Universidade Federal da Paraíba
Centro de Informática
Graduação em Engenharia da Computação

A Unified Matheuristic for a Class of Vehicle Routing Problems Under Time and Demand Uncertainty

Carlos Vinícius Costa Neves

João Pessoa - PB
2023

Carlos Vinícius Costa Neves

A Unified Matheuristic for a Class of Vehicle Routing Problems Under Time and Demand Uncertainty

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal da Paraíba (UFPB), como requisito para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: Prof. Dr. Anand Subramanian

João Pessoa - PB
2023

Catalogação na publicação
Seção de Catalogação e Classificação

N518m Neves, Carlos Vinícius Costa.
Uma matheurística unificada para uma classe de problemas de roteamento de veículos com demandas e tempos incertos / Carlos Vinícius Costa Neves. - João Pessoa, 2023.
46 f. : il.

Orientação: Anand Subramanian.
TCC (Graduação) - UFPB/CI.

1. Roteamento de veículos. 2. Incertezas. 3. Otimização robusta. 4. Matheurística. I. Subramanian, Anand. II. Título.

UFPB/CI CDU 004

Elaborado por Michelle de Kássia Fonseca Barbosa - CRB-738

Carlos Vinícius Costa Neves

A Unified Matheuristic for a Class of Vehicle Routing Problems Under Time and Demand Uncertainty

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal da Paraíba (UFPB), como requisito para obtenção do grau de Bacharel em Engenharia da Computação.

Trabalho aprovado. João Pessoa - PB, 17 de novembro de 2023:

gov.br Documento assinado digitalmente
ANAND SUBRAMANIAN
Data: 24/11/2023 18:18:28-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Anand Subramanian
Orientador

gov.br Documento assinado digitalmente
TEOBALDO LEITE BULHOES JUNIOR
Data: 24/11/2023 20:31:03-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Teobaldo Bulhões
Examinador

gov.br Documento assinado digitalmente
BRUNO PETRATO BRUCK
Data: 24/11/2023 21:14:01-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Bruno Bruck
Examinador

João Pessoa - PB
2023

Resumo

Este trabalho aborda uma classe de PRVs com demandas e tempos de viagem incertos, considerando múltiplos atributos como frotas heterogêneas, múltiplos depósitos e janelas de tempo. Assume-se que as demandas e tempos de viagem são variáveis aleatórias que pertencem a um conjunto de incertezas com *budget*, e o problema é abordado sob o escopo da otimização robusta. Para resolver o problema, uma matheurística com *multi-start* que combina o *framework Iterated Local Search* com um método exato de particionamento de conjuntos foi empregada. Estruturas de dados foram incorporadas ao método de busca local para tornar mais eficiente a checagem de viabilidade de movimentos. Experimentos computacionais foram feitos em conjuntos de instâncias de *benchmark* para vários casos particulares, como o PRV com janelas de tempo e o PRV com frota heterogênea. Assim como em outros trabalhos, as instâncias foram ligeiramente modificadas para acomodar as incertezas de demanda e tempo e para aferir a performance do algoritmo. No geral, os resultados são promissores, já que o método proposto alcança as melhores soluções conhecidas na maioria dos casos, e melhora diversas instâncias em aberto.

Palavras-chave: Roteamento de Veículos. Incertezas. Otimização Robusta. Matheurística.

Abstract

This work address a class of VRPs under uncertain demands and travel times, considering multiple attributes such as heterogeneous fleet, multiple depots, and time windows. We assume customer demands and travel times are random variables that belong to a budgeted uncertainty set, and approach the problems under the robust optimization paradigm. To solve them, we implement a multi-start matheuristic that combines the iterated local search framework with an exact set partitioning procedure. Data structures were incorporated into the local search procedure to make movement feasibility checking more efficient. We conduct computational experiments over several sets of benchmark instances for many particular cases, such as VRPTW, HFVRP, and the classical CVRP. As also done in related works, we perform slight modifications (e.g., increase vehicle capacity, generate uncertainty sets) in each instance to account for uncertainties and properly assess performance in the robust counterparts. Overall, the results are promising, as our method achieves best-known solutions in most cases, as well as improves many of the open instances.

Keywords: Vehicle Routing. Uncertainties. Robust Optimization. Matheuristic.

List of Tables

List of Figures

Contents

1 Introduction

1.1 Preliminaries

The Vehicle Routing Problem (VRP) is one of the most famous problems in Combinatorial Optimization (CO). A VRP asks for a set of routes with minimal cost to serve a group of customers — each with specific product demands — by using a fleet of vehicles. Here, "cost" is an intentionally broad term, as its definition heavily depends on the goal. For instance, when planning a route, one may consider (*i*) its transportation costs, (*ii*) the amount of pollution emitted by the vehicle, (*iii*) the total duration of services, or some other aspect. More often than not, practical applications impose additional constraints, including time windows that restrict arrival times at customers, vehicle capacities that limit the quantity of products a vehicle can transport, and constraints on the size of the vehicle fleet.

The VRP is widely known to be $\mathcal{NP}$-hard, as one of its special cases, the Traveling Salesman Problem (TSP) is also known to be $\mathcal{NP}$-hard (GAREY; JOHNSON, 1979). As such, there are currently no efficient (polynomial time) algorithms to solve the VRP to optimality as of yet. Furthermore, in the likely scenario where $\mathcal{P} \neq \mathcal{NP}$, that will remain to be the case. Nevertheless, the utility of the VRP in a wide range of real-life applications has motivated extensive research efforts towards solving it ever since its introduction by (DANTZIG; RAMSER, 1959). Today, many methods can achieve provably optimal solutions for instances with hundreds of customers, while other approaches are able to provide satisfactory solutions for instances involving thousands of customers; both in reasonable amounts of time.

Algorithms for dealing with $\mathcal{NP}$-hard CO problems can be categorized as either exact or heuristic. In the former case, provably optimal solutions are obtained through exhaustive search. Since exhaustive search takes exponential time in the worst case, efforts are often geared towards limiting the search space. Heuristic algorithms, on the other hand, employ educated guesses to find high-quality solutions within reasonable time. Some heuristics, labeled metaheuristics, are even general enough to act as frameworks for addressing entire classes of problems. Further yet, one may enhance a heuristic by combining it with mathematical programming. One such heuristic is called a matheuristic.

Many algorithms for solving VRPs are primarily designed under the assumption that data (e.g., customer demands, traveling times between customers) is known for certain beforehand. While that is indeed the case on occasion, most practical CO problems are, at least to some extent, subject to uncertainty. More importantly, solutions obtained through methods that disregard uncertainty may become invalid. As an example, consider

a scenario where all demands in a planned route are up 10% higher than previously thought. The unexpected increase in load could potentially violate the vehicle's capacity constraint, rendering the route infeasible. The decision-maker is then faced with the choice of either attempting a quick repair of the solution, or solving the entire problem again. For distinction, we hereafter use the term "deterministic" to denote optimization problems that do not account for uncertainties.

One possible way to address uncertainty is Stochastic Optimization (SO). In SO, uncertain parameters are considered as random variables with known probability distributions. The constraints and the objective are then treated as functions of the expected values of these random variables. Typically, probability distributions must be discretized so they can be represented with finitely many scenarios, which can be done with a Monte Carlo approach. Generally, high-quality of solutions and algorithmic complexity are directly tied to a higher number of sampled scenarios.

Another approach is Robust Optimization (RO) (Aharon Ben-Tal; Laurent El Ghaoui; Arkadi Nemirovski, 2009). In RO, the problem's parameters are assumed to be contained within a well-defined uncertainty set. A solution is deemed feasible if and only if it is feasible for all scenarios belonging to said uncertainty set. One advantage is that no knowledge about the data distributions is required. The decision-maker is given flexibility by being able to tailor the uncertainty set to their needs: broader uncertainty sets lead to costly solutions protected against many scenarios, while more strict uncertainty sets lead to less protected, yet less expensive solutions. Another advantage of RO is that tractability is often transferable, that is, the robust counterpart of a CO problem is likely to remain tractable as long as the deterministic version is.

In this work, we propose a matheuristic for the Heterogeneous Fleet VRP with Time Windows (HFVRPTW) under time and demand uncertainty. To address uncertainties, we employ a RO-based approach.

1.2 Motivation

Most VRPs arise as a necessity in supply-chain systems, as they can be used to manage the delivery of finished goods to end consumers. In fact, transportation volumes worldwide are so huge that plannings — most of which are done manually to this day — which provide savings of even 2% would have significant positive impact (HASLE; LIE; QUAK, 2007). Other non-obvious real-life applications of VRPs include robotics, circuit board assembly, and even DNA sequencing (GUTIN et al., 2007). Given such a wide range of applications, it is crucial to consider the challenges posed by uncertainty in this context.

It is worth noting that in the specific case of VRP, uncertainty is commonly observed on travel times and service times, as well as on customer demands (SUNGUR;

ORDóñEZ; DESSOUKY, 2008; MUNARI et al., 2019). Since predicting variation of uncertain data greatly increases the complexity of route planning, there is a notable a gap in the existing VRP literature concerning algorithms that incorporate uncertainty. Studies in this area, which are relatively recent, show that the potential practical benefits of addressing uncertainty in route planning are substantial (GENDREAU; JABALI; REI, 2014; JAILLET; QI; SIM, 2016; MUNARI et al., 2019).

As previously stated, a RO-based approach requires a well-defined uncertainty set. While many types of uncertainty sets with distinct properties have been proposed, cardinality constrained uncertainty sets (BERTSIMAS; SIM, 2003; BERTSIMAS; SIM, 2004) are certainly among the most used for solving VRPs under RO. Their linear structure allows efficient implementation into mathematical models through either dualization or linearization of recursive equations. Moreover, they offer a straightforward interpretation, granting the decision-maker direct control over the number of problem parameters that can reach their worst-case value.

Currently, there are several state-of-the-art heuristic methods to solve deterministic VRPs. One such approach is Unified Hybrid Genetic Search (UHGS), proposed in (VIDAL et al., 2013). UHGS operates by choosing two individuals an initial population, where each individual is a giant tour — a single trip that visits all customers, starting from the depot. The two individuals are combined into a single giant tour, which is then optimally split into routes that form an actual solution. Said solution is then improved through well-known local search operators. One potential drawback of UHGS is that it may be difficult to adapt the splitting algorithm to account for uncertainties. Moreover, when dealing with heterogeneous, limited fleets, the problem of optimally splitting a giant tour reduces to a shortest path problem with resource constraints, which is $\mathcal{NP}$-hard.

Another method, Slack Induction by String Removals (SISRs), proposed in (CHRISTIAENS; BERGHE, 2020), uses a ruin-and-recreate approach. During the ruining phase, related customers are removed from the solution, introducing spatial and capacity slacks. Absent customers are then reinserted into the partial solution using a greedy algorithm. While SISRs is reportedly applicable to several problems associated to the capacitated VRP (CVRP), we argue that incorporating heterogeneous, possibly limited fleets into the main algorithm may be a nontrivial task. Additionally, if time and demand uncertainties are considered, not only may the changes in spatial and capacity slacks become harder to manage, but the search-space of feasible solutions may also become more restricted and difficult to navigate.

One more metaheuristic that is worthy of attention is Iterated Local Search (ILS) (LOURENçO; MARTIN; STüTZLE, 2019). As the name suggests, ILS works by iteratively improving the current solution through local search. In case no further improvements can be made, the best solution undergoes a slight perturbation, and the local search is executed again. This process is repeated until a predefined stopping criterion is met.

ILS was sucessfully employed to solve deterministic VRPs with heterogeneous fleets on multiple occasions (SUBRAMANIAN et al., 2012; PENNA et al., 2019; MáXIMO; CORDEAU; NASCIMENTO, 2022). Another advantage of ILS is its independence from overly complex subroutines. Moreover, provided that one already has an implementation of ILS for a deterministic CO problem, adapting to its robust counterpart is theoretically straightforward: generally, it suffices to modify the solution feasibility checking part of the local search procedure. Of course, in the context of RO, this highly depends on the chosen uncertainty set, and may require additional data structures and algorithms if one wishes not to compromise performance.

1.3 Objectives

Objectives of this work are as follows.

- Define the uncertainty sets of travel times and customer demands.
- Review the approaches for VRPs with uncertainty proposed in the literature.
- Devise a local-search and integer programming based matheuristic for solving a class of VRPs.
- Conduct experiments to test the proposed method on instances from literature.

1.4 Organization

The remainder of this work is organized as follows.

- Chapter 2 provides a review of related works.
- Chapter 3 provides a formal description to the addressed class of problems.
- Chapter 4 gives a mathematical formulation for the class of problems.
- Chapter 5 describes the proposed matheuristic.
- Chapter 6 presents the computational results of our method.
- Chapter 7 contains the concluding remarks of this work.

2 Literature Review

2.1 The Cardinality Constrained Uncertainty Set

The cardinality constrained uncertainty set was proposed in (BERTSIMAS; SIM, 2004). In line with the common practice in RO, it defines a polytope that contains all possible realizations of the problem parameters, and a solution is considered feasible if and only if it is feasible for all realizations. The size of the set is regulated by a scalar, often called "budget", which controls the number of parameters that may reach their worst-case values. For instance, if the budget of the demand uncertainty set is equal to 5, up to 5 customer demands may reach their worst-case scenario. Since we are mainly concerned with time and demand uncertainties, it is prudent to consider two separate, independent cardinality constrained uncertainty sets, as will be shown in the next chapter.

To formally define the cardinality constrained set, consider a vector $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$, whose components may represent customer demands or traveling times, for example. Furthermore, let $\bar{y}, \hat{y} \in \mathbb{R}^n$ be two vectors of constants. The cardinality constrained set, which contains all possible realizations of y and is controlled by the budget Γ, is defined as

$$\mathcal{U} = \left\{ y \in \mathbb{R}^n \mid y_i = \bar{y}_i + \hat{y}_i \gamma_i, \sum_{i \in V} \gamma_i \leq \Gamma, \gamma_i \in [0,1], i \in \{1, \ldots, n\} \right\}. \tag{2.1}$$

2.2 Approaches to solve VRPs with uncertainty

The following sections contain a review on heuristic and exact approaches to solve VRPs under uncertainty. Reviewed works are summarized within Table 1.

2.2.1 Exact Approaches for VRPs Under Uncertainty

State-of-the-art approaches for deterministic VRPs often serve as a basis for exact approaches addressing VRPs under uncertainty. What may be observed in the following works is that exact algorithms in the robust vehicle routing literature primarily differ in their methods of incorporating uncertainty into the problem. For instance, in certain column-generation based approaches, uncertainty handling is delegated to the pricing problem. Meanwhile, other methods introduce uncertainty through addition of valid inequalities. Likewise, in compact formulations, uncertainty may be incorporated either through the dualization or linearization of recursive equations.

(SUNGUR; ORDóñEZ; DESSOUKY, 2008) was the first work to use RO to solve a VRP. To handle the VRP with Stochastic Demands, they use the well-known MTZ

mathematical formulation for the CVRP. The authors identify the vehicle load inequalities as the only set of constraints that requires modification to accomodate uncertainty, whereas the rest of the model remains unchanged. They then observe that for a solution to be robust, it suffices for it to be feasible in the worst-case scenario, that is, the scenario where the the sum of demands is maximal. By employing duality in this implicit optimization problem, the authors derive a compact, equivalent mathematical model. (ORDóñEZ, 2010) extends the approach to incorporate traveling cost uncertainty with cardinality constrained sets, as well as other classes of uncertainty, such as vehicle-dependent uncertainty sets. Among other results, both works report that robust solutions tend to cleverly distribute vehicle capacities so that multiple distinct scenarios can be accomodated.

Similarly, (PESSOA et al., 2021) also address the robust CVRP. They reformulate the problem as a deterministic heterogeneous fleet CVRP where a customer's demand is determined by the vehicle that serves them. To treat the problem, Branch-Cut-and-Price (BCP) techniques for the heterogeneous CVRP were employed. The paper demonstrates how to strengthen the robust rounded capacity inequalities proposed by (GOUNARIS; WIESEMANN; FLOUDAS, 2013), while also also providing significant speed-ups to the method by presenting general conditions under which pricing subproblems are infeasible, and may thus be avoided. They were able to solve most open instances to optimality. Insights on the usefulness of robust solutions may also be found within their reported results.

(AGRA et al., 2012) proposes a formulation for the VRP with Time Windows (VRPTW) with uncertainty traveling times. Their formulation works by extending the network induced by the original graph, enabling to keep track of the position at which customers are visited. Coupled with dualization of the time windows constraints, this allows to express the problem through a mathematical model whose size is polynomially related to the instance length. (AGRA et al., 2013) propose a more elaborate Branch-and-Cut (BC) algorithm for the same problem. Their cutting planes scheme, which relies on generalization of path inequalities to handle uncertainty, alongside scenario reduction techniques, reportedly yields faster solution times than that of (AGRA et al., 2012). Another example of a dualization-based formulation may be found in (VEGA; MUNARI; MORABITO, 2019), which discusses the VRP with multiple deliveryman and uncertain demands.

(LEE; LEE; PARK, 2012) studies the VRP with Deadlines (VRPD), which is essentially a stricter version of VRPTW where all nodes have the same release time, and only deadlines may vary. The authors propose a path-based exponential formulation to the problem. This formulation is then solved with a Dantzig-Wolfe decomposition approach. Unlike the already mentioned works, however, uncertainty is encapsulated into the subproblems. More precisely, instead of embedding uncertainty into the constraints, the authors treat the pricing problem as a robust shortest path problem with resource

constraints (SPPRC). They solve it by using a tailored labeling algorithm. One drawback of the method is that it can not be directly applied to the VRPTW, since potential waiting times between trips — which are absent in VRPD — may absorb traveling time deviations, compromising their labeling algorithm.

A tailored labeling algorithm was also proposed by (MUNARI et al., 2019) to address the more general robust VRPTW through BCP. Their BCP was able to solve instances with up to 100 customers. Furthermore, extensive simulations show that the probability of capacity and time window constraints being violated by the robust solutions are significantly lower than that of the deterministic ones. The increase in cost incurred by robust solutions was also observed to be low, indicating a generally good trade-off. In the same vein, (LU; GZARA, 2019) proposes a BCP for the robust VRPTW that hinges on a tailored labeling algorithm coupled with 2-path and subset row inequalities to strengthen the master problem. A similar approach is also used by (VEGA; MUNARI; MORABITO, 2020) to solve the robust VRPTW with Multiple Deliveryman.

For a more generalized approach to the robust VRPTW by means of BCP, readers are referred to (WANG; SUBRAMANYAM; GOUNARIS, 2022). The authors demonstrate, for certain uncertainty sets, how to construct a reduced version containing finitely many scenarios. Contrasting the four previous works, the pricing problem is treated as a deterministic SPPRC where each resource constraint is associated to a scenario in the reduced uncertainty set. Note that this only guarantees partially robust solutions. To ensure robust feasibility, rounded capacity inequalities are added to the master problem. This not only allowed the authors encapsulate multiple different uncertainty sets for both demands and traveling times, but also enabled using the already well-established generic package for solving VRPs proposed by (PESSOA et al., 2020). Their computational experiments show that the approach is competitive.

To our knowledge, (MUNARI et al., 2019) were also the first to use linearization of recursive equations arising from the cardinality constrained uncertainty set (AGRA et al., 2013) to propose a compact formulation for the VRPTW. As opposed to dualization-based formulations such as that of (AGRA et al., 2012), which tend to be prohibitive on instances with more than 20 customers, their compact formulation provided feasible solutions — several of which are optimal — for instances with up to 50 customers. One possible explanation is that dualizing uncertainty introduces an excessive amount of variables and constraints to the model, whereas linearizing recursive equations requires fewer, more interrelated additional variables, yielding a stronger formulation. Linearization of recursive equations is also used in (VEGA; MUNARI; MORABITO, 2020) to solve the robust VRPTWMD. In (CAMPOS, 2022), the same approach is employed to incorporate the knapsack uncertainty set to the same problem. One key observation about the last case is that the size of the resulting formulation is pseudo-polynomial, and it relies in discretization of knapsack size as well as travel time and demand deviations.

More recently, (BARTOLINI et al., 2021) showed that, in the case of the knapsack uncertainty set, the worst-case scenario associated to traveling times in a route exhibits a specific structure. By exploiting said structure, they are able to verify feasibility of a route in polynomial time. They use this approach in a two-phase algorithm to solve the robust TSP with Time Windows (TSPTW). During phase 1, the dual of the continuous set partitioning formulation is solved by a column-generation based algorithm. Phase 2 finds the optimal tour by executing a specialized dynamic programming (DP) algorithm that uses knowledge of the reduced costs induced by the best dual solution obtained in phase 1 to fathom of suboptimal paths.

2.2.2 Heuristics for VRPs Under Uncertainty

The works discussed in this section highlight the diversity of proposed metaheuristics in the robust VRP literature. Despite that, most of these approaches depend on a local search engine. Consequently, polyhedral uncertainty sets, including the cardinality-constrained set and the knapsack set, are frequently utilized, as they possess properties that simplify the task of evaluating robust feasibility. Taking uncertainties into consideration also tends to reduce search space size, as fewer solutions remain feasible across all realizations of the uncertainty set. Hence, most approaches incorporate strategies to avoid local optima, such as tabu lists and perturbation mechanisms.

(HU et al., 2018) study a VRPTW with uncertain times and demands, and whose objective is lexicographic, with a primary focus on minimizing the number of vehicles used. To address the problem, they employ the Adaptative Variable Neighborhood Search (AVNS) metaheuristic, and uncertainties are modeled through a slightly modified version of the cardinality constrained uncertainty set where each route has its own time and demand budgets, which are linearly proportional to the number of serviced customers. Local search is used to improve solutions. In their approach, infeasible solutions are allowed, but their cost is penalized. When evaluating a movement, violation is calculated exactly, through (i) sorting demands in the case of capacity constraints and (ii) DP in the case of time-windows constraints, as discussed earlier.

Deviating from most works in the robust VRP literature, (SOLANO-CHARRIS; PRINS; SANTOS, 2015) and (SOLANO-CHARRIS; PRINS; SANTOS, 2016) provide examples of problems where uncertainty has no impact on feasibility. The former addresses the CVRP under traveling time uncertainty, whereas the latter investigates a bi-objective variant of the CVRP under time and demand uncertainty — the objective is to minimize both total traveling time and unmet demands. The two works model uncertainty as a discrete set containing finitely many scenarios, and the objectives are defined using a min-max criterion, that is, the goal is to minimize the worst-case scenario of the cost function. Both works propose multiple metaheuristics to solve the problems, all of which

deeply rely in local search, whose asymptotic complexity is significantly impacted by the number of scenarios.

In addition to their aforementioned BCP algorithm, (PESSOA et al., 2021) also provide a primal heuristic for the CVRP under demand uncertainty based on ILS. The constructive phase of the algorithm is rather straightforward, and works by greedily splitting a randomly generated single-route solution. The local search step relies on neighborhoods such as crossover (2-OPT*) and single customer moves. A key contribution in their heuristic procedure is based on the observation that a single customer transferral can be decomposed into two 2-OPT* moves. The authors further demonstrate a computationally efficient lower bound for the worst-case total demand under both considered uncertainty sets for a 2-OPT* move. If this lower bound exceeds the vehicle's capacity, exact move evaluation may be skipped, providing a substantial speed-up to the heuristic.

The work of (GOUNARIS et al., 2016) introduces the use of disjoint budget sets for modeling uncertainty in VRPs. Budget disjoint sets offer an intuitive view of uncertainty by dividing the set of customers into regions, and imposing an upper limit on the sum of demand deviations of each region. The paper further provides data structures to efficiently compute total demand of a route in the worst-case upon removal or addition of a single customer. Since most inter-route local search operators involve an exchange of customer sets, this allows tabu search to be done efficiently by decomposing movements into sequences of single-customer removals or additions. (YU; ANH; BALDACCI, 2023) also employ the disjoint budget set to solve the VRP with Cross-Docking (VRPCD) under demand uncertainty using Adaptive Large Neighborhood Search (ALNS). They exploit a strategy similar to that of (GOUNARIS et al., 2016) by decomposing destroy and repair operators into single-customer additions or removals, and showing how to efficiently check solution feasibility upon such modifications.

Authors of (SUBRAMANYAM; REPOUSSIS; GOUNARIS, 2018) extend the Adaptive Memory Programming (AMP) framework proposed in (GOUNARIS et al., 2016) to a broader class of VRPs with uncertain demands. The targeted class of problems encompasses multiple aspects, such as heterogeneous limited fleets, and multiple depots. Their approach is divided into two phases: initialization and exploitation. In the initialization phase, a reference set is populated with several feasible solutions obtained with a constructive heuristic. Each solution is then improved through tabu search. Likewise, each iteration of the exploitation phase starts with the construction of a provisional solution by combination of features from the best solutions in the reference set. The provisional solution is then improved through tabu search and inserted in the reference set. The exploitation phase continues until a time limit is reached, and the best solution is returned.

As is done in (GOUNARIS et al., 2016) for knapsack sets, (SUBRAMANYAM; REPOUSSIS; GOUNARIS, 2018) demonstrate, for a variety of uncertainty sets, how to compute the change in total route demand in the worst-case scenario upon the insertion

or removal of a single customer. Note that while this facilitates generalization to any uncertainty set, the reported asymptotic complexities of feasibility evaluation are often not optimal, as each local search move must be decomposed into sequences of removals and additions. If one desires to efficiently evaluate move feasibility, neighborhood-tailored procedures may be necessary for each uncertainty set.

Finally, their extensive experiments include a robust analysis that illustrates the behavior of each uncertainty set, as well as quantifies the impact of robustness on the cost of solutions. The results seem to suggest that AMP performs better than ILS, especially in the deterministic case. It is important to note, however, that the comparison was conducted in a custom implementation that lacks some elements of the ILS approach proposed in (PENNA; SUBRAMANIAN; OCHI, 2013), such as an exact set partitioning method. The authors also propose an integer programming formulation that relies on a generalized version of rounded capacity cuts for CVRP. They further argue that their heuristic solutions are either optimal or near-optimal, as they are often close to the best dual bounds provided by their mathematical formulation.

Table 1 – Overview of VRPs under uncertainty

Work	**Variant**	**Approach**	**Uncertain Parameters**
(SUNGUR; ORDóñEZ; DESSOUKY, 2008)	CVRP	RO model	demands
(ORDóñEZ, 2010)	CVRP	RO model	demands, times, costs
(AGRA et al., 2012)	VRPTW	RO model	times
(AGRA et al., 2013)	VRPTW	BC	times
(GOUNARIS et al., 2016)	CVRP	BC	demands
(LEE; LEE; PARK, 2012)	VRPD	BCP	demands, times
(LU; GZARA, 2019)	VRPTW	BCP	demands
(MUNARI et al., 2019)	VRPTW	BCP	demands, times
(VEGA; MUNARI; MORABITO, 2019)	VRPMD	RO model	demands
(VEGA; MUNARI; MORABITO, 2020)	VRPTWMD	BCP	demands, times
(CAMPOS, 2022)	VRPTW	RO model, BC	demands, times
(BARTOLINI et al., 2021)	TSPTW	BCP	times
(PESSOA et al., 2021)	CVRP	BCP, ILS	demands
(WANG; SUBRAMANYAM; GOUNARIS, 2022)	VRPTW	BCP	demands, times
(SOLANO-CHARRIS; PRINS; SANTOS, 2015)	CVRP	RO model, MS-ILS-GT	costs
(GOUNARIS et al., 2016)	CVRP	AMP	demands
(SOLANO-CHARRIS; PRINS; SANTOS, 2016)	bi-VRP	MOEA, NSGAII	demands, costs
(SUBRAMANYAM; REPOUSSIS; GOUNARIS, 2018)	HFVRP	AMP	demands
(HU et al., 2018)	VRPTW	AVNS	demands, times
(YU; ANH; BALDACCI, 2023)	VRPCD	ALNS	demands
This work	HFVRPTW	ILS	demands, times

3 Problem Definition

Let $V_C = \{1, \ldots, n\}$, $V_D = \{n+1, \ldots, n+m\}$, and $V'_D = \{(n+1)+m, \ldots, (n+m)+m\}$ be the set of n customers, m depots, and m depot copies, respectively. Note that the depots $n+d \in V_D$ e $(n+d)+m \in V'_D$ are identical for all $d \in \{1, \ldots, m\}$. In addition, for all $d \in \{1, \ldots, m\}$, let $A_d = \{(i,j) \mid i \in V_C \cup \{n+d\}, j \in V_C \cup \{n+d+m\}, i \neq j\}$ be the set of arcs connecting customers in in V_C to one another, as well as to the depot in $n+d \in V_D$ and its copy $n+d+m \in V'_D$. Thus, we define a directed graph $G = (V, A)$ such that $V = V_C \cup V_D \cup V'_D$, and $A = \bigcup_{d=1}^{m} A_d$. Each arc $a = (i,j) \in A$ is associated with a traveling cost c_a and a traveling time t_a. Furthermore, let K be the set of available vehicle types. Each vehicle type $k \in K$ has a capacity Q_k, and incurs *(i)* a variable cost α_k for each unity of distance traveled and *(ii)* a fixed cost β_k for each vehicle used. The amount of vehicles of type $k \in K$ which may be used is has an upper limit U_k. Finally, each customer $j \in V_C$ has a demand q_j, a service time s_j, and a time window $[w_j^a, w_j^b]$. For the sake of simplicity, we assume $q_j = 0$ if $j \in V_D \cup V'_D$.

We aim to find a set of routes with minimal cost — which may be composed of both fixed costs and variable costs — such that: (i) each route is traversed to exactly one vehicle $k \in K$, which departs from a depot in V_D at time 0, and ends in its corresponding copy in V'_D after servicing a subset of customers in V_C; (ii) the sum of demands from customers in a route must not exceeded the capacity Q_k of its associated vehicle $k \in K$; (iii) all customer time windows must be satisfied; (iv) each customer must be serviced exactly once; (v) the upper limit U_k on the amount of routes associated to a vehicle of type $k \in K$ must be respected.

We emphasize that both demands and traveling times are uncertain. Hence, similar to the last chapter, we hereby provide a formal description of the time and demand uncertainty sets. Let q and t be the demand and travel time vectors, respectively. Similarly, let $\bar{q}$ and $\bar{t}$, be the "nominal" demand and travel time vectors, and let $\hat{q}$ and $\hat{t}$ be the demand and travel time deviations. The demand and time uncertainty sets, respectively $\mathcal{U}^q$ and $\mathcal{U}^t$, are defined by the following equations.

$$\mathcal{U}^q = \left\{ q \in \mathbb{R}^{|V_C|} \mid q_i = \bar{q}_i + \hat{q}_i \gamma_i^q, \sum_{i \in V_C} \gamma_i^q \leq \Gamma^q, 0 \leq \gamma_i^q \leq 1, i \in V_C \right\} \tag{3.1}$$

$$\mathcal{U}^t = \left\{ t \in \mathbb{R}^{|A|} \mid t_{ij} = \bar{t}_{ij} + \hat{t}_{ij} \gamma_{ij}^t, \sum_{(i,j) \in A} \gamma_{ij}^t \leq \Gamma^t, 0 \leq \gamma_{ij}^t \leq 1, (i,j) \in A \right\} \tag{3.2}$$

Note that if $\Gamma^d = 0$ and $\Gamma^t = 0$, the sets $\mathcal{U}^d$ and $\mathcal{U}^t$ become singletons, which means demands and traveling times are fixed. In this specific case, the problem reduces to its deterministic version.

A well-known result in RO involving linear constraints establishes that, for polyhedral uncertainty sets like $\mathcal{U}^d$ and $\mathcal{U}^t$, only scenarios corresponding to extreme points have to be inspected to verify feasibility. This allows one to employ specialized algorithms to find such scenarios and considerably speed up the verification of robust feasibility. For instance, it is known that the worst-case scenario associated to customer demands is achieved when the Γ^d highest demands reach their worst-case, which can be easily determined by a sorting algorithm. Likewise, as shown in (AGRA et al., 2013; MUNARI et al., 2019), the earliest time a vehicle can arrive in customer v_j when up to Γ^t traveling times reach their worst-case scenario, labeled w_{v_j}, can be computed through DP by using the following equation. Feasibility can then be determined by comparing said vairable to the upper limit of the time window of customer v_j.

$$w_{v_j,\gamma} = \begin{cases} w^a_{v_0}, & \text{se } j = 0; \\ \max\{w^a_{v_j}, w_{v_{j-1},\gamma}, +s_{v_{j-1}} + \bar{t}_{v_{j-1},v_j}\}, & \text{se } \gamma = 0; \\ \max\{w^a_{v_j}, w_{v_{j-1},\gamma}, +s_{v_{j-1}} + \bar{t}_{v_{j-1},v_j}, & \\ w_{v_{j-1},\gamma-1} + s_{v_{j-1}} + \bar{t}_{v_{j-1},v_j} + \hat{t}_{v_{j-1},v_j}\}, & \text{otherwise.} \end{cases} \tag{3.3}$$

4 Mathematical Formulation

Let x_a^{kd} be a binary variable that assumes value 1 iff a vehicle of type $k \in K$ which departed from depot $d \in V_D$ traverses the arc $a \in A_d$. In addition, the continuous variable $u_{a\gamma}$ stores the total load of the vehicle that traverses arc $a \in A$ when up to γ customers assume their worst-case demand. Likewise, the continuous variable $w_{a\gamma}$ stores the arrival time at customer j when up to γ traveling times deviate. We present a compact formulation for the problem beginning with the objective function (4.1), which computes the sum of variable and fixed costs.

$$\text{minimize} \sum_{k\in K}\sum_{d=1}^{m}\sum_{a\in A_d} \alpha_k c_a x_a^{kd} + \sum_{k\in K}\sum_{d=1}^{m}\sum_{a\in\delta^+(n+d)} \beta_k x_a^{kd} \tag{4.1}$$

Constraints (4.2)–(4.3), shown below, are recurrent in VRP literature. They guarantee that each customer is visited exactly once, and that the vehicle finishes at the same depot from which it departed. Constraints (4.4) impose an upper limit on the amount of vehicles of each type, whereas constraints (4.5) delimit the domain of variables x_a^{kd}.

$$\sum_{k\in K}\sum_{d=1}^{m}\sum_{a\in\delta_d^-(j)} x_a^{kd} = 1 \qquad j \in V_C \tag{4.2}$$

$$\sum_{a\in\delta_d^+(i)} x_a^{kd} = \sum_{a\in\delta_d^-(i)} x_a^{kd} \qquad k \in K, d \in \{1,\dots,m\}, i \in V_C \tag{4.3}$$

$$\sum_{d=1}^{m}\sum_{a\in\delta^+(n+d)} x_a^{kd} \le U_k \qquad k \in K \tag{4.4}$$

$$x_a^{kd} \in \mathbb{Z}_+ \qquad k \in K, d \in \{1,\dots,m\}, a \in A_d \tag{4.5}$$

The next set of constraints generalizes load flow inequalities to the robust case. This is achieved by linearizing recursive equations associated with the demand uncertainty set, as initially proposed by (MUNARI et al., 2019). Specifically, constraints (4.6)–(4.7) guarantee an accurate computation of vehicle loads in the worst-case scenario, and constraints (4.8) ensure that vehicle capacities are respected. Additionally, constraints (4.9) define the domain of variables $u_{a\gamma}$.

$$\sum_{a\in d^+(j)} u_{a\gamma} \ge \sum_{a\in\delta^-(j)} u_{a\gamma} + \bar{q}_j \qquad j \in V_C, \gamma = 0,\dots,\Gamma^q \tag{4.6}$$

$$\sum_{a\in d^+(j)} u_{a\gamma} \ge \sum_{a\in\delta^-(j)} u_{a\gamma-1} + \bar{q}_j + \hat{q}_j \qquad j \in V_C, \gamma = 1,\dots,\Gamma^q \tag{4.7}$$

$$u_{a\gamma} \le \sum_{k\in K}\sum_{d\in V(a)} Q_k x_a^{kd} \qquad a \in A, \gamma = 0,\dots,\Gamma^q \tag{4.8}$$

$$u_{a\gamma} \ge 0 \qquad a \in A, \gamma = 0,\dots,\Gamma^q \tag{4.9}$$

The following constraints are responsible for the time aspect of the problem, and were derived using the same approach as the previous ones. Constraints (4.10)–(4.11) ensure that service starting times are correctly computed in the worst-case scenario, and constraints (4.12) guarantee adherence to customer time windows. Lastly, constraints (4.13) delimit the domain of variables $w_{j\gamma}$.

$$w_{j\gamma} \geq w_{i\gamma} + \bar{t}_a + s_i - M \left(1 - \sum_{k \in K} \sum_{d \in V_D(a)} x_a^{kd}\right) \quad \gamma = 0, \ldots, \Gamma^t, a = (i,j) \in A \tag{4.10}$$

$$w_{j\gamma} \geq w_{i\gamma-1} + s_i + \bar{t}_a + \hat{t}_a - M \left(1 - \sum_{k \in K} \sum_{d \in V_D(a)} x_a^{kd}\right) \quad \gamma = 1, \ldots, \Gamma^t, a = (i,j) \in A \tag{4.11}$$

$$w_j^a \leq w_{j\gamma} \leq w_j^b \qquad j \in V_C, \gamma = 0, \ldots, \Gamma^t \tag{4.12}$$

$$w_{j\gamma} \geq 0 \qquad j \in V_C, \gamma = 0, \ldots, \Gamma^t \tag{4.13}$$

5 Proposed Algorithm

The proposed matheuristic combines the ILS metaheuristic (LOURENçO; MARTIN; STüTZLE, 2019) with a set partitioning formulation. The main procedure is described in Algorithm 1. First, a solution constructed by a greedy heuristic (line 5) is improved through the **ILS**. If the cost of solution s' is strictly less than f^*, the current best solution s^* is updated. The process is repeated until a stopping criterion — either a time limit or a certain number of iterations) — is met. Finally, the **SetPartitioning** procedure is used to obtain an optimal combination of routes which were found during local searches and stored in a data structure called pool (line 10). The **SetPartitioning** process is executed until the best solution s^* can not be further improved (lines 11–14). Finally, the procedure returns the best solution s^* (line 15).

The main components of the **ILS** procedure (e.g., constructive procedure, local search, perturbation) are explained in the remainder of this chapter.

Algorithm 1: ILS-SP

```
1  procedure ILS-SP (s*, MaxSPTime)
2      f* ← ∞
3      RoutePool ← {}
4      while stopping criterion not met do
5          s ← Construction()
6          s' ← s
7          s' ← ILS(I_ILS, s, s', RoutePool)
8          if f(s') < f* then
9              s*, f* ← s', f(s')
10     s' ← SetPartitioning(s*, RoutePool, MaxSPTime)
11     do
12         s* ← SetPartitioning(s', RoutePool, MaxSPTime)
13         s', f* ← s*, f(s*)
14     while f(s') < f*
15     return s*
```

5.1 Constructive heuristic

Initial solutions are obtained through the parallel insertion method proposed by (SUBRAMANIAN; UCHOA; OCHI, 2013). First, one empty route is generated for each vehicle. Each route is randomly filled with a single customer. Remaining customers are added greedily so as to minimize traveling costs. During the procedure, both capacity

and time windows constraints are temporarily relaxed, resulting in potentially infeasible solutions. We describe a way to address infeasibility in the next section.

5.2 Penalization of infeasible solutions

The constructive heuristic does not ensure robust feasibility, that is, some routes may violate both capacity and time windows constraints in the worst-case scenario. To account for that, infeasible routes are penalized according to how much they violate said constraints. A route is feasible if and only if its total amount of violation is equal to zero. The sum of route violations is then embedded into the cost of the solution, which is evaluated under a lexicographic criterion. Hence, if the solution returned by the constructive heuristic is infeasible, the local search procedure is encouraged to find a feasible solution first. Once a feasible solution is found, the local search procedure behaves as expected, solely improving the sum of traveling costs.

To show how solutions are penalized, let s be a solution, while R_s is its set of routes. Furthermore, let c_r be the cost of route $r \in R_s$, which includes traveling times and fixed costs. Finally, let $q(r)$ be the total demand of route r in the worst-case, whereas w_{v_j,Γ^t} is the earliest possible arrival time at node v_j when up to Γ^t traveling times deviate; both of which are computed as described in the previous chapter. The capacity and time windows violations of route r, respectively V_r^q and V_r^t, are defined by the following equations.

$$V_r^q = \max\{0, q(r) - C\} \tag{5.1}$$

$$V_r^t = \sum_{v_j \in r} \max\{0, w_{v_j,\Gamma^t} - w_{v_j}^b\} \tag{5.2}$$

The total cost of s is then calculated as

$$f(s) = \sum_{r \in R_s} (c_r + \beta_1 V_r^q + \beta_2 V_r^t), \tag{5.3}$$

where the constant penalty factors, β_1 and β_2 are high enough to impose a lexicographical ordering on solution costs such that violations are prioritized. In other words, the cost of an infeasible solution will always be worse than that of any feasible solution.

5.3 Local Search

The local search procedure is based on Variable Neighborhood Descent with random neighborhood ordering (RVND) (SUBRAMANIAN et al., 2010). The procedure works hierarchically. First, an inter-route neighborhood structure randomly selected from a list is thoroughly explored, and the best improving move, if any, is applied. In case the

explored inter-route neighborhood improves the current solution, the list of inter-route neighborhoods is repopulated with all inter-route neighborhoods, and modified routes are then improved through a intra-route local search. Otherwise, the inter-route neighborhood is removed from the list. The intra-route search has its own list of neighborhoods, and behaves similarly, except that neighborhoods are removed from the list regardless of whether the solution was improved or not. When the intra-route neighborhood list is emptied, the process is repeated, and continues until the inter-route neighborhood list is empty.

Most neighboorhood structures employed — some of which are tailored for specific problem variants — are common in VRP literature. They are described as follows.

- **Shift(1,0)** – a customer is removed from a route r_1 and inserted into another route r_2
- **Shift(2,0)** – a block of two consecutive customers is removed from a route r_1 and inserted into another route r_2
- **Swap(1,1)** – Two customers from different routes are exchanged
- **Swap(2,1)** – A block of two consecutive customers is exchanged with a single customer from a different route
- **Swap(2,2)** – Two blocks of two consecutive customers, each from a different route, are exchanged
- **Cross** – Two arcs, each from a different route, are removed, and the resulting solution is reconstructed
- **Shift-k** – An arbitrarily sized block of consecutive customers is moved to the end of a different route
- **ShiftDepot** – A route is transferred from one depot to another
- **SwapDepot** – The depots of two different routes are exchanged
- **Exchange** – Two customers from the same route have their positions exchanged
- **Reinsertion** – A single customer is moved to a different position in the same route
- **Or-opt-2** – A block of two consecutive customers is moved to a different position in the same route
- **Or-opt-3** – A block of three consecutive customers is moved to a different position in the same route

As mentioned in Chapter 4, solution feasibility in terms of vehicle capacities may be assessed by sorting the demand deviations of a route in non-increasing order. To avoid doing this for every move evaluation, a data structure containing sorted demand deviations is maintained for each route. Whenever a move that swaps two arbitrarily sized customer subsets from different routes is evaluated, a simple algorithm is use to compute the new sum of demands in the worst-case for each one of the involved routes. By knowing the sorted demand deviations, the worst-case total demand can be computed in $\mathcal{O}(l^2)$ time, where l is the sum of the sizes of each subset. As for the time windows constraints, the DP approach mentioned in Chapter 4 is employed.

5.4 Perturbation

Two different perturbation mechanisms were used. The first one, called **MultipleShift**, applies a randomly chosen movement from the **Shift(1,0)** neighborhood. The second one, called **MultipleSwap**, executes a random movement from the **Swap(1,1)** neighborhood. Whenever the perturbation must be executed within the **ILS** procedure, exactly one of the two mentioned methods is randomly performed one to three times. In the specific case where the vehicle fleet is unlimited, a third perturbation mechanism, denoted **Split** may also be used. The **Split** procedure works by sequentially transferring customers from a route r to a new route associated with a vehicle picked at random. This process is repeated until r is empty. The route r is then removed, and the new generated routes are added to the solution.

5.5 Set partitioning

The **SetPartitioning** procedure attempts to construct better solutions by using a pool of routes obtained during local searches. The procedure is based on the well-known set partitioning formulation for VRPs. Each route r in the pool R is associated to a binary decision λ_r, which assumes value 1 iff said route is used in the solution. The formulation is expressed as follows.

$$\text{minimize} \sum_{r \in R} c_r \lambda_r \tag{5.4}$$

$$\text{subject to} \sum_{r \in R_i} \lambda_r = 1 \qquad i \in V' \tag{5.5}$$

$$\lambda_r \in \{0, 1\} \qquad r \in R. \tag{5.6}$$

The objective function (5.4) aims to minimize the total cost of the solution. Constraints (5.5) ensure each customer is visited by exactly one vehicle. Finally, constraints (5.6) define the domain of the decision variables.

In case the vehicle fleet is limited (i.e., $U_k < +\infty$ for some $k \in K$), an additional set of constraints must be added to the model to ensure feasibility, yielding the following formulation.

$$(5.4) \tag{5.7}$$

$$\text{subject to } (5.5), (5.6) \tag{5.8}$$

$$\sum_{r \in R, \text{type}(r)=k'} \lambda_r \leq U_{k'} \qquad k \in K \tag{5.9}$$

Constraints (5.9) ensure that the limit on the number of vehicles of each type is respected.

Algorithm 2 outlines the **SetPartitioning** procedure. Initialization involves generating a Mixed Integer Program (MIP) based on one of the set partitioning formulations, selected in accordance with the specific problem variant (lines 2–4). Whenever the MIP solver identifies an integer solution, the IncumbentCallback() procedure is invoked. Within this procedure, the ILS metaheuristic is called with the incumbent solution as a starting point. If ILS manages to improve the incumbent solution, the cutoff value of the MIP solver is updated (lines 8–12). For a depiction of the procedure, the reader is referred to Figure 1, which shows how ILS may be used to stop branching on the branch-and-bound tree created by the MIP solver.

Algorithm 2: Set Partitioning

```
1  procedure SetPartitioning (s*, PoolRotas, MaxSPTime)
2      SPModel ← GenerateMIP(PoolRotas)
3      MIPStart ← s*
4      Cutoff ← f(s*)
5      s* ← MIPSolver(SPModel, MIPStart, Cutoff, MaxSPTime,
         IncumbentCallback())
6      return s*
7  procedure IncumbentCallback(Incumbent Solution)
8      s ← Incumbent Solution
9      s' ← s
10     s ← ILS(I_ILS, s, s', NULL)
11     if f(s) < f(s*) then
12         s*, Cutoff ← s, f(s)
```

5.6 Acceleration Mechanisms

To avoid unnecessary computations, we use the auxiliary data structures proposed in (SUBRAMANIAN et al., 2012; PENNA et al., 2019). One such data structure is `NeighbStatus`, which informs, for a certain route and neighborhood, if the route has been modified after the neighborhood has failed to find an improvement move involving the

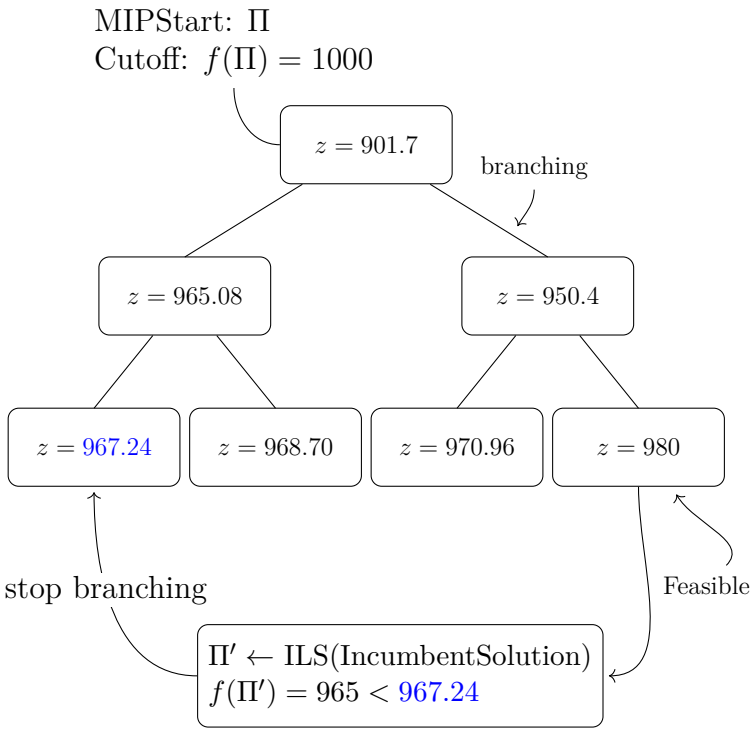

Figure 1 – Depiction of the **SetPartitioning** procedure

same route. The authors also propose several data structures to store useful information such as cumulative demands and the maximal demand in a route. When the current solution is feasible, these data structures preemptively provide indication on whether or not a move is potentially feasible.

In the specific case of the VRPTW, we also employ some of the acceleration mechanisms proposed by (MALHEIROS et al., 2021), which are described below.

- **Memoization Move Descriptor (MMD)** – An extension of the Static Move Descriptor (SMD) method (ZACHARIADIS; KIRANOUDIS, 2010; BEEK et al., 2018). During local search, we store a description of the best move for each route (or pair of routes). Before exploring a neighborhood structure on a certain route, we check whether it was previously attempted. If so, the best improving move is applied. Otherwise, the neighborhood is thoroughly explored and the best move — if any — is stored.

- **Memoization Solution Descriptor (MSD)** – Stores local optima visited throughout the algorithm. In case a previously visited local optimum is found, the current ILS iteration is aborted. This avoids exhaustively attempting to improve unpromising solutions to no avail.

6 Computational Results

The algorithm was coded in linguagem C++ (g++ 9.4.0) and executed in an Intel® Core™ i7-3770 CPU 3.40GHz with 16 GB of RAM running Ubuntu Linux 20.04. The set partitioning models were implemented with CPLEX 20.01. To demonstrate the different attributes that the method accomodates, we conducted computational experiments on instances originating from several VRP variants, which are listed below.

- **CVRP** – The vehicle fleet is homogeneous and only capacity constraints are present
- **VRPTW** – Similar to CVRP, but each customer also has a service time and a time window
- **FSMFD** – A heterogeneous, limited fleet of vehicles is available, and each vehicle type has a nonzero fixed cost and a nonzero variable cost
- **FSMD** – Similar to FSMFD, but all fixed costs are null
- **FSMF** – Similar to FSMFD, but all variable costs are null
- **HVRPD** – Similar to FSMD, but the fleet of vehicles is limited
- **HVRPFD** – Similar to FSMFD, but the fleet of vehicles is limited
- **MDVRP** – The vehicle fleet is homogeneous, and contrasting all other previous versions, multiple depots are also available

In the following sections, details on how instances were generated, and descriptions of parameter settings used for experiments are provided for each variant. Experimental results are also discussed. For tables containing extended results, the reader is referred to Appendix A.

6.1 CVRP

Experiments for the CVRP were conducted on the classical instance sets A,B,E,F,M and P, initially devised for the deterministic version. Instances for the robust version were generated as follows. First, the capacity of the vehicles is set to $Q = \lfloor Q_{min} \times (1 - \tau) + Q_{max} \times \tau \rfloor$, where τ was set to 0.3 by default, and the values Q_{min} and Q_{max} were taken from (PESSOA et al., 2021) for each instance. The demand budget was defined as $\Gamma^q = \rho \times \text{numberOfCustomers/numberOfVehicles}$, where ρ was set to 0.75. Finally, for each customer $i \in V_C$, their nominal demand $\bar{q}_i$ is equal to the corresponding instance

demand, whilst their demand deviation was calculated as $\hat{q}_i = \mu\bar{d}_i$, where $\mu = 0.3$. As in (PESSOA et al., 2021), other value configurations for τ, ρ and μ were also used to assess algorithm behavior on different uncertainty set parametrizations. Such configurations are shown in Table 2.

Table 2 – Different configurations for τ, ρ and μ

Config.	***Main***	***Low*** $\hat{q}$	***High*** $\hat{q}$	***Low*** Γ^q	***High*** Γ^q	***Low*** Q	***High*** Q
μ	0.3	0.1	0.5	0.3	0.3	0.3	0.3
ρ	0.75	0.75	0.75	0.5	1	0.75	0.75
τ	0.3	0.3	0.3	0.3	0.3	0.15	0.6

For the CVRP variant, the stopping criterion for Algorithm 1 is running exactly 50 iterations of the **ILS** procedure. The number of times the ILS metaheuristic may fail consecutively before stopping was set to numberOfCustomers + 5 × numberOfVehicles. The time limit for the set partitioning MIP was set to 60 seconds. The algorithm was executed 10 times for each instance, and results were compared to those of the exact approach proposed in (PESSOA et al., 2021). Table 3 shows summarized results for the benchmark set under the main configuration from Table 2. Note that not only the algorithm was able to finish within reasonable times on average, but average gaps are also relatively low. In fact, the algortihm manage to achieve the best known solution for all instances, and new upper bounds were found for two instances.

Table 3 – Aggregated results for the main configuration

Inst. type	**Avg. cost**	**Avg. gap (%)**	**Avg. time (s)**
A	1061.39	0.01	16.44
B	985.24	0.01	172.14
E	666.99	0.03	47.13
F	486.0	0.0	10.04
M	974.15	0.02	127.78
P	596.09	0.0	17.2

Average costs and running times for configurations other than the main one can be found in Table 4. Configurations *Low* Γ^q, *Low* $\hat{q}$ e *High* Q are less restrictive, and consequently, they all exhibit smaller costs on average, while also requiring less time to solve. The *Low* Q configuration, requires substantially more time than the already mentioned noes, since smaller vehicle capacities reduce the search space, making local search harder. A similar phenomenon may be observed in the *High* Γ^q configuration, since a higher demand budget also makes the instance more restrictive and reduces the search space. Feasibility checking on instances with a higher budget will also require more computational time for some neighborhoods (such as **Cross**), for instance. The *High* $\hat{q}$ configuration was the most challenging one since, in most cases, no feasible solutions could be found.

Table 4 – Aggregated results for different configurations

Inst.	Low Γ		High Γ		Low $\hat{d}$		High $\hat{d}$		Low C		High C	
	Avg.	**Time (s)**	**Avg.**	**Time (s)**	**Avg.**	**Time (s)**	**Avg.**	**Time (s)**	**Avg.**	**Time (s)**	**Avg.**	**Time (s)**
A	1025.08	15.47	1133.51	476.0	968.49	17.43	—	—	1080.99	22.25	1033.26	16.05
B	953.47	19.46	1017.88	401.88	871.95	15.65	—	—	1003.36	375.89	959.49	25.57
E	652.18	117.06	664.9	1632.97	626.34	35.86	500.5	1.79	641.67	49.05	657.42	39.89
F	483.0	9.05	486.5	10.61	469.0	8.67	—	—	493.0	12.58	476.0	9.19
M	933.5	112.63	1016.9	782.71	880.7	101.82	—	—	981.7	138.96	946.0	115.64
P	576.17	13.97	575.06	27.33	558.24	15.41	207.5	0.21	607.43	23.41	584.3	15.77

6.2 VRPTW

Experiments for the VRPTW were conducted on the Solomon benchmark set. Instances for the robust version were generated as in (MUNARI et al., 2019). Given a customer $i \in V_C$, its demand deviation was calculated as $\hat{q}_i = \lfloor \alpha^q \times \bar{q}_i \rfloor$. Similarly, for a given arc $a = (i, j)$, its associated traveling time deviation was defined as $\hat{t}_{ij} = 0.1 \times \lfloor 10 \times \alpha^t \times \bar{t}_{ij} \rfloor$. For each original instance from the benchmark set, multiple instances for the robust counterpart were generated by varying the demand and time budgets $\Gamma^q, \Gamma^t \in \{0, 1, 5, 10\}$ and the factors $\alpha^q, \alpha^t \in \{0, 0.1, 0.25, 0.5\}$.

The algorithm parameters used in the VRPTW variant were identical to the ones described in the previous section. The algorithm was executed 5 times on each instance, and performance was compared to the exact method proposed by (MUNARI et al., 2019). Results for instances with 100 customers can be found within Table 5, which aggregates average costs and times obtained for each budget configuration and instance type.

As expected, higher budget values negatively impact solution costs, as they make instances more restrictive. The same phenomenon is also observed in average times, specially in the case of the time budget Γ^t. The reason for this is probably that the asymptotic complexity of the DP algorithm used to check robust feasibility in terms of time is a function of $|V_C|$ and Γ^t. Note that the algorithm performed significantly better than the approach of (MUNARI et al., 2019) for R1 and R2 instances. This is probably due to the random (as opposed to clustered) distribution of customer locations, which can be a limiting factor for exact methods. Overall, our algorithm achieved the best known solution for approximately 80% of instances, and new upper bounds were found for 14% of instances.

6.3 HFVRP, FSM and MDVRP

The instances used on experiments with the HFVRP and FSM variants were originally proposed by (GOLDEN et al., 1984), while instances for the MDVRP were introduced by (CORDEAU; GENDREAU; LAPORTE, 1997). Adaptations to the robust versions were made exactly as in (SUBRAMANYAM; REPOUSSIS; GOUNARIS, 2018). Nominal customer demands were inherited from the original instances, and the demand

Table 5 – Results for Solomon instances with 100 customers

		C1			R1			RC1		
Γ^d	Γ^t	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Time (s)**
0	0	826.70	-0.001	3.9	1173.74	0.014	53.2	1334.51	0.002	49.5
1	0	924.33	0.041	8.2	1174.69	-0.002	57.1	1348.39	0.049	58.7
5	0	1042.57	-0.083	20.5	1180.97	-0.055	74.3	1386.50	-0.130	71.6
10	0	1056.39	-0.083	23.3	1186.65	-0.057	78.2	1402.85	0.068	81.2
0	1	846.73	-0.149	4.8	1154.17	-0.011	67.2	1400.48	0.074	50.6
0	5	862.05	-0.118	8.6	1195.33	0.008	91.1	1470.80	-0.126	62.7
0	10	862.40	-0.141	12.0	1203.42	0.013	129.1	1481.87	0.057	94.9
1	1	926.34	-0.076	11.2	1155.05	0.061	75.2	1412.70	0.067	61.4
5	5	1045.70	-0.108	28.2	1197.90	0.062	118.7	1492.02	-0.057	91.7
10	10	1059.81	-0.085	43.9	1206.99	-0.072	162.4	1506.26	0.001	115.7
		C2			R2			RC2		
Γ^d	Γ^t	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Tempo (s)**
0	0	587.38	0.000	12.1	873.35	0.085	44.5	1002.53	0.207	22.2
1	0	587.37	0.000	12.0	873.38	-1.369	43.8	1002.65	0.169	22.4
5	0	593.66	-0.126	12.5	873.32	-0.316	49.5	1002.48	0.153	26.7
10	0	613.40	-0.944	14.7	873.46	-1.334	55.0	1002.40	0.070	28.8
0	1	593.43	0.000	15.5	876.69	-1.110	53.8	1006.89	0.077	25.3
0	5	599.81	0.000	33.8	879.75	-1.647	87.8	1011.49	0.066	42.4
0	10	599.81	0.000	62.8	880.18	-1.814	127.5	1012.05	-0.042	62.3
1	1	593.43	0.000	15.2	876.41	-1.926	51.8	1006.80	0.086	27.5
5	5	604.78	-0.251	28.5	879.88	-2.067	90.2	1011.08	0.014	44.0
10	10	622.25	-1.459	44.7	880.49	-1.984	143.5	1012.05	-0.119	67.0

deviation of customer i is calculated as $\hat{q}_i = \alpha^q \times \bar{d}_i$, with α set to 0.1. The demand budget was defined as $\Gamma^q = \beta \times |V_C|$, where $\beta = 0.2$. To ensure feasibility in the worst-case scenario, vehicle capacities were increased by 10%.

To compare our method to the metaheuristic propopsed by (SUBRAMANYAM; REPOUSSIS; GOUNARIS, 2018), we executed the algortihm 10 times on each instance. The maximum number of times the ILS metaheuristic is allowed to fail was set to 300, and the same time limit of the AMP from (SUBRAMANYAM; REPOUSSIS; GOUNARIS, 2018) (3000 seconds) was used as the stopping criterion of Algorithm 1. Since (SUBRAMANYAM; REPOUSSIS; GOUNARIS, 2018) only reports the cost of the best solution from 10 runs, no assessment can be made about the quality of their solutions on average when compared to ours. Thus, we opted to report the gap of our best solutions with respect to theirs. The summarized results — which were aggregated on all instances — can be found in Table 6, while more detailed results may be found in A.

Table 6 – Summarized results for HFVRP, FSM and MDVRP

Variant	**Avg. cost**	**Avg. gap (%)**	**Improved BKSs**
FSMF	4046.96	-0.01	2
FSMD	989.22	0.08	1
FSMFD	4273.27	-0.09	3
HVRPFD	5125.84	0.07	1
HVRPD	1230.72	0.22	0
MDVRP	1005.9	0.33	1

As observed in Table 6, our algorithm achieved negative gaps on FSMF and FSMFD instances on average. One possible explanation is that our algorithm excels at finding optimal compositions of vehicle types when nonzero fixed costs are considered. However, since this phenomenon is not observed in the HVRPFD variant, it is possible that the presence of upper limits on fleet sizes is a limiting factor to our method. Additionally, it is noteworthy that our method found the best-known solution for each instance at least once, with HVRPD being the only variant for which no better upper bounds could be found.

7 Concluding Remarks

In this work, a class of heterogeneous fleet VRPs was studied within the framework of RO. We formally defined We formally defined a generalized problem that incorporates attributes such as time windows, heterogeneous fleets, and multiple depots. Cardinality-constrained uncertainty sets were employed to model time and demand uncertainty. We applied a matheuristic that combines Iterated Local Search (ILS) with a set partitioning formulation to solve the problem. While the method is not entirely novel in the context of deterministic VRPs, it is noteworthy that theoretical results related to RO had to be leveraged to make it viable when uncertainties are considered. Moreover, in most cases, our method achieved the best-known solutions, and improved upper bounds were obtained for instances of several robust VRP variants.

For future work, our intention is to incorporate other uncertainty sets, such as the knapsack uncertainty set, into the method. Additionally, we aim to generate instances that encompass all attributes outlined in the problem definition. An intersting line of research would be to investigate the impact of various instance characteristics, such as the number of depots or the limit on the number of vehicles, on the price of robustness.

Bibliography

AGRA, A. et al. Layered Formulation for the Robust Vehicle Routing Problem with Time Windows. In: HUTCHISON, D. et al. (Ed.). *Combinatorial Optimization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. v. 7422, p. 249–260. ISBN 978-3-642-32146-7 978-3-642-32147-4. Series Title: Lecture Notes in Computer Science. Disponível em: <http://link.springer.com/10.1007/978-3-642-32147-4_23>.

AGRA, A. et al. The robust vehicle routing problem with time windows. *Computers & Operations Research*, v. 40, n. 3, p. 856–866, mar. 2013. ISSN 03050548. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0305054812002134>.

Aharon Ben-Tal; Laurent El Ghaoui; Arkadi Nemirovski. *Robust Optimization*. [s.n.], 2009. (Princeton series in applied mathematics). ISBN 978-0-691-14368-2. Disponível em: <https://press.princeton.edu/books/hardcover/9780691143682/robust-optimization>.

BARTOLINI, E. et al. The Robust Traveling Salesman Problem with Time Windows Under Knapsack-Constrained Travel Time Uncertainty. *Transportation Science*, v. 55, n. 2, p. 371–394, mar. 2021. ISSN 0041-1655, 1526-5447. Disponível em: <http://pubsonline.informs.org/doi/10.1287/trsc.2020.1011>.

BEEK, O. et al. An Efficient Implementation of a Static Move Descriptor-based Local Search Heuristic. *Computers & Operations Research*, v. 94, p. 1–10, jun. 2018. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054818300078>.

BERTSIMAS, D.; SIM, M. Robust discrete optimization and network flows. *Mathematical Programming*, v. 98, n. 1, p. 49–71, set. 2003. ISSN 1436-4646. Disponível em: <https://doi.org/10.1007/s10107-003-0396-4>.

BERTSIMAS, D.; SIM, M. The Price of Robustness. *Operations Research*, v. 52, n. 1, p. 35–53, fev. 2004. ISSN 0030-364X. Publisher: INFORMS. Disponível em: <https://pubsonline.informs.org/doi/10.1287/opre.1030.0065>.

CAMPOS, R. A. de. *Aircraft routing under uncertainty via robust optimization*. Tese (Doutorado) — Universidade Federal de São Carlos, 2022.

CHRISTIAENS, J.; BERGHE, G. V. Slack Induction by String Removals for Vehicle Routing Problems. *Transportation Science*, v. 54, n. 2, p. 417–433, mar. 2020. ISSN 0041-1655. Publisher: INFORMS. Disponível em: <https://pubsonline.informs.org/doi/abs/10.1287/trsc.2019.0914>.

CORDEAU, J.-F.; GENDREAU, M.; LAPORTE, G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, v. 30, n. 2, p. 105–119, 1997. ISSN 1097-0037. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291097-0037%28199709%2930%3A2%3C105%3A%3AAID-NET5%3E3.0.CO%3B2-G. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0037%28199709%2930%3A2%3C105%3A%3AAID-NET5%3E3.0.CO%3B2-G>.

DANTZIG, G. B.; RAMSER, J. H. The Truck Dispatching Problem. *Management Science*, v. 6, n. 1, p. 80–91, out. 1959. ISSN 0025-1909. Publisher: INFORMS. Disponível em: <https://pubsonline.informs.org/doi/10.1287/mnsc.6.1.80>.

GAREY, M. R.; JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. First edition. New York u.a: W. H. Freeman, 1979. ISBN 978-0-7167-1045-5.

GENDREAU, M.; JABALI, O.; REI, W. Chapter 8: Stochastic Vehicle Routing Problems. In: *Vehicle Routing*. Society for Industrial and Applied Mathematics, 2014, (MOS-SIAM Series on Optimization). p. 213–239. ISBN 978-1-61197-358-7. Disponível em: <https://epubs.siam.org/doi/10.1137/1.9781611973594.ch8>.

GOLDEN, B. et al. The fleet size and mix vehicle routing problem. *Computers & Operations Research*, v. 11, n. 1, p. 49–66, jan. 1984. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/0305054884900078>.

GOUNARIS, C. E. et al. An Adaptive Memory Programming Framework for the Robust Capacitated Vehicle Routing Problem. *Transportation Science*, v. 50, n. 4, p. 1239–1260, nov. 2016. ISSN 0041-1655, 1526-5447. Disponível em: <http://pubsonline.informs.org/doi/10.1287/trsc.2014.0559>.

GOUNARIS, C. E.; WIESEMANN, W.; FLOUDAS, C. A. The Robust Capacitated Vehicle Routing Problem Under Demand Uncertainty. *Operations Research*, v. 61, n. 3, p. 677–693, jun. 2013. ISSN 0030-364X, 1526-5463. Disponível em: <https://pubsonline.informs.org/doi/10.1287/opre.1120.1136>.

GUTIN, G. et al. (Ed.). *The Traveling Salesman Problem and Its Variations*. Boston, MA: Springer US, 2007. v. 12. (Combinatorial Optimization, v. 12). ISBN 978-0-387-44459-8 978-0-306-48213-7. Disponível em: <http://link.springer.com/10.1007/b101971>.

HASLE, G.; LIE, K.-A.; QUAK, E. (Ed.). *Geometric Modelling, Numerical Simulation, and Optimization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. ISBN 978-3-540-68782-5. Disponível em: <http://link.springer.com/10.1007/978-3-540-68783-2>.

HU, C. et al. Robust vehicle routing problem with hard time windows under demand and travel time uncertainty. *Computers & Operations Research*, v. 94, p. 139–153, jun. 2018. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054818300376>.

JAILLET, P.; QI, J.; SIM, M. Routing Optimization Under Uncertainty. *Operations Research*, v. 64, n. 1, p. 186–200, fev. 2016. ISSN 0030-364X. Publisher: INFORMS. Disponível em: <https://pubsonline.informs.org/doi/10.1287/opre.2015.1462>.

LEE, C.; LEE, K.; PARK, S. Robust vehicle routing problem with deadlines and travel time/demand uncertainty. *Journal of the Operational Research Society*, v. 63, n. 9, p. 1294–1306, set. 2012. ISSN 0160-5682, 1476-9360. Disponível em: <https://www.tandfonline.com/doi/full/10.1057/jors.2011.136>.

LOURENçO, H. R.; MARTIN, O. C.; STüTZLE, T. Iterated Local Search: Framework and Applications. In: GENDREAU, M.; POTVIN, J.-Y. (Ed.). *Handbook of Metaheuristics*. Cham: Springer International Publishing, 2019, (International Series in Operations

Research & Management Science). p. 129–168. ISBN 978-3-319-91086-4. Disponível em: <https://doi.org/10.1007/978-3-319-91086-4_5>.

LU, D.; GZARA, F. The robust vehicle routing problem with time windows: Solution by branch and price and cut. *European Journal of Operational Research*, v. 275, n. 3, p. 925–938, jun. 2019. ISSN 03772217. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0377221718310713>.

MALHEIROS, I. et al. A hybrid algorithm for the multi-depot heterogeneous dial-a-ride problem. *Computers & Operations Research*, v. 129, p. 105196, maio 2021. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054820303130>.

MUNARI, P. et al. The Robust Vehicle Routing Problem with Time Windows: Compact Formulation and Branch-Price-and-Cut Method. *Transportation Science*, v. 53, n. 4, p. 1043–1066, jul. 2019. ISSN 0041-1655, 1526-5447. Disponível em: <http://pubsonline.informs.org/doi/10.1287/trsc.2018.0886>.

MáXIMO, V. R.; CORDEAU, J.-F.; NASCIMENTO, M. C. An adaptive iterated local search heuristic for the Heterogeneous Fleet Vehicle Routing Problem. *Computers & Operations Research*, v. 148, p. 105954, dez. 2022. ISSN 03050548. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0305054822002052>.

ORDóñEZ, F. Robust Vehicle Routing. In: *Risk and Optimization in an Uncertain World*. INFORMS, 2010, (INFORMS TutORials in Operations Research). p. 153–178. ISBN 978-0-9843378-0-4. Section: 7. Disponível em: <https://pubsonline.informs.org/doi/10.1287/educ.1100.0078>.

PENNA, P. H. V.; SUBRAMANIAN, A.; OCHI, L. S. An Iterated Local Search heuristic for the Heterogeneous Fleet Vehicle Routing Problem. *Journal of Heuristics*, v. 19, n. 2, p. 201–232, abr. 2013. ISSN 1381-1231, 1572-9397. Disponível em: <http://link.springer.com/10.1007/s10732-011-9186-y>.

PENNA, P. H. V. et al. A hybrid heuristic for a broad class of vehicle routing problems with heterogeneous fleet. *Annals of Operations Research*, v. 273, n. 1, p. 5–74, fev. 2019. ISSN 1572-9338. Disponível em: <https://doi.org/10.1007/s10479-017-2642-9>.

PESSOA, A. et al. A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, v. 183, n. 1-2, p. 483–523, set. 2020. ISSN 0025-5610, 1436-4646. Disponível em: <https://link.springer.com/10.1007/s10107-020-01523-z>.

PESSOA, A. A. et al. Branch-Cut-and-Price for the Robust Capacitated Vehicle Routing Problem with Knapsack Uncertainty. *Operations Research*, v. 69, n. 3, p. 739–754, maio 2021. ISSN 0030-364X, 1526-5463. Disponível em: <http://pubsonline.informs.org/doi/10.1287/opre.2020.2035>.

SOLANO-CHARRIS, E.; PRINS, C.; SANTOS, A. C. Local search based metaheuristics for the robust vehicle routing problem with discrete scenarios. *Applied Soft Computing*, v. 32, p. 518–531, jul. 2015. ISSN 1568-4946. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568494615002380>.

SOLANO-CHARRIS, E. L.; PRINS, C.; SANTOS, A. C. Solving the bi-objective Robust Vehicle Routing Problem with uncertain costs and demands. *RAIRO - Operations Research*, v. 50, n. 4-5, p. 689–714, out. 2016. ISSN 0399-0559, 1290-3868. Disponível em: <http://www.rairo-ro.org/10.1051/ro/2016048>.

SUBRAMANIAN, A. et al. A parallel heuristic for the Vehicle Routing Problem with Simultaneous Pickup and Delivery. *Computers & Operations Research*, v. 37, n. 11, p. 1899–1911, nov. 2010. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054809002779>.

SUBRAMANIAN, A. et al. A hybrid algorithm for the Heterogeneous Fleet Vehicle Routing Problem. *European Journal of Operational Research*, v. 221, n. 2, p. 285–295, set. 2012. ISSN 03772217. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0377221712002093>.

SUBRAMANIAN, A.; UCHOA, E.; OCHI, L. S. A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, v. 40, n. 10, p. 2519–2531, out. 2013. ISSN 03050548. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S030505481300021X>.

SUBRAMANYAM, A.; REPOUSSIS, P. P.; GOUNARIS, C. E. *Robust optimization of a broad class of heterogeneous vehicle routing problems under demand uncertainty*. arXiv, 2018. ArXiv:1810.04348 [cs, math]. Disponível em: <http://arxiv.org/abs/1810.04348>.

SUNGUR, I.; ORDóñEZ, F.; DESSOUKY, M. A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions*, v. 40, n. 5, p. 509–523, mar. 2008. ISSN 0740-817X. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/07408170701745378. Disponível em: <https://doi.org/10.1080/07408170701745378>.

VEGA, J. D. L.; MUNARI, P.; MORABITO, R. Robust optimization for the vehicle routing problem with multiple deliverymen. *Central European Journal of Operations Research*, v. 27, n. 4, p. 905–936, dez. 2019. ISSN 1613-9178. Disponível em: <https://doi.org/10.1007/s10100-017-0511-x>.

VEGA, J. D. L.; MUNARI, P.; MORABITO, R. Exact approaches to the robust vehicle routing problem with time windows and multiple deliverymen. *Computers & Operations Research*, v. 124, p. 105062, dez. 2020. ISSN 03050548. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0305054820301799>.

VIDAL, T. et al. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research*, v. 40, n. 1, p. 475–489, jan. 2013. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054812001645>.

WANG, A.; SUBRAMANYAM, A.; GOUNARIS, C. E. Robust vehicle routing under uncertainty via branch-price-and-cut. *Optimization and Engineering*, v. 23, n. 4, p. 1895–1948, dez. 2022. ISSN 1573-2924. Disponível em: <https://doi.org/10.1007/s11081-021-09680-6>.

YU, V. F.; ANH, P. T.; BALDACCI, R. A robust optimization approach for the vehicle routing problem with cross-docking under demand uncertainty. *Transportation*

Research Part E: Logistics and Transportation Review, v. 173, p. 103106, maio 2023. ISSN 1366-5545. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1366554523000947>.

ZACHARIADIS, E. E.; KIRANOUDIS, C. T. A strategy for reducing the computational complexity of local search-based methods for the vehicle routing problem. *Computers & Operations Research*, v. 37, n. 12, p. 2089–2105, dez. 2010. ISSN 0305-0548. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0305054810000535>.

APPENDIX A – Extended Results

Detailed results for variants shown in Chapter 6 can be found within the following tables.

Table 7 – Results for the robust CVRP under the main parameter configuration

Inst.	**Best**	**Avg.**	***Gap*** **(%)**	**Time (s)**	**Inst.**	**Best**	**Avg.**	***Gap*** **(%)**	**Time (s)**
A-n32-k5	857	857.0	0.0	2.42	B-n64-k9	865	865.2	-0.09	29.4
A-n33-k5	675	675.0	0.0	2.14	B-n66-k9	1319	1319.0	0.0	34.44
A-n33-k6	758	758.0	0.0	2.84	B-n67-k10	1086	1086.0	0.0	24.68
A-n34-k5	776	776.0	0.0	2.18	B-n68-k9	1298	1298.0	0.0	41.52
A-n36-k5	823	823.0	0.0	3.51	B-n78-k10	1261	1266.6	0.444	3632.43
A-n37-k5	706	706.0	0.0	2.81	E-n101-k14	1109	1111	0.18	227.99
A-n37-k6	948	948.0	0.0	4.93	E-n101-k8	826	826.3	0.036	105.68
A-n38-k5	714	714.0	0.0	3.33	E-n13-k4	277	277.0	0.0	0.15
A-n39-k5	818	818.0	0.0	4.09	E-n22-k4	373	373.0	0.0	0.49
A-n39-k6	850	850.0	0.0	3.64	E-n23-k3	570	570.0	0.0	0.38
A-n44-k6	930	930.0	0.0	6.62	E-n30-k3	495	495.0	0.0	0.95
A-n45-k6	918	918.0	0.0	6.24	E-n31-k7	379	379.0	0.0	2.64
A-n45-k7	1163	1163.0	0.0	9.0	E-n33-k4	836	836.0	0.0	1.7
A-n46-k7	988	988.0	0.0	7.71	E-n51-k5	519	519.0	0.0	8.15
A-n48-k7	1129	1129.0	0.0	9.52	E-n76-k10	830	830.2	0.024	85.04
A-n53-k7	1019	1019.0	0.0	13.72	E-n76-k14	1020	1021.4	0.137	90.63
A-n54-k7	1169	1169.0	0.0	15.45	E-n76-k7	697	697.0	0.0	41.91
A-n55-k9	1107	1107.0	0.0	16.84	E-n76-k8	736	736.0	0.0	46.92
A-n60-k9	1408	1408.0	0.0	29.32	F-n45-k4	736	736.0	0.0	4.77
A-n61-k9	1022	1022.0	0.0	22.44	F-n72-k4	236	236.0	0.0	15.31
A-n62-k8	1339	1339.1	0.007	37.43	M-n101-k10	918	918.3	0.033	84.36
A-n63-k10	1348	1348.0	0.0	30.61	M-n121-k7	1030	1030.0	0.0	171.21
A-n63-k9	1618	1619.2	0.074	39.96	P-n101-k4	681	681.0	0.0	59.98
A-n64-k9	1414	1415.8	0.127	27.1	P-n19-k2	195	195.0	0.0	0.15
A-n65-k9	1184	1184.0	0.0	26.13	P-n20-k2	208	208.0	0.0	0.19
A-n69-k9	1177	1177.0	0.0	34.78	P-n21-k2	208	208.0	0.0	0.23
A-n80-k10	1795	1796.5	0.084	79.17	P-n22-k2	213	213.0	0.0	0.24
B-n31-k5	694	694.0	0.0	1.89	P-n22-k8	601	601.0	0.0	0.99
B-n34-k5	789	789.0	0.0	3.07	P-n23-k8	527	527.0	0.0	1.21
B-n35-k5	986	986.0	0.0	2.69	P-n40-k5	468	468.0	0.0	3.62
B-n38-k6	823	823.0	0.0	3.47	P-n45-k5	512	512.0	0.0	5.83
B-n39-k5	561	561.0	0.0	3.12	P-n50-k10	695	695.0	0.0	13.28
B-n41-k6	838	838.1	0.012	5.14	P-n50-k7	563	563.0	0.0	10.43
B-n43-k6	779	779.0	0.0	8.03	P-n50-k8	614	614.0	0.0	13.87
B-n44-k7	943	943.0	0.0	9.25	P-n51-k10	736	736.0	0.0	13.85
B-n45-k5	739	739.0	0.0	7.82	P-n55-k10	718	718.0	0.0	20.57
B-n45-k6	668	668.0	0.0	4.98	P-n55-k15	945	945.0	0.0	23.63
B-n50-k7	758	758.0	0.0	7.64	P-n55-k7	583	583.0	0.0	14.33
B-n50-k8	1330	1330.1	0.008	14.55	P-n55-k8	624	624.0	0.0	15.06
B-n51-k7	1027	1027.0	0.0	12.89	P-n60-k10	755	755.0	0.0	24.33
B-n52-k7	775	775.0	0.0	7.78	P-n60-k15	1020	1020.0	0.0	31.97
B-n56-k7	740	740.0	0.0	19.33	P-n65-k10	809	809.0	0.0	38.23
B-n57-k7	1132	1132.0	-0.088	15.19	P-n70-k10	824	824.0	0.0	46.54
B-n57-k9	1656	1656.0	0.0	25.09	P-n76-k4	590	590.0	0.0	26.29
B-n63-k10	1587	1587.6	0.038	44.86	P-n76-k5	621	621.0	0.0	30.74

Table 8 – Results for Solomon instances with 25 customers

		C1			R1			RC1		
Γ^d	Γ^t	**Custo**	**Gap (%)**	**Tempo (s)**	**Custo**	**Gap (%)**	**Tempo (s)**	**Custo**	**Gap (%)**	**Tempo (s)**
0	0	190.59	0.000	0.2	463.37	0.000	0.2	350.24	0.000	0.2
1	0	202.68	0.000	0.2	463.42	0.011	0.2	360.91	0.222	0.2
5	0	238.50	0.000	0.2	463.42	0.011	0.2	434.39	0.547	0.3
10	0	239.08	0.018	0.2	463.42	0.011	0.2	439.54	0.399	0.2
0	1	192.34	0.000	0.2	1058.55	0.001	0.2	728.99	0.207	0.3
0	5	195.83	0.000	0.3	1068.76	0.030	0.3	755.64	0.000	0.4
0	10	195.83	0.000	0.4	1068.99	0.024	0.3	764.80	0.015	0.5
1	1	202.87	0.004	0.2	1058.55	0.001	0.2	739.60	0.047	0.3
5	5	238.95	0.011	0.3	1068.70	0.017	0.3	815.18	0.188	0.4
10	10	239.73	0.025	0.5	1068.88	0.000	0.3	818.32	0.696	0.5
		C2			R2			RC2		
Γ^d	Γ^t	**Custo**	**Gap (%)**	**Tempo (s)**	**Custo**	**Gap (%)**	**Tempo (s)**	**Custo**	**Gap (%)**	**Tempo (s)**
0	0	214.52	0.033	0.2	382.27	0.031	0.1	319.27	0.000	0.1
1	0	214.51	0.027	0.2	382.15	0.002	0.1	319.28	0.000	0.1
5	0	214.47	0.011	0.2	382.15	0.002	0.1	319.28	0.000	0.1
10	0	214.45	0.000	0.2	382.15	0.000	0.1	319.28	0.000	0.1
0	1	214.52	0.005	0.2	383.86	0.000	0.1	319.55	0.000	0.2
0	5	214.57	0.000	0.3	384.85	0.046	0.2	319.72	0.000	0.2
0	10	214.58	0.005	0.4	384.96	0.019	0.2	319.88	0.000	0.3
1	1	214.54	0.016	0.2	383.87	0.004	0.2	319.55	0.000	0.1
5	5	214.58	0.005	0.3	384.82	0.005	0.2	319.72	0.000	0.2
10	10	214.59	0.011	0.4	384.95	0.017	0.3	319.88	0.000	0.3

Table 9 – Results for Solomon instances with 50 customers

		C1			R1			RC1		
Γ^d	Γ^t	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Time (s)**
0	0	361.69	0.000	0.9	766.25	0.019	1.6	730.83	0.055	1.2
1	0	407.05	0.121	1.2	766.16	0.003	1.6	809.55	-0.112	1.4
5	0	454.87	0.155	1.2	766.97	0.022	1.9	845.61	0.240	1.6
10	0	459.43	0.051	1.5	767.93	0.034	1.7	852.40	0.419	1.5
0	1	365.08	0.000	1.1	2491.83	0.018	1.8	1161.56	0.139	1.5
0	5	370.30	0.000	1.7	2514.28	0.039	2.7	1221.81	0.049	2.0
0	10	370.30	0.000	2.5	2516.89	0.020	3.5	1227.07	0.094	3.0
1	1	407.93	0.162	1.5	2491.92	0.020	2.0	1215.91	-0.069	1.5
5	5	455.11	0.136	2.2	2514.75	0.009	3.1	1267.92	0.255	2.4
10	10	460.16	0.062	3.4	2517.27	0.060	3.8	1270.90	0.126	3.3
		C2			R2			RC2		
Γ^d	Γ^t	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Time (s)**	**Cost**	**Gap (%)**	**Time (s)**
0	0	357.50	0.000	0.8	615.73	0.036	1.7	573.34	0.348	0.7
1	0	357.50	0.000	0.8	615.63	0.020	1.8	573.33	0.346	0.7
5	0	360.69	0.000	0.9	616.45	-0.005	2.0	573.66	0.406	0.7
10	0	367.92	-0.038	0.9	615.68	0.022	2.0	573.37	0.353	0.7
0	1	357.50	0.000	1.1	619.66	0.005	2.2	577.09	0.179	0.7
0	5	357.50	0.000	2.0	620.80	0.038	3.1	577.74	0.161	1.1
0	10	357.50	0.000	3.1	620.90	-0.072	4.3	578.44	0.297	1.5
1	1	357.50	0.000	1.1	619.62	-0.001	2.1	577.19	0.197	0.7
5	5	360.69	-0.018	2.0	620.76	0.034	3.2	578.14	0.244	1.1
10	10	368.02	-0.002	3.0	620.95	0.023	4.4	578.43	0.295	1.6

Table 10 – Results for the FSMF, FSMD and FSMFD variants

	FSMF			FSMD			FSMFD		
Instance	**Avg.**	**Best**	**Gap (%)**	**Avg.**	**Best**	**Gap (%)**	**Avg.**	**Best**	**Gap (%)**
3	951.61	951.61	0	623.22	623.22	0	1144.22	1144.22	0
4	6437.33	6437.33	0	380.71	380.71	0	6437.33	6437.33	0
5	988.63	988.63	0	742.85	742.85	0	1312.65	**1274.22**	-3.6
6	6516.47	6516.47	0	406.19	406.19	0	6516.47	6516.47	0
13	2408.77	2406.36	0	1491.86	1491.86	0	2964.64	2964.64	0
14	9119.03	9119.03	0	603.21	603.21	0	9126.90	9126.90	0
15	2586.37	2586.37	0	999.82	999.82	0	2634.96	2634.96	0
16	2724.92	2720.43	0	1131.00	1131.00	0	3168.91	3168.91	0
17	1746.94	1746.92	0.72	1040.93	1038.60	0	2005.07	2004.48	0
18	2372.22	2372.85	0.13	1805.61	1801.40	0.03	3149.30	**3148.99**	-0.1
19	8664.53	**8661.81**	-0.012	1107.88	1105.44	0	8664.60	8662.90	0
20	4046.66	**4040.52**	-0.497	1537.31	1533.71	0.03	4154.14	**4153.84**	-0.35

Table 11 – Results for the HVRPD and HVRPFD variants

	HVRPD		HVRPFD	
Inst.	**AMP**	**ILSSP**	**AMP**	**ILSSP**
13	1517,84	1517,84	3,185.09	3,185.09
14	606,67	606,67	10,106.67	10,106.67
15	1015,29	1015,29	3,065.29	3,065.29
16	1144,94	1144,94	3,265.41	3,265.41
17	1061,96	1061,96	2,076.96	2,076.96
18	1823,58	1823,58	3,745.10	**3743.58**
19	1120,34	1120,34	10,420.34	10,420.34
20	1534,17	1534,17	4,834.17	4,834.17

Table 12 – Results for the MDVRP variant

	MDVRP	
Inst.	**HAMP**	**ILSSP**
1	576.87	576.87
2	473.22	473.22
3	641.19	641.19
4	999.21	999.21
5	750.03	750.03
6	877.26	**876.50**
7	881.97	881.97
12	1,318.95	1,318.95
15	2,505.42	2,505.42