Submitted to *Transportation Science*
manuscript (Please, provide the mansucript number!)

An Adaptive Memory Programming Framework for the Robust Capacitated Vehicle Routing Problem

Chrysanthos E. Gounaris
Department of Chemical Engineering, Carnegie Mellon University, PA 15213, USA, gounaris@cmu.edu

Panagiotis P. Repoussis
Howe School of Technology Management, Stevens Institute of Technology, Hoboken, NJ 07030, USA, prepouss@stevens.edu

Christos D. Tarantilis
Department of Management Science & Technology, Athens University of Economics & Business, Greece, tarantil@aueb.gr

Wolfram Wiesemann
Imperial College Business School, Imperial College London, London, SW7 2AZ, UK, ww@imperial.ac.uk

Christodoulos A. Floudas
Department of Chemical and Biological Engineering, Princeton University, NJ 08544, USA, floudas@princeton.edu

We present an Adaptive Memory Programming (AMP) metaheuristic to address the Robust Capacitated Vehicle Routing Problem under demand uncertainty. Contrary to its deterministic counterpart, the robust formulation allows for uncertain customer demands, and the objective is to determine a minimum cost delivery plan that is feasible for all demand realizations within a prespecified uncertainty set. A crucial step in our heuristic is to verify the robust feasibility of a candidate route. For generic uncertainty sets, this step requires the solution of a convex optimization problem, which becomes computationally prohibitive for large instances. We present two classes of uncertainty sets for which route feasibility can be established much more efficiently. While we discuss our implementation in the context of the AMP framework, our techniques readily extend to other metaheuristics. Computational studies on standard literature benchmarks with up to 483 customers and 38 vehicles demonstrate that the proposed approach is able to quickly provide high quality solutions. In the process, we obtain new best solutions for a total of 123 benchmark instances.

Key words: Vehicle Routing; Robust Optimization; Adaptive Memory Programming
History: Received September 2012

1. Introduction

Vehicle routing problems arise in a large variety of practical contexts, particularly in the areas of freight transportation and logistics. In broad terms, vehicle routing problems concern the distribution of goods and/or services between production facilities, distribution centers and end-customers. Numerous variants and applications of this problem have been proposed in the literature, differing in the considered time scale, the objectives to be optimized and the operational constraints involved (Baldacci et al. 2012, 2010, Cordeau et al. 2007, Golden et al. 2008, Toth and Vigo 2002).

Despite these differences, all vehicle routing problems share in common that they determine an optimal assignment of customer orders to a fleet of vehicles, as well as the sequencing of deliveries over a prespecified time horizon. Most frequently, the objective is to minimize the transportation costs, expressed in terms of onetime (*e.g.*, fleet size) and/or recurring costs (*e.g.*, distance traveled).

One of the most extensively studied classes of vehicle routing problems is the Capacitated Vehicle Routing Problem (CVRP), which has been investigated for more than five decades (Laporte 2009). The CVRP concerns the cost-optimal delivery of a product from a single depot to a set of customers through a number of capacity-constrained vehicles. Traditionally, the literature on the CVRP assumes that the problem data (*e.g.*, the customer demands, service times, vehicle capacities and transportation costs) is known with certainty at the time the problem is solved. However, in many real-life applications this data is subject to significant uncertainty, and their precise values are only observed gradually during the execution of a delivery plan. Anticipating this uncertainty at the design stage is crucial in order to determine realistic delivery plans and avoid severe penalties—both contractually and in terms of lost customer goodwill—for failing to provide a reliable service.

Traditionally, decision problems with uncertain problem data are formulated as stochastic programs (Birge and Louveaux 2011, Prékopa 1995, Shapiro et al. 2009) or as Markov decision processes (Bertsekas 2007, Puterman 1994). Both methodologies model the uncertain problem data as random variables that follow a known distribution. The goal is to optimize a risk measure (such as the expected value, the variance or the conditional value-at-risk of some cost function), subject to the satisfaction of side constraints (either almost surely or with high probability). The decisions may need to be chosen before the uncertain problem data is known (so-called *here-and-now* decisions), or they can adapt to the observed values of the problem data (so-called *wait-and-see* decisions). While stochastic programming and Markov decision processes have been very successful at addressing a wide variety of uncertainty-affected decision problems, they suffer from two shortcomings. Firstly, they assume that the probability distribution governing the uncertain problem data is known precisely, which is rarely the case in practice. Secondly, both methodologies are affected by the curse of dimensionality, which impacts their computational tractability. In the context of the CVRP, stochastic programming and Markov decision processes are thus primarily suited for small and medium-sized problem instances where sufficient historical information is available to estimate the probability distribution underlying the uncertain problem data.

In recent years, robust optimization has been proposed as an alternative paradigm to address both of these shortcomings. Similar to stochastic programming and Markov decision processes, robust optimization models the uncertain problem data as random variables. However, robust optimization assumes that only partial information about the distribution of the uncertain problem data is available, for example its support, symmetry properties or some of the moments (Ben-Tal and Nemirovski 1999, Bertsimas and Sim 2004, Kuhn et al. 2011). Robust optimization determines solutions that perform best either in expectation, with high probability or almost surely, assuming that the problem parameters are governed by the worst probability distribution that is compatible with the available information. It has been shown that robust optimization problems can be reformulated as non-stochastic models in which the uncertain problem parameters are assumed to reside in some *uncertainty set*, and where the goal is to optimize in view of the worst possible parameter setting within this uncertainty set (Chen et al. 2010, Delage and Ye 2010). The resulting non-stochastic models often enjoy similar tractability properties as the deterministic version of the problem, which makes robust optimization very attractive from a computational viewpoint. In the context of the CVRP, robust optimization may be particularly well-suited to address larger problem instances where little or no historical information about the uncertain problem parameters is available. For reviews of the robust optimization literature, we refer to Ben-Tal et al. (2009), Bertsimas and Thiele (2006), Bertsimas et al. (2011), Li et al. (2011a) and Li et al. (2011b).

Unlike deterministic variants of the vehicle routing problem, which have been studied extensively, vehicle routing under uncertainty has received much less attention in the literature. To date, most of the contributions to vehicle routing under uncertainty employ the stochastic programming methodology (Birge and Louveaux 2011, Prékopa 1995, Shapiro et al. 2009). Recent contributions to vehicle routing with stochastic customer demands include Yang et al. (2000), where routes are equipped with preventive restocking points for the vehicles, Erera et al. (2010), Laporte et al. (2002) and Secomandi and Margot (2009), which consider return trips to the depot whenever a customer demand cannot be served, and Ak and Erera (2007), where customers can be swapped reactively between vehicles. Problems with stochastic travel times are studied in Kenyon and Morton (2003), while Adelman (2004) and Kleywegt et al. (2004) combine the vehicle routing problem with aspects of stochastic inventory control. Bent and Hentenryck (2004), Goodson et al. (2013), Hvattum et al. (2006) and Smith et al. (2010) study dynamic problems where customer requests arrive over time and the very presence of customers is subject to uncertainty. For reviews of the stochastic vehicle routing literature, we refer to Cordeau et al. (2007), Gendreau et al. (1996) and Toth and Vigo (2002).

To our knowledge, the first solution procedure for the robust CVRP (RCVRP) with uncertain customer demands and travel times has been proposed in Sungur et al. (2008). The authors determine vehicle routes that satisfy the vehicle capacities and specified delivery time windows for all possible realizations of the uncertain problem data. Variants of the model were applied to a bioterrorism emergency planning problem (Shen et al. 2009) and a courier delivery problem (Sungur et al. 2010). The formulation from Sungur et al. (2008) optimizes in view of the scenario where all customer demands and travel times attain their worst-case realizations simultaneously, which may be overly conservative for practical purposes. Ordóñez (2010) alleviates this issue by considering upper bounds on the customer demands and travel times experienced by any vehicle. Recently, Gounaris et al. (2013) considered the RCVRP where the customer demands can be supported on a generic polyhedron. They develop exact robust counterparts for a number of well-known formulations for the deterministic CVRP, and they propose algorithms to efficiently derive Robust Rounded Capacity Inequalities (Gounaris et al. 2012) for use as cutting planes in a branch-and-cut solution procedure. Finally, Agra et al. (2013) study travel time uncertainty in the context of the vehicle routing problem with time windows. They present two robust formulations and apply them to a ship routing and scheduling problem.

Both the deterministic and the robust CVRP can be cast as mixed-integer linear optimization problems. The CVRP generalizes the well-known Travelling Salesman Problem but—unlike the latter, where instances with thousands of nodes can often be solved to optimality—CVRP instances with more than one hundred customers still pose a formidable challenge for exact solution procedures (Baldacci et al. 2010). Exact solution methods for the CVRP are typically based on extensions of branch-and-cut (Lysgaard et al. 2004), branch-and-cut-and-price (Fukasawa et al. 2006) or set partitioning approaches (Baldacci et al. 2008). Due to the computational challenges involved in solving the CVRP, a number of heuristic methods have been developed, such as iterative improvement local search algorithms (Toth and Vigo 2003, Xu and Kelly 1996), evolutionary algorithms (Prins 2004, Reimann et al. 2004) and hybrid metaheuristic schemes, such as Memetic Algorithms (Nagata and Bräysy 2009) and Adaptive Memory Programming (Tarantilis 2005). These methods have made significant contributions towards solving large-scale and very large-scale problem instances (Kytöjoki et al. 2007, Li et al. 2005).

In this paper, we propose an Adaptive Memory Programming (AMP) metaheuristic to solve the RCVRP under demand uncertainty. AMP is a general-purpose metaheuristic framework that focuses on the exploitation of strategic memory components (Glover 1997). It has been successfully applied to a range of difficult combinatorial optimization problems, in particular in the vehicle routing domain (Repoussis and Tarantilis 2010, Rochat and Taillard 1995, Tarantilis and Kiranoudis 2002, Tarantilis 2005). Based on the intuition that high-quality locally optimal solutions share common features and components (such as common customer visiting sequences), the goal of AMP is to exploit a set of long term memories for the iterative construction of new provisional solutions. These solutions are used as the basis for restarting and intensifying the search, while adaptive learning mechanisms are applied to update and manipulate the memory structures (Taillard 2001).

Despite their overall sophistication with regards to constructing and recombining solutions, most metaheuristic approaches for vehicle routing problems adopt local search procedures to improve the quality of the obtained solutions. The search involves repeated local moves, that is, transitions from one solution to another reachable (neighboring) one. A critical aspect during this process is to verify route feasibility, and in some cases quantify the degree of route infeasibility, in order to evaluate the effect of a local move. The AMP framework proposed in this paper, for example, grows and manipulates a reference set of elite solutions by means of performing search trajectories that iteratively emerge from the provisional solutions. This process results in the repeated construction of candidate routes whose robust feasibility needs to be verified.

In the deterministic CVRP, verifying the feasibility of a route requires summing up the customer demands along the route and checking whether the sum exceeds the capacity of the vehicle. In the RCVRP, on the other hand, verifying route feasibility requires the solution of a convex optimization problem, which becomes computationally demanding for large instances. It is therefore imperative to identify properties that allow us to establish route feasibility more efficiently. To that end, this paper studies two classes of uncertainty sets for which robust route feasibility can be established very efficiently. While we discuss our implementation in the context of an AMP metaheuristic, our methods readily extend to other metaheuristics. We present numerical results on well-known benchmark data sets from the literature that illustrate the computational performance of our proposed solution approach.

We can summarize the contributions of this paper as follows.

1. We develop an AMP metaheuristic for the RCVRP under demand uncertainty. To our best knowledge, this is the first application of a metaheuristic framework on a robust variant of a vehicle routing problem.

2. We extend earlier AMP frameworks for the deterministic CVRP and discuss a number of implementation enhancements that appear to improve performance in the RCVRP setting. Most prominently, (a) we present a novel mechanism to identify and select elite solution components of varying size, and (b) we employ a new augmented objective function that facilitates the exploration of both feasible and infeasible regions of the solution space during the local search process.

3. A crucial step in any RCVRP heuristic is the verification of route feasibility. We present two classes of uncertainty sets that allow us to establish route feasibility in a very efficient way. Our results generalize earlier findings on disjoint budget uncertainty sets to the broader class of inclusion-constrained budget uncertainty sets, which allow us to model a wider range of dependencies between the customer demands.

4. We demonstrate the effectiveness of the proposed framework via a comprehensive study on small-, medium- and large-scale benchmark instances from the literature. In the process, we identify new best solutions for a total of 123 instances.

5. We demonstrate the usefulness of our heuristic for initializing exact search procedures. By feeding high-quality heuristic solutions into a previously developed branch-and-cut algorithm, we

solve 3 benchmark instances to certified optimality for the first time, as well as obtain new best lower bounds for an additional 31 instances.

The remainder of the paper is structured as follows. Section 2 defines the RCVRP and introduces some notation. Section 3 discusses the verification of route feasibility, which is critical for the computational efficiency of the AMP framework presented in Section 4. Section 5 reports computational results, and we provide some concluding remarks in Section 6.

2. The Robust Capacitated Vehicle Routing Problem

An instance of the deterministic CVRP is described by a complete, directed and weighted graph $G=(V,A,c)$ with node set $V=\{0,1,\ldots,n\}$, arc set $A=V\times V$ and nonnegative arc weights $c: A\mapsto \mathbb{R}_+$. Node 0 represents the unique depot, whereas the nodes $i\in\{1,\ldots,n\}$ refer to customers with known nonnegative demands $\boldsymbol{q}_i\in\mathbb{R}_+$ for the product. We denote the vector of all customer demands by $\boldsymbol{q}\in\mathbb{R}^n_+$, and we refer to the set of customer nodes as $V_C=V\setminus\{0\}$. The arc weight $c(i,j)$ describes the transportation costs incurred by any vehicle that traverses the arc $(i,j)\in A$ (*e.g.*, fuel, labor and insurance). The depot is equipped with m homogeneous vehicles $K=\{1,\ldots,m\}$, each of which can transport up to Q units of the product.

A candidate solution to the deterministic CVRP is given by a *set of routes* $\boldsymbol{R}=(\boldsymbol{R}_1,\ldots,\boldsymbol{R}_m)$, where $\boldsymbol{R}_k=(\boldsymbol{R}_{k,0},\boldsymbol{R}_{k,1},\ldots,\boldsymbol{R}_{k,n_k},\boldsymbol{R}_{k,n_k+1})$ represents the route of the k-th vehicle. For each route $\boldsymbol{R}_k$, we have $\boldsymbol{R}_{k,0}=\boldsymbol{R}_{k,n_k+1}=0$, that is, the route has to start and end at the depot node. The nodes $\boldsymbol{R}_{k,1},\ldots,\boldsymbol{R}_{k,n_k}\in V_C$ represent the customers served on the k-th route in the order in which they are being served. We stipulate that a route set $\boldsymbol{R}$ forms a partition of the customer set V_C, that is, each customer is served on exactly one route, and neither split deliveries nor unserved customer demands are permitted. We say that a route set $\boldsymbol{R}$ is *feasible* if each route $\boldsymbol{R}_k$ satisfies the capacity constraint of the k-th vehicle, that is, if $\sum_{l=1}^{n_K}\boldsymbol{q}_{\boldsymbol{R}_{k,l}}\leq Q$ for all $k\in K$. Sometimes we additionally impose route duration constraints. In these cases, we interpret $c(i,j)$ as the travel time from node i to j, and we require the cumulative duration of each route $\boldsymbol{R}_k$ to be bounded above by $T\in\mathbb{R}_+$, that is, $\sum_{l=0}^{n_k}c(\boldsymbol{R}_{k,l},\boldsymbol{R}_{k,l+1})+\sum_{l=1}^{n_k}s(\boldsymbol{R}_{k,l})\leq T$ for all $k\in K$, where $s(i)$, $i\in V_C$, denotes the service time of customer i. In either case, we denote by $\mathcal{R}(\boldsymbol{q})$ the set of all feasible route sets for the demand vector $\boldsymbol{q}$. We can then define the CVRP as follows.

$$\begin{aligned} \underset{\boldsymbol{R}}{\text{minimize}} \quad & c(\boldsymbol{R})=\sum_{k\in K}\sum_{l=0}^{n_k}c(\boldsymbol{R}_{k,l},\boldsymbol{R}_{k,l+1}) \\ \text{subject to} \quad & \boldsymbol{R}\in\mathcal{R}(\boldsymbol{q}). \end{aligned} \tag{CVRP}$$

In this problem, the objective function minimizes the cumulative transportation costs (or travel times) along all routes. The constraint ensures that the route set is feasible; that is, all routes start and end at the depot node, all customer demands are being served, there are no split deliveries, all routes complete within the duration limit (if imposed), and the capacity constraints are satisfied for all vehicles.

Contrary to its deterministic counterpart, the robust CVRP assumes that the customer demands $\boldsymbol{q}$ are no longer known precisely, but that they are merely known to lie in some uncertainty set $\mathcal{Q} \subseteq \mathbb{R}^n_+$. In the absence of historical data, this uncertainty set can be chosen based on subjective information from domain experts, for example as polyhedra or ellipsoids around some nominal demand values. Ben-Tal and Nemirovski (2000) have shown that such uncertainty sets can successfully immunize optimization problems against imprecise knowledge of the problem parameters. If, on the hand, historical data about the customer demands is available, then the uncertainty set can be designed using statistical results. The construction of suitable uncertainty sets has been discussed, amongst others, by Bandi and Bertsimas (2012), Ben-Tal et al. (2009), Chen et al. (2010), Gounaris et al. (2013), Li et al. (2011a,b) and Wiesemann et al. (2013). We will revisit this point in Section 3. We remark that the uncertainty set $\mathcal{Q}$ is usually not rectangular, that is, it does not typically admit the possibility that all customer demands attain their maximum values simultaneously. Once the uncertainty set $\mathcal{Q}$ has been chosen, we follow the robust optimization paradigm and seek to determine a set of routes that remains feasible for all possible demand realizations $\boldsymbol{q} \in \mathcal{Q}$. To this end, we call a route set *robust feasible* if each route $\boldsymbol{R}_k$ satisfies the capacity constraint of the k-th vehicle for all possible demand realizations, that is, $\sum_{l=1}^{n_K} \boldsymbol{q}_{\boldsymbol{R}_{k,l}} \leq Q$ for all $k \in K$ and all $\boldsymbol{q} \in \mathcal{Q}$. Again, we may sometimes impose additional constraints on the route durations. In analogy to the set $\mathcal{R}(\boldsymbol{q})$, we denote by $\mathcal{R}(\mathcal{Q})$ the set of all robust feasible route sets for the uncertainty set $\mathcal{Q}$. The robust CVRP can then be formulated as follows.

$$\begin{aligned} \underset{\boldsymbol{R}}{\text{minimize}} \quad & c(\boldsymbol{R}) = \sum_{k \in K} \sum_{l=0}^{n_k} c(\boldsymbol{R}_{k,l}, \boldsymbol{R}_{k,l+1}) \\ \text{subject to} \quad & \boldsymbol{R} \in \mathcal{R}(\mathcal{Q}). \end{aligned} \tag{RCVRP}$$

The objective function is the same as in the deterministic case, but the constraint now ensures that the route set is robust feasible, that is, all routes start and end at the depot node, all customer demands are being served, there are no split deliveries, all routes complete within the duration limit (if imposed), and the capacity constraints are satisfied for all vehicles under any possible demand realization $\boldsymbol{q} \in \mathcal{Q}$.

In the robust CVRP, the set of routes $\boldsymbol{R}$ is chosen as a here-and-now decision, that is, $\boldsymbol{R}$ is selected before the realization of the uncertain demands is observed. Alternatively, one could envision dynamic formulations in which the routes are adapted whenever some of the uncertain demands have been observed. Examples include detours to the depot that are inserted whenever a customer's demand exceeds the remaining vehicle load, the option to preventively restock a vehicle during its route or the decision to reactively swap customers between vehicle routes (see Section 1).

The robust CVRP constitutes a conservative formulation for the CVRP under demand uncertainty since it requires feasibility for all possible demand realizations $\boldsymbol{q} \in \mathcal{Q}$. One can envisage alternative models that require satisfaction of the vehicle capacity constraints in expectation or with a sufficiently high probability. Linearity of the expectation operator implies that formulations which meet the capacity constraints in expectation are equivalent to instances of the deterministic CVRP where we identify the customer demand vector $\boldsymbol{q}$ with the vector of expected customer demands. There is a strong connection between the robust CVRP and chance-constrained vehicle routing problems in which the vehicle capacities must be satisfied with a pre-specified probability. Indeed, by choosing the uncertainty set $\mathcal{Q}$ appropriately, we can interpret the robust CVRP as a chance-constrained vehicle routing problem in which the probability distribution governing the customer demands is itself subject to uncertainty, see Gounaris et al. (2013).

3. Verifying the Robust Feasibility of Vehicle Routes

In this section, we assume that we are given a set of routes $\boldsymbol{R}$ (*i.e.*, a partition of the customer set), and we wish to establish whether $\boldsymbol{R}$ is robust feasible with respect to the capacity constraints. To this end, we need to verify the satisfaction of the following m semi-infinite constraints:

$$\sum_{l=1}^{n_k} \boldsymbol{q}_{\boldsymbol{R}_{k,l}} \leq Q \qquad \forall k \in K, \, \forall \boldsymbol{q} \in \mathcal{Q}$$

By construction, these constraints are satisfied if and only if

$$\max_{\boldsymbol{q} \in \mathcal{Q}} \sum_{l=1}^{n_k} \boldsymbol{q}_{\boldsymbol{R}_{k,l}} \leq Q \qquad \forall k \in K, \tag{1}$$

that is, if for each route $k \in K$, the maximum cumulative customer demands $\sum_{l=1}^{n_k} \boldsymbol{q}_{\boldsymbol{R}_{k,l}}$ over all demand realizations $\boldsymbol{q} \in \mathcal{Q}$ does not exceed the vehicle's capacity Q. We can assume that $\mathcal{Q} \subseteq [\mathbf{0}, Q \cdot \mathbf{e}]$, where $\mathbf{e} \in \mathbb{R}^n$ denotes the vector of all ones, since otherwise the RVRP instance would be trivially infeasible. Moreover, we can without loss of generality assume that the uncertainty set $\mathcal{Q}$ is convex. In fact, since the objective function of the embedded maximization problem in (1) is linear, the optimal value of this problem does not change if we replace the feasible region with its convex hull. Thus, in general, verifying the robust feasibility of a route set requires the solution of m convex optimization problems.

In the following, we assume that the uncertainty set $\mathcal{Q}$ is a polyhedron. In this case, the embedded maximization problem in (1) constitutes a linear program. If there is no exploitable sparsity structure, the solution of such linear programs with M inequality constraints and N variables requires on average $\mathcal{O}(MN^2)$ arithmetic operations using interior point methods or variants of the simplex algorithm, see Bertsimas and Tsitsiklis (1997) or Boyd and Vandenberghe (2004). Moreover, if we know the maximum of $\sum_{l \in S_1} \boldsymbol{q}_l$ over $\mathcal{Q}$ and we are interested in the maximum of $\sum_{l \in S_2} \boldsymbol{q}_l$ over $\mathcal{Q}$, where $S_1, S_2 \subseteq V_C$ are "sufficiently similar" sets, then we can solve the second linear program much faster if we warm-start the algorithm from the solution of the first linear program.

In the remainder of this section, we consider two special classes of polyhedral uncertainty sets—budget and factor model uncertainty sets—that are relevant in practice and that allow us to verify the robust feasibility of vehicle routes much faster than for generic polyhedral uncertainty sets.

3.1. Budget Uncertainty Sets

We consider budget uncertainty sets of the form

$$\mathcal{Q} = \left\{ \boldsymbol{q} \in \mathbb{R}^n_+ : \boldsymbol{q} \in \left[\underline{\boldsymbol{q}}, \overline{\boldsymbol{q}}\right], \ \sum_{i \in B_l} \boldsymbol{q}_i \leq \boldsymbol{b}_l \text{ for } l = 1, \ldots, L \right\}, \tag{2}$$

which constitute the intersection of the n-dimensional hyperrectangle $\left[\underline{\boldsymbol{q}}, \overline{\boldsymbol{q}}\right]$ with L budget constraints involving customer subsets $B_l \subseteq V_C$. In this definition, $\underline{\boldsymbol{q}}, \overline{\boldsymbol{q}} \in \mathbb{R}^n_+$, $L \in \mathbb{N}$, $B_l \subseteq V_C$ and $\boldsymbol{b}_l \in \mathbb{R}_+$, $l \in \mathcal{L} = \{1, \ldots, L\}$, constitute parameters which need to be selected by the modeler. To exclude empty uncertainty sets, we require that $\underline{\boldsymbol{q}} \leq \overline{\boldsymbol{q}}$ component-wise and $\boldsymbol{b}_l \geq \sum_{i \in B_l} \underline{\boldsymbol{q}}_i$ for all $l \in \mathcal{L}$. Without loss of generality, we further assume that $B_l \neq B_{l'}$ for all $l, l' \in \mathcal{L}$, $l \neq l'$.

Budget uncertainty sets of the type (2) reflect the belief that the demand $\boldsymbol{q}_i$ of customer $i \in V_C$ can vary between the lower and upper bounds $\underline{\boldsymbol{q}}_i$ and $\overline{\boldsymbol{q}}_i$. However, unless the customer demands exhibit perfect correlations, it is unlikely that all customer demands simultaneously attain extreme values. The uncertainty set therefore imposes limits on the cumulative customer demands over various subsets B_l of the customer set V_C. The parameters of the uncertainty set can be specified using domain knowledge or statistical arguments. For example, under mild technical conditions, the central limit theorem shows that if the customer demands $\boldsymbol{q}_i$, $i \in V_C$, constitute independent and identically distributed random variables with mean μ and variance σ^2, then for sufficiently large $|B_l|$, the cumulative customer demands in the budget set B_l satisfy

$$\sum_{i \in B_l} \boldsymbol{q}_i \ \leq \ |B_l| \, \mu + \sqrt{|B_l|} \, \sigma \, \Phi^{-1}(1 - \beta) \tag{3}$$

with probability β. Here, $\Phi^{-1}(\cdot)$ denotes the inverse cumulative distribution function of the standard normal distribution. One readily verifies that the inequality (3) can be incorporated as a budget constraint of the form $\sum_{i \in B_l} \boldsymbol{q}_i \leq \boldsymbol{b}_l$ in the uncertainty set (2).

Budget uncertainty sets have been first proposed by Bertsimas and Sim (2004) for the special case of a single budget constraint; that is, the case where $L = 1$ and $B_1 = V_C$. It has been shown in Gounaris et al. (2013) that under these special conditions the RCVRP reduces to an instance of the deterministic CVRP. In the remainder of this section, we therefore focus on instances of the uncertainty set (2) where $L > 1$, in which case the RCVRP no longer admits an equivalent reformulation as a deterministic CVRP instance.

Generic Budget Uncertainty Sets. The maximization of $\sum_{i\in S} \boldsymbol{q}_i$ over a budget uncertainty set of the form (2) amounts to the solution of a fractional packing problem (Garg and Könemann 2007). Ignoring polylogarithmic factors and assuming that $L \le n$, a $(1+\epsilon)$-approximation to the fractional packing problem can be determined in time $\mathcal{O}(\epsilon^{-2}Ln)$, see Young (2001).

Inclusion-Constrained Budget Uncertainty Sets. We now consider a subclass of budget uncertainty sets where the budget sets B_l, $l \in \mathcal{L}$, satisfy the following *inclusion condition*:

(**IC**) For all $l, l' \in \mathcal{L}$, the budget sets B_l and $B_{l'}$ satisfy $B_l \subseteq B_{l'}$, $B_{l'} \subseteq B_l$ or $B_{l'} \cap B_l = \emptyset$.

Thus, for any two budget sets B_l and $B_{l'}$ we have that either one set is a subset of the other one, or the two sets are disjoint. An example of a budget uncertainty set that satisfies condition (**IC**) is:

$$\mathcal{Q} = \left\{ \boldsymbol{q} \in \mathbb{R}^4_+ : \boldsymbol{q} \in [1,3]^3, \ \boldsymbol{q}_1 + \boldsymbol{q}_2 \le 3, \ \boldsymbol{q}_3 + \boldsymbol{q}_4 \le 3, \ \boldsymbol{q}_1 + \boldsymbol{q}_2 + \boldsymbol{q}_3 + \boldsymbol{q}_4 \le 5 \right\}$$

One readily verifies that (**IC**) establishes a partial order on the budget sets $\{B_l\}_{l\in\mathcal{L}}$ with respect to the subset relation $\subseteq$, with the additional property that incomparable budget sets are disjoint. The inclusion condition is closely related to the notion of laminar families in combinatorial optimization (Schrijver 2003), and a similar condition has been used recently to study distributionally robust optimization problems (Wiesemann et al. 2013, Xu and Mannor 2012). Without loss of generality, we assume from now on that the budget sets are ordered according to this subset relation, that is, for all $l, l' \in \mathcal{L}$, $l \le l'$, we have $B_l \subseteq B_{l'}$ or $B_l \cap B_{l'} = \emptyset$.

We want to show that if (**IC**) is satisfied, then we can maximize $\sum_{i\in S} \boldsymbol{q}_i$ over the uncertainty set (2) by solving a maximum flow problem. A maximum flow problem is defined through a directed, weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \gamma)$ whose node set $\mathcal{V}$ contains a designated source node s and sink node t. The arcs $\mathcal{A} \subseteq \mathcal{V} \times \mathcal{V}$ can be interpreted as pipes with nonnegative capacities $\gamma(i,j) \in \mathbb{R}_+$, $(i,j) \in \mathcal{A}$. The goal is to maximize the cumulative flow from the source to the sink, subject to flow conservation at the intermediate nodes $i \in \mathcal{V} \setminus \{s,t\}$ and satisfaction of the arc capacities γ. For further information about maximum flow problems, we refer to Ahuja et al. (1993).

Let us fix an instance of the uncertainty set (2), together with a nonempty set of customer nodes $S \subseteq V_C$. We construct an instance of the maximum flow problem as follows. The node set $\mathcal{V}$ contains the source node s and the sink node t, as well as nodes s_i for each customer $i \in S$ and nodes β_l^0 and β_l^1 for each budget $l \in \mathcal{L}$. We connect the source node s to each customer node s_i through an arc with capacity $\gamma(s, s_i) = \overline{\boldsymbol{q}}_i - \underline{\boldsymbol{q}}_i$. These links ensure that none of the customer demands exceeds its upper bound. Now we consider the budget sets B_l in the order of ascending subscripts l. For those customers $i \in B_l$ that do not take part in any of the previous budget sets $B_1, \ldots, B_{l-1}$, we introduce an arc from s_i to β_l^0 with capacity $\gamma(s_i, \beta_l^0) = \infty$.[1] We now consider the previous budget sets $B_{l'}$, $l' = 1, \ldots, l-1$, that share some customers with B_l (that is, $B_{l'} \cap B_l \neq \emptyset$). For each of these budget sets we introduce an arc from $\beta_{l'}^1$ to β_l^0 with capacity $\gamma(\beta_{l'}^1, \beta_l^0) = \infty$, provided that there is no budget set $B_{l''} \neq B_{l'}, B_l$ such that $B_{l'} \subseteq B_{l''} \subseteq B_l$. Intuitively, we thus connect the node β_l^0 corresponding to the budget set B_l with all nodes $\beta_{l'}$ that correspond to budget sets $B_{l'}$ that are "immediate predecessors" of B_l with respect to the subset relation $\subseteq$. We also add an arc from β_l^0 to β_l^1 with capacity $\gamma(\beta_l^0, \beta_l^1) = \boldsymbol{b}_l - \sum_{i \in B_l} \underline{\boldsymbol{q}}_i$. This arc ensures that the l-th budget constraint is satisfied. Once we have added arcs for all budget sets B_l, $l \in \mathcal{L}$, we introduce arcs from β_l^1 to the sink node t for all those budget sets B_l that satisfy $B_l \cap B_{l'} = \emptyset$ for all $l' = l+1, \ldots, L$, that is, for all those budget sets B_l that are maximal with respect to the $\subseteq$ relation. For each of these arcs, we set the capacity to $\gamma(\beta_l^1, t) = \infty$.

We now formalize this idea. For a given instance of the uncertainty set (2), we define

$$\mathcal{P}(l) = \{l' \in \{1, \ldots, l-1\} : B_{l'} \subseteq B_l \text{ and } B_{l'} \cap B_{l''} = \emptyset \text{ for all } l'' \in \{l'+1, \ldots, l-1\}\} \qquad \text{for } l \in \mathcal{L}$$

as the index set of *direct predecessor budget sets* of B_l, that is, those budget sets $B_{l'} \subseteq B_l$ for which there is no budget set $B_{l''} \neq B_{l'}, B_l$ such that $B_{l'} \subseteq B_{l''} \subseteq B_l$. We also denote by $\mathcal{T} = \{l \in \mathcal{L} : B_l \cap B_{l'} = \emptyset \text{ for all } l' \in \{l+1, \ldots, L\}\}$ the set of *terminal budget sets* B_l, that is, those budget sets that are not contained in any other budget set.

For a nonempty customer set $S \subseteq V_C$, the node set $\mathcal{V}$ of $\mathcal{G}$ is given by

$$\mathcal{V} = \{s, t\} \cup \{s_i : i \in S\} \cup \left\{\beta_l^0, \beta_l^1 : l \in \mathcal{L}\right\}.$$

The arc set $\mathcal{A}$ is then given by the union of $\mathcal{A}_0$, $\mathcal{A}_l$, $l \in \mathcal{L}$, and $\mathcal{A}_{L+1}$, where

$$\mathcal{A}_0 = \{(s, s_i) : i \in S\} \cup \left\{(s_i, t) : i \in S \setminus \bigcup_{l \in \mathcal{L}} B_l\right\}$$

with arc weights $\gamma(s, s_i) = \overline{\boldsymbol{q}}_i - \underline{\boldsymbol{q}}_i$ and $\gamma(s_i, t) = \infty$,

$$\mathcal{A}_l = \left\{(\beta_{l'}^1, \beta_l^0) : l' \in \mathcal{P}(l)\right\} \cup \left\{(s_i, \beta_l^0) : i \in B_l \setminus \bigcup_{l' < l} B_{l'}\right\} \cup \left\{(\beta_l^0, \beta_l^1)\right\} \qquad \text{for } l \in \mathcal{L}$$

with arc weights $\gamma(\beta_{l'}^1, \beta_l^0) = \gamma(s_i, \beta_l^0) = \infty$ and $\gamma(\beta_l^0, \beta_l^1) = \boldsymbol{b}_l - \sum_{i \in B_l} \underline{\boldsymbol{q}}_i$, as well as

$$\mathcal{A}_{L+1} = \left\{(\beta_l^1, t) : B_l \in \mathcal{T}\right\}$$

with arc weights $\gamma(\beta_l^1, t) = \infty$. The construction of the graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \gamma)$ is illustrated in Figure 1.

The following auxiliary result analyzes the structure of the graph $\mathcal{G}$.

[1] Whenever we assign a capacity of ∞ to an arc, this can be replaced with a sufficiently large finite number, for example $M = \sum_{i \in S} \left(\overline{\boldsymbol{q}}_i - \underline{\boldsymbol{q}}_i\right)$.

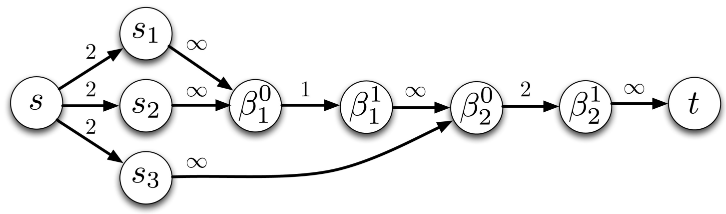

Figure 1 **Construction of the graph** $\mathcal{G}$ **for** $\mathcal{Q} = \left\{ \boldsymbol{q} \in [1,3]^3 : \boldsymbol{q}_1 + \boldsymbol{q}_2 \leq 3,\ \boldsymbol{q}_1 + \boldsymbol{q}_2 + \boldsymbol{q}_3 \leq 5 \right\}$ **and** $S = \{1,2,3\}$**.**

LEMMA 1. *Assume that condition (**IC**) is satisfied. Then, for each $i \in S$, the graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \gamma)$ contains exactly one path from the source s to the sink t via node s_i, and this path is of the form*

$$P_i = (s, s_i, \beta^0_{l_1}, \beta^1_{l_1}, \beta^0_{l_2}, \beta^1_{l_2}, \ldots, \beta^0_{l_{\sigma_i}}, \beta^1_{l_{\sigma_i}}, t),$$

where $l_1 < l_2 < \ldots < l_{\sigma_i}$ are the budgets sets B_l that contain customer i. Moreover, we have $\sigma_i = 0$ if and only if customer i does not participate in any budget set, that is, if $i \notin \bigcup_{l \in \mathcal{L}} B_l$.

Proof. If $i \in S \setminus \bigcup_{l \in \mathcal{L}} B_l$, then the only arc emanating from node s_i is $(s_i, t) \in \mathcal{A}_0$, and the assertion follows from the fact that $(s, s_i) \in \mathcal{A}_0$. Assume now that $i \in \bigcup_{l \in \mathcal{L}} B_l$ and that the graph $\mathcal{G}$ contains the path P_i, that is, $(s, s_i), (s_i, \beta^0_{l_1}), \ldots, (\beta^1_{l_{\sigma_i}}, t) \in \mathcal{A}$. By construction of the arc set $\mathcal{A}$, we have $i \in B_{l_1}$ since $(s_i, \beta^0_{l_1}) \in \mathcal{A}_{l_1}$. Likewise, we have $i \in B_{l_s}$, $s = 2, \ldots, \sigma_i$, because $(\beta^1_{l_{s-1}}, \beta^0_{l_s}) \in \mathcal{A}_{l_s}$. We thus conclude that $l_1 < l_2 < \ldots < l_{\sigma_i}$ are the budgets sets B_l that contain customer i.

To show the reverse direction, assume that customer $i \in S$ participates in the budget sets $B_{l_1}, B_{l_2}, \ldots, B_{l_{\sigma_i}}$, $l_1 < l_2 < \ldots < l_{\sigma_i}$, and that $i \notin B_{l'}$ for all $l' \in \mathcal{L} \setminus \{l_1, l_2, \ldots, l_{\sigma_i}\}$. In that case, the only arc emanating from s_i is $(s_i, \beta^0_{l_1}) \in \mathcal{A}_{l_1}$. Likewise, the definition of $\mathcal{P}(l_2)$ implies that the only arc emanating from $\beta^1_{l_1}$ is $(\beta^1_{l_1}, \beta^0_{l_2}) \in \mathcal{A}_{l_2}$. In the same way, one readily shows by induction over s that the only arc emanating from $\beta^1_{l_s}$, $s = 2, \ldots, \sigma_i - 1$, is $(\beta^1_{l_s}, \beta^0_{l_{s+1}}) \in \mathcal{A}_{l_{s+1}}$. Finally, we have $B_{l_{\sigma_i}} \in \mathcal{T}$, for otherwise $i \in B_{l'}$ for some $l' \in \{l_{\sigma_i} + 1, \ldots, L\}$. Hence, the only arc emanating from $\beta^1_{l_{\sigma_i}}$ is $(\beta^1_{l_{\sigma_i}}, t) \in \mathcal{A}_{L+1}$. We have thus shown that if customer $i \in S$ participates in the budget sets $B_{l_1}, B_{l_2}, \ldots, B_{l_\sigma}$, then P_i is the only (s,t)-path via node s_i. This completes the proof. $\square$

In the remainder of this section, we use the shorthand notation $(j,k) \in P_i$ to denote that (j,k) is an arc on the path P_i specified in Lemma 1, that is, $(j,k) \in \left\{ (s, s_i), (s_i, \beta^0_{l_1}), \ldots, (\beta^1_{l_{\sigma_i}}, t) \right\}$. We now show that there is a one-to-one correspondence between maximal flows in $\mathcal{G}$ and maximizers of $\sum_{i \in S} \boldsymbol{q}_i$ over the uncertainty set (2).

PROPOSITION 1. *Assume that condition (**IC**) is satisfied, and let $Q^\star$ denote the value of a maximal flow in the graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \gamma)$. Then $Q^\star + \sum_{i \in S} \underline{\boldsymbol{q}}_i$ is the maximum of $\sum_{i \in S} \boldsymbol{q}_i$ over (2).*

Proof. Assume that $\boldsymbol{q}^\star \in \mathbb{R}^n_+$ maximizes $\sum_{i \in S} \boldsymbol{q}_i$ over (2). For each customer $i \in S$, Lemma 1 ensures that there is a unique (s,t)-path via node s_i that is of the form $P_i =$

$(s, s_i, \beta^0_{l_1}, \beta^1_{l_1}, \beta^0_{l_2}, \beta^1_{l_2}, \ldots, \beta^0_{l_{\sigma_i}}, \beta^1_{l_{\sigma_i}}, t)$. We define an (s,t)-flow $f_i : \mathcal{A} \mapsto \mathbb{R}_+$ for each customer i via $f_i(j,k) = \boldsymbol{q}^\star_i - \underline{\boldsymbol{q}}_i$ for $(j,k) \in P_i$ and $f_i(j,k) = 0$ for $(j,k) \in \mathcal{A} \setminus P_i$. By construction, each (s,t)-flow $f_i(\cdot)$, $i \in S$, satisfies flow conservation and all arc capacities. Consider now the cumulative flow $f : \mathcal{A} \mapsto \mathbb{R}_+$ defined through $f(j,k) = \sum_{i \in S} f_i(j,k)$ for all $(j,k) \in \mathcal{A}$. By construction, $f(\cdot)$ satisfies flow conservation. Thus, we only have to check that all arc capacities are satisfied. Since the customer flows $f_i(\cdot)$, $i \in S$, do not share any of the arcs (s, s_i), (s_i, β^0_l) or (s_i, t), we only need to verify the satisfaction of the capacities of the arcs (β^0_l, β^1_l) and (β^1_l, t). For each such arc (β^0_l, β^1_l), we have

$$f(\beta^0_l, \beta^1_l) \;=\; \sum_{i \in S} f_i(\beta^0_l, \beta^1_l) \;=\; \sum_{i \in B_l} (\boldsymbol{q}^\star_i - \underline{\boldsymbol{q}}_i) \;\leq\; \boldsymbol{b}_l - \sum_{i \in B_l} \underline{\boldsymbol{q}}_i \;=\; \gamma(\beta^0_l, \beta^1_l).$$

Here, the first equality follows from the definition of $f(\cdot)$, the second one from the definition of the customer flows $f_i(\cdot)$ and Lemma 1, the inequality holds because $\boldsymbol{q}^\star \in \mathcal{Q}$, and the last equality follows from the definition of $\gamma(\beta^0_l, \beta^1_l)$. A similar argument shows that the capacities of the arcs (β^1_l, t) are satisfied as well. Thus, the flow $f(\cdot)$ is feasible and has the value $\sum_{i \in S} (\boldsymbol{q}^\star_i - \underline{\boldsymbol{q}}_i)$. We have therefore shown that the value of a maximal flow in the graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \gamma)$ is at least as large as the maximum of $\sum_{i \in S} (\boldsymbol{q}^\star_i - \underline{\boldsymbol{q}}_i)$ over (2).

We now show that the value of a maximal flow in $\mathcal{G}$ is also at most as large as the maximum of $\sum_{i \in S} (\boldsymbol{q}^\star_i - \underline{\boldsymbol{q}}_i)$ over (2). To this end, note that Lemma 1 implies that the set of all (s,t)-paths is given by $\{P_i\}_{i \in S}$. Due to the optimality of $\boldsymbol{q}^\star$, for every customer i we either have $\boldsymbol{q}^\star_i = \overline{\boldsymbol{q}}_i$ or there is a budget set B_l such that $i \in B_l$ and $\sum_{j \in B_l} \boldsymbol{q}^\star_j = \boldsymbol{b}_l$. Hence, for every path P_i, either $f_i(s, s_i) = \gamma(s, s_i)$ or $f_i(\beta^0_{l_s}, \beta^1_{l_s}) = \gamma(\beta^0_{l_s}, \beta^1_{l_s})$ for some $s = 1, \ldots, \sigma_i$. We can then construct an (s,t)-cut of the same value as the flow $f(\cdot)$. The max-flow-min-cut theorem (Ahuja et al. 1993) now implies that $\mathcal{G}$ does not contain a larger flow than $f(\cdot)$. This concludes the proof. $\square$

We remark that both Lemma 1 and Proposition 1 rely on the assumption that the inclusion condition (**IC**) is satisfied. One can readily construct instances of the uncertainty set $\mathcal{Q}$ that violate (**IC**) and for which the results of Lemma 1 and Proposition 1 no longer hold.

The maximum flow problem defined on the graph $\mathcal{G}$ can readily be solved with standard algorithms. For example, variants of the push-relabel maximum flow algorithm can find a maximal flow in time $\mathcal{O}((n+L)^3)$, and implementations of Dinic's blocking flow algorithm determine an optimal solution in time $\mathcal{O}((n+L)^2 \log(n+L))$, see Ahuja et al. (1993). We now show that due to the specific structure of the graph $\mathcal{G}$, we can find a maximal flow much faster.

Proposition 2. *Assume that condition (**IC**) is satisfied. Then, a maximal flow in $\mathcal{G}=(\mathcal{V},\mathcal{A},\gamma)$ is given by $f^\star:\mathcal{A}\mapsto\mathbb{R}_+$, where $f^\star(j,k)=\sum_{i\in S} f_i^\star(j,k)$ for all $(j,k)\in\mathcal{A}$, and each $f_i^\star:\mathcal{A}\mapsto\mathbb{R}_+$ is defined through*

$$f_i^\star(j,k)=\min\left\{\overline{\boldsymbol{q}}_i-\underline{\boldsymbol{q}}_i,\ \min\left\{\boldsymbol{b}_l-\sum_{i'\in B_l}\underline{\boldsymbol{q}}_{i'}-\sum_{\substack{i'\in B_l,\\ i'<i}} f_{i'}^\star(s,s_{i'})\ :\ l\in\mathcal{L}\text{ such that } i\in B_l\right\}\right\}\qquad\forall (j,k)\in P_i,\tag{4}$$

where P_i is defined in Lemma 1, as well as $f_i^\star(j,k)=0$ for all $(j,k)\in\mathcal{A}\setminus P_i$. Note that the recursion is well-defined as each flow $f_i^\star(\cdot)$ is defined in terms of the flows $f_{i'}^\star(\cdot)$ for $i'=1,\ldots,i-1$.

Proof. Assume that $S=\{i_1,i_2,\ldots,i_\sigma\}$ with $i_1,i_2,\ldots,i_\sigma\in V_C$ and $i_1<i_2<\ldots<i_\sigma$. Since $\underline{\boldsymbol{q}}_{i_1}\le\overline{\boldsymbol{q}}_{i_1}$ and $\sum_{i'\in B_l}\underline{\boldsymbol{q}}_{i'}\le\boldsymbol{b}_l$ for all budget sets $l\in\mathcal{L}$ that contain customer i_1, the flow $f_{i_1}^\star(\cdot)$ is nonnegative. By construction, $f_{i_1}^\star(\cdot)$ also satisfies the capacities of all arcs. Finally, since all arcs in P_{i_1} carry the same flow, $f_{i_1}^\star(\cdot)$ satisfies flow conservation. We thus conclude that $f_{i_1}^\star(\cdot)$ represents a feasible flow in $\mathcal{G}$. By the same argument, one can show via induction over $s=2,\ldots,\sigma$ that all flows resulting from the partial sums $\sum_{s'=1}^{s} f_{i_{s'}}^\star(\cdot)$ are nonnegative, satisfy the capacities of all arcs and flow conservation. We thus conclude that $f^\star(\cdot)$ is a feasible flow in $\mathcal{G}$.

We now show that $f^\star(\cdot)$ is indeed a maximal flow in $\mathcal{G}$. To this end, note that Lemma 1 implies that the set of all (s,t)-paths is given by $\{P_i\}_{i\in S}$. For each flow resulting from a partial sum $\sum_{s'=1}^{s} f_{i_{s'}}^\star(\cdot)$ and each path $P_{i_{s'}}$, $s'=1,\ldots,s$, there is at least one arc $(j,k)\in P_{i_{s'}}$ for which $\sum_{s'=1}^{s} f_{i_{s'}}^\star(j,k)=\gamma(j,k)$. Indeed, if the minimum in (4) is attained by the first term $\overline{\boldsymbol{q}}_{i_{s'}}-\underline{\boldsymbol{q}}_{i_{s'}}$, then we have $\sum_{s'=1}^{s} f_{i_{s'}}^\star(s,s_{i_{s'}})=\gamma(s,s_{i_{s'}})$. Likewise, if the minimum in (4) is attained by the term $\boldsymbol{b}_l-\sum_{i'}\underline{\boldsymbol{q}}_{i'}-\sum_{i'} f_{i'}^\star(s,s_{i'})$ for some budget set $l\in\mathcal{L}$ that contains customer i, then we have $\sum_{s'=1}^{s} f_{i_{s'}}^\star(\beta_l^0,\beta_l^1)=\gamma(\beta_l^0,\beta_l^1)$. We thus conclude that for each path P_i in $f^\star(\cdot)$, $i\in S$, there is an arc $(j,k)\in P_i$ where $f^\star(j,k)=\gamma(j,k)$. Since $\{P_i\}_{i\in S}$ constitutes the set of all (s,t)-paths, we can then construct an (s,t)-cut with the same value as $f^\star(\cdot)$. The max-flow-min-cut theorem (Ahuja et al. 1993) now implies that $\mathcal{G}$ does not contain a larger flow than $f^\star(\cdot)$. $\square$

While Proposition 2 solves the maximization of $\sum_{i\in S}\boldsymbol{q}_i$ over (2) through a maximum flow problem, we can now abstract from flow graphs and directly calculate the maximizer of $\sum_{i\in S}\boldsymbol{q}_i$ over (2).

Corollary 1. *The maximum of $\sum_{i\in S}\boldsymbol{q}_i$ over (2) is given by $\boldsymbol{q}^\star$ defined through*

$$\boldsymbol{q}_i^\star=\min\left\{\overline{\boldsymbol{q}}_i,\ \min\left\{\boldsymbol{b}_l-\sum_{\substack{i'\in B_l,\\ i'<i}}\boldsymbol{q}_{i'}^\star\ :\ l\in\mathcal{L}\text{ such that } i\in B_l\right\}\right\}\qquad\text{for } i\in S,$$

as well as $\boldsymbol{q}_i^\star=\underline{\boldsymbol{q}}_i$ for $i\notin S$. Using appropriate data structures, the solution $\boldsymbol{q}^\star$ can be computed in time $\mathcal{O}(nL)$.

Note that we cannot expect to maximize $\sum_{i\in S}\boldsymbol{q}_i$ over (2) in less than $\mathcal{O}(nL)$ operations without making assumptions regarding the sparsity of the uncertainty set $\mathcal{Q}$. This is the case because each budget set B_l can contain $\mathcal{O}(n)$ customers in general, and there are L such budget sets.

Disjoint Budget Uncertainty Sets. We call a budget uncertainty set *disjoint* if $B_l \cap B_{l'} = \emptyset$ for all $l, l' \in \mathcal{L}$. By construction, disjoint budget uncertainty sets satisfy the condition (**IC**). It turns out, however, that we can maximize $\sum_{i\in S}\boldsymbol{q}_i$ more efficiently if the budget uncertainty set is disjoint.

PROPOSITION 3 **(Gounaris et al. (2013))**. *Assume that the sets $\{B_l\}_{l=1}^L$ in (2) are disjoint, that is, $B_l \cap B_{l'} = \emptyset$ for $l \neq l'$. Then, for any customer subset $S \subseteq V_C$, the maximum of $\sum_{i\in S}\boldsymbol{q}_i$ over $\mathcal{Q}$ from (2) is given by*

$$\sum_{i\in S}\underline{\boldsymbol{q}}_i + \sum_{l=1}^{L}\min\left\{\boldsymbol{b}_l - \sum_{i\in B_l}\underline{\boldsymbol{q}}_i,\ \sum_{i\in S\cap B_l}\left(\overline{\boldsymbol{q}}_i - \underline{\boldsymbol{q}}_i\right)\right\} + \sum_{i\in S\setminus\bigcup_{l=1}^{L}B_l}\left(\overline{\boldsymbol{q}}_i - \underline{\boldsymbol{q}}_i\right).$$

Using appropriate data structures, the maximizer of $\sum_{i\in S}\boldsymbol{q}_i$ can be computed in time $\mathcal{O}(|S|)$.

Proposition 3 can be proved using linear programming duality arguments. While this result allows us to maximize $\sum_{i\in S}\boldsymbol{q}_i$ very efficiently for disjoint budget uncertainty sets, we can further speed up computations if we need to repeatedly solve the maximization problem for customer sets that are sufficiently similar. Indeed, for two customer subsets $S_1, S_2 \subseteq V_C$, the maximum of $\sum_{i\in S_2}\boldsymbol{q}_i$ over (2) can be calculated from the maximum of $\sum_{i\in S_1}\boldsymbol{q}_i$ over (2) in time $\mathcal{O}(|S_1 \setminus S_2| + |S_2 \setminus S_1|)$. In particular, if S_2 results from S_1 through the addition or removal of a single customer, then the maximum of $\sum_{i\in S_2}\boldsymbol{q}_i$ can be calculated from the maximum of $\sum_{i\in S_1}\boldsymbol{q}_i$ in constant time $\mathcal{O}(1)$. This is crucial for metaheuristic frameworks, which heavily rely on local search procedures that iteratively perturb a given subset of customers.

Let us formalize this idea. For a given customer set S, we store in $z = \max_{\boldsymbol{q}\in\mathcal{Q}}\sum_{i\in S}\boldsymbol{q}_i$ the maximum cumulative customer demands over S. Moreover, for each budget $l \in L$ the variable $\boldsymbol{\rho}_l = \sum_{i\in S\cap B_l}\left(\overline{\boldsymbol{q}}_i - \underline{\boldsymbol{q}}_i\right)$ stores the sum of differences between upper and lower demand bounds of those customers in S that participate in the budget B_l. The variable $\pi = \sum_{i\in S\bigcap\bigcup_{l=1}^{L}B_l}\underline{\boldsymbol{q}}_i + \sum_{i\in S\setminus\bigcup_{l=1}^{L}B_l}\overline{\boldsymbol{q}}_i$ stores the sum of lower bounds of all customers in S that participate in some budget $l \in L$, minus the sum of upper bounds of those customers in S that do not participate in any budget. We typically start with $S = \emptyset$, in which case $(z, \boldsymbol{\rho}, \pi) = \mathbf{0}$.

If we want to add a customer $i \notin S$ to S that participates in the budget $l_i \in \mathcal{L}$, then we update the quantities π, $\boldsymbol{\rho}_{l_i}$ and z via $\pi^{\text{new}} \leftarrow \pi^{\text{old}} + \underline{\boldsymbol{q}}_i$, $\boldsymbol{\rho}_{l_i}^{\text{new}} \leftarrow \boldsymbol{\rho}_{l_i}^{\text{old}} + (\overline{\boldsymbol{q}}_i - \underline{\boldsymbol{q}}_i)$ and $z^{\text{new}} \leftarrow z^{\text{old}} + (\pi^{\text{new}} - \pi^{\text{old}}) + (\boldsymbol{\rho}_{l_i}^{\text{new}} - \boldsymbol{\rho}_{l_i}^{\text{old}})$. If customer i does not participate in any budget, then we only update π and z via $\pi^{\text{new}} \leftarrow \pi^{\text{old}} + \overline{\boldsymbol{q}}_i$ and $z^{\text{new}} \leftarrow z^{\text{old}} + (\pi^{\text{new}} - \pi^{\text{old}})$. Likewise, if we want to remove a customer $i \in S$ that participates in the budget $l_i \in \mathcal{L}$, then we update the quantities π, $\boldsymbol{\rho}_{l_i}$ and z via $\pi^{\text{new}} \leftarrow \pi^{\text{old}} - \underline{\boldsymbol{q}}_i$, $\boldsymbol{\rho}_{l_i}^{\text{new}} \leftarrow \boldsymbol{\rho}_{l_i}^{\text{old}} - (\overline{\boldsymbol{q}}_i + \underline{\boldsymbol{q}}_i)$ and $z^{\text{new}} \leftarrow z^{\text{old}} + (\pi^{\text{new}} - \pi^{\text{old}}) + (\boldsymbol{\rho}_{l_i}^{\text{new}} - \boldsymbol{\rho}_{l_i}^{\text{old}})$. If customer i does not participate in any budget, then we only update π and z via $\pi^{\text{new}} \leftarrow \pi^{\text{old}} - \overline{q}_c$ and $z^{\text{new}} \leftarrow z^{\text{old}} + (\pi^{\text{new}} - \pi^{\text{old}})$. This way, for every customer subset S the value of z equals the maximum of $\sum_{i \in S} \boldsymbol{q}_i$ over $\boldsymbol{q} \in \mathcal{Q}$.

Disjoint budget uncertainty sets allow us to impose upper bounds on the customer demands encountered in different geographical regions. As we discuss in Section 5, this can help to avoid overly conservative route sets that hedge against highly unlikely concentrations of customer demands in specific regions. Inclusion-constrained budget uncertainty sets represent a natural generalization of this concept. They allow us to impose a hierarchy of upper bounds on the customer demands. One could, for example, simultaneously impose upper bounds on the customer demands encountered in every municipality (finest granularity), county and state (coarsest granularity). Under mild assumptions, the central limit theorem implies that higher and lower demand realizations tend to cancel each other out, which allows us to impose upper demand bounds at coarser levels of granularity that are tighter than the sums of upper demand bounds at finer levels.

3.2. Factor Model Uncertainty Sets

We now assume that the uncertainty set takes the form

$$\mathcal{Q} = \left\{ \boldsymbol{q} \in \mathbb{R}^n_+ : \boldsymbol{q} = \boldsymbol{q}^0 + \boldsymbol{\Gamma}\boldsymbol{\xi} \text{ for some } \boldsymbol{\xi} \in \Xi \right\}, \tag{5a}$$

where

$$\Xi = \left\{ \boldsymbol{\xi} \in \mathbb{R}^F : \boldsymbol{\xi} \in [-\mathbf{e}, +\mathbf{e}], \ \mathbf{e}^\top \boldsymbol{\xi} \in [-\beta F, +\beta F] \right\}. \tag{5b}$$

In this definition, $\boldsymbol{q}^0 \in \mathbb{R}^n_+$, $\boldsymbol{\Gamma} \in \mathbb{R}^{n \times F}$, $F \in \mathbb{N}$ and $\beta \in [0, 1]$ constitute parameters which need to be selected by the modeler. The vector $\mathbf{e} \in \mathbb{R}^F$ denotes the vector of all ones.

The uncertainty set (5) stipulates that the customer demands $\boldsymbol{q}$ are distributed around a nominal demand vector $\boldsymbol{q}^0$, subject to an additive disturbance $\boldsymbol{\Gamma}\boldsymbol{\xi}$. We can interpret $\boldsymbol{\xi}$ as a vector of F independent factors that attain values in the unit hypercube. The linear operator $\boldsymbol{\Gamma}\boldsymbol{\xi}$ allows us to model correlations among the customer demands through linear combinations of these factors. The constraint $\mathbf{e}^\top \boldsymbol{\xi} \in [-\beta F, +\beta F]$ reflects the belief that not all factors attain extreme values at the same time. If $\beta = 0$, then the constraint requires that as many factors ξ_f will be above $\mathbf{0} \in \mathbb{R}^F$ as there will be below $\mathbf{0}$. This special case is known as the "zero-net-alpha adjustment" in robust portfolio optimization, see Ceria and Stubbs (2006). If $\beta = 1$, then the factor uncertainty set Ξ reduces to an F-dimensional hypercube. If records of historical customer demands are available, then β can be chosen using statistical arguments (*cf.* Section 3.1).

The following result shows that the satisfaction of constraint (1) can be verified efficiently if the uncertainty set $\mathcal{Q}$ is of the form (5).

PROPOSITION 4 **(Gounaris et al. (2013))**. *For a customer set $S \subseteq V_C$, assume that $f_1, \ldots, f_F$ represents an ordering of the factors $\boldsymbol{\xi}_f$ in* (5b) *according to non-increasing marginal demands*

$$\sum_{i \in S} \boldsymbol{\Gamma}_{if_1} \geq \sum_{i \in S} \boldsymbol{\Gamma}_{if_2} \geq \ldots \geq \sum_{i \in S} \boldsymbol{\Gamma}_{if_F}.$$

Then, the maximum of $\sum_{i \in S} \boldsymbol{q}_i$ over the polytope defined in (5) *is given by*

$$\sum_{i \in S} \boldsymbol{q}_i^0 + \min \left\{ \sum_{f=1}^{F} \left| \sum_{i \in S} \boldsymbol{\Gamma}_{if} - \lambda \right| + \beta F \left| \lambda \right| : \lambda \in \left\{ 0, \sum_{i \in S} \boldsymbol{\Gamma}_{if_{\ell^+}}, \sum_{i \in S} \boldsymbol{\Gamma}_{if_{\ell^-}} \right\} \right\}, \tag{6}$$

where $\ell^+ = \lceil (1+\beta)F/2 \rceil$ and $\ell^- = \max\{\lceil (1-\beta)F/2 \rceil, 1\}$. Using appropriate data structures, the maximizer of $\sum_{i \in S} \boldsymbol{q}_i$ can be computed in time $\mathcal{O}(F\,|S| + F \log F)$.

We remark that $F \ll n$ for typical factor models, which means that we can maximize $\sum_{i \in S} \boldsymbol{q}_i$ over the polytope (5) very efficiently. The proof of Proposition 4 is based on the following intuition. Maximizing $\sum_{i \in S} \boldsymbol{q}_i$ over the uncertainty set (5) is equivalent to the optimization problem

$$\begin{array}{ll} \underset{\boldsymbol{\xi} \in \mathbb{R}^F}{\text{maximize}} & \displaystyle\sum_{i \in S} \left[\boldsymbol{q}_i^0 + \sum_{f=1}^{F} \boldsymbol{\Gamma}_{if} \boldsymbol{\xi}_f \right] \\ \text{subject to} & -\mathbf{e} \leq \boldsymbol{\xi} \leq +\mathbf{e} \\ & -\beta F \leq \mathbf{e}^\top \boldsymbol{\xi} \leq +\beta F. \end{array}$$

If we dualize this problem, then we obtain a linear minimization problem with $2F+2$ nonnegative variables and F constraints. The specific structure of the problem allows us to remove all but one variable. The resulting problem minimizes a piecewise linear and convex function over a one-dimensional domain. We thus know that the optimum must be attained at a breakpoint of this function. A closer examination reveals that only three breakpoints qualify as optimal solutions, and these correspond to the cases $\lambda \in \left\{ 0, \sum_{i \in S} \boldsymbol{\Gamma}_{if_{\ell^+}}, \sum_{i \in S} \boldsymbol{\Gamma}_{if_{\ell^-}} \right\}$ in Proposition 4. For a formal proof of the proposition, the reader is referred to Gounaris et al. (2013).

We now describe an incremental procedure that allows us to calculate the maximum of $\sum_{i \in S_2} \boldsymbol{q}_i$ very efficiently from the maximum of $\sum_{i \in S_1} \boldsymbol{q}_i$ if S_2 results from S_1 through the addition or removal of a single customer. To this end, we define the constant vector $\boldsymbol{\xi}^{\text{wc}} \in \mathbb{R}^F$ as follows. Let $\tau = \lfloor (F - \lfloor \beta F \rfloor)/2 \rfloor$ and $\sigma = \lfloor \beta F \rfloor + \tau$. If $\sigma + \tau = F$, then $\boldsymbol{\xi}^{\text{wc}} = (1, \ldots, 1, -1, \ldots, -1)^\top$, where the first σ and the last τ components are 1 and -1, respectively. If $\sigma + \tau \neq F$, then we set $\boldsymbol{\xi}^{\text{wc}} = (1, \ldots, 1, \beta F - \lfloor \beta F \rfloor, -1, \ldots, -1)^\top$, where again the first σ and the last τ components are 1 and -1, respectively. By construction, the vector $\boldsymbol{\xi}^{\text{wc}}$ satisfies $-\mathbf{e} \leq \boldsymbol{\xi}^{\text{wc}} \leq +\mathbf{e}$ and $-\beta F \leq \mathbf{e}^\top \boldsymbol{\xi}^{\text{wc}} \leq +\beta F$. During the incremental search, the variable $z = \max_{\boldsymbol{q} \in \mathcal{Q}} \sum_{i \in S} \boldsymbol{q}_i$ stores the maximum cumulative customer demands over S, $\boldsymbol{\gamma}_f = \sum_{i \in S} \boldsymbol{\xi}_f^{\text{wc}} \boldsymbol{\Gamma}_{if}$ stores the total disturbances for each factor $f \in F$ in S, and the variable $\kappa = \sum_{i \in S} \boldsymbol{q}_i^0$ stores the cumulative nominal demands in S. We start with an empty customer subset $S = \emptyset$, and therefore $(z, \boldsymbol{\gamma}, \kappa) = \mathbf{0}$ initially.

If we add a customer $i \notin S$ to S, then we conduct the updates $\boldsymbol{\gamma}_f^{\text{new}} \leftarrow \boldsymbol{\gamma}_f^{\text{old}} + \boldsymbol{\xi}_f^{\text{wc}} \boldsymbol{\Gamma}_{if}$, $f \in \{1, \ldots, F\}$, and $\kappa^{\text{new}} \leftarrow \kappa^{\text{old}} + \boldsymbol{q}_i^0$. We then sort the components of $\boldsymbol{\gamma}$ in decreasing order, and we set $z^{\text{new}} \leftarrow \kappa^{\text{new}} + \sum_{f=1}^{F} \boldsymbol{\gamma}_f^{\text{old}}$. Similarly, if we remove a customer $i \in S$ from S, then we conduct the updates $\boldsymbol{\gamma}_f^{\text{new}} \leftarrow \boldsymbol{\gamma}_f^{\text{old}} - \boldsymbol{\xi}_f^{\text{wc}} \boldsymbol{\Gamma}_{if}$, $f \in \{1, \ldots, F\}$, and $\kappa^{\text{new}} \leftarrow \kappa^{\text{old}} - \boldsymbol{q}_i^0$. We again sort the components of $\boldsymbol{\gamma}$ in decreasing order, and we set $z^{\text{new}} \leftarrow \kappa^{\text{new}} + \sum_{f=1}^{F} \boldsymbol{\gamma}_f^{\text{old}}$. Note that the sorting can typically be done efficiently as the previous order of the components of $\boldsymbol{\gamma}$ is likely to be "almost sorted."

4. An AMP Framework for the Robust CVRP

In this section, we propose an AMP metaheuristic that incorporates a number of novel elements regarding *(i)* the systematic identification, selection and combination of promising solution components, *(ii)* the manipulation and updating of the set of elite solutions, *(iii)* the construction of combinations of multiple elite solutions, and *(iv)* the improvement of the quality of provisional solutions generated during the search process via local search. The adaptive memory refers to a reference set of feasible solutions that is populated and updated by elite solutions, following a deterministic set of rules. The key idea is to keep track of the "elite components" of the solutions visited during the search, and to use them as building blocks for the construction of new provisional solutions. To that end, search diversification is achieved by combining elite components of multiple reference solutions in ways that have not been encountered in the search history, while the search gradually intensifies as the reference set evolves with strictly improving solutions, and the solution's elite components tend to more often belong to solutions from a limited number of regions of the solution space.

From the implementation viewpoint, the proposed solution framework consists of two phases, namely the initialization and the exploitation phase (see Algorithm 1). The *initialization phase* (Lines 3 through 14) generates a reference set $\mathcal{P}$ of high quality solutions. During this phase it is important to ensure that the initial reference solutions are adequately diversified so as to provide a good initial sampling of promising areas of the solution space. This is achieved via the greedy randomized savings heuristic (Lines 4 and 5) presented in Section 4.1, while the solutions generated are further improved by the Tabu Search algorithm (Line 6) described in Section 4.3. Once the initialization phase has been completed, the *exploitation phase* (Lines 15 through 26) manipulates $\mathcal{P}$ through an exploration of search trajectories initiated from new provisional solutions. In particular, at each iteration the elite components of the reference solutions are systematically identified and, based on deterministic criteria, a subset of them is selected as an intermediate solution $\boldsymbol{R}^0$ (Line 16) from which the generation of the final provisional solution $\boldsymbol{R}^\dagger$ is initiated (Line 17). This procedure is described in Section 4.2. The provisional solution $\boldsymbol{R}^\dagger$ is then used to restart the Tabu Search algorithm (Line 18) described in Section 4.3. To that end, the best encountered feasible solution

Algorithm 1 Adaptive Memory Programming

Input: μ, ν, ζ, η, and θ (user-defined parameters)

$\mathcal{P} \leftarrow \emptyset$, $\boldsymbol{R}^B \leftarrow \emptyset$, $c(\boldsymbol{R}^B) = +\infty$
Start Timer t
`// Initialization Phase`
while $|\mathcal{P}| < \mu$ **do**
    `// Construction of initial solution`
    $\boldsymbol{R}^0 \leftarrow ((0,1,0),(0,2,0),\cdots,(0,m,0))$
    $\boldsymbol{R}^0 \leftarrow$ Greedy Randomized Savings Heuristic($\boldsymbol{R}^0, \eta$)
    `// Improvement via local search`
    $\boldsymbol{R}' \leftarrow$ Tabu Search($\boldsymbol{R}^0, \nu, \zeta$)
    `// If not feasible, do not consider`
    if Not Feasible($\boldsymbol{R}'$) **then**
        continue
    end if
    `// Update incumbent`
    if $c(\boldsymbol{R}') < c(\boldsymbol{R}^B)$ **then**
        $\boldsymbol{R}^B \leftarrow \boldsymbol{R}'$
    end if
    `// Update reference set`
    $\mathcal{P} \leftarrow \mathcal{P} \cup \boldsymbol{R}'$
end while
`// Exploitation Phase`
while $t < t_{lim}$ **do**
    `// Construction of provisional solution`
    $\boldsymbol{R}^0 \leftarrow$ Selection of Elite Components($\mathcal{P}, \theta$)
    $\boldsymbol{R}^\dagger \leftarrow$ Greedy Randomized Savings Heuristic($\boldsymbol{R}^0, \eta$)
    `// Improvement via local search`
    $\boldsymbol{R}' \leftarrow$ Tabu Search($\boldsymbol{R}^\dagger, \nu, \zeta$)
    `// If not feasible, do not consider`
    if Not Feasible($\boldsymbol{R}'$) **then**
        continue
    end if
    `// Update incumbent`
    if $c(\boldsymbol{R}') < c(\boldsymbol{R}^B)$ **then**
        $\boldsymbol{R}^B \leftarrow \boldsymbol{R}'$
    end if
    `// Update reference set`
    $\mathcal{P} \leftarrow$ Reference Set Update Method($\mathcal{P}, \boldsymbol{R}'$)
end while
return $\boldsymbol{R}^B$

$\boldsymbol{R}'$ updates the reference set (Line 25), if quality and diversity criteria are met. This process is regulated by the reference set update method described in Section 4.4.

Our AMP solution framework is controlled through five parameters, namely the size μ of the reference set $\mathcal{P}$, the size η of the restricted candidate list (see Section 4.1), the parameter θ used during the selection of elite components (see Section 4.2), and the parameters ν and ζ that control the local search process (see Section 4.3). The algorithm terminates after a pre-specified time limit t_{lim} is reached (Line 15), at which point the best encountered feasible solution $\boldsymbol{R}^B$ is returned (Line 27).

The above described AMP framework introduces few user-defined parameters, as compared to similar VRP metaheuristic approaches. Furthermore, and contrary to many recent metaheuristic algorithms proposed in the VRP literature, our framework does not incorporate any instance-specific features, spatiotemporal decomposition schemes or heuristic restriction procedures to accelerate the neighborhood search process. As described in Gendreau and Tarantilis (2010), such mechanisms may have a strong impact on the efficiency and scalability towards large-scale problem instances; however, they are hard to implement for practical applications. To that end, the proposed solution approach is generic and could be applied without any modification for both deterministic and robust problem settings. In the literature, there is an evident lack for efficient solution approaches with a wider applicability towards problem extensions that combine multiple features. From this viewpoint, our approach can be regarded as relatively simple to implement and adopt.

Our experience with developing a metaheuristic framework for the RCVRP reveals that the combination of robust capacity, duration and fixed fleet-size constraints makes the construction of feasible solutions very difficult using traditional insertion- and savings-based construction heuristics. For this reason, we decided to treat infeasibility indirectly and to use a hierarchical penalized cost function to allow the local search to enter the infeasible region. Similar approaches for the control exploration of feasible and infeasible regions are widely used, and as reported in Vidal et al. (2012), they are found to enhance the performance of the search process. Finally, it is worth to note that the proposed mechanisms for selecting solution components of varying size generalize those of earlier AMP heuristics applied for the deterministic CVRP in Tarantilis and Kiranoudis (2002) and Tarantilis (2005), while they provide more consistent performance.

4.1. Greedy Randomized Savings Heuristic

As mentioned earlier, it is important during the initialization phase to ensure an adequate level of diversity among the solutions in the initial reference set (Tarantilis 2005). For this purpose, we use a generalized savings construction heuristic to generate initial solutions (Lines 4 and 5 in

Algorithm 1) coupled with a probabilistic mechanism, similar to that proposed in Repoussis et al. (2010).

Given an initial partial solution (*e.g.*, a solution where one vehicle is assigned to each customer or to subsets of customers), savings heuristics iteratively merge pairs of routes according to a savings metric (Clarke and Wright 1964): for two vehicle routes $\boldsymbol{R}_k = (\boldsymbol{R}_{k,0}, \ldots, \boldsymbol{R}_{k,i}, \boldsymbol{R}_{k,j}, \ldots, \boldsymbol{R}_{k,n_k+1})$ and $\boldsymbol{R}_{k'} = (\boldsymbol{R}_{k',0}, \boldsymbol{R}_{k',i'}, \ldots, \boldsymbol{R}_{k',j'}, \boldsymbol{R}_{k',n_{k'}+1})$, the savings function (assuming symmetric transportation costs) evaluates to $c(i,j) + c(0,i') + c(j', n_{k'}+1) - c(i,i') - c(j,j')$. This process is repeated until no further savings can be obtained or when a predefined number of routes is reached. Based on this scheme, various merging combinations can be obtained.

During the route merging process, priority is given to merging combinations with positive savings that are robust feasible. If a predefined number of routes must be reached, however, both negative savings and infeasible merging combinations are considered as well. Moreover, the merging of single-customer routes with routes containing at least two customers is encouraged by multiplying the savings with a random parameter from the range $[1.1, 1.6]$. We also incorporate a greedy randomized mechanism (Resende and Ribeiro 2010). In particular, the savings for each possible pair of routes are ordered in a so-called restricted savings list, which contains the route pairs resulting in the highest savings. At each iteration, a random pair is selected from the list, and the corresponding routes are merged. In our implementation, the restricted savings list is cardinality-based and fixed to a predefined size η. Parameter η determines the extent of randomization and greediness during the construction process. From our numerical experiments we conclude that a range of values $\eta \in [8, 12]$ is suitable for the RCVRP instances we consider. In our implementation, we set $\eta = 10$.

Given that we always add and remove a constant number of edges, it is straightforward to evaluate the savings in the transportation costs as well as to verify, if necessary, that the route duration constraints are satisfied (simply check whether $\sum_{l=0}^{n_k} c(\boldsymbol{R}_{k,l}, \boldsymbol{R}_{k,l+1}) + \sum_{l=1}^{n_k} s(\boldsymbol{R}_{k,l}) \leq T$ for all $k \in K$). Verifying the robust satisfaction of the capacity constraints is more involved, but can be done efficiently for the cases discussed in Section 3.

4.2. Generation of Provisional Solutions

The general aim of the exploitation phase is to generate new provisional solutions by combining components encountered in reference solutions. For this purpose, a systematic approach is applied periodically to select and isolate (based on deterministic criteria) from the reference set $\mathcal{P}$ a set of solution components (*i.e.* subroutes) that are good candidates to participate in the provisional solution (Line 16 in Algorithm 1). An *elite component* refers to a subroute (*i.e.*, an ordered subset of customers) that appears "sufficiently frequently" in $\mathcal{P}$. We define the *length* len_b of an elite component b to be the number of customers in the subroute, and we restrict len_b to the interval

$[2, n]$. We note that an elite component may be as large as a complete route and as short as an individual arc linking two customers. Singleton customers, on the other hand, do not qualify as elite components.

The number of reference solutions in which the elite component is encountered varies as well. We apply a lower cutoff, which enforces that a solution component is considered only if it appears in at least a minimum number of reference solutions f_{min}. According to our tests, a range of $f_{min} \in [0.15\mu, 0.25\mu]$ is appropriate for the RCVRP instances we consider, and we use $f_{min} = 0.20\mu$ in our implementation.

The final set of extracted components should consist of non-overlapping customer sets in order to be suitable for recombination in the new solution. To that end, we assign a score to each elite component according to a metric that is described later in this section. We then sort the list of elite components according to their score, and we start adopting components from the top of the list. Once an elite component is extracted from the list, all remaining elite components that share at least one customer with the former are removed from further consideration, as adopting those components would result in the inclusion of duplicate customers in the final set.

Finally, given the extracted elite components, an intermediate solution is generated by assigning a vehicle to each of the selected subroutes as well as to any singleton customer that does not participate in any of the selected elite components. By construction, the intermediate solution is robust feasible, apart from potentially violating the number m of available vehicles. The reason for this is that all elite components stem from robust feasible reference solutions and the customer visiting sequences remain unchanged, ensuring that both the capacity and duration constraints are satisfied. At this point, the greedy randomized savings heuristic from Section 4.1 is employed to obtain the new (finalized) provisional solution (Line 17 in Algorithm 1). It is worth mentioning that the proposed probabilistic route merging scheme will often provide different combinations of the elite components that have not been encountered in the search history. Note also that at the end of the construction process the provisional solution can be either feasible or infeasible.

Calculation of Component Scores. The component scoring metric for each elite component b is defined as

$$sc_b = \left(\sum_{\boldsymbol{R} \in \mathcal{P}} w_{\boldsymbol{R}} \mathcal{I}_{\boldsymbol{R},b} \right) / \, (1-\theta)^{\mathrm{len}_b - 2},$$

where $\mathcal{I}_{\boldsymbol{R},b}$ is an indicator function taking the value of 1 if the elite component participates in the solution $\boldsymbol{R}$ and 0 otherwise, $w_{\boldsymbol{R}}$ are factors that weight the participation in each solution, and the parameter $\theta \in (0,1)$ quantifies the adoption of longer elite components at the expense of their shorter subsets.

We adopt two different strategies for the selection of the weights $w_{\boldsymbol{R}}$. In the first strategy, the weight $w_{\boldsymbol{R}}$ attributed to each solution $\boldsymbol{R} \in \mathcal{P}$ is calculated as $w_{\boldsymbol{R}} = \frac{\text{diss}(\boldsymbol{R}, \boldsymbol{R}^*)}{\max_{\boldsymbol{R}' \in \mathcal{P}} \text{diss}(\boldsymbol{R}', \boldsymbol{R}^*)}$, where $\text{diss}(\boldsymbol{R}, \boldsymbol{R}^*)$ is the dissimilarity between solution $\boldsymbol{R}$ and the currently best solution $\boldsymbol{R}^*$, measured in terms of the total number of different arcs (also known as broken-pairs distance). This strategy enhances diversification and guides the search towards distant and possible unexplored regions with respect to the current best solution. In the second strategy, the weight accounts for the cumulative transportation costs incurred by the solution: $w_{\boldsymbol{R}} = \frac{\max_{\boldsymbol{R}' \in \mathcal{P}} c(\boldsymbol{R}') - c(\boldsymbol{R})}{\max_{\boldsymbol{R}' \in \mathcal{P}} c(\boldsymbol{R}') - \min_{\boldsymbol{R}' \in \mathcal{P}} c(\boldsymbol{R}')}$. This strategy myopically intensifies and directs the search effort to regions close to the best solutions in $\mathcal{P}$. In our implementation, we use these strategy interchangeably with equal selection probability. More elaborate strategic oscillation schemes—based on current search progress—can be adopted to better balance the needs for exploration and exploitation.

With regards to parameter θ, we select the moderate value of $\theta = 0.2$ which we have identified to work well for the instances we consider. Note that very small values of θ (*i.e.*, $\theta \to 0$) would cause the elite component adoption to predominantly involve components of length $\text{len}_b = 2$ (*i.e.*, individual arcs) which appear in the reference solution at least as many times as the longer subroutes.

On the other hand, large values of θ (*i.e.*, $\theta \to 1$) would imply that only complete routes in the solution reference set would be considered, which results in insufficient diversification. In other words, although selecting elite components with a large number of nodes is desirable from the perspective of readily recombining the elite components into new high quality provisional solutions, the right balance needs to be sought between appearance frequency and component length.

4.3. Tabu Search

In an effort to intensify the search close to promising regions of the solution space, the initial and provisional solutions are further improved by means of a Tabu Search algorithm (Lines 6 and 18 in Algorithm 1). As described in Glover (1997), Tabu Search explores the solution space by iteratively perturbing a solution $\boldsymbol{R}$ to the best admissible solution $\boldsymbol{R}'$ within a subset $\Omega_Y(\boldsymbol{R})$ of a preselected neighborhood structure Y. To that end, a short term memory (also known as a *tabu list*) keeps track of the attributes of the most recently visited solutions in the search history and prevents revisiting them for a predefined number of iterations ν (*tabu tenure*). The tabu status of an admissible neighboring solution can be overridden only if predefined aspiration criteria are met. This iterative local search procedure is repeated until a termination condition is met, at which point the best encountered solution is returned.

Although the Tabu Search algorithm accounts for a significant portion of the overall computational burden, the method is essential for the progression towards high-quality regions of the solution space. From the implementation viewpoint, three aspects determine the computational efficiency and overall performance, namely *(i)* the definition of the search space, *(ii)* the selection of the neighborhood structures and associated evaluation methods, and *(iii)* the definitions of the tabu list, aspiration criteria, and the termination condition. Below, we provide an overview of our implementation with regards to each of these three aspects.

4.3.1. Search Space. The search space is defined such that the method can handle both feasible and infeasible solutions, and the solutions are evaluated lexicographically based on feasibility. Let $\boldsymbol{R}=(\boldsymbol{R}_1,\dots,\boldsymbol{R}_m)$ be a set of routes. If all routes $\boldsymbol{R}_k$ are robust feasible, the search attempts to improve the original objective function (*i.e.*, $c(\boldsymbol{R})$, the cumulative transportation costs), and is confined to the space of robust feasible route sets. As such, once a feasible solution is reached during the local search process, we do not allow the search to re-enter the infeasible region, admitting only those neighboring solutions that are feasible. On the other hand, infeasibility may occur due to violations of the route durations and/or the capacity constraints of the vehicles. In these cases, the search space is defined with respect to the total distance traveled, compounded by the weighted sum of route duration and vehicle capacity violations.

Let $d(\boldsymbol{R})$ and $g(\boldsymbol{R})$ be the cumulative route duration and vehicle capacity violations, respectively, and let ψ^D and ψ^Q be the corresponding weight coefficients. The new augmented objective function $h(\boldsymbol{R})$ is defined as

$$h(\boldsymbol{R})=c(\boldsymbol{R})+\psi^D d(\boldsymbol{R})+\psi^Q g(\boldsymbol{R}). \tag{7}$$

The route duration violation of the k-th vehicle evaluates to $\max\{0,c(\boldsymbol{R}_k)+s(\boldsymbol{R}_k)-T\}$, where $c(\boldsymbol{R}_k)=\sum_{l=0}^{n_k} c(\boldsymbol{R}_{k,l},\boldsymbol{R}_{k,l+1})$ and $s(\boldsymbol{R}_k)=\sum_{l=1}^{n_k} s(\boldsymbol{R}_{k,l})$ are the cumulative travel times and services times along the route, respectively, and T is the maximum allowable route duration. In total, the cumulative route duration violation $d(\boldsymbol{R})$ can be expressed as

$$d(\boldsymbol{R})=\sum_{k\in K}\max\{0,c(\boldsymbol{R}_k)+s(\boldsymbol{R}_k)-T\}. \tag{8}$$

Similarly, the cumulative vehicle capacity violation $g(\boldsymbol{R})$ can be defined as

$$g(\boldsymbol{R})=\sum_{k\in K}\max\{0,\max_{\boldsymbol{q}\in\mathcal{Q}}\sum_{l=1}^{n_k}\boldsymbol{q}_{\boldsymbol{R}_{k,l}}-Q\}. \tag{9}$$

In the literature, augmented objective functions with similar penalty terms are widely adopted for the exploration of both feasible and infeasible regions (Tarantilis et al. 2012). However, in the context of the RCVRP, verifying robust feasibility is more involved as we need to evaluate the inner maximization in (9), which itself amounts to an optimization problem. As we will discuss in Section 4.3.2, our AMP procedure uses both intra-route and inter-route moves. Intra-route moves alter the order in which customers are visited, but they do not modify the set of customers that participate in the route. Thus, the capacity violation remains the same and no capacity check has to be performed. On the other hand, inter-route moves, which account for the majority of the postulated moves, require us to recalculate the capacity violation. Hence, it is important to employ an implementation that enables fast "on-the-fly" computations, such as the one that was detailed in Section 3.

The performance of the proposed local search scheme crucially depends on the initial values and the subsequent readjustment of the weight coefficients. In our implementation, we initially set the coefficients to $\psi^D = 10$ and $\psi^Q = 1$, and we increment either of the coefficients during the search via $\psi^D \leftarrow \psi^D + 10$ or $\psi^Q \leftarrow \psi^Q + 1$ if the current solution violates a route duration or a vehicle capacity, respectively. On the other hand, if the solution becomes feasible, we re-initialize the coefficients to $\psi^D = 10$ and $\psi^Q = 1$. This selection reflects our computational experience, where it appears rather difficult to restore feasibility for large scale long-haul problem instances that prescribe tight route duration restrictions.

4.3.2. Neighborhood Structures and Associated Evaluation Methods. At each iteration of the local search process, the best admissible (feasible or infeasible) neighbor $\boldsymbol{R}'$ (*i.e.*, $\min_{\boldsymbol{R}' \in \Omega_Y(\boldsymbol{R})} \{h(\boldsymbol{R}')\}$) replaces the current solution $\boldsymbol{R}$. In our implementation, we use neighborhood structures based on ordinary edge-exchange local moves, namely intra- and inter-route *2-Opt*, *1-0 Relocate* and *1-1 Exchange* (Aarts and Lenstra 2003, Gendreau and Tarantilis 2010). The selection of neighborhood structures at each iteration of the Tabu Search is random, with equal selection probability. Our algorithm does not exploit any potential spatiotemporal structure.

We remark that the aforementioned neighborhood structures involve the addition and deletion of a constant number of edges, and that the overall size of each neighborhood is quadratic. In particular, the intra-route 2-Opt involves the substitution of a total of 2 edges within a single route, or similarly, the inversion of the visiting sequence of a segment $(\boldsymbol{R}_{k,i}, \ldots, \boldsymbol{R}_{k,j})$. On the other hand, the inter-route 2-Opt neighborhood (also known as 2-Opt*) swaps the end segments of two vehicle routes $\boldsymbol{R}_k$ and $\boldsymbol{R}_{k'}$ (*i.e.*, $(\boldsymbol{R}_{k,l}, \ldots, \boldsymbol{R}_{k,n_k+1})$ and $(\boldsymbol{R}_{k',l'}, \ldots, \boldsymbol{R}_{k',n_{k'}+1})$) without reversing the order of customers. Finally, a 1-0 Relocate move involves the removal of one customer from its current position and its insertion into a different position of either the same or a separate route, while a 1-1 Exchange move swaps the positions of two customers.

We traverse the above described neighborhood structures in lexicographic order and apply an early pruning mechanism that is based on both feasibility and gain. Feasibility checks and computations of violations can be performed efficiently for the route durations. To that end, we store in memory the vehicles' arrival times at each of the customers and update these quantities whenever a local move is adopted. The verification of robust feasibility with respect to the capacity constraints is more demanding. However, due to the nature of the search procedure that explores the space in an incremental fashion, we seldom have to compute the capacity violation of a customer set $S \subseteq V_C$ *from scratch*. Instead, most of the time we are interested in computing the incremental difference in violation that results from adding a customer $i \notin S$ to a set S or removing a customer $j \in S$ from the set S, which can be done efficiently using the methods described in Section 3. For example, in the case of an inter-route 1-0 Relocate move, one may simply compute the incremental violation difference between customer subsets $\boldsymbol{R}_k$ and $\boldsymbol{R}_k \setminus \{\boldsymbol{R}_{k,i}\}$, where $k \in K$ and $i \in V_C$, as well as the difference between $\boldsymbol{R}_{k'}$ and $\boldsymbol{R}_{k'} \cup \{\boldsymbol{R}_{k,i}\}$, where $k' \in K$. The former difference is negative (*i.e.*, a move towards feasibility), while the latter difference is positive (*i.e.*, a move towards infeasibility). Similar arguments can be used to break down the inter-route 2-Opt and 1-1 Exchange moves into elementary additions and removals of customers. Moreover, the same concept can be readily applied and/or extended to other more complex neighborhood structures, such as λ-interchange and ejection chains. Note that in the case of 2-Opt* moves, which involve the relocation of potentially large subroutes, it is important to account for the length of the involved subroutes and implement the computation so as to minimize the number of required customer additions and removals (considering also the possibility of computing the violation from scratch).

4.3.3. Tabu List, Aspiration Criteria, and Termination Condition. The primary goal of a tabu list is to avoid cycling, at least in the short term. For this purpose, both the forward and reverse local move attributes that correspond to edges being added and deleted are stored in the tabu list, and the addition and deletion of these edges is restricted for a number of iterations ν. The tabu status is overridden if an improvement is observed with respect to the best encountered solution (aspiration criterion). The termination condition imposes a maximum number of iterations ζ without observing any further improvement.

Regarding the parameter settings, the literature typically adopts a value $\nu \in [20, 40]$ for intensification search, and this range seems to fit well for the RCVRP instances we consider. Furthermore, one may expect that large values of ζ (our termination condition) will increase the performance. However, there is a trade off between efficiency and effectiveness, since large values of ζ may result in excessive runtimes at an insignificant added benefit. Overall, it seems that values less than 300 provide a consistent performance across most problem instances. Based on these observations, we set $\nu = 30$ and $\zeta = 100$.

4.4. Reference Set Update Method

The size of the reference set $\mathcal{P}$ progressively increases during the initialization phase up to a maximum value of μ. After that, the method switches to the exploitation phase, where the size of $\mathcal{P}$ is kept constant by replacing older solutions with more recently encountered ones (see Algorithm 1).

In the process, it is important to ensure an appropriate balance between quality and diversity among the reference solutions. Such a balance allows for a more efficient and effective exploration of the solution space. Furthermore, it has been reported that an appropriately diverse collection of reference solutions can in many cases contain useful information about the structural aspects of the optimal solutions (Tarantilis et al. 2012).

On this basis, the reference set update method used in the exploitation phase adopts a deterministic set of rules that account for attractiveness both in terms of the total transportation costs as well as the level of dissimilarity between the reference solutions and the currently best solution. We now formalize this idea. Let $\boldsymbol{R}$ denote a route set that is a candidate for insertion into $\mathcal{P}$, $\boldsymbol{R}'$ any reference solution of $\mathcal{P}$, and $\boldsymbol{R}^B$ and $\boldsymbol{R}^W$ the best and the worst of solutions of $\mathcal{P}$, respectively. If $c(\boldsymbol{R}) < c(\boldsymbol{R}^B)$, then $\boldsymbol{R}$ replaces $\boldsymbol{R}^W$ in $\mathcal{P}$. Otherwise, if there exists some $\boldsymbol{R}'$ such that $c(\boldsymbol{R}) < c(\boldsymbol{R}')$ and $\operatorname{diss}(\boldsymbol{R}, \boldsymbol{R}^B) > \operatorname{diss}(\boldsymbol{R}', \boldsymbol{R}^B)$, then $\boldsymbol{R}$ replaces $\boldsymbol{R}'$ in $\mathcal{P}$. We remark that only feasible solutions are considered for insertion into the reference set. In our implementation, we set $\mu = 15$, which appears to be a fairly robust value.

5. Computational Results

We begin our computational study with the 180 RCVRP benchmark instances introduced in Gounaris et al. (2013). These instances originate from standard CVRP benchmark problems used in the literature (see NEO Research Group (2012) for a compilation of data files) and correspond to small- and medium-sized instances with up to 150 customers and 15 vehicles. In these instances, it is assumed that the uncertain customer demands are supported on uncertainty sets of the form of budget and factor model sets similar to the ones described in Section 3. In particular, we use the budget uncertainty set

$$\mathcal{Q}_{\mathcal{B}} = \left\{ \boldsymbol{q} \in \left[(1-\alpha)\boldsymbol{q}^0, (1+\alpha)\boldsymbol{q}^0\right] \,:\, \sum_{i\in\Omega}(\boldsymbol{q}_i - \boldsymbol{q}_i^0) \leq \beta \sum_{i\in\Omega}\left[(1+\alpha)\boldsymbol{q}_i^0 - \boldsymbol{q}_i^0\right] \;\forall\, \Omega \in \{NE, NW, SW, SE\} \right\},$$

where NE, NW, SW, SE refer to the four geographic quadrants defined by the customer coordinates from the benchmark datasets. The support $\mathcal{Q}_{\mathcal{B}}$ stipulates that the customer demands deviate by at most $\alpha \cdot 100\%$ from their nominal values $\boldsymbol{q}^0$ specified in the benchmark problems. Moreover, the cumulative demand in each quadrant does not exceed its nominal value by more than $\beta \cdot 100\%$. In addition, we also consider the following factor model uncertainty set for the customer demands:

$$\mathcal{Q}_{\mathcal{F}} = \left\{ \boldsymbol{q} \in \left[(1-\alpha)\boldsymbol{q}^0, (1+\alpha)\boldsymbol{q}^0\right] \,:\, \exists\, \boldsymbol{\xi} \in \Xi \text{ such that } \boldsymbol{q}_i = \left(1 + \alpha \sum_{f=1}^{4} \boldsymbol{\gamma}_{if}\boldsymbol{\xi}_f\right)\boldsymbol{q}_i^0 \;\forall\, i \in V_C \right\},$$

where

$$\Xi = \left\{ \boldsymbol{\xi} \in \mathbb{R}^4 \,:\, \boldsymbol{\xi} \in [-\mathbf{e}, +\mathbf{e}]\,,\; \mathbf{e}^\top \boldsymbol{\xi} \in [-4\beta, +4\beta] \right\}.$$

The interpretation of parameter α is the same as in the case of $\mathcal{Q}_{\mathcal{B}}$. In $\mathcal{Q}_{\mathcal{F}}$, we model the demand of customer i as a convex combination of $f = 1, \ldots, 4$ factors that can be interpreted as quadrant demands. The weights $\boldsymbol{\gamma}_{if}$ of this convex combination reflect the relative proximity of customer i to each quadrant. More precisely, we set $\boldsymbol{\gamma}_{if} = \boldsymbol{\rho}_{if} / \sum_{f'=1}^{4} \boldsymbol{\rho}_{if'}$, where $\boldsymbol{\rho}_{if}$ measures the inverse distance between customer i and the centroid of quadrant f. The cumulative demand over all quadrants is assumed to deviate from its nominal value by at most $\beta \cdot 100\%$. Note that for both $\mathcal{Q}_{\mathcal{B}}$ and $\mathcal{Q}_{\mathcal{F}}$, we retrieve a rectangular support if we set $\beta = 1$ and the deterministic CVRP if we choose $\alpha = 0$. From now on, we set $\alpha = 0.1$ and $\beta = 0.5$. Later in this section, we perform a sensitivity analysis and demonstrate that the metaheuristic retains its efficiency across the full range of values for these parameters.

We first report the best solutions identified by our AMP framework after a time limit of 1h CPU.[2] Table 1 presents these results for both $\mathcal{Q}_{\mathcal{B}}$ and $\mathcal{Q}_{\mathcal{F}}$. To illustrate the quality of the obtained solutions, we also compare them with the upper bounds reported in Table EC.3 of Gounaris et al. (2013), which have been obtained with a branch-and-cut method. We observe that out of the total of 180 instances, the metaheuristic was able to obtain improved best-known solutions for 49 instances, while it was able to match or come very close to the previously best-known solutions in all remaining cases.

We now turn our attention to the performance of the metaheuristic in terms of CPU time requirements. Although the results reported above correspond to the best solutions obtained after a time limit of 1h CPU, this amount of time may be overly generous for all but the most difficult instances. In fact, in many cases the metaheuristic has found a very good solution in an early stage of the search process and spent the remaining time attempting to improve this solution. In order to better appreciate this fact, we compute the relative differences between the best solutions obtained at given time marks and the overall best known solution (*i.e.*, the best solution after 1h CPU). Table 2 reports the aggregated results. We observe that within 5s CPU the heuristic has found a solution that is within 1% of the overall best, while there is practically no improvement beyond 5m CPU. At this point, it is probably more reasonable to restart the algorithm with a different random seed rather than to spend additional time trying to improve the solution at hand. We further note that no significant performance difference is observed between the budget and factor model uncertainty sets.

In fact, the performance of the new AMP framework is not only insensitive to the type of uncertainty set, but it also does not seem to be affected by the actual range and magnitude of the anticipated uncertainty. To show this, we select various values for the parameters α and β and

[2] Throughout the computational study, we use a PC with a single-core Intel 2.66GHz processor and 3GB RAM.

Table 1 Best solutions obtained after 1h CPU for 90 benchmark problems and the two supports $\mathcal{Q}_{\mathcal{B}}$ and $\mathcal{Q}_{\mathcal{F}}$, compared to the previously best-known upper bounds (BKUB) reported in Gounaris et al. (2013). Boldface numbers represent improvements over the BKUB, and asterisks indicate that the BKUB is already optimal.

Problem	Budgets ($\mathcal{Q}_{\mathcal{B}}$)		Factor model ($\mathcal{Q}_{\mathcal{F}}$)		Problem	Budgets ($\mathcal{Q}_{\mathcal{B}}$)		Factor model ($\mathcal{Q}_{\mathcal{F}}$)		Problem	Budgets ($\mathcal{Q}_{\mathcal{B}}$)		Factor model ($\mathcal{Q}_{\mathcal{F}}$)	
	BHS†	BKUB††	BHS	BKUB		BHS	BKUB	BHS	BKUB		BHS	BKUB	BHS	BKUB
A-n32-k5	748	748*	748	748*	B-n39-k5	532	532*	529	529*	F-n45-k4	722	718*	722	714*
A-n33-k5	642	642*	631	631*	B-n41-k6	796	796*	791	791*	F-n72-k4	232	232*	232	232*
A-n33-k6	717	717*	710	710*	B-n43-k6	681	681*	680	680*	F-n135-k7	**1,138**	1,149	1,086	1,086
A-n34-k5	715	715*	702	702*	B-n44-k7	835	835*	835	835*	M-n101-k10	809	809*	804	804*
A-n36-k5	755	755*	766	766*	B-n45-k5	701	701	691	680*	M-n121-k7	**995**	1,054	**987**	1,014
A-n37-k5	650	650*	648	648*	B-n45-k6	660	660*	657	657*	M-n151-k12	**1005**	1,219	**991**	1,154
A-n37-k6	892	892*	892	892	B-n50-k7	679	679*	699	699*	P-n16-k8	439	439*	439	439*
A-n38-k5	706	704*	693	693*	B-n50-k8	**1,224**	1,228	**1,217**	1,221	P-n19-k2	195	195*	195	195*
A-n39-k5	777	777*	772	772*	B-n51-k7	962	962	928	928*	P-n20-k2	208	208*	208	208*
A-n39-k6	787	787*	786	786*	B-n52-k7	675	675*	670	670*	P-n21-k2	208	208*	208	208*
A-n44-k6	**911**	917	892	892*	B-n56-k7	623	623*	623	623*	P-n22-k2	213	213*	213	213*
A-n45-k6	896	896*	891	891*	B-n57-k7	1,055	1,055	1,052	1,052	P-n22-k8	537	537*	557	557*
A-n46-k7	888	888*	883	883*	B-n57-k9	1,557	1,540*	1,541	1,539*	P-n23-k8	504	504	503	503
A-n48-k7	1,033	1,033	1,034	1,034	B-n63-k10	**1,407**	1,411	1,405	1,405	P-n40-k5	447	447*	447	447*
A-n53-k7	974	974	974	967*	B-n64-k9	803	803*	803	803*	P-n45-k5	501	501*	494	494*
A-n54-k7	1,126	1,119	**1,104**	1,106	B-n66-k9	**1,251**	1,339	1,226	1,210	P-n50-k7	541	539	538	537
A-n55-k9	1,037	1,030	1,007	1,007*	B-n67-k10	1,008	1,007	**1,005**	1,009	P-n50-k8	592	592	590	588
A-n60-k9	**1,280**	1,298	**1,264**	1,269	B-n68-k9	**1,213**	1,215	1,197	1,191	P-n50-k10	656	656	**656**	658
A-n61-k9	983	983	977	977	B-n78-k10	1,134	1,131	1,130	1,130	P-n51-k10	**707**	712	698	698
A-n62-k8	**1,236**	1,278	**1,201**	1,320	E-n22-k4	373	373*	373	373*	P-n55-k7	549	549	548	544
A-n63-k9	**1,505**	1,752	**1,500**	1,728	E-n23-k3	563	563*	544	544*	P-n55-k8	574	572	568	568
A-n63-k10	**1,244**	1,262	**1,229**	1,229	E-n30-k3	475	475*	492	492*	P-n55-k10	**672**	674	663	663
A-n64-k9	**1,341**	1,535	**1,319**	1,500	E-n33-k4	814	814*	814	814*	P-n55-k15	**891**	900	877	877
A-n65-k9	1,106	1,106	1,094	1,094*	E-n51-k5	516	516*	516	516*	P-n60-k10	715	712	**707**	712
A-n69-k9	**1,116**	1,119	**1,096**	1,107	E-n76-k7	670	661	**666**	672	P-n60-k15	**936**	957	**916**	935
A-n80-k10	**1,668**	1,986	**1,664**	1,945	E-n76-k8	**710**	758	**703**	706	P-n65-k10	**766**	769	**762**	768
B-n31-k5	651	651*	651	651*	E-n76-k10	**803**	887	**788**	842	P-n70-k10	**797**	818	**783**	790
B-n34-k5	768	768	755	748*	E-n76-k14	**953**	1,068	**958**	1,089	P-n76-k4	590	590*	590	590*
B-n35-k5	883	883*	883	883*	E-n101-k8	**794**	846	**783**	786	P-n76-k5	619	616	618	615
B-n38-k6	729	729*	729	729*	E-n101-k14	**1,049**	1,171	**1,018**	1,140	P-n101-k4	678	673*	676	673*

†BHS: Best heuristic solution after 1h CPU. ††BKUB: Best-known upper bound from Gounaris et al. (2013).

Table 2 Average progress of heuristic solutions across the 90 benchmark problems from Table 1. This table reports the average relative differences to the overall best solutions obtained by the heuristic after 1h CPU.

Support	Time (CPU)						
	1s	5s	10s	30s	1m	5m	10m
$\mathcal{Q}_{\mathcal{B}}$	1.37%	0.59%	0.38%	0.21%	0.12%	0.05%	0.04%
$\mathcal{Q}_{\mathcal{F}}$	1.49%	0.65%	0.42%	0.29%	0.23%	0.05%	0.02%

Table 3 Impact of α and β in the average progress of heuristic solutions across all 180 benchmark instances from Table 1 (including both supports $\mathcal{Q}_{\mathcal{B}}$ and $\mathcal{Q}_{\mathcal{F}}$). This table reports the average relative differences to the overall best solutions obtained by the heuristic after 1h CPU.

(α, β)	Time (CPU)						
	1s	5s	10s	30s	1m	5m	10m
$(0.0, 0.0)^{\dagger}$	1.19%	0.46%	0.34%	0.21%	0.17%	0.07%	0.05%
$(0.1, 0.0)$	1.13%	0.65%	0.39%	0.21%	0.13%	0.05%	0.02%
$(0.1, 0.5)$	1.43%	0.62%	0.40%	0.25%	0.17%	0.05%	0.03%
$(0.1, 1.0)^{\dagger}$	1.30%	0.42%	0.39%	0.21%	0.11%	0.05%	0.03%
$(0.2, 0.0)$	1.27%	0.57%	0.41%	0.25%	0.18%	0.07%	0.05%
$(0.2, 0.5)$	1.27%	0.72%	0.31%	0.32%	0.22%	0.08%	0.04%
$(0.2, 1.0)^{\dagger}$	1.30%	0.76%	0.53%	0.31%	0.24%	0.10%	0.05%

†Although these problems reduce to deterministic CVRP instances, they are solved with our RCVRP metaheuristic for a fair comparison.

resolve all instances. We present the aggregated results in Table 3. We observe that the differences approach their terminal value of zero at about the same rate for all parameter values, an indication that the heuristic performs comparably in all cases.

Determining high-quality heuristic solutions is of course very important in practice. But how important are high-quality solutions if one is merely interested in proving optimality? In an effort to answer this question, we undertake the following experiment. We perform a single run of the AMP metaheuristic with a short time limit of 5m CPU. We take the best heuristic solution obtained within this time limit, feed it as an initial incumbent into the branch-and-cut framework of Gounaris et al. (2013) and record the performance difference relative to the original branch-and-cut scheme. In particular, we record the residual lower and upper bounds after 24h CPU and compare them with the ones reported in Gounaris et al. (2013). To ensure a fair comparison, we have used the same code, computer, MILP solver (CPLEX 12.1 IBM Corp. (2009)) and problem instances that were employed in the previous study, the only difference being the introduction of the heuristic solutions as starting incumbents for the MILP solver. Table 4 presents the results of this experiment. We restrict ourselves to those 88 instances which had not been solved to global optimality by the original branch-and-cut scheme (*i.e.*, for which there was a residual LB–UB gap after 24h CPU). As the table shows, the a priori knowledge of a heuristic solution enabled the exact framework to obtain new optimality certificates for 3 instances and improve the previously best-known lower and upper bounds in a total of 31 and 54 instances, respectively.

We have shown that the metaheuristic developed in this paper is efficient for small- and medium-sized instances, which is consistent with the performance of the AMP metaheuristic from Section 4 when applied to the deterministic CVRP and other related problems (Repoussis and Tarantilis 2010). In particular, we observe that the presence of uncertainty does not significantly degrade the performance of the AMP scheme. We have further shown that the metaheuristic can help to improve the performance of exact solution frameworks. We now investigate the performance of our metaheuristic on large-scale RCVRP instances that would be challenging for exact algorithms. To that end, we apply our approach to the two standard large-scale CVRP benchmark suites used in the literature (Christofides et al. 1979, Golden et al. 1998). These instances involve up to 483 customers and 38 vehicles. In analogy to Gounaris et al. (2013), we increase the vehicle capacities by 20% in order to accommodate the uncertainty in the customer demands.[3] The results are presented in Table 5. For each of the benchmark instances and each of the two uncertainty sets $\mathcal{Q}_{\mathcal{B}}$ and $\mathcal{Q}_{\mathcal{F}}$, we conducted 10 independent runs that were initialized with different random seeds. All runs were conducted with the same parameter settings $\alpha = 0.1$ and $\beta = 0.5$. We report the "best-best," "average-best" and "worst-best" solutions found after a time limit of 1h CPU as well as the time required to obtain the best solution found. For completeness, the table also lists the number of customers, number of vehicles, route duration limit, homogeneous service time for each customer, as well as the homogeneous vehicle capacity for each instance. We observe that our framework provides feasible solutions to all of these instances within the allotted time limit. The relatively small differences between "best-best" and "worst-best" solutions indicate that the AMP framework is reasonably robust against the random seed initialization and can be expected to perform efficiently even when run only for a single time. Furthermore, the reported runtimes indicate that in most cases the metaheuristic was still making progress when the time limit was reached, and it could thus benefit from additional computational resources.

We conclude this section by quantifying the average incease in the total cost of an RCVRP solution compared to its deterministic CVRP counterpart. To this end, we focus on the 14 problem sets introduced by Christofides et al. (1979) and select various combinations of parameter settings (α, β). In each case, we run the AMP framework as before[4], record the best heuristic solution encountered and compare it with the solution of the deterministic CVRP instance (that is, the

[3] The benchmark instances were originally proposed for the deterministic CVRP, and they are designed to be capacity-tight in this setting. Hence, the original vehicle capacity specifications would lead to infeasible instances even for small choices of α and β (see also Gounaris et al. (2013) for further justification).

[4] Ten independent runs starting from different random seeds; 1h CPU time limit for each run.

Table 4 Residual LB–UB gap after 24h CPU for those instances for which optimality certificates were not available prior to this study. Previously known bounds are reported in Gounaris et al. (2013). Boldface numbers indicate improved bounds, and asterisks denote new certificates of optimality.

Problem	Budgets ($\mathcal{Q}_\mathcal{B}$)		Factor model ($\mathcal{Q}_\mathcal{F}$)	
	AMP + B&C[†]	B&C[††]	AMP + B&C	B&C
A-n37-k6		(no gap)	**892***	889–892
A-n44-k6	883–**909**	885–917		(no gap)
A-n48-k7	1,008–1,033	1,012–1,033	1,007–**1,033**	1,009–1,034
A-n53-k7	964–974	966–974		(no gap)
A-n54-k7	1,059–**1,106**	1,065–1,119	1,060–**1,097**	1,063–1,106
A-n55-k9	1,008–1,030	1,010–1,030		(no gap)
A-n60-k9	**1,222–1,280**	1,221–1,298	**1,222–1,264**	1,221–1,269
A-n61-k9	954–983	957–983	954–**974**	956–977
A-n62-k8	1,159–**1,214**	1,162–1,278	1,155–**1,201**	1,156–1,320
A-n63-k9	**1,426–1,505**	1,415–1,752	**1,418–1,498**	1,412–1,728
A-n63-k10	**1,187–1,240**	1,186–1,262	1,184–**1,222**	1,187–1,229
A-n64-k9	**1,249–1,325**	1,240–1,535	1,242–**1,314**	1,244–1,500
A-n65-k9	1,090–1,106	1,091–1,106		(no gap)
A-n69-k9	**1,059–1,109**	1,057–1,119	**1,053–1,096**	1,050–1,107
A-n80-k10	**1,566–1,662**	1,561–1,986	**1,565–1,645**	1,560–1,945
B-n34-k5	748–768	749–768		(no gap)
B-n45-k5	692–701	693–701		(no gap)
B-n50-k8	1,117–**1,224**	1,163–1,228	1,144–**1,217**	1,188–1,221
B-n51-k7	939–**961**	939–962		(no gap)
B-n57-k7	1,006–1,055	1,026–1,055	1,013–1,052	1,024–1,052
B-n63-k10	1,370–**1,407**	1,370–1,411	**1,378**–1,405	1,377–1,405
B-n66-k9	**1,203–1,251**	1,198–1,339	**1,210***	1,205–1,210
B-n67-k10	**983**–1,007	982–1,007	981–**1,001**	981–1,009
B-n68-k9	**1,170–1,213**	1,169–1,215	1,173–1,191	1,173–1,191

Problem	Budgets ($\mathcal{Q}_\mathcal{B}$)		Factor model ($\mathcal{Q}_\mathcal{F}$)	
	AMP + B&C	B&C	AMP + B&C	B&C
B-n78-k10	1,124–1,131	1,126–1,131	1,120–1,130	1,124–1,130
E-n76-k7	644–661	645–661	643–**661**	643–672
E-n76-k8	**681–709**	680–758	681–**700**	682–706
E-n76-k10	**756–796**	755–887	**752–782**	750–842
E-n76-k14	**896–952**	895–1,068	884–**952**	884–1,089
E-n101-k8	**767–789**	765–846	**767–783**	766–786
E-n101-k14	**972–1,011**	971–1,171	**966–1,009**	964–1,140
F-n135-k7	1,085–**1,129**	1,085–1,149	1,081–1,086	1,082–1,086
M-n121-k7	**963–994**	961–1,054	957–**982**	957–1,014
M-n151-k12	**936–993**	935–1,219	**931–986**	928–1,154
P-n23-k8	494–504	496–504	489–503	492–503
P-n50-k7	530–539	531–539	529–537	531–537
P-n50-k8	572–592	574–592	569–588	573–588
P-n50-k10	633–656	635–656	629–**656**	631–658
P-n51-k10	675–**707**	678–712	670–698	672–698
P-n55-k7	539–549	542–549	**544***	540–544
P-n55-k8	559–572	563–572	**560**–568	559–568
P-n55-k10	636–**670**	638–674	633–**657**	636–663
P-n55-k15	**843–889**	841–900	834–877	834–877
P-n60-k10	680–712	683–712	677–**705**	677–712
P-n60-k15	874–**931**	875–957	863–**916**	864–935
P-n65-k10	729–**765**	729–769	**725–761**	724–768
P-n70-k10	754–**785**	755–818	**750–783**	749–790
P-n76-k5	607–616	609–616	605–615	607–615

[†] After 5m CPU of AMP plus 24h CPU of branch-and-cut. [††] After 24h CPU of branch-and-cut, as reported in Gounaris et al. (2013).

Table 5 **Heuristic solutions obtained for the large-scale literature benchmark problems and the two supports $\mathcal{Q}_\mathcal{B}$ and $\mathcal{Q}_\mathcal{F}$. Each problem is solved 10 times starting from a different random seed. The time limit is set to 1h CPU for each run.**

Problem†	Num.	Num.	Dur.	Serv.	Veh.	Budgets ($\mathcal{Q}_\mathcal{B}$)				Factor model ($\mathcal{Q}_\mathcal{F}$)			
	Cust.	Veh.	Lim.	Time	Cap.	Best	Avg.	Worst	t_{best}(s)	Best	Avg.	Worst	t_{best}(s)
cmt-01	50	5	∞	0	192	519.43	519.45	519.49	0	519.43	519.44	519.49	0
cmt-02	75	10	∞	0	168	807.15	810.08	813.73	8	785.86	793.25	805.11	9
cmt-03	100	8	∞	0	240	803.33	806.91	813.16	142	794.52	799.83	810.23	237
cmt-04	150	12	∞	0	240	1,012.80	1,027.55	1,039.16	1,337	1,006.33	1,017.56	1,031.88	1,441
cmt-05	199	16	∞	0	240	1,254.38	1,266.11	1,276.94	3,214	1,243.27	1,254.04	1,269.06	1,500
cmt-06	50	6	200	10	192	555.43	555.43	555.43	2	555.43	555.43	555.43	1
cmt-07	75	11	160	10	168	902.01	902.16	903.57	21	901.40	901.40	901.40	4
cmt-08	100	9	230	10	240	865.50	867.10	870.69	31	865.50	866.54	872.32	7
cmt-09	150	14	200	10	240	1,167.06	1,174.08	1,188.71	2,551	1,163.81	1,171.41	1,178.33	920
cmt-10	199	18	200	10	240	1,412.10	1,419.88	1,435.33	798	1,410.65	1,415.15	1,422.33	2,027
cmt-11	120	7	∞	0	240	1,005.10	1,006.08	1,007.51	261	994.63	996.03	1,000.45	45
cmt-12	100	10	∞	0	240	808.90	809.43	811.30	2	804.08	804.08	804.08	3
cmt-13	120	11	720	50	240	1,547.06	1,558.19	1,566.64	95	1,544.90	1,550.57	1,558.64	3,087
cmt-14	100	11	1040	90	240	847.43	847.43	847.43	2	835.11	835.31	836.70	1
gol-01	240	9	650	0	660	5,694.68	5,736.31	5,786.97	2,876	5,698.06	5,742.69	5,781.46	682
gol-02	320	10	900	0	840	8,557.12	8,644.37	8,765.58	584	8,544.31	8,595.61	8,669.85	3,362
gol-03	400	10	1200	0	1080	11,362.36	11,556.63	11,769.02	423	11,423.06	11,537.64	11,622.89	1,894
gol-04	480	10	1600	0	1200	14,134.17	14,293.78	14,479.57	3,505	13,975.76	14,182.68	14,351.98	3,413
gol-05	200	5	1800	0	1080	6,466.68	6,560.29	6,738.52	818	6,460.98	6,574.13	6,665.41	492
gol-06	280	7	1500	0	1080	8,414.28	8,516.33	8,592.39	1,666	8,415.21	8,530.00	8,595.63	2,625
gol-07	360	9	1300	0	1080	10,266.87	10,409.49	10,574.95	3,012	10,203.57	10,333.52	10,411.69	3,583
gol-08	440	11	1200	0	1080	12,078.23	12,202.96	12,306.87	1,830	12,074.27	12,166.00	12,253.10	3,046
gol-09	255	14	∞	0	1200	570.63	577.92	583.60	1,224	562.65	568.35	572.25	3,537
gol-10	323	16	∞	0	1200	736.41	743.12	746.36	2,750	724.61	731.17	735.08	3,151
gol-11	399	18	∞	0	1200	925.88	933.12	940.25	2,347	912.17	918.26	927.10	3,043
gol-12	483	19	∞	0	1200	1,181.15	1,192.50	1,213.49	2,099	1,114.85	1,127.18	1,135.78	3,332
gol-13	252	26	∞	0	1200	844.05	849.95	853.71	3,424	838.49	841.72	845.25	1,286
gol-14	320	30	∞	0	1200	1,080.59	1,084.47	1,088.18	2,218	1,070.32	1,075.00	1,082.29	2,589
gol-15	396	33	∞	0	1200	1,341.21	1,352.83	1,358.16	3,109	1,331.96	1,337.92	1,343.65	2,560
gol-16	480	37	∞	0	1200	1,641.95	1,677.06	1,699.75	3,569	1,629.12	1,635.50	1,647.92	3,139
gol-17	240	22	∞	0	240	713.68	716.97	721.15	2,689	701.42	705.45	708.65	2,974
gol-18	300	27	∞	0	240	1,012.90	1,015.86	1,017.29	2,132	991.54	996.41	1,002.97	2,445
gol-19	360	33	∞	0	240	1,389.32	1,399.93	1,404.73	2,160	1,369.57	1,373.81	1,376.83	2,373
gol-20	420	38	∞	0	240	1,860.84	1,871.24	1,876.39	2,910	1,827.02	1,832.72	1,841.28	3,569

†Instances "cmt" are from Christofides et al. (1979), while instances "gol" are from Golden et al. (1998).

parameter setting $(\alpha, \beta) = (0.0, 0.0)$). The results of this sensitivity analysis are reported in Table 6. We observe that the robust solutions are slightly more expensive than their deterministic counterparts, with values monotonically increasing as the size of the uncertainty set increases. The largest increases are exhibited for parameter setting $(\alpha, \beta) = (0.2, 1.0)$ and are 4.37% and 4.09% for the uncertainty supports $\mathcal{Q}_\mathcal{B}$ and $\mathcal{Q}_\mathcal{F}$, respectively. This indicates that selecting modestly more expensive routes (in the order of 5%) suffices to immunize the delivery schedule for a considerable random increase in customer demands (up to 20%). No significant difference can be observed between the two support types. These findings are in agreement with what has been reported in Gounaris et al. (2013) for different problem instances.

Table 6 Increase in transportation costs relative to the deterministic CVRP solutions for the supports $\mathcal{Q}_\mathcal{B}$ (left) and $\mathcal{Q}_\mathcal{F}$ (right). The tables report the averages across the 14 benchmark instances from Christofides et al. (1979).

$\alpha\backslash\beta$	0.00	0.25	0.50	0.75	1.00
0.00	0.00%	0.00%	0.00%	0.00%	0.00%
0.05	0.78%	0.94%	0.96%	0.97%	1.00%
0.10	1.51%	1.73%	1.81%	1.81%	1.91%
0.15	2.45%	2.72%	2.83%	2.84%	2.86%
0.20	3.84%	4.11%	4.17%	4.22%	4.37%

$\alpha\backslash\beta$	0.00	0.25	0.50	0.75	1.00
0.00	0.00%	0.00%	0.00%	0.00%	0.00%
0.05	0.26%	0.45%	0.52%	0.59%	0.75%
0.10	0.62%	0.87%	1.04%	1.28%	1.80%
0.15	0.93%	1.24%	1.84%	2.15%	2.71%
0.20	1.34%	1.91%	2.65%	3.21%	4.09%

6. Conclusions

In this paper we have developed an Adaptive Memory Programming metaheuristic for the Robust Capacitated Vehicle Routing Problem (RCVRP) under demand uncertainty. Metaheuristics for vehicle routing problems have in common that their efficiency and effectiveness strongly depends on the neighborhood size and speed of neighborhood evaluation. While route feasibility can be checked very efficiently in deterministic problems, verification of robust route feasibility in the RCVRP requires the solution of an optimization problem for each candidate route. We have presented two classes of polyhedral uncertainty sets for which route feasibility can be determined very efficiently: budget sets and factor models. Our numerical results demonstrate the effectiveness of our framework on benchmark instances from the literature with up to 483 customers and 38 vehicles. We have identified new best solutions for 123 instances and, by combining our heuristic with a branch-and-cut algorithm, we have solved 3 instances to global optimality for the first time.

Acknowledgments

The authors gratefully acknowledge support from the National Science Foundation and the Research Centre of Athens University of Economics and Business.

References

Aarts, E., J. K. Lenstra. 2003. *Local Search in Combinatorial Optimization*. Princeton University Press, Princeton NJ, USA.

Adelman, D. 2004. A price-directed approach to stochastic inventory/routing. *Operations Research* **52**(4) 499–514.

Agra, A., M. Christiansen, R. Figueiredo, L.M. Hvattum, M. Poss, C. Requejo. 2013. The robust vehicle routing problem with time windows. *Computers & Operations Research* **40**(3) 856–866.

Ahuja, R. K., T. L. Magnanti, J. B. Orlin. 1993. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Upper Saddle River NJ, USA.

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

Baldacci, R., N. Christofides, A. Mingozzi. 2008. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming Ser. A* **115**(2) 351–385.

Baldacci, R., A. Mingozzi, R. Roberti. 2012. Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research* **218**(1) 1–6.

Baldacci, R., P. Toth, D. Vigo. 2010. Exact algorithms for routing problems under vehicle capacity constraints. *Annals of Operations Research* **175**(1) 213–245.

Bandi, C., D. Bertsimas. 2012. Tractable stochastic analysis in high dimensions via robust optimization. *Mathematical Programming Ser. B* **134**(1) 23–70.

Ben-Tal, A., L. El Ghaoui, A. Nemirovski. 2009. *Robust Optimization*. Princeton University Press, Princeton NJ, USA.

Ben-Tal, A., A. Nemirovski. 1999. Robust solutions of uncertain linear programs. *Operations Research Letters* **25**(1) 1–13.

Ben-Tal, A., A. Nemirovski. 2000. Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical Programming Ser. A* **88**(3) 411–424.

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

Bertsekas, D. P. 2007. *Dynamic Programming and Optimal Control*, vol. 2. Athena Scientific.

Bertsimas, D., D. B. Brown, C. Caramanis. 2011. Theory and applications of robust optimization. *SIAM Review* **53**(3) 464–501.

Bertsimas, D., M. Sim. 2004. The price of robustness. *Operations Research* **52**(1) 35–53.

Bertsimas, D., A. Thiele. 2006. Robust and data-driven optimization: Modern decision making under uncertainty. M. P. Johnson, B. Norman, N. Secomandi, eds., *INFORMS TutORials in Operations Research: Models, Methods, and Applications for Innovative Decision Making*, chap. 4. INFORMS, Hanover MD, USA, 95–122.

Bertsimas, D., J. N. Tsitsiklis. 1997. *Introduction to Linear Optimization*. Athena Scientific, Belmont MA, USA.

Birge, J. R., F. Louveaux. 2011. *Introduction to Stochastic Programming*. 2nd ed. Springer, New York NY, USA.

Boyd, S., L. Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press, Cambridge, UK.

Ceria, S., R. A. Stubbs. 2006. Incorporating estimation errors into portfolio selection: Robust portfolio construction. Axioma Research Paper 3, Axioma, Inc.

Chen, W., M. Sim, J. Sun, C.-P. Teo. 2010. From CVaR to uncertainty set: Implications in joint chance-constrained optimization. *Operations Research* **58**(2) 470–485.

Christofides, N., A. Mingozzi, P. Toth. 1979. The vehicle routing problem. N. Christofides, A. Mingozzi, P. Toth, C. Sandi, eds., *Combinatorial Optimization*. Wiley, Chichester, UK, 315–338.

Clarke, G., J. W. Wright. 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research* **12**(4) 568–589.

Cordeau, J.-F., G. Laporte, M. W. P. Savelsbergh, D. Vigo. 2007. Vehicle routing. C. Barnhart, G. Laporte, eds., *Transportation*, *Handbooks in Operations Research and Management Science*, vol. 14, chap. 6. North-Holland, Amsterdam, The Netherlands, 367–428.

Delage, E., Y. Ye. 2010. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research* **58**(3) 596–612.

Erera, A. L., J. C. Morales, M. W. P. Savelsbergh. 2010. The vehicle routing problem with stochastic demand and duration constraints. *Transportation Science* **44**(4) 474–492.

Fukasawa, R., H. Longo, J. Lysgaard, M. P. de Aragão, M. Reis, E. Uchoa, R. F. Werneck. 2006. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming Ser. A* **106**(3) 491–511.

Garg, N., J. Könemann. 2007. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM Journal on Computing* **37**(2) 630–652.

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

Gendreau, M., C.D. Tarantilis. 2010. Solving large-scale vehicle routing problems with time windows: The state-of-the-art. Technical Report CIRRELT-2010-04, Université de Motréal.

Glover, F. 1997. Tabu search and adaptive memory programming - advances, applications and challenges. R.S. Barr, R.V. Helgason, J.L. Kennington, eds., *Interfaces in Computer Science and Operation Research: Advances in Metaheuristics*. Kluwer, Boston MA, US, 1–75.

Golden, B., S. Raghavan, E. Wasil, eds. 2008. *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, New York NY, USA.

Golden, B. L., E. A. Wasil, J. P. Kelly, I. M. Chao. 1998. The impact of metaheuristics on solving the vehicle routing problem: Algorithms, problem sets, and computational results. T. G. Crainic, G. Laporte, eds., *Fleet Management and Logistics*. Kluwer, Boston MA, USA, 33–56.

Goodson, J.C., J.W. Ohlmann, B.W. Thomas. 2013. Rollout policies for dynamic solutions to the multivehicle routing problem with stochastic demand and duration limits. *Operations Research* **61**(4) 138–154.

Gounaris, C. E., W. Wiesemann, C. A. Floudas. 2012. The robust rounded capacity inequalities. *Book of Extended Abstracts of the 5th International Workshop on Freight Transportation and Logistics, ODYSSEUS-2012*. 38–41.

Gounaris, C. E., W. Wiesemann, C. A. Floudas. 2013. The robust capacitated vehicle routing problem under demand uncertainty. *Operations Research,* **61**(3) 677–693.

Hvattum, L.M., A. Lokketangen, G. Laporte. 2006. Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic. *Transportation Science* **40**(4) 421–438.

IBM Corp. 2009. ILOG CPLEX Optimization Suite 12.

Kenyon, A. S., D. P. Morton. 2003. Stochastic vehicle routing with random travel times. *Transportation Science* **37**(1) 69–82.

Kleywegt, A. J., V. S. Nori, M. W. P. Savelsbergh. 2004. Dynamic programming approximations for a stochastic inventory routing problem. *Transportation Science* **38**(1) 42–70.

Kuhn, D., W. Wiesemann, A. Georghiou. 2011. Primal and dual linear decision rules in stochastic and robust optimization. *Mathematical Programming Ser. A* **130**(1) 177–209.

Kytöjoki, J., T. Nuortio, O. Bräysy, M. Gendreau. 2007. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research* **34**(9) 2743–2757.

Laporte, G. 2009. Fifty years of vehicle routing. *Transportation Science* **43**(4) 408–416.

Laporte, G., F. Louveaux, L. Van Hamme. 2002. An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research* **50**(3) 415–423.

Li, F., B. Golden, E. Wasil. 2005. Very large-scale vehicle routing: New test problems, algorithms, and results. *Computers & Operations Research* **32**(5) 1165–1179.

Li, Z., R. Ding, C. A. Floudas. 2011a. A comparative theoretical and computational study on robust counterpart optimization: I. robust linear optimization and robust mixed integer linear optimization. *Industrial & Engineering Chemistry Research* **50**(18) 10567–10603.

Li, Z., Q. Tang, C. A. Floudas. 2011b. A comparative theoretical and computational study on robust counterpart optimization: II. probabilistic guarantees on constraint satisfaction. *Industrial & Engineering Chemistry Research* **51**(19) 6769–6788.

Lysgaard, J., A. N. Letchford, R. W. Eglese. 2004. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming Ser. A* **100**(2) 423–445.

Nagata, Y., O. Bräysy. 2009. Edge assembly based memetic algorithm for the capacitated vehicle routing problem. *Networks* **54**(4) 205–214.

NEO Research Group, Universidad de Málaga. 2012. `http://neo.lcc.uma.es/vrp/`.

Ordóñez, F. 2010. Robust vehicle routing. J. J. Hasenbein, ed., *INFORMS TutORials in Operations Research*, chap. 7. INFORMS, Hanover MD, USA, 153–178.

Prékopa, A. 1995. *Stochastic Programming*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Prins, C. 2004. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research* **31**(12) 1985–2002.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Reimann, M., K. Doerner, R. F. Hartl. 2004. D-ants: savings based ants divide and conquer the vehicle routing problem. *Computers & Operations Research* **31**(4) 563–591.

Repoussis, P.P., C.D. Tarantilis. 2010. Solving the fleet size and mix vehicle routing problem with time windows via adaptive memory programming. *Transportation Research Part C* **18** 695–712.

Repoussis, P.P., C.D. Tarantilis, O. Braysy, G. Ioannou. 2010. A hybrid evolution strategy for the open vehicle routing problem. *Computers & Operations Research* **37**(3) 443–455.

Resende, M.G.C., C.C. Ribeiro. 2010. Greedy randomized adaptive search procedures: Advances and applications. M. Gendreau, J.-Y. Potvin, eds., *Handbook of Metaheuristics*. Springer Publishers, New York NY, USA, 281–317.

Rochat, Y., E.D. Taillard. 1995. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics* **1**(1) 147–167.

Schrijver, A. 2003. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer.

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

Shapiro, A., D. Dentcheva, A. Ruszczyński. 2009. *Lectures on Stochastic Programming: Modeling and Theory*. MPS-SIAM Series on Optimiation, SIAM, Philadelphia PA, USA.

Shen, Z., M. M. Dessouky, F. Ordóñez. 2009. A two-stage vehicle routing model for large-scale bioterrorism emergencies. *Networks* **54**(4) 255–269.

Smith, S. L., M. Pavone, F. Bullo, E. Frazzoli. 2010. Dynamic vehicle routing with priority classes of stochastic demands. *SIAM Journal on Control and Optimization* **48**(5) 3224–3245.

Sungur, I., F. Ordóñez, M. M. Dessouky. 2008. A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions* **40**(5) 509–523.

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

Taillard, E.D. 2001. Adaptive memory programming: A unified view of metaheuristics. *European Journal of Operational Research* **135**(1) 1–16.

Tarantilis, C.D. 2005. Solving the vehicle routing problem with adaptive memory programming methodology. *Computers & Operations Research* **32**(9) 2309–2327.

Tarantilis, C.D., A. Anagnostopoulou, P.P. Repoussis. 2012. Adaptive path relinking for vehicle routing and scheduling problems with product returns. *Transportation Science,* Accepted for publication.

Tarantilis, C.D., C.T. Kiranoudis. 2002. Boneroute: An adaptive memory-based method for effective fleet management. *Annals of Operations Research* **115**(1-4) 227–241.

Toth, P., D. Vigo, eds. 2002. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications, SIAM, Philadelphia PA, USA.

Toth, P., D. Vigo. 2003. The granular tabu search (and its application to the vehicle routing problem). *INFORMS Journal on Computing* **15**(4) 333–348.

Vidal, T., T.G. Crainic, M. Gendreau, N. Lahrichi, W. Rei. 2012. A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. *Operations Research* **60(3)** 611–624.

Wiesemann, W., D. Kuhn, M. Sim. 2013. Distributionally robust convex optimization. Available on Optimization Online.

Xu, H., S. Mannor. 2012. Distributionally robust Markov decision processes. *Mathematics of Operations Research* **37**(2) 288–300.

Xu, J., J. P. Kelly. 1996. A new network flow-based tabu search heuristic for the vehicle routing problem. *Transportation Science* **30**(4) 379–393.

Yang, W.-H., K. Mathur, R. H. Ballou. 2000. Stochastic vehicle routing problem with restocking. *Transportation Science* **34**(1) 99–112.

Young, N. E. 2001. Sequential and parallel algorithms for mixed packing and covering. *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*. 538–546.