successful}\} \\ &= \Pr\{r_{v_q} \text{is successful} | \mathrm{r}_{\mathrm{v}_{\mathrm{q}-1}} \text{is successful}\} \Pr\{\mathrm{r}_{\mathrm{v}_{\mathrm{q}-1}} \text{is successful}\},\end{aligned}$$

where we have used the fact that the probability $\Pr\{r_{v_q} \text{is successful}|\mathrm{r}_{\mathrm{v}_{\mathrm{q}-1}}$ is not successful$\}$ is zero. Applying the same reasoning to $r_{v_{q-1}}$, $r_{v_{q-2}}$, ..., $r_{v_1}$ we can write:

$$\Pr\{r \text{ is successful}\} = \prod_{\mathrm{i}=1}^{\mathrm{q}} \Pr\{\mathrm{r}_{\mathrm{v}_\mathrm{i}} \text{is successful } | \mathrm{r}_{\mathrm{v}_{\mathrm{i}-1}} \text{is successful}\}, \tag{26}$$

where we have used $\Pr\{r_{v_1}|\text{is successful}\} = 1$ and $\Pr\{r_{v_0}\text{is successful}\} = 1$. Now, substituting (26) into the definition (8) of $\beta_r$, we can write

$$\beta_r = -\ln\left(\prod_{i=1}^{q} \Pr\{r_{v_i} \text{ is successful}|r_{v_{i-1}} \text{ is successful}\}\right).$$

Observing that $\{r_{v_i}$ is successful$\}$ implies $\{r_{v_{i-1}}$ is successful$\}$ and using properties of the conditional probability and (23) we can write

$$\begin{aligned}\beta_r &= -\ln\left(\prod_{i=1}^{q}\Pr(\{r_{v_i}\ \text{is}\ \text{successful}\} \wedge \{r_{v_{i-1}}\ \text{is}\ \text{successful}\}\right)/\\ &\qquad \Pr\{r_{v_{i-1}}\ \text{is}\ \text{successful}\})\\ &= -\ln\left(\prod_{i=1}^{q}\Pr\{r_{v_i}\ \text{is}\ \text{successful}\}/\Pr\{r_{v_{i-1}}\ \text{is}\ \text{successful}\}\right)\\ &= -\ln\left(\prod_{i=1}^{q}\bar{M}_{v_i}(b_{v_i})/\bar{M}_{v_{i-1}}(b_{v_{i-1}})\right)\\ &= \sum_{i=1}^{q} p_{v_{i-1},v_i}.\end{aligned}$$

□

We can now prove the following proposition.

**Proposition 4.1** (Reduced cost decomposition) Consider any feasible route $r = (v_0, \ldots, v_q, v_{q+1})$ where $v_0$ and $v_{q+1}$ represent the depot 0 and its subroutes $r_{v_i} = (v_0, \ldots, v_i)$, $1 \le i \le q$, starting at the origin and ending at node $v_i$. Then, its reduced cost can be expressed as

$$\bar{c}_r = \sum_{i=1}^{q+1} \bar{c}_{v_{i-1},v_i},$$

where

$$\bar{c}_{v_{i-1},v_i} = c_{v_{i-1},v_i} - \gamma_{v_i} + \delta p_{v_{i-1},v_i}, i = 1, \ldots, q \,\text{and}\, \mathrm{c}_{\mathrm{v_q},\mathrm{v_{q+1}}} = \mathrm{c}_{\mathrm{v_q},\mathrm{v_{q+1}}} = \mathrm{c}_{\mathrm{v_q},0}.$$

*Proof* Consider Eq. (25) and recall that $c_r = \sum_{i=1}^{q+1} c_{v_{i-1},v_i}$. Consequently, by substituting (25) into (21) and rearranging, we can express the reduced cost in the form:

$$\bar{c}_r = \sum_{i=1}^{q}(c_{v_{i-1},v_i} - \gamma_{v_i} + \delta p_{v_{i-1},v_i}) + c_{v_q,v_{q+1}} = \sum_{i=1}^{q+1} \bar{c}_{v_{i-1},v_i},$$

which proves the proposition. □

Proposition 4.1 shows that the reduced cost of a route can be decomposed into single arc contributions. Hence, it is straightforward to see that the reduced-cost extension function is defined by $C_j = C_i + \bar{c}_{ij}$. Notice that the reduced-cost extension function depends not only on the label component $C$, as in the case of VRPTW, but also on the label components $\bar{M}_i(a_i), \ldots, \bar{M}_i(b_i)$.

*Dominance* As previously mentioned, a key element for the labeling algorithm is the ability to discard non-useful labels. Consider two feasible partial routes $r^1$ and $r^2$, both ending in a given node $i \in V_c$ and represented by the labels $E_i^1$ and $E_i^2$.

Similarly to the VRPTW, in the context of the VRPTW-ST, we say that $E_i^1$ dominates $E_i^2$ when:

(1) Any feasible extension $e$ of $r^2$ ending at a given node $j$ is also feasible for $r^1$, and
(2) For any such extension $e$, the inequality $C_j^1 \leq C_j^2$ holds, where $C_j^l$ is the reduced cost of the route obtained by extending route $r^l$, $l = 1, 2$.

If the inequality in (2) is satisfied at equality for all feasible extensions $e$, then these labels dominate each other and at most one can be deleted. Because these two conditions are difficult to verify, we derive below sufficient conditions to identify dominated labels. These conditions are based in part on the following lemma.

**Lemma 4.2** *(Stochastic dominance). If two routes $r^1$ and $r^2$ are such that their labels satisfy $\bar{M}_i^1(z) \geq \bar{M}_i^2(z)$ for all $z \in [a_i, b_i]$, then for any common feasible extension e ending at a given node j,* the extended labels satisfy $\bar{M}_j^1(z) \geq \bar{M}_j^2(z)$ *for all $z \in [a_j, b_j]$.*

*Proof* First consider the case of a single-arc extension along arc $(i, j)$. In this case, using Equation (24) and the hypotheses of the lemma, we can write

$$\begin{aligned} \bar{M}_j^1(z) &= \sum_{k \in \mathbb{N}} d_i(k) \bar{M}_i^1(z - t_{ij} - k) \\ &\geq \sum_{k \in \mathbb{N}} d_i(k) \bar{M}_i^2(z - t_{ij} - k) = \bar{M}_j^2(z) \end{aligned}$$

for all $z \in [a_j, b_j]$, which proves the lemma in the case of a single-arc extension. To prove the lemma in the case of a more general extension, it suffices to recursively apply the single-arc result to all arcs in the extension. □

It is worth noticing that the concept of stochastic dominance has been introduced and used in more general contexts than in this paper [see for example Wellman et al. (1995)]. In our setting, however, Lemma 4.2 assures that stochastic dominance holds in spite of the presence of hard time windows.

To present the dominance rule and to simplify the notation, we express the reduced cost $\bar{c}_r$ of a partial route $r$ ending in node $i \in V_c$ as $\bar{c}_r = \rho_i - \delta \ln \bar{M}_i(b_i)$, obtained by setting $\rho_i := c_r - \sum_{h \in r} \gamma_h$ and substituting this into (21), together with (8) and (22).

**Proposition 4.2** (Dominance rule). If $r^1$ and $r^2$ are such that

(i) $\rho_i^1 \leq \rho_i^2$,
(ii) $V_i^{1j} \leq V_i^{2j}$ for all $j \in V_c$,
(iii) $\bar{M}_i^1(z) \geq \bar{M}_i^2(z)$, for all $z \in [a_i, b_i]$,

then $r^1$ dominates $r^2$ in the sense specified by conditions (1) and (2).

*Proof* We need to show that conditions (1) and (2) are satisfied under the hypotheses (i), (ii), and (iii). Consider an extension $e$ of routes $r^1$ and $r^2$, resulting in routes denoted $r^1 \oplus e$ and $r^2 \oplus e$, respectively. Assume that $r^2 \oplus e$ is feasible, that is, it is elementary and its success probability is greater than or equal to $\alpha$. From hypotheses (ii) and (iii), Lemma 4.2, and the fact that the extension functions of the $V^j$ components are nondecreasing, it follows that $r^1 \oplus e$ is also feasible and condition 1) is met.

Now let $\mathcal{A}(e) = \{(i,j) \in A | (i,j) \text{belongs to e}\}$ and assume that extension $e$ ends in node $j$. We can write:

$$\begin{aligned} C_j^1 &= \rho_i^1 + \sum_{(h,k)\in\mathcal{A}(e)} (c_{hk} - \gamma_k) - \delta \ln \bar{M}_j^1(b_j) \\ &\leq \rho_i^2 + \sum_{(h,k)\in\mathcal{A}(e)} (c_{hk} - \gamma_k) - \delta \ln \bar{M}_j^1(b_j) \\ &\leq \rho_i^2 + \sum_{(h,k)\in\mathcal{A}(e)} (c_{hk} - \gamma_k) - \delta \ln \bar{M}_j^2(b_j) = C_j^2, \end{aligned}$$

where the first inequality follows from ($i$), and the second follows from Lemma 4.2, together with the fact that logarithms are increasing functions and $\delta \geq 0$. Thus, condition 2) is satisfied. □

## 4.2 Acceleration strategies

To improve the efficiency of our algorithm, we adapted several known techniques from the literature. Below we briefly describe these accelerating strategies.

### *4.2.1 The ng-path relaxation*

The ESPPRC is strongly NP-hard (see Dror 1994) and it must be solved many times during a branch-price-and-cut algorithm. Much work has been done to speed up its solution, including the total or partial relaxation of the path elementarity. We adopt a path relaxation called the *ng*-path introduced in Baldacci et al. (2011). This approach associates a subset $N_i \subseteq V_c$ with each customer $i \in V_c$, such that $i \in N_i$ and $|N_i| \leq \Delta$, where $\Delta$ is a given integer parameter. Given a partial route $r = (v_0, \ldots, v_q)$, the subsets $N_i$ allow us to define a new subset $\Pi(r)$ whose elements are prevented from being extension candidates for $r$. The subset $\Pi(r)$ is defined as $\Pi(r) = \{v_i \in r | v_i \in \bigcap_{l=i+1}^{q} N_{v_l}, i = 1, \ldots, q-1\} \cup \{v_q\}$. If $\Delta$ is too small, the *ng*-paths may contain many cycles. If $\Delta = |V_c|$, the *ng*-paths are elementary but computationally expensive to handle.

We implemented the *ng*-path relaxation by modifying the extension function for the $V_1, \ldots, V_n$ label components. Specifically, we set $V_i = 0$ for all $i \notin \Pi(r)$ that are still reachable from $v_q$. In our implementation we eliminate 2-cycles according to the procedure described in Irnich and Villeneuve (2006). As a final remark, we build subsets $N_i$ by taking the nearest $\Delta - 1$ customers, and we experimented with three different values of $\Delta$. Preliminary computational tests with different values of $\Delta$ showed a slight advantage for $\Delta = 10$.

### *4.2.2 Decremental state space relaxation*

The decremental state space relaxation was introduced independently by Boland et al. (2006) and Righini and Salani (2008). This technique initially solves the subproblem without considering any elementarity requirements, i.e., without considering any label component $V^1, \ldots, V^n$. If the computed shortest path is non-elementary, some of the corresponding label components are added to avoid non-elementarity and the subproblem is solved again. This process is iterated until an elementary shortest path is found. In a column-generation context, however, the algorithm may be stopped either if a negative reduced cost elementary path is found, or if no negative reduced cost (cyclic) path is found. As proposed in Desaulniers et al. (2008), instead of restarting the decremental state space with an empty set of visit-label components at each iteration of the column generation, we use the components from the previous iteration. In our implementation we straightforwardly adapt this technique to *ng*-paths.

### *4.2.3 Heuristic dynamic programming*

To speed up the exact dynamic programming algorithm, we use two common techniques. The first temporarily eliminates unpromising arcs from the arc set $A$. This operation depends on the current value of the dual variables. If no routes with negative reduced costs are found, the number of arcs is progressively increased until all the arcs are considered. See Desaulniers et al. (2008) for a more detailed description of this technique.

The second technique consists in applying an aggressive dominance rules by eliminating a large number of labels, some of them possibly yielding shortest paths. Similarly, to the previous technique, if no route with a negative reduced cost is found, we try again with a weaker dominance rule. In our algorithm we define two parameters $V_{\max}$ and $M_{\max}$ indicating that the dominance comparison must be done on a subset of the conditions in Proposition 4.2, specifically on (i), on the $V_{\max}$ label components in (ii), and on the $M_{\max}$ label components in (iii), according to the following mechanism. Initially, $V_{\max} = 10$ (the actual number of visit-label components considered might be lower because of the interaction with the decremental state space relaxation) and $M_{\max} = 1$ (we start by considering the last label component, indicating the total route probability). If no routes with negative reduced costs are found, $V_{\max}$ is increased in steps of 10 and the process iterates. If at the end of these iterations no routes with negative reduced costs are found, $M_{\max}$ is progressively increased by $M_{\text{tot}}/M_{\text{res}}$, where $M_{\text{tot}}$ is the maximum number of $M(\cdot)$ at a given customer, and $M_{\text{res}}$ is a parameter set to 4 in our algorithm. This mechanism iterates until either a route with a negative reduced cost is found or all the $M(\cdot)$ label components have been considered.

#### 4.2.4 Tabu search column generator

To find columns with negative reduced costs, it is not always necessary to solve the ESPPRC by dynamic programming. In fact, in many cases, good heuristics can be effective. In our algorithm, we adopt a method similar to that presented in Desaulniers et al. (2008); it solves the ESPPRC by a multi-start tabu search algorithm. The method is based on two moves: the insertion and deletion of individual customers from a given solution. We adapt the procedure to our case by restricting the explored solution space to solutions that are feasible with respect to the worst-case service-time scenario. To diversify the search, the algorithm is run several times with different initial solutions and is limited to a maximum of $I_{\max}$ iterations for each run. The set of initial solutions considered are given by the routes in the basis of the current RMP. These routes are good starting points because they have reduced costs of zero. In our experiments we tested $I_{\max} = 5, 15, 25$. Preliminary computational tests showed that this technique gave a significant average improvement in the computational time (ranging from 10% to more than 50%). The best results were obtained with $I_{\max} = 15$. Given that our tabu search procedure does not involve any randomness, experimental replication was unecessary.

### 4.3 Cutting planes: subset-row inequalities

Once a valid lower bound and the corresponding solution have been obtained by column generation at a given node of the branch-and-bound search tree, we strengthen the bound by looking for violated valid inequalities. We consider a family of inequalities introduced in Jepsen et al. (2008) called *subset-row* inequalities that can be seen as Chvátal–Gomory rank I cuts. In the general case, subset-row inequalities are expressed as:

$$\sum_{r \in \mathcal{R}} \left\lfloor \frac{1}{k} \sum_{i \in S} a_{ir} \right\rfloor x_r \leq \left\lfloor \frac{|S|}{k} \right\rfloor, \quad \forall S \subseteq V_c, 2 \leq k \leq |S|.$$

Similarly to Jepsen et al. (2008), we consider only the cuts obtained with $|S| = 3$ and $k = 2$ because they are easier to find. In this case, the cuts can be rewritten as:

$$\sum_{r \in \mathcal{R}_S} x_r \leq 1, \forall S \subseteq V_c : |S| = 3, \tag{27}$$

where $\mathcal{R}_S$ is the subset of paths visiting at least two customers in $S$. In a column generation method, the addition of subset-row inequalities to the RMP requires several adjustments to the subproblem, as well as careful management when the number of added cuts increases. We follow the implementation of Desaulniers et al. (2008), to which we refer for the details.

### 4.4 Branching strategies

To find integer solutions, we apply the following branching rules whenever required. First, we branch on the total number of vehicles used. If this number is integer, we branch on the arc-flow variables. In this case, we select the arc $(i, j)$ with flow closest to 0.5. To fix the flow on this arc to 0, we simply remove $(i, j)$ from set $A$. To fix it to 1, we remove all arcs $(i, \ell), \ell \neq j$ and all arcs $(\ell, j), \ell \neq i$ from $A$. The corresponding columns in the RMP are deleted. Finally, the branch-and-bound tree is explored using a best-first strategy.

## 5 Computational experiments

We performed two sets of experiments to investigate different aspects of our work, namely (1) tests on benchmark instances; (2) evaluation under a variety of conditions, such as different probability distributions and reliability thresholds, and comparison with a deterministic approach.

The experiments were performed on machines running Linux (Suse) and an Intel Core i7-2600 3.40 GHz CPU with 16 G of RAM.

### 5.1 Instance set

Following the current practice in testing algorithmic methods for VRP-related problems, we derived our benchmark instances from Solomon's well-known database (1987). It is made up of six instance classes: R1, RC1, C1, R2, RC2, C2. All Solomon's instances contain 100 customers located on a 100*100 square. The prefixes of the class names indicate how the customer locations are chosen: R for random, C for clustered, and RC for hybrid random and clustered. The numeric suffix (1 or 2) is related to the average time-window width compared to the average traveling time: class names ending with 1 usually have narrower time windows. There are between 8 and 12 instances for each class, for a total of 85.

We derived several instance families. For all of them we disregarded the vehicle capacity and the customer demands. The time horizon was discretized in intervals of 0.1 minutes. Then, starting from the original service times, we constructed several discrete triangular distributions and several values for the reliability threshold. Our instance families are as follows:

1. *Basic* Symmetric triangular distribution with median equal to the deterministic value of the service time, i.e., 10 for R and RC classes and 90 for C classes, and support intervals of $\{8.0, 8.1, \ldots, 12.0\}$ for R and RC classes and $\{70.0, 70.1, \ldots, 110.0\}$ for C classes. The reliability threshold is set to $\alpha = 95\%$.
2. *Low-probability* Similar to the previous family, but with $\alpha = 85\%$.
3. *Large-support* Similar to the Basic family, but the support of the probability distributions is considerably larger: $\{5.0, 5.1, \ldots, 15.0\}$ for R and RC classes, and $\{45.0, 45.1, \ldots, 135.0\}$ for C classes.

4. *Positive-skewed* This is a variant of the Large-support family. The distributions have the same support as the Large-support, but different medians: 7.0 for R and RC classes and 63.0 for C classes. This represents practical situations where service times have large variability, like for Large-support, but they are more likely to be short.

It is worth noticing that we also considered hybrid instances where different service probability distributions were associated to different subsets of customers. However, preliminary results did not provide elements that could not be inferred by the analysis of the four cases reported below. Therefore, hybrid instances have not been further investigated.

Finally, our method was able to solve very few 100-customer instances, so we followed common practice by reducing the size of Solomon's instances by considering only the first 25 customers (for suffixes 1 and 2) and 50 customers (for suffix 1 only).

## 5.2 Algorithmic behavior on benchmark instances

This section reports the results obtained by our algorithm for the benchmark instances of the VRPTW-ST in the four classes introduced in Sect. 5.1. The main goal is to analyze the impact of different problem settings on the efficiency of our algorithm.

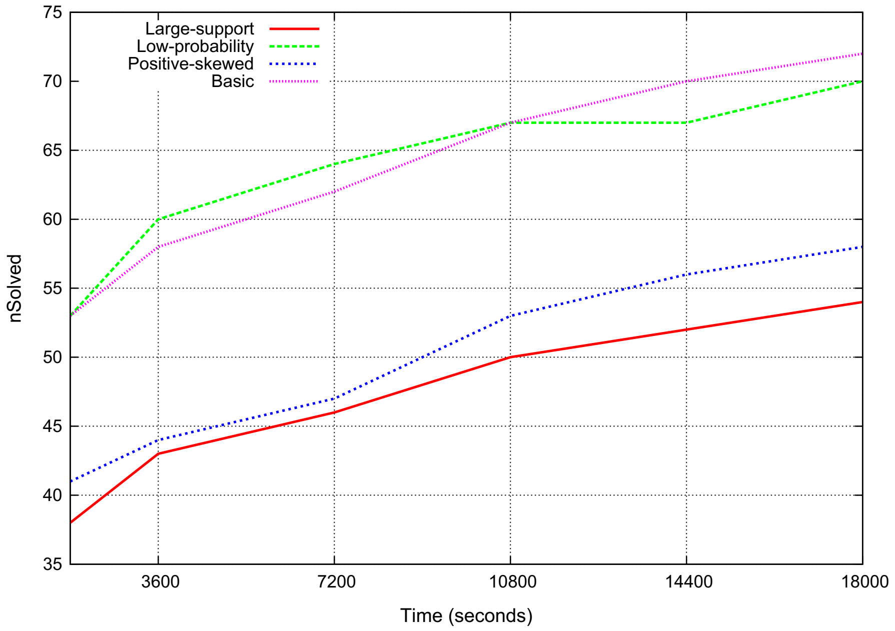

**Fig. 1** Number of solved instances by time

Figure 1 reports the evolution of the number of solved instances, aggregated over class type, over time. The *x*-axis indicates the computing time in seconds, and the *y*-axis indicates the number of instances solved. The first observation we can make is that problems with a larger probability-distribution support, i.e., Large-support and Positive-skewed, are much harder than those with a smaller support, i.e., Basic and Low-probability. One of the reasons for this behavior is that the arrival-time probability distributions at the customers usually have larger supports when the service-time probability distributions have larger supports. This fact has two general consequences: (1) the memory consumption is larger; and (2) it is harder to find dominated labels because there are more conditions to be verified. Another reason is that, on average, the root gap increases for large service-time distributions and this slows the convergence. When comparing the Large-support and Positive-skewed classes, we observe that the former are slightly more difficult than the latter, and this seems to be linked to better root-gap values for the Positive-skewed case. There is no significant difference in performance between the Basic and Low-probability classes, although Low-probability seems slightly more difficult given the larger feasible region and the related harder fathoming process in the dynamic programming algorithm. We furthermore observe that around 80% of the instances are solved within the first hour of computation.

More detailed results are shown in Tables 1, 2, 3, and 4. The data are grouped by instance type and number of customers, and several levels of aggregation are reported. Columns 1, 2, and 3 identify the instance type, the class, and the number of customers. Columns 4 and 5 report the mean and standard deviation of the number of vehicles used in the optimal solution. Columns 6 and 7 report the mean and standard deviation of the route success probability. Columns 8 and 9 report the mean and standard deviation of the computing times, while the last two columns report the total number of instances in the group and the number of instances solved to optimality in 5 hours of computing time. Each column reports aggregated results for a particular instance group identified by the first three columns. For example, row 1 reports aggregated results for instances belonging to the C1 family with 25 customers. A missing value in one of the first three columns indicates that the data have been aggregated at higher levels. For example, row 4 reports aggregated data for all the instances in C1, R1, and RC1 with 25 customers; row 9 reports aggregated data for all the instances in C1, R1, and RC1; and the last row aggregates over all the instances. The results mostly confirm the highlighted tendencies.

## 5.3 Model behavior

In this section, we analyze the behavior of the proposed model, and we do this in two ways. First, we compare variations in solution cost, number of vehicles used, and success probability when testing our model on the four instance families. Second, we compare the solutions obtained by our probabilistic model with two plans obtained by solving the deterministic VRPTW model. In the first deterministic case we set the service time to the median of the corresponding stochastic instance, and in the second we set the service time to the corresponding worst-case value.

**Table 1** Algorithmic behavior on the Basic instance family

| type | class | n | nVehAvg | nVehStd | SuccProbAvg | SuccProbStd | TimeAvg | TimeStd | total | count |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | 3.2 | 0.4 | 98.3 | 1.7 | 2633.1 | 3291.7 | 9 | 9 |
| 1 | R | 25 | 5.0 | 1.3 | 99.4 | 1.1 | 13.8 | 22.3 | 12 | 12 |
| 1 | RC | 25 | 3.3 | 0.4 | 99.3 | 1.1 | 41.7 | 55.1 | 8 | 8 |
| 1 |  | 25 | 4.0 | 1.2 | 99.0 | 1.4 | 834.4 | 2195.4 | 29 | 29 |
| 1 | C | 50 | 5.6 | 0.8 | 97.4 | 1.6 | 8763.7 | 6031.3 | 9 | 5 |
| 1 | R | 50 | 8.5 | 2.0 | 97.7 | 1.7 | 1123.0 | 1449.7 | 12 | 11 |
| 1 | RC | 50 | 6.6 | 1.0 | 98.6 | 1.5 | 3008.9 | 4105.4 | 8 | 7 |
| 1 |  | 50 | 7.3 | 1.9 | 97.9 | 1.7 | 3358.0 | 4777.2 | 29 | 23 |
| 1 |  |  | 5.4 | 2.3 | 98.5 | 1.7 | 1950.6 | 3788.5 | 58 | 52 |
| 2 | C | 25 | 2.0 | 0.0 | 99.8 | 0.4 | 3515.6 | 5147.3 | 8 | 7 |
| 2 | R | 25 | 2.9 | 0.7 | 99.4 | 1.1 | 2580.4 | 4605.5 | 11 | 10 |
| 2 | RC | 25 | 3.0 | 0.0 | 100.0 | 0.0 | 2594.9 | 4733.2 | 8 | 5 |
| 2 |  | 25 | 2.6 | 0.6 | 99.7 | 0.8 | 2881.3 | 4832.3 | 27 | 22 |
|  |  |  | 4.6 | 2.3 | 98.9 | 1.6 | 2227.3 | 4148.4 | 85 | 74 |

**Table 2** Algorithmic behavior on the Low-probability instance family

| type | class | n | nVehAvg | nVehStd | SuccProbAvg | SuccProbStd | TimeAvg | TimeStd | total | count |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | 3.0 | 0.0 | 95.4 | 4.9 | 5232.8 | 6983.3 | 9 | 8 |
| 1 | R | 25 | 5.0 | 1.3 | 97.5 | 4.4 | 15.2 | 27.2 | 12 | 12 |
| 1 | RC | 25 | 3.3 | 0.4 | 99.3 | 1.1 | 32.4 | 41.1 | 8 | 8 |
| 1 |  | 25 | 3.9 | 1.3 | 97.4 | 4.2 | 1510.8 | 4413.1 | 29 | 28 |
| 1 | C | 50 | 5.0 | 0.0 | 92.2 | 1.4 | 3244.5 | 4065.2 | 9 | 4 |
| 1 | R | 50 | 8.5 | 2.0 | 93.7 | 3.9 | 1186.3 | 1585.7 | 12 | 11 |
| 1 | RC | 50 | 6.6 | 1.0 | 94.9 | 4.3 | 3284.4 | 3197.3 | 8 | 7 |
| 1 |  | 50 | 7.2 | 2.0 | 93.8 | 3.8 | 2228.1 | 2932.6 | 29 | 22 |
| 1 |  |  | 5.4 | 2.3 | 95.8 | 4.4 | 1826.4 | 3849.3 | 58 | 50 |
| 2 | C | 25 | 2.0 | 0.0 | 99.8 | 0.4 | 1703.7 | 2124.2 | 8 | 6 |
| 2 | R | 25 | 2.9 | 0.7 | 99.4 | 1.1 | 2923.1 | 5426.1 | 11 | 10 |
| 2 | RC | 25 | 3.0 | 0.0 | 100.0 | 0.0 | 497.4 | 729.1 | 8 | 4 |
| 2 |  | 25 | 2.7 | 0.7 | 99.7 | 0.9 | 2072.1 | 4132.8 | 27 | 20 |
|  |  |  | 4.6 | 2.4 | 96.9 | 4.1 | 1896.6 | 3933.9 | 85 | 70 |

**Table 3** Algorithmic behavior on the Large-support instance family

| type | class | n | nVehAvg | nVehStd | SuccProbAvg | SuccProbStd | TimeAvg | TimeStd | total | count |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | 4.0 | 0.9 | 96.7 | 1.7 | 6121.7 | 6386.5 | 9 | 5 |
| 1 | R | 25 | 5.4 | 1.4 | 98.1 | 1.3 | 119.5 | 245.9 | 12 | 12 |
| 1 | RC | 25 | 3.4 | 0.5 | 97.2 | 1.6 | 476.8 | 479.1 | 8 | 8 |
| 1 |  | 25 | 4.5 | 1.4 | 97.6 | 1.6 | 1434.3 | 3711.8 | 29 | 25 |
| 1 | C | 50 | 8.0 | 0.0 | 96.3 | 0.0 | 7271.5 | 0.0 | 9 | 1 |
| 1 | R | 50 | 9.3 | 2.2 | 96.5 | 0.6 | 1846.6 | 1793.7 | 12 | 9 |
| 1 | RC | 50 | 7.8 | 1.1 | 95.5 | 0.1 | 8201.2 | 5510.0 | 8 | 4 |
| 1 |  | 50 | 8.8 | 2.0 | 96.2 | 0.7 | 4049.7 | 4419.1 | 29 | 14 |
| 1 |  |  | 6.0 | 2.7 | 97.1 | 1.5 | 2373.1 | 4173.3 | 58 | 39 |
| 2 | C | 25 | 2.0 | 0.0 | 99.1 | 1.5 | 6331.6 | 4019.5 | 8 | 4 |
| 2 | R | 25 | 3.0 | 0.7 | 100.0 | 0.0 | 2264.0 | 3947.0 | 11 | 9 |
| 2 | RC | 25 | 3.0 | 0.0 | 99.8 | 0.4 | 1037.4 | 1454.9 | 8 | 4 |
| 2 |  | 25 | 2.8 | 0.6 | 99.7 | 0.8 | 2932.5 | 4043.2 | 27 | 17 |
|  |  |  | 5.0 | 2.7 | 97.9 | 1.8 | 2542.9 | 4142.2 | 85 | 56 |

Table 5 compares the results obtained by running our algorithm on instances with the same service-time probability distributions and two different reliability thresholds $\alpha = (0.85, 0.95)$, i.e., Low-probability and Basic. Specifically, Table 5 shows the relative variation in the results obtained for the Low-probability family, and it uses as a reference the values obtained for the Basic family. The values are aggregated as before and the columns report the following data. Columns 1, 2, and 3 identify the instance type, the class, and the number of customers. Columns 4 and 5

**Table 4** Algorithmic behavior on the Positive-skewed instance family

| type | class | n | nVehAvg | nVehStd | SuccProbAvg | SuccProbStd | TimeAvg | TimeStd | total | count |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | 3.8 | 1.0 | 98.1 | 1.8 | 4854.7 | 5430.1 | 9 | 5 |
| 1 | R | 25 | 5.3 | 1.5 | 98.1 | 1.3 | 86.3 | 145.5 | 12 | 12 |
| 1 | RC | 25 | 3.3 | 0.4 | 96.8 | 1.4 | 742.7 | 1589.9 | 8 | 8 |
| 1 | | 25 | 4.3 | 1.5 | 97.7 | 1.6 | 1250.0 | 3169.8 | 29 | 25 |
| 1 | C | 50 | 6.0 | 1.4 | 96.0 | 1.2 | 8644.4 | 4798.8 | 9 | 3 |
| 1 | R | 50 | 9.2 | 2.3 | 96.1 | 1.1 | 3543.2 | 4523.5 | 12 | 9 |
| 1 | RC | 50 | 7.2 | 1.5 | 96.1 | 1.0 | 6015.2 | 5150.1 | 8 | 5 |
| 1 | | 50 | 8.1 | 2.4 | 96.1 | 1.1 | 5170.5 | 5141.9 | 29 | 17 |
| 1 | | | 5.8 | 2.6 | 97.1 | 1.6 | 2836.9 | 4515.0 | 58 | 42 |
| 2 | C | 25 | 2.0 | 0.0 | 99.4 | 0.6 | 4841.2 | 4741.2 | 8 | 4 |
| 2 | R | 25 | 3.1 | 0.6 | 99.7 | 0.9 | 1183.5 | 2305.4 | 11 | 8 |
| 2 | RC | 25 | 3.0 | 0.0 | 100.0 | 0.1 | 1017.8 | 1230.9 | 8 | 4 |
| 2 | | 25 | 2.8 | 0.6 | 99.7 | 0.8 | 2056.5 | 3353.4 | 27 | 16 |
| | | | 5.0 | 2.6 | 97.8 | 1.8 | 2621.6 | 4240.9 | 85 | 58 |

**Table 5** Comparison between Low-probability and Basic instance families

| type | class | n | %CostDAvg | %CostDStd | %VehDAvg | %VehDStd | %SuccDAvg | %SuccDStd | count | countD |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | -0.9 | 1.2 | -25.0 | 43.3 | -2.92444 | 3.91583 | 8 | -1 |
| 1 | R | 25 | -0.3 | 0.6 | 0.0 | 0.0 | -1.93500 | 3.60071 | 12 | 0 |
| 1 | RC | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 8 | 0 |
| 1 | | 25 | -0.4 | 0.8 | -7.1 | 25.8 | -1.66484 | 3.34874 | 28 | -1 |
| 1 | C | 50 | -0.1 | 0.1 | 0.0 | 0.0 | -5.09193 | 2.49325 | 3 | 0 |
| 1 | R | 50 | -0.2 | 0.2 | 0.0 | 42.6 | -3.98612 | 3.94469 | 11 | 0 |
| 1 | RC | 50 | -0.1 | 0.2 | 0.0 | 0.0 | -3.70317 | 4.81969 | 7 | 0 |
| 1 | | 50 | -0.1 | 0.2 | 0.0 | 30.9 | -4.04978 | 4.12059 | 21 | 0 |
| 1 | | | -0.3 | 0.6 | -4.1 | 28.3 | -2.68696 | 3.88302 | 49 | -1 |
| 2 | C | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 6 | 0 |
| 2 | R | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 10 | 0 |
| 2 | RC | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 4 | -1 |
| 2 | | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 20 | -1 |
| | | | -0.2 | 0.5 | -2.9 | 23.9 | -1.90813 | 3.49192 | 69 | -2 |

**Table 6** Comparison between Basic and Large-support instance families

| type | class | n | %CostDAvg | %CostDStd | %VehDAvg | %VehDStd | %SuccDAvg | %SuccDStd | count | countD |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | 15.4 | 12.7 | 60.0 | 49.0 | -0.91052 | 1.27116 | 5 | -4 |
| 1 | R | 25 | 3.0 | 4.0 | 41.7 | 49.3 | -1.32654 | 1.92608 | 12 | 0 |
| 1 | RC | 25 | 3.7 | 5.0 | 12.5 | 33.1 | -2.04566 | 1.89631 | 8 | 0 |
| 1 | | 25 | 5.7 | 8.4 | 36.0 | 48.0 | -1.47346 | 1.85286 | 25 | -4 |
| 1 | C | 50 | | | | | | | 0 | -3 |
| 1 | R | 50 | 4.6 | 3.7 | 44.4 | 83.1 | -1.34917 | 2.27829 | 9 | -2 |
| 1 | RC | 50 | 4.1 | 1.6 | 50.0 | 50.0 | -2.42705 | 1.69911 | 4 | -3 |
| 1 | | 50 | 4.4 | 3.3 | 46.2 | 74.6 | -1.68082 | 2.17469 | 13 | -8 |
| 1 | | | 5.3 | 7.1 | 39.5 | 58.7 | -1.54440 | 1.97135 | 38 | -12 |
| 2 | C | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.34183 | 0.50852 | 3 | -3 |
| 2 | R | 25 | 0.3 | 0.5 | 0.0 | 0.0 | 0.72698 | 1.21418 | 8 | -2 |
| 2 | RC | 25 | 0.0 | 0.0 | 0.0 | 0.0 | -0.24850 | 0.43041 | 4 | -1 |
| 2 | | 25 | 0.1 | 0.4 | 0.0 | 0.0 | 0.38982 | 1.02816 | 15 | -6 |
| | | | 3.8 | 6.5 | 28.3 | 52.8 | -0.99698 | 1.96079 | 53 | -18 |

report the mean and standard deviation of the percentage variation in the solution cost. Columns 6 and 7 report the mean and standard deviation of the percentage variation in the number of vehicles used. Columns 8 and 9 report the mean and standard deviation of the success probability, while Columns 10 and 11 report the number of instances in the comparison (only solutions that are optimal for both settings are considered) and the difference in the number of instances solved for each group. The results show an average solution cost improvement of 0.2%, which is correlated with an average decrease in the vehicles used, and corresponds approximately to a 2% deterioration in the average success probability, which is one order of magnitude higher. This suggests that weakening the constraint on the success probability would not necessarily imply a significant gain in terms of the solution cost. Finally, for $\alpha = 0.85$, our algorithm failed to find the optimal solution of two instances that were solved to optimality for $\alpha = 0.95$, and this confirms that the problems are more difficult on average.

Table 6 compares the results obtained by running our algorithm on instances with different service-time probability distributions, and the same reliability threshold $\alpha = 0.95$, i.e., the Basic and Large-support families. Specifically, Table 6 shows the relative variation in the results obtained for the Large-support family, and it uses as a reference the values obtained for the Basic family. The results show that when the uncertainty is higher, the solution cost and the number of vehicles used might be considerably higher for the same reliability threshold.

Table 7 compares the results obtained by running our algorithm on instances with Large-support and Positive-skewed service-time probability distributions and the same reliability threshold $\alpha = 0.95$. Specifically, Table 7 shows the relative variation in the results obtained for the Positively skewed distribution, and it uses as a reference the values obtained for the Large-support family. The results show that although the support of both probability distributions is large, the Positive-skewed distribution has more potential for reductions in the cost and vehicle utilization.

Tables 8 and 9 compare the solutions obtained by the stochastic model and the deterministic VRPTW model with the service time set to the median of the probability distribution. Specifically, Table 8 shows the relative variation in the results obtained for the deterministic model, and it uses as a reference the values obtained for the Basic family. Table 9 shows the results obtained by the same deterministic model, and it uses as a reference the values obtained for the Large-support family. The analysis is restricted to the solved instances only. The two tables display similar trends, but the Large-support family has more variation. Generally speaking, the results show that deterministic planning with median values consistently lowers the success probability, while the solution costs improve only marginally. There are also type-specific differences: the advantages of the stochastic model are evident for type 1 but almost negligible for type 2. This is because type-2 instances have wide time windows, so service-time uncertainty has little effect on feasibility. When the time windows are narrower, the stochastic model becomes important. This can be confirmed by inspecting, for instance, row 9 of both tables where aggregated results are shown for type-1 instances only: an average improvement of 2.3% in the solution cost has a corresponding decrease of 21% in the success probability when the reference is the Basic family, and an average improvement of 6.8% in the solution cost has a corresponding decrease of 44% in the success probability when the reference is the Large-support family. These values imply that the deterministic plan has a success probability below 75% for the Basic family and below 50% for the Large-support family, and such low values are not acceptable for most practical applications.

We end our analysis with Tables 10 and 11, which compare the solutions obtained by the stochastic model and the deterministic VRPTW model with the service time set to the worst-case time of the probability distribution. Specifically,

**Table 7** Comparison between Postively skewed and Large-support instance families

| type | class | n | %CostDAvg | %CostDStd | %VehDAvg | %VehDStd | %SuccDAvg | %SuccDStd | count | countD |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | -0.7 | 0.5 | 0.0 | 0.0 | 0.69870 | 1.77200 | 4 | 0 |
| 1 | R | 25 | -0.5 | 0.7 | -16.7 | 37.3 | 0.00942 | 1.80405 | 12 | 0 |
| 1 | RC | 25 | -3.0 | 4.2 | -12.5 | 33.1 | -0.37952 | 2.07888 | 8 | 0 |
| 1 |  | 25 | -1.4 | 2.7 | -12.5 | 33.1 | -0.00535 | 1.92889 | 24 | 0 |
| 1 | C | 50 | -3.2 | 0.0 | 0.0 | 0.0 | -1.20090 | 0.00000 | 1 | 2 |
| 1 | R | 50 | -2.6 | 2.7 | -11.1 | 31.4 | -0.33536 | 1.18514 | 9 | 0 |
| 1 | RC | 50 | -2.6 | 1.1 | 0.0 | 0.0 | 0.45972 | 1.09671 | 4 | 1 |
| 1 |  | 50 | -2.6 | 2.2 | -7.1 | 25.8 | -0.17001 | 1.20556 | 14 | 3 |
| 1 |  |  | -1.8 | 2.6 | -10.5 | 30.7 | -0.06602 | 1.70048 | 38 | 3 |
| 2 | C | 25 | -0.0 | 0.0 | 0.0 | 0.0 | -0.50783 | 0.74314 | 3 | 1 |
| 2 | R | 25 | -0.0 | 0.1 | 0.0 | 0.0 | -0.34969 | 0.92519 | 8 | 0 |
| 2 | RC | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.20800 | 0.36027 | 4 | 0 |
| 2 |  | 25 | -0.0 | 0.1 | 0.0 | 0.0 | -0.23260 | 0.82208 | 15 | 1 |
|  |  |  | -1.3 | 2.4 | -7.5 | 26.4 | -0.11316 | 1.50670 | 53 | 4 |

**Table 8** Comparison between deterministic approach (median values) and stochastic (Basic family)

| type | class | n | %CostDAvg | %CostDStd | %VehDAvg | %VehDStd | %SuccDAvg | %SuccDStd | count | countD |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | -7.2 | 9.6 | -22.2 | 41.6 | -14.95029 | 14.30632 | 9 | 0 |
| 1 | R | 25 | -0.3 | 0.6 | 8.3 | 27.6 | -5.71342 | 10.36817 | 12 | 0 |
| 1 | RC | 25 | -0.3 | 0.5 | 0.0 | 0.0 | -14.63575 | 16.35132 | 8 | 0 |
| 1 |  | 25 | -2.5 | 6.2 | -3.4 | 32.0 | -11.04137 | 14.20596 | 29 | 0 |
| 1 | C | 50 | -6.4 | 10.9 | -50.0 | 86.6 | -19.94655 | 25.81939 | 4 | 5 |
| 1 | R | 50 | -1.4 | 1.0 | -27.3 | 61.7 | -48.76202 | 21.79792 | 11 | 0 |
| 1 | RC | 50 | -1.0 | 1.9 | 0.0 | 0.0 | -23.69179 | 23.26045 | 7 | 1 |
| 1 |  | 50 | -2.2 | 5.2 | -22.7 | 59.8 | -35.54595 | 26.59363 | 22 | 6 |
| 1 |  |  | -2.3 | 5.8 | -11.8 | 47.1 | -21.61197 | 23.81429 | 51 | 6 |
| 2 | C | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00053 | 0.00119 | 6 | 2 |
| 2 | R | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 10 | 1 |
| 2 | RC | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 5 | 3 |
| 2 |  | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00015 | 0.00068 | 21 | 6 |
|  |  |  | -1.7 | 5.0 | -8.3 | 40.0 | -15.30844 | 22.32059 | 72 | 12 |

**Table 9** Comparison between deterministic approach (median values) and stochastic (Large-support family)

| type | class | n | %CostDAvg | %CostDStd | %VehDAvg | %VehDStd | %SuccDAvg | %SuccDStd | count | countD |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | -19.9 | 16.7 | -100.0 | 89.4 | -39.35158 | 33.97154 | 5 | 4 |
| 1 | R | 25 | -3.1 | 3.4 | -33.3 | 47.1 | -23.12847 | 20.30033 | 12 | 0 |
| 1 | RC | 25 | -3.6 | 4.2 | -12.5 | 33.1 | -34.56294 | 18.85600 | 8 | 0 |
| 1 |  | 25 | -6.7 | 10.5 | -40.0 | 63.2 | -30.03212 | 24.27375 | 25 | 4 |
| 1 | C | 50 | -33.3 | 0.0 | -300.0 | 0.0 | -93.01458 | 0.00000 | 1 | 8 |
| 1 | R | 50 | -5.6 | 2.7 | -77.8 | 62.9 | -75.77688 | 13.19437 | 9 | 2 |
| 1 | RC | 50 | -4.1 | 1.4 | -50.0 | 50.0 | -56.44372 | 8.65402 | 4 | 4 |
| 1 |  | 50 | -7.2 | 7.6 | -85.7 | 83.3 | -71.48438 | 15.58537 | 14 | 14 |
| 1 |  |  | -6.8 | 9.6 | -56.4 | 74.4 | -44.91242 | 29.33082 | 39 | 18 |
| 2 | C | 25 | -0.0 | 0.0 | 0.0 | 0.0 | -6.33533 | 8.98435 | 3 | 5 |
| 2 | R | 25 | -0.3 | 0.5 | 0.0 | 0.0 | -7.22580 | 9.74890 | 8 | 3 |
| 2 | RC | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 4 | 4 |
| 2 |  | 25 | -0.1 | 0.4 | 0.0 | 0.0 | -5.12083 | 8.74546 | 15 | 12 |
|  |  |  | -5.0 | 8.7 | -40.7 | 68.1 | -33.85920 | 30.98746 | 54 | 30 |

Table 10 shows the relative variation in the results obtained for the deterministic model, and it uses as a reference the values obtained for the Basic family. Table 11 shows the results obtained for the deterministic model, and it uses as a reference the values obtained for the Large-support family. As for the previous case, we restrict our analysis to the solved instances. The two tables display similar trends, but the Large-support family has more variation. Generally speaking, the results show that the deterministic plan with worst-case values consistently increases the solution

**Table 10** Comparison between deterministic approach (worst case values) and stochastic (Basic family)

| type | class | n | %CostDAvg | %CostDStd | %VehDAvg | %VehDStd | %SuccDAvg | %SuccDStd | count | countD |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | 18.4 | 10.4 | 33.3 | 47.1 | 1.71514 | 1.74803 | 9 | 0 |
| 1 | R | 25 | 1.3 | 2.4 | 16.7 | 37.3 | 0.55087 | 1.09193 | 12 | 0 |
| 1 | RC | 25 | 3.9 | 4.9 | 12.5 | 33.1 | 0.72861 | 1.10276 | 8 | 0 |
| 1 |  | 25 | 7.3 | 10.0 | 20.7 | 40.5 | 0.96123 | 1.42763 | 29 | 0 |
| 1 | C | 50 | 26.8 | 3.2 | 100.0 | 0.0 | 3.28417 | 1.18930 | 3 | 2 |
| 1 | R | 50 | 1.4 | 0.9 | 18.2 | 57.5 | 2.33595 | 1.75452 | 11 | 1 |
| 1 | RC | 50 | 7.6 | 3.9 | 66.7 | 47.1 | 1.44015 | 1.61291 | 6 | 0 |
| 1 |  | 50 | 7.0 | 9.1 | 45.0 | 58.9 | 2.20944 | 1.74508 | 20 | 3 |
| 1 |  |  | 7.2 | 9.6 | 30.6 | 50.3 | 1.47070 | 1.68095 | 49 | 3 |
| 2 | C | 25 | 3.8 | 5.5 | -16.7 | 37.3 | 0.17732 | 0.39506 | 6 | 2 |
| 2 | R | 25 | 0.2 | 0.4 | 0.0 | 0.0 | 0.58158 | 1.12425 | 10 | 1 |
| 2 | RC | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 5 | 3 |
| 2 |  | 25 | 1.2 | 3.4 | -4.8 | 21.3 | 0.32760 | 0.84214 | 21 | 6 |
|  |  |  | 5.4 | 8.7 | 20.0 | 46.6 | 1.12777 | 1.57005 | 70 | 9 |

**Table 11** Comparison between deterministic approach (worst case values) and stochastic (Large-support family)

| type | class | n | %CostDAvg | %CostDStd | %VehDAvg | %VehDStd | %SuccDAvg | %SuccDStd | count | countD |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | C | 25 | 19.1 | 11.6 | 80.0 | 40.0 | 3.25466 | 1.71267 | 5 | 3 |
| 1 | R | 25 | 3.0 | 2.0 | 33.3 | 47.1 | 1.87741 | 1.26070 | 12 | 0 |
| 1 | RC | 25 | 19.9 | 6.7 | 100.0 | 50.0 | 2.77428 | 1.57894 | 8 | 0 |
| 1 |  | 25 | 11.6 | 10.6 | 64.0 | 55.7 | 2.43986 | 1.57090 | 25 | 3 |
| 1 | C | 50 | 14.9 | 0.0 | 200.0 | 0.0 | 3.68960 | 0.00000 | 1 | 4 |
| 1 | R | 50 | 3.2 | 2.1 | 66.7 | 47.1 | 3.52046 | 0.64401 | 9 | 3 |
| 1 | RC | 50 | 10.7 | 10.0 | 125.0 | 43.3 | 4.50800 | 0.09573 | 4 | 1 |
| 1 |  | 50 | 6.2 | 7.0 | 92.9 | 59.3 | 3.81469 | 0.68070 | 14 | 8 |
| 1 |  |  | 9.6 | 9.8 | 74.4 | 58.7 | 2.93339 | 1.47755 | 39 | 11 |
| 2 | C | 25 | 8.1 | 4.0 | 0.0 | 0.0 | 0.01173 | 0.01659 | 3 | 5 |
| 2 | R | 25 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00000 | 0.00000 | 8 | 3 |
| 2 | RC | 25 | 0.3 | 0.5 | 0.0 | 0.0 | 0.24850 | 0.43041 | 4 | 4 |
| 2 |  | 25 | 1.7 | 3.7 | 0.0 | 0.0 | 0.06861 | 0.24748 | 15 | 12 |
|  |  |  | 7.4 | 9.3 | 53.7 | 60.0 | 2.13762 | 1.80005 | 54 | 23 |

cost, while the success probability improves only marginally. There are again type-specific differences. This can be confirmed by inspecting, for instance, row 9 of both tables where aggregated results are shown for type-1 instances only: an average improvement of 1.4% in the success probability has a corresponding increase of 7.2% in the solution cost when the reference is the Basic family, and an average improvement of 2.9% in the success probability has a corresponding increase of 9.6% in the solution cost when the reference is the Large-support family.

## 6 Conclusion

In this paper, we have presented the VRPTW-ST, which is an important stochastic variant of the VRPTW where service times are stochastic and time windows are hard. We formulate the problem as a set partitioning model with a probabilistic constraint on the global success probability of the route plan. We solve the VRPTW-ST by a branch-price-and-cut algorithm. The results show that our method is effective, and that a stochastic chance-constrained model has advantages over deterministic models.

One of the limitations of our approach is the high dimension of the label in the dynamic programming algorithm. Future research might focus on suitable aggregate representations of the planning horizon accompanied by new modified dominance rules.

## References

Adulyasak Y, Jaillet P (2014) Models and algorithms for stochastic and robust vehicle routing with deadlines. Transp Sci 50(2):608–626

Baldacci R, Mingozzi A, Roberti R (2011) New route relaxation and pricing strategies for the vehicle routing problem. Oper Res 59(5):1269–1283

Bertsimas DJ (1992) A vehicle routing problem with stochastic demand. Oper Res 40(3):574–585

Boland N, Dethridge J, Dumitrescu I (2006) Accelerated label setting algorithms for the elementary resource constrained shortest path problem. Oper Res Lett 34(1):58–68

Campbell AM, Thomas BW (2008) Probabilistic traveling salesman problem with deadlines. Transp Sci 42(1):1–21

Chang T-S, Wan Y-W, Ooi WT (2009) A stochastic dynamic traveling salesman problem with hard time windows. Eur J Oper Res 198(3):748–759

Cortés C, Gendreau M, Rousseau L-M, Souyris S, Weintraub A (2014) Branch-and-price and constraint programming for solving a real-life technician dispatching problem. Eur J Oper Res 238(1):300–312

Desaulniers G, Lessard F, Hadjar A (2008) Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. Transp Sci 42(3):387–404

Dror M (1994) Note on the complexity of the shortest path models for column generation in VRPTW. Oper Res 42(5):977–978

Errico F, Desaulniers G, Gendreau M, Rei W, Rousseau LM (2015) A priori optimization with recourse for the vehicle routing problem with hard time windows and stochastic service times. Eur J Oper Res 249(1):55–66

Feillet D, Dejax P, Gendreau M, Gueguen C (2004) An exact algorithm for the elementary shortest path problem with resource constraints: application to some vehicle routing problems. Networks 44(3):216–229

Gendreau M, Jabali O, Rei W (2014) Stochastic vehicle routing problems. Vehicle routing: problems, methods, and applications, 2nd edn, Chapter 8. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, pp 213–239

Gendreau M, Laporte G, Séguin R (1995) An exact algorithm for the vehicle routing problem with stochastic demands and customers. Transp Sci 29(2):143–155

Irnich S, Desaulniers G (2005) Shortest path problems with resource constraints, Chapter 2, GERAD 25th Anniversary Series. Springer, Berlin, pp 33–65

Irnich S, Villeneuve D (2006) The shortest-path problem with resource constraints and k-cycle elimination for $k \geq 3$. INFORMS J Comput 18(3):391–406

Jepsen M, Petersen B, Spoorendonk S, Pisinger D (2008) Subset-row inequalities applied to the vehicle-routing problem with time windows. Oper Res 56(2):497–511

Jula H, Dessouky M, Ionnou PA (2006) Truck route planning in nonstationary stochastic networks with time windows at customer locations. IEEE Trans Intell Transp Syst 7(1):51–62

Kenyon AS, Morton DP (2003) Stochastic vehicle routing with random travel times. Transp Sci 37(1):69–82

Lambert V, Laporte G, Louveaux FV (1993) Designing collection routes through bank branches. Comput Oper Res 20(7):783–791

Laporte G, Louveaux FV, Hamme L (2002) An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. Oper Res 50(3):415–423

Laporte G, Louveaux FV, Mercure H (1992) The vehicle routing problem with stochastic travel times. Transp Sci 26(3):161–170

Lee C, Lee K, Park S (2012) Robust vehicle routing problem with deadlines and travel time/demand uncertainty. J Oper Res Soc 60:1294–1306

Lei H, Laporte G, Guo B (2012) A generalized variable neighborhood search heuristic for the capacitated vehicle routing problem with stochastic service times. Top 20:99–118

Li X, Tian P, Leung SCH (2010) Vehicle routing problems with time windows and stochastic travel and service times: models and algorithm. Int J Prod Econ 125(1):137–145

Lübbecke ME, Desrosiers J (2005) Selected topics in column generation. Oper Res 53(6):1007–1023

Righini G, Salani M (2008) New dynamic programming algorithms for the resource constrained elementary shortest path problem. Networks 51(3):155–170

Solomon MM (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. Oper Res 35:254–265

Souyris S, Cortés CE, Ordóñez F, Weintraub A (2013) A robust optimization approach to dispatching technicians under stochastic service times. Optim Lett 7(7):1549–1568

Sungur I, Ren Y, Ordóñez F, Dessouky M, Zhong H (2010) A model and algorithm for the courier delivery problem with uncertainty. Transp Sci 44(2):193–205

Taş D, Dellaert N, van Woensel T, de Kok T (2012) Vehicle routing problem with stochastic travel times including soft time windows and service costs. Technical report, Beta Research School for Operations Management and Logistics. WP 364

Taş D, Gendreau M, Dellaert N, van Woensel T, de Kok A (2014) Vehicle routing with soft time windows and stochastic travel times: a column generation and branch-and-price solution approach. Eur J Oper Res 236(3):789–799

Wang X, Regan CA (2001) Assignment models for local truckload trucking problems with stochastic service times and time window constraints. Transp Res Rec 1771:61–68

Wellman M, Ford M, Larson K (1995) Path planning under time-dependent uncertainty. In: Proceedings of the eleventh conference on uncertainty in artificial intelligence, August 18–20, 1995. Montréal, pp 532–539