UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

PAULO VITOR SILVESTRIN

An efficient heuristic for the Multi-compartment Vehicle Routing Problem

Thesis presented in partial fulfillment
of the requirements for the degree of
Master of Computer Science

Advisor: Prof. Dr. Marcus Rolf Peter Ritt

Porto Alegre
October 2016

CIP — CATALOGING-IN-PUBLICATION

Silvestrin, Paulo Vitor

An efficient heuristic for the Multi-compartment Vehicle Routing Problem / Paulo Vitor Silvestrin. – Porto Alegre: PPGC da UFRGS, 2016.

71 f.: il.

Thesis (Master) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2016. Advisor: Marcus Rolf Peter Ritt.

1. Vehicle routing problem. 2. Multi-compartment. 3. Algorithm. 4. Heuristic. 5. Tabu Search. I. Ritt, Marcus Rolf Peter. II. Título.

ABSTRACT

We study a variant of the vehicle routing problem that allows vehicles with multiple compartments. The need for multiple compartments frequently arises in practical applications when there are several products of different quality or type, that must be kept or handled separately. The resulting problem is called the multi-compartment vehicle routing problem (MCVRP). We propose a tabu search heuristic and embed it into an iterated local search to solve the MCVRP. In several experiments we analyze the performance of the iterated tabu search and compare it with results from the literature. We find that it consistently produces solutions that are better than existing heuristic algorithms.

Keywords: Vehicle routing problem. Multi-compartment. Algorithm. Heuristic. Tabu Search.

Uma heurística eficiente para o problema de roteamento de veículos com múltiplos compartimentos

RESUMO

Este trabalho apresenta uma variação do problema de roteamento de veículos que permite o uso de veículos com múltiplos compartimentos. A necessidade de veículos com múltiplos compartimentos surge com frequência em aplicações práticas quando uma série de produtos, que possuem diferentes qualidades ou tipo, precisam ser transportados mas não podem ser misturados. Este problema é chamado na literatura de roteamento de veículos com múltiplos compartimentos (PRVMC). Nós propomos uma heurística busca tabu implementada em uma busca local iterada para resolver este problema. Experimentos foram feitos para avaliar a performance da busca tabu iterada e os resultados obtidos foram comparados com os resultados disponíveis na literatura. O algoritimo proposto é capaz de encontrar soluções melhores e em menos tempo de processamento que as heurísticas existentes.

Palavras-chave: Roteamento de veículos, Múltiplos compartimentos, Heurística, Busca tabu.

LIST OF ABBREVIATIONS AND ACRONYMS

CVRP Capacitated Vehicle Routring Problem

ILS Iterated Local Search

ITS Iterated Tabu Search

MCVRP Multi-Compartment Vehicle Routing Problem

TS Tabu Search

VRP Vehicle Routing Problem

VRPTW Vehicle Routing Problem with Time Windows

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

Operations research (OR) is a discipline that aims to find optimal or near-optimal solutions to complex decision-making problems with a focus on practical applications. For this reason, operations research deals with demands that come directly from problems that arise in the industry. Transportation related problems are among the main subjects of research due to their great impact on costs and efficiency in a variety of industries. The fundamental problem studied in OR that deals with transportation is the vehicle routing problem (VRP).

The VRP consists in finding the optimal set of routes where there is a set of customers with a certain demand that must be attended by a visit of a vehicle that departs from a depot. The VRP is a difficult combinatorial optimization problem where the first algorithm proposed dates from the end of 1950's and we still have real-world instances that the state-of-the-art algorithms cannot solve.

For the different kinds of industry the transportation problems may differ in terms of vehicle characteristics, special handling of the product and specific requirements from customers. This work focuses on the case where the customer demand for different types of products that must be kept separated for some particular reason. Thus, the vehicle capacity is divided in multiple compartments to be able to attend the demands for more than one product in a single visit. This problem is called in the literature the multi-compartment vehicle routing problem (MCVRP). In this work we also consider a variant of this problem where the customer demand for all product types must be attended in one single visit, called the MCVRP without splitting (MCVRP-WS).

1.1 Research Objectives and Contributions

This work focuses on researching the state-of-the-art and proposing new heuristics for the MCVRP and the MCVRP-WS. It also involves studing the best heuristic techniques available for the VRP.

The major contribution of this work is the proposal of a new heuristic for the MCVRP which is an adaptation of a successful algorithm for the VRP. Our algorithm performs better in terms of accuracy and speed compared to the available ones.

1.2 Overview of the Dissertation

This dissertation is organized as follows. In the next chapter we present a study of heuristics for the VRP. Chapter 3 is a study of the MCVRP and the MCVRP-WS concerning real-world applications, mathematical formulation and the state-of-the-art algorithms. In Chapter 4 we propose an efficient tabu search for the MCVRP-WS. In Chapter 5 we present an iterated tabu search for the MCVRP that outperforms the state-of-the-art algorithms. We conclude and discuss future work in Chapter 6.

2 THE VEHICLE ROUTING PROBLEM

The vehicle routing problem (VRP) consists in, given a set of customers with demands and a depot with a limited amount of vehicles, finding the shortest path routes that attend all the customers demands. This problem was proposed in the end of the 1950s by (DANTZIG; RAMSER, 1959) and since then it has been intensively studied by the scientific community due to its strong practical importance in many application fields, as well as interest as a difficult combinatorial optimization problem.

The VRP generalizes the $\mathcal{NP}$-hard traveling salesman problem (TSP), thus is $\mathcal{NP}$-hard as well, which means that no polynomial algorithm can solve it unless $\mathcal{P} = \mathcal{NP}$. Although a lot of scientific effort have been applied and great advances were reached, state-of-the-art exact algorithms barely solve VRP instances with more than 150 customers and their computational time is often not viable for practical proposes (SUBRAMANIAN; UCHOA; OCHI, 2013). Thus, most of the last 10 years of studies for the VRP rely on heuristic algorithms to find good solutions for real-world problems in a small amount of time (LAPORTE; ROPKE; VIDAL, 2014). State-of-the-art heuristics can find solutions of instances with 200 up to 400 customers with an average gap from the optimal solution of less than 1% in less than 10 minutes (VIDAL et al., 2013).

Practical applications of the VRP are cases where a set of places must be visited and the necessary routes are a decision variable. The decision for the optimal routes can consider minimal use of fuel, least amount of vehicles possible, save of resources as time and money or the routes that result in less environmental impact. This kind of problem arises in different industries such as transportation, logistics, communication, manufacturing military, and so on. To attend the necessities of different industries the VRP was extended adding some constraints and attributes, which results in variants of the VRP such as (a complete survey about the VRP variants can be found in (VIDAL et al., 2013)):

- *Capacitated VRP (CVRP)*: this is the classical variation and the most studied one. Here the a set of customers have demands that must be attended in one single visit by a fleet of identical vehicles with a fixed capacity. All vehicle routes start and end in one depot.
- *VRP with Time Windows (VRPTW)*: The customer demand must be attended within a defined time interval.
- *VRP with Pickup and Delivery (VRPPD)*: A number of goods must be moved from

pickup locations to delivery locations.

- *Heterogeneous VRP (HVRP)*: In this variant the fleet of vehicles is heterogeneous, so the vehicles may have different capacities from each other.
- *Periodic VRP (PVRP)*: In this case customers have repetitive demands over multiple days. Given a planning period of D days one must determine which customers will be visited and the shortest routes for each day.
- *Multi-depot VRP (MDVRP)*: The vehicles can depart from more than one depot. In the main definition the vehicle destination must be at the origin depot, but in variants called *non-fixed* problems the route can finish in any depot.
- *Multi-compartment VRP (MCVRP)*: in this variation the vehicles capacity is divided in more than one compartment and the customers have demands for different types of products where each product type has a dedicated compartment.

Each of these VRP variants are challenging fields of research. In this chapter we will focus on the CVRP that is the most studied variant of the VRP.

In the next section we present the formal definition of the CVRP. In Section 2.2 we describe the most used sets of instances. Section 2.3 is an overview of the most used metaheuristics and a brief description of successful work using them. We conclude this chapter in Section 2.4.

2.1 Problem Definition

The CVRP consists in a fleet of identical vehicles with fixed loading capacity C and customers that have a defined amount of demand c_i. We are given a set of locations $V = \{V_0\} \cup V_+$, where V_0 is the *depot*, and $V_+ = \{V_1, \ldots, V_n\}$ is the set of *customers*. Each pair of locations $i, j \in V$ has a travel time d_{ij}. Each customer may have additionally a drop time t_i, i.e. the time needed to load or unload the demand. All the customer demand must be attended in one visit.

The objective is to find the shortest possible set of routes that satisfy all customers demands and respect the vehicle capacity. The CVRP can be formulated as follows:

$$\textbf{minimize} \quad \sum_{i,j \in V} \sum_{k \in [r]} (d_{ij} + t_j) x_{ijk}, \tag{2.1}$$

$$\textbf{subject to} \quad \sum_{i \in V} \sum_{k \in [r]} x_{ijk} = 1, \qquad \forall j \in V \setminus \{V_0\}, \tag{2.2}$$

$$\sum_{i \in V} x_{ijk} = \sum_{i \in V} x_{jik}, \qquad \forall j \in V, k \in [r], \tag{2.3}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1, \qquad \forall S \subseteq V \setminus \{V_0\}, |S| \geq 2, k \in [r], \tag{2.4}$$

$$\sum_{i,j \in V} c_j x_{ijk} \leq C, \qquad \forall k \in [r], \tag{2.5}$$

$$x_{ijk} \in \{0, 1\}, \qquad \forall i, j \in V, k \in [r]. \tag{2.6}$$

In this formulation we minimize the total travel time (2.1). By constraint (2.2) every customer has to be attended exactly once in some route. Constraint (2.3) establishes flow conservation, and constraint (2.4) eliminates subroutes that do not include the depot. The capacity are guaranteed by (2.5).

2.2 Instances

An instance for the CVRP contains a set of customers and a depot that is the starting point of each route. Every customer, including the depot, have a well-defined distance from each other. Usually they have coordinates in a 2D space, then the Euclidean distance is used. The instance also determines the maximum fleet size and vehicle capacity, the maximum route length and the drop time.

A lot of different instances can be found in the literature. The instances have from 10 to 1200 customers (in the most recent ones). They also vary in depot positioning (centered, corner or random) and customer positioning (clustered, symmetric or real case). In this work we focus on the instances proposed by (CHRISTOFIDES; MINGOZZI; TOTH, 1979) and (GOLDEN et al., 1998), which are two of the most common data sets for the VRP (LAPORTE; ROPKE; VIDAL, 2014).

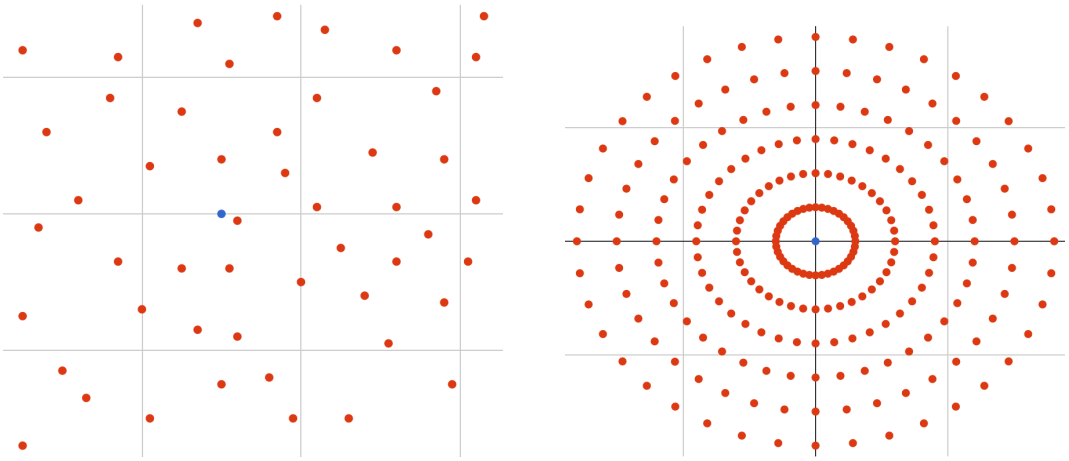
The instance set CMT from (CHRISTOFIDES; MINGOZZI; TOTH, 1979) contains 14 instances with 50 to 200 customers. The instance set GWKC from (GOLDEN et al., 1998) contains 20 instances with 240 to 483 customers distributed spatially in sym-

metric patterns (see Figure 2.1 for an example). Table 2.1 shows the size of the instances (n), the limit number of routes or fleet size (K), the maximum route length constraint (d), the drop time value (T), the vehicle capacity (Q) and the shortest known total route length (best known value, BKV).

Table 2.1: Details of the CMT and GWKC instances.

inst	n	K	d	T	Q	BKV
CMT1	50	5	∞	0	160	524.61
CMT2	75	10	∞	0	140	835.26
CMT3	100	8	∞	0	200	826.14
CMT4	150	12	∞	0	200	1028.42
CMT5	199	17	∞	0	200	1291.29
CMT6	50	6	6200	10	160	555.43
CMT7	75	11	160	10	140	909.67
CMT8	100	9	230	10	200	865.95
CMT9	150	14	200	10	200	1162.55
CMT10	199	18	200	10	200	1395.85
CMT11	120	7	∞	0	200	1042.11
CMT12	100	10	∞	0	200	819.56
CMT13	120	11	720	50	200	1541.14
CMT14	100	11	1040	90	200	866.37
GWKC1	240	9	650	0	550	5623.47
GWKC2	320	10	900	0	700	8404.61
GWKC3	400	9	1200	0	900	11036.20
GWKC4	480	10	1600	0	1000	13590.00
GWKC5	200	5	1800	0	900	6460.98
GWKC6	280	7	1500	0	900	8412.90
GWKC7	360	8	1300	0	900	10102.70
GWKC8	440	10	1200	0	900	11635.30
GWKC9	255	14	∞	0	1000	579.71
GWKC10	323	16	∞	0	1000	735.66
GWKC11	399	17	∞	0	1000	912.03
GWKC12	483	19	∞	0	1000	1101.50
GWKC13	252	26	∞	0	1000	857.19
GWKC14	320	29	∞	0	1000	1080.55
GWKC15	396	33	∞	0	1000	1337.87
GWKC16	480	36	∞	0	1000	1611.56
GWKC17	240	22	∞	0	200	707.76
GWKC18	300	27	∞	0	200	995.13
GWKC19	360	33	∞	0	200	1365.60
GWKC20	420	38	∞	0	200	1817.59

Figure 2.1: This figure shows the distribution of clientes in CMT and GWKC instances. On the left is the CMT1 instance, on the right is the GWKC1. The red dots are the clients and the blue dot is the depot. The images are from (CVRPLIB, 2016).



2.3 Heuristic algorithms

The VRP is a hard combinatorial optimization problem and exact algorithms are able to solve only relatively small instances. Thus, search heuristics methods are used to reach high quality solutions on instances with a large amount of customers. The heuristics methods for the VRP can be classified in constructive heuristics and improvement heuristics, the latter is divided in local search and population based heuristics.

2.3.1 Constructive Heuristics

The constructive heuristics for the VRP are used to generate a solution for a given instance usually in a short computational time. It starts from an empty solution and in each iteration keeps adding new costumers to the routes until all the customers are visited. It is often used to generate a start solution to improvement heuristics.

2.3.1.1 Sweep Algorithm

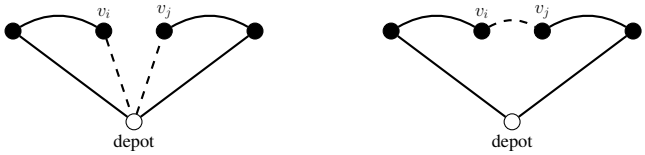
This heuristic can be classified as a insertion heuristic because customers are inserted in the solution one by one always keeping the solution feasible. The algorithm is called sweep because the insertion order is defined by the angle that the customer makes with an arbitrary axis centred at the depot. Feasible routes are created by rotating an imaginary line, with one end tied at the depot, and when a customer is touched it is inserted in the current route. If the maximum route length or vehicle capacity constraints are violated

a new route is created. Different start positions can result in different solutions, then an improvement in this heuristic is to run the sweep algorithm once for each customer as the reference point and use the best solution found. This heuristic is very simple and can find solutions for the CMT instances with an average deviation of 7.01% from the best known values.

2.3.1.2 Savings Heuristic

This heuristic is widely used in VRP problems because of its simplicity, speed and it often obtains good results, in average 6.71% above the optimal solution in CMT instances (see (CORDEAU et al., 2002)). Consider a route which visits customer V_i last, and another route which visits customer V_j first. We can join these routes by going directly from V_i to V_j. This results in *savings* of $s_{ij} = d_{V_i,V_0} + d_{V_0,V_j} - d_{V_i,V_j}$. The Figure 2.2 exemplify the join of two routes. The heuristic proposed by (CLARKE; WRIGHT, 1964) determines the savings s_{ij} for each pair of customers V_i and V_j, and sorts them in a non-increasing order. Then, the algorithm creates one route for each customer, starting at the depot, visiting only this customer, and then returning to the depot. Finally, it visits the savings list in the sorted order cyclically, and repeatedly applies feasible joins, until no such join is possible. A join is feasible for a saving s_{ij} if two routes with endpoints V_i and V_j exist and the resulting route respect the vehicle capacity and route length constraints.

Figure 2.2: Savings heuristics route join example.



2.3.2 Improvement Heuristics

Improvement heuristics are algorithms that receive a solution as input, which is modified by performing a series of operations to obtain a new one. The objective is to find improved solutions by modifications and evaluations that are made in a systematic way. Improvement heuristics for the VRP perform intra-route and inter-route moves to modify the current solution. These moves generate a pool of neighbour solutions that is called *search space*. We will present the main moves in the next section. Explore all the neighbourhood often requires too much operations, thus some techniques that considers

geographical restrictions to avoid moves between distant customers are used to reduce the search space. One of them is the generalized insertion procedure GENI that is a kind of intra-route move, which is presented in more details in the Subsection 2.3.2.2.

Usually, improvement heuristics allow worse solutions and even unfeasible solutions during the search to escape from local minimum with the aim of finding better solutions after a few more moves. Therefore, metaheuristics must be used to find high quality solutions. We present the most common metaheuristics for the VRP in Sections 2.3.2.3, 2.3.2.4, 2.3.2.5 and 2.3.2.6.

2.3.2.1 Neighborhoods

The neighborhoods in VRP are called *moves* which are operations that transform a solution s in another solution s' that shares some characteristics of s. The moves can be separated into two major categories *intra-route moves* and *inter-route moves*. The intra-route moves, also known as single-route neighborhood, affect only one route of the current solution at a time. They permute the customers within the route with the objective of optimize the route length. Thus, traveling salesman problem (TSP) neighborhoods can be used as intra-route moves for the VRP.

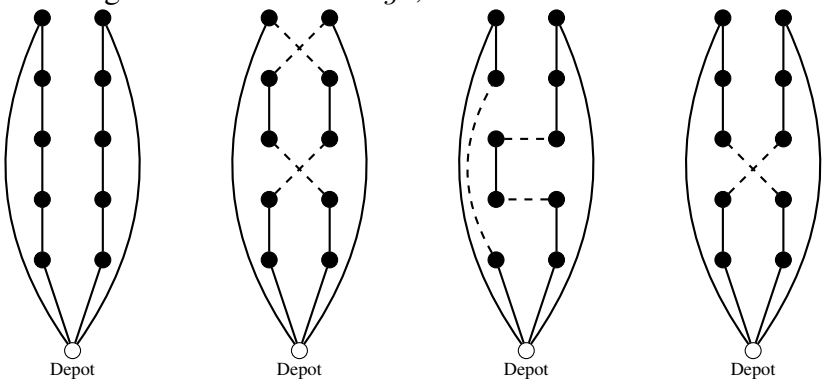
The most used intra-route move in modern heuristics is the 2-OPT proposed by (LIN, 1965), where two edges are removed and two new edges are created to complete the route.

The inter-route moves, also known as multiroute neighborhood, move customers from one route to another involving two or more routes. The most common moves are $exchange$, $relocate$, and 2-OPT*.

The $exchange$ move, also known as $swap$, consists in choosing one or more consecutive customers in two different routes and then exchange their positions. In the $relocate$ neighborhood one or more customers are moved from one route to another route. The 2-OPT* is similar to the 2-OPT, two edges from two different routes are removed and new ones are inserted. The Figure 2.3 shows an example of each inter-route move described above.

It is important to notice that, in order to have an efficient algorithm, the evaluation of the resulting cost of a move must be made in a few operations. The move must be applied to the solution only after the neighborhood evaluation.

Figure 2.3: The $exchange$, $relocate$ and 2-OPT* moves.



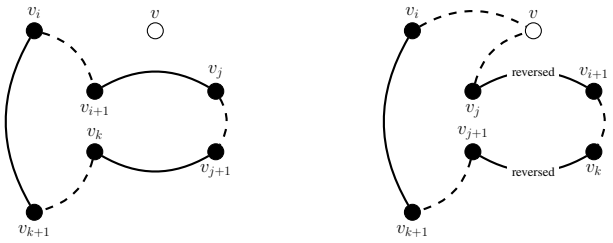
(a) Original routes (b) $Exchange$ (c) $Relocate$ (d) 2-OPT*

2.3.2.2 GENI

The generalized insertion procedure, known in the literature as GENI, was proposed by (Gendreau, Michel and Hertz, Alain and Laporte, 1992) for the traveling salesman problem. The GENI algorithm is widely used in VRP heuristics to insert customers to routes or to remove customers from routes. Together with the insertion or removal of a vertex GENI applies a subset of 3-opt and 4-opt moves to the route. Since the complete exploration of the neighborhood would be too expensive, $O(n^4)$ operations for the 4-opt moves, only the q-nearest vertices. Considering the fact that the q-nearest list must be updated after a insertion or removal, the GENI has time complexity $O(nq^4 + n^2)$.

Given that the $N_q(v)$ are the q-nearest vertices of v the $v_i, v_j \in N_q(v)$, $v_k \in N_q(v_{i+1})$ and $v_l \in N_p(v_{j+1})$. The GENI 3-opt moves are called *Type I*. In *Type I insertion*, Figure 2.4, the vertex v is inserted between v_i and v_j; the edges (v_i, v_{i+1}), (v_j, v_{j+1}) and (v_k, v_{k+1}) are deleted and the new edges (v, v_i), (v, v_j), $(v_i + 1, v_k)$ and $(v_{j+1}, v_{k+1}$ are created.

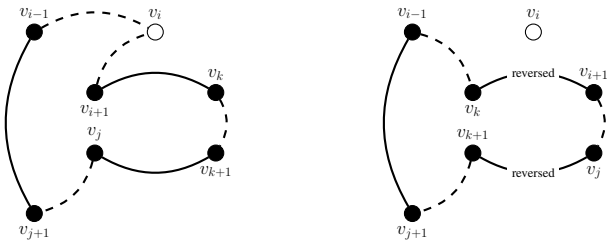
Figure 2.4: GENI Type I insertion.



In *Type I removal*, Figure 2.5, the vertex v_i together with the edges (v_{i-1}, v_i) and (v_i, v_{i+1}) are removed. The other edges removed are (v_k, v_{k+1}) and (v_j, v_{j+1}). Then, the edges (v_{i-1}, v_k), (v_{i+1}, v_j) and (v_{k+1}, v_{j+1}) are created.

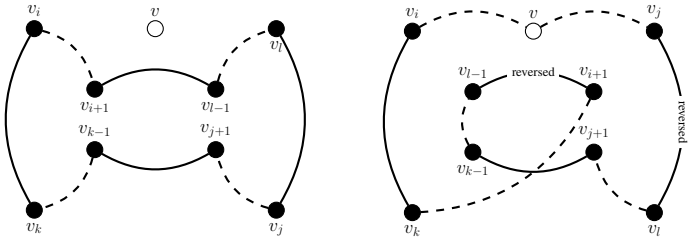
The GENI 4-opt moves are called *Type II*. The *Type II insertion*, Figure 2.6, con-

Figure 2.5: GENI Type I removal.



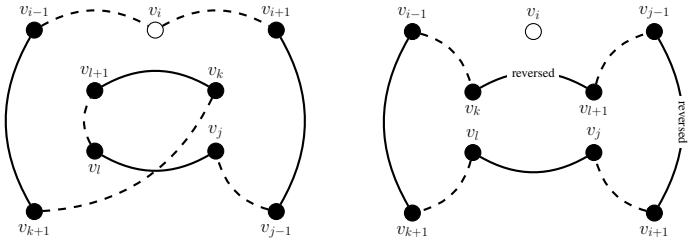
sists in inserting the vertex v between the vertices v_i and v_j where the edges removed are (v_i, v_{i+1}), (v_{l-1}, v_l), (v_j, v_{j+1}) and (v_{k-1}, v_k), then the new edges are (v_i, v), (v, v_j), (v_l, v_{j-1}), (v_{k-1}, v_{l-1}) and (v_{i+1}, v_k).

Figure 2.6: GENI Type II insertion.



For the *Type II removal*, Figure 2.7, the edges removed are (v_{i-1}, v_i), (v_i, v_{i+1}), (v_{j-1}, v_j), (v_l, v_{l+1}) and (v_k, v_{k+1}). The new edges are (v_{i-1}, v_k), (v_{l+1}, v_{j-1}), (v_{i+1}, v_j) and (v_l, v_{k+1}).

Figure 2.7: GENI Type II removal.



2.3.2.3 Simulated Annealing

The simulated annealing is a simple heuristic that has two main components: a neighborhood generator and an acceptor. The neighborhood generator must generate a random neighbor solution s' of a given solution s. The acceptor defines if the next solution used in the neighborhood generator will be s' or s. The solution s' is always accepted if s' is better s, $f(s') < f(s)$, otherwise s' is accepted with probability $e^{-(f(s')-f(s))/T}$. Here, T is the so-called temperature which decreases in each iteration to intensify the search around a promising solution. See (OSMAN, 1993) for an application of the simulated

annealing in the VRP.

The Record-to-record travel is a variation of the simulated annealing that is also called deterministic annealing. In this algorithm the solution s' is only accepted if its cost does not exceed the best solution found so far by a factor σ. Usually σ is a little more than 1 (e.g. $\sigma = 1.05$) (LAPORTE; ROPKE; VIDAL, 2014).

2.3.2.4 Variable neighborhood search

The variable neighborhood search (VNS), as the name suggests, is based on the fact that different neighborhood operators applied to the same solution can result in different local optimal solutions. Thus, the VNS needs a set of different neighborhoods that usually is ordered in increasing size of search space. The first neighborhood is applied to the initial solution until reach the local minimum, then the next neighborhood operator is applied to the resulting solution. When the last neighborhood was applied the first one is tried again and it goes until no of the neighborhood operators can find a better solution. The use of multiple neighborhoods diversify the search because each one explores different regions in the search space.

The paper (KYTÖJOKI et al., 2007) proposes an efficient VNS implementation for the VRP that is able to find results near to the best known value in a few seconds. Their algorithm use the inter-route neighborhoods $relocate$, 2-OPT* and $exchange$, in this order. Also, after each inter-route neighborhood, the intra-route neighborhoods 2-OPT, Or-OPT and 3-OPT are applied to the modified routes.

2.3.2.5 Tabu Search

Tabu search (TS) is one of the most used meta-heuristic for the VRP with a variety of implementations proposed in the pasts years (LAPORTE; ROPKE; VIDAL, 2014). It is a meta-heuristic which guides a local search through the search space and it is one of the most successful heuristics for vehicle routing problems (CORDEAU et al., 2002). TS is a local search that starts with an initial solution and moves to the next solution after visiting a given neighborhood, the algorithm goes until a stop criterion is reached. The main feature of the TS is that it allows non-improving moves and avoids cycling by storing recent moves in a short-term memory and declaring moves that return to previous solutions as tabu (GLOVER; LAGUNA, 1997). A common tabu criteria for the VRP is that a customer is not allowed to go back to its previous route for θ iterations when moved

from one route to another.

A classical TS implementation for the VRP was proposed by (GENDREAU; HERTZ; LAPORTE, 1994), they use the GENI heuristic for insertions and removals in the $relocate$ neighbourhood. The main feature of their algorithm is consider infeasible solutions during the search, it means that solutions that violate the vehicle capacity or route length are also visited, but the excess amount is penalized in the objective function by a factor that increases in each iteration.

Other good implementation of TS for the VRP is the granular tabu search from (TOTH; VIGO, 2003). The idea is to remove unpromising edges whose cost exceeds a threshold. This threshold in based in a good solution determined by a fast heuristic.

2.3.2.6 Iterated Local Search

The iterated local search (ILS) metaheuristic can be seen as a framework where any local search procedure can be embedded in it. (CORDEAU; MAISCHBERGER, 2012) and (CHEN; HUANG; DONG, 2010) are two successful examples of ILS for the VRP, the first has used a tabu search as local search procedure and the last has used a variable neighborhood descent.

The ILS consists in three components: a perturbation procedure, a local search heuristic and an acceptance criterion. The perturbation procedure modifies the solution with the aim of jumping the search to different places in the search space. The perturbation must be done with care, if it is to strong the algorithm is reduced to a random start. In the acceptance criterion is selected the solution that will be used in the next iteration. The algorithm, as shown in Algorithm 1, starts from some local minimum and repeatedly applies a perturbation to escape from it followed by a tabu search to find another local minimum, until some stopping criterion is satisfied.

The iterated variable neighborhood search algorithm proposed by (CHEN; HUANG; DONG, 2010) the perturbation procedure is a exchange move where two route are randomly chosen as well the amount of customers exchanged in each route. The acceptance criterion selects the best known solution so far if it was not improved in 50 consecutive iterations, otherwise it uses a technique similar to simulated annealing to select between the last used solution or the solution returned by the VND.

Algorithm 1 Iterated tabu search

```
1: function ITS( )
2:     s_0 ← GenerateInitialSolution
3:     s* ← localSearch(s_0)
4:     s** ← s*
5:     for termination condition do
6:         s' ← perturb(s*)
7:         s'* ← localSearch(s')
8:         if f(s'*) < f(s**) then
9:             s** ← s'*
10:        end if
11:        s* ← acceptanceCriterion(s'*, s*, s**)
12:    end for
13:    return the best solution s* found during search
14: end function
```

2.3.3 Population Based Heuristics

Population based heuristics, also known as evolutionary algorithms, use methods that are inspired by evolution of species, such as reproduction, mutation, recombination and selection. The algorithm starts with a population, which is a pool of solutions, then in each iteration a series of strategies are applied so that the individuals, or solutions, are combined or modified with the aim of finding better solutions.

The two most successful population based metaheuristic are genetic algorithm and ant colony optimization that we describe in the next sections. Even though, population based heuristics are widely studied, all known successful VRP heuristic of this type use some kind of local search to intensify the search in promising solutions (LAPORTE; ROPKE; VIDAL, 2014). The resulting algorithms are so-called memetic algorithm (MOSCATO; COTTA, 2010) and hybridized ant colony algorithm (ABDULKADER; GAJPAL; ELMEKKAWY, 2015).

2.3.3.1 Genetic Algorithm

The genetic algorithm (GA) starts by a population of solutions, these solutions are referred in GA as *chromosomes*. Then, new solutions, or individuals, are created using two operators called *crossover* and *mutation* applied to the chosen solutions in order to improve then or diversify the population. After a sufficient amount of new solutions a new population is selected and the algorithm starts again. It goes until a termination condition is satisfied.

The first competitive GA that could outperforms the existing powerful tabu search methods was proposed by (PRINS, 2004). In the proposed GA, the *chromosome* representation is a simple sequence of clients without route delimiters. This representation is also called *giant-tours* because it can be seen as the order in which a vehicle must visit all customers. Then a optimal split procedure is applied to divide the giant-tour in routes following the tour order and respecting the vehicle capacity and route length constraints. This technique is still used in state-of-the-art algorithms as (VIDAL et al., 2012).

The GA proposed by (NAGATA; BRÄYSY, 2009) is one of the best heuristics for the VRP. They use as initial population an amount of 100 solutions. The solutions are generated by a procedure similar to the savings heuristic where routes are joined in a random order together with a local search procedure to improve the solution quality. They use a crossover technique called *edge assembly crossover* (EAX) that was proposed by the same author for the traveling salesman problem and adapted to the VRP. The EAX is a five step procedure that receives two parent solution and creates only one child solution based on the parent characteristics. The EAX can generate infeasible solutions that violates the vehicle capacity constraint, when it happens a repairing procedure is applied. The repairing procedure is a best improvement local search that considers only moves that reduce the overcapacity violation. Finally, a first improvement local search using the 2-OPT, $Relocate$ and $Exchange$ neighborhoods is applied to improve the resulting solution. The modification and the local search procedures are the *mutation* components of the GA.

2.3.3.2 Ant Colony Optimization

The inspiring source of this metaheuristic proposed by (DORIGO; CARO; GAMBARDELLA, 1999) is the pheromone used by ants as trail marker and as communication medium. This algorithm explores a technique of learning based on the history of previous visited solutions. The ant colony optimization (ACO) algorithms consist in iterations over three main step: generation of a pool of solutions (ants) considering the pheromone information, update the pheromone information also called pheromone evaporation and *daemon actions* that is a local search or adjusts to the pheromone to intensify the search in promising solutions.

(DOERNER et al., 2004) is a successful implementation of this metaheuristic for the VRP. They propose a ACO that uses the savings heuristic (see Section 2.3.1.2) to construct the solutions. Instead of always join the routes with biggest saving, they use

the pheromones to increase the probability of join more promising routes. They compare three different approaches to handle with pheromones: *rank based ant system* where only the best E ants leave pheromone, *min-max ant system* where only the best ant leave pheromone but there is a upper and lower bounds on the pheromone values and finally *and colony system* where also the best ant leaves pheromone but the global evaporation is also restricted to arcs in the best solution found. For all solutions found a local search using $exchange$ and 2-OPT neighborhood is applied to guarantee that they are a local minimum.

2.4 Conclusion

The vehicle routing problem has been subject of studies for more than 50 years, but great advances were reached in the past 10 years. The new algorithms are able to solve bigger instances in less and less computational time. The industry together with the academy continue to bring new instances and challenging constraints and variations that turns the VRP an always interesting and important topic of research. An open challenge is design heuristic algorithms for the VRP that have a good score in four characteristics: speed, accuracy, flexibility and simplicity. Such an algorithm is hard to design because usually speed and accuracy came in detriment of flexibility and simplicity.

3 THE MULTI-COMPARTMENT VEHICLE ROUTING PROBLEM

The problem called the *multi-compartment vehicle routing problem* (MCVRP), also found as VRP with compartments (VRPC) and multi-compartment delivery problem (MCDP) in the literature, extends the well-known vehicle routing problem (VRP) by allowing the co-delivery (or co-collection) of different products that must be kept in separate compartments. The MCVRP is a single depot vehicle routing problem for multiple product types given a homogeneous fleet. Each customer may have different demand for each product type, and the vehicles have multiple compartments of different sizes dedicated for each product type. As in the standard vehicle routing problem, the aim is to satisfy the demands of every customer such that the total travel time of all vehicles is minimized. The MCVRP generalizes the $\mathcal{NP}$-hard Capacitated VRP, and thus is also $\mathcal{NP}$-hard. The largest instances that state-of-the-art exact approaches for MCVRP could solve contain about 50 customers (COELHO; LAPORTE, 2015). Thus larger problems, or problems with additional constraints are usually solved by heuristic algorithms.

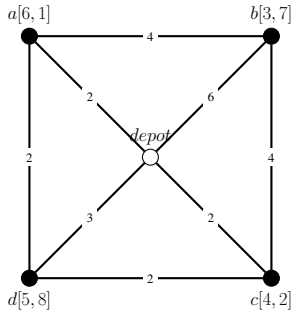
The MCVRP has an important feature that is the *split demand* which allows customers to be visited multiple times, but the demand for each product must be attended in a single visit. It is important to notice that the split demands feature is different from split delivery (SDVRP) ((ARCHETTI N. BIANCHESSI, 2014)) where a demand can be attended for multiple vehicles and may be greater than the compartment capacity.

We present in Figure 3.1 an example, proposed in (El Fallahi; PRINS; Wolfler Calvo, 2008), to better understand the difference between the VRP, the SDVRP and the MCVRP. The customers a, b, c, d have demands for two different products and the depot is the square in the center. The vehicles have capacity 18 in two compartments of capacity 9. For the VRP and the SDVRP suppose that the products can be mixed together then the customers demands is the sum of the two products demand. The shortest length to attend all demands for the VRP is 22 in three routes: $(a[7], b[10]), (c[6]), (d[13])$; for the SDVRP the length is 19 in two routes: $(a[2], b[10], c[6]), (a[5], d[13])$; and for the MCVRP the length is 20 in two routes: $(a[6, 0], b[3, 7], c[0, 2]), (a[0, 1], d[5, 8], c[4, 0])$.

In summary, the MCVRP has the following characteristics:

- Single Depot: all vehicles must depart and arrive at the same place;
- Homogeneous Fleet: all vehicles are identical (equal amount of compartments and capacity);
- Multiple product types: each product type has a dedicated compartment;

Figure 3.1: Example to explain the difference between the VRP, the SDVRP and the MCVRP.



- Split demands: customer demands for different product types may be attended by different vehicles;
- Route duration limit: usually these problems have maximum route duration constraint, it means that the time from the depot, visiting the customers (including the drop time), and return to the depot must not exceeds a maximum time.

This chapter is organized as follows. The next section address real-world application, we formally define the problem and present the mathematical formulation in Section 3.2. The instances available in literature are described in Section 3.4. In the Section 3.5 we present the main heuristic algorithms found in literature. We conclude in Section 3.6.

3.1 Real-world Applications

Real-world applications for this problem are cases where products must be transported in different compartments for some particular reason. In this section we present MCVRP applications in real-world cases found in the literature. In these applications usually other constraints are added to attend the reality of each problem.

In the diary factory the raw milk collection use temperature-controlled vehicles that have many compartments and consume more fuel than regular vehicles. Raw milk is a very perishable product so the delivery time is not only a cost factor but also an important factor for the quality of the final product. Milks from different collection center cannot mixed in the same compartment as well milks of different types. (SETHANAN; PITAKASO, 2016) proposes a differential evolution algorithm for this problem that, besides the traveling cost, also considers the vehicle cleaning cost in the objective function. (MENDOZA et al., 2010) studied a memetic algorithm for the MCVRP with stochastic

demands that was motivated by this real-world problem.

The delivery of groceries to convenience stores often use vehicles with multi-compartments. Unlike a supermarket, convenience stores carry most of their inventory in the front of the store. These space limitations impose the need to a very tight control of inventories which requires small orders of different product types. These orders must be attended by a single distributor that can deliver dry, refrigerates and frozen items together in the same vehicle. A compartment can often be a box filled with dry ice or in large trucks spaces separated by bulkheads. This problem was studied in (CHAJAKIS; GUIGNARD, 2003) where optimization models were proposed for two possible cargo space layouts.

The MCVRP is also applied to transportation of live animals to slaughterhouses, this problem is known as Livestock collection problem (OPPEN; KKETANGEN, 2008). It consists in transportation of living animals from farmers to one slaughterhouse. Some interesting constraints arise in this problem: animals cannot be transported continuously for more than 8 hours; mixing animals with and without horns or animals of substantially different size are not allowed; and the visit order must follow a health status of the animals. (OPPEN; KKETANGEN, 2008) proposes a tabu search for this specific problem.

In (LAHYANI et al., 2015) the authors propose a model and an exact approach to solve a rich MCVRP that arises in olive oil collection process in Tunisia. Olive oil have three different grades known as extra, virgin and lampante. The different grades must be kept separated during the transportation, then vehicles with compartments are necessary. Other important constraint in this problem is that it is forbidden to load extra and virgin oil immediately after lampante oil in the same compartment, unless it has been cleaned.

Next we present the two main applications of the MCVRP that is the fuel delivery and waste collection.

3.1.1 Fuel delivery

The fuel delivery problem is the most studied application of the MCVRP. The fuel distribution consists in several different product types that must be delivered by compartmentalized vehicles to customers with several tanks. The vehicles often are not equipped with debit meters, which implies that whenever a deliver is made the full contempt of the compartment must be emptied. The vehicles have compartments with fixed sizes and the products are incompatible with each other, then they must be delivered in different compartments. There are no incompatibilities between product and compartment, it means

that each product can be allocated any compartment where the packing arise as a sub-problem. Often customers order large demands and one compartment receives only one product for one customer so the main problem is assign orders to vehicle compartments. Once it is done the routing part is rather easy, one TSP with few stops must be solved for each truck.

A study about different problems that arise in petroleum companies can be found in (COELHO; LAPORTE, 2015). They propose a classification where compartments and tanks can be split or unsplit, where a split tank may receive deliveries from different vehicles, likewise, a split compartment can deliver the load to different compartments. In (COELHO; LAPORTE, 2015) also present the mathematical model and an exact algorithm for these variants of the MCVRP including multi-period.

In (DERIGS et al., 2010) the MCVRP is called VRP with compartments (VRPC) and define it as an abstract problem covering different applications that can occur in retail and petrol industries. They consider that certain products pair must not be loaded in the same compartment and certain products cannot be transported in certain compartments. That work also provide a study of several heuristics to solve VRPC.

(CORNILLIER et al., 2008) propose a multi-phase heuristic for a variant of the VRPC where one must determine, for each day of the planning horizon, how much of each product should be delivered to each station, how to load these products into vehicle compartments and how to plan vehicle routes. The station do not specify fixed visit dates and delivery amounts, these decisions are optimized by the distributor.

The paper (POPOVIC; VIDOVIC; RADIVOJEVIC, 2012) treat the problem fuel delivery problem as Inventory Routing Problem (IRP) where the distributor has the responsibility of clients inventory management, given the clients (or stations) order quantity and time of delivery. Then the distributor can better utilize the vehicles.

A real case of the fuel delivery problem is related in (AVELLA; BOCCIA; SFORZA, 2004). Each customer has an order and a frequency of one or more days. One must determine the routes and delivery plan for each truck where each order must be satisfied by the following day and the total cost is minimum. The paper presents a heuristic and an exact algorithm to solve this problem.

In petrol station replenishment often is found problems with multiple depot and time windows as presented in (CORNILLIER; BOCTOR; RENAUD, 2012; BENANTAR; OUAFI, 2012). In (MENDOZA et al., 2010) stochastic demands has been introduced to the MCVRP, i.e. the exact value of demands is not known at the moment when

the routes are planned, to obtain the MCVRPSD.

3.1.2 Waste collection systems

The MCVRP arises in the waste collection systems by the necessity of collecting different types of waste that can not be mixed. The total cost of recycling can be reduced if the waste separation process is eliminated and this can be done in collection site with disposal bins for each kind of waste. Then, for transportation, vehicles with compartments are required to keep the types of waste separated.

(ELBEK; WØHLK, 2016) describes a real case of glass and paper waste collection in Denmark. They use vehicles with two compartments and variable capacities. The objective is ensure that the cubes were the waste is placed are emptied before being overfilled. (REED; YIANNAKOU; EVERING, 2014) also studied the waste collection in the UK. They propose an ant colony algorithm that we will see in details in this work.

The glass waste collection is also the context of study in (HENKE; SPERANZA; WÄSCHER, 2015), they study a special case that occurs in Germany where glass of different colors must be kept separated. In this context the trucks allow the use of bulkheads in predefined positions, it will split the loading space in compartments of variable size. Then, the number of compartments can be identical to the number of products types but can also be smaller. They propose a variable neighborhood search that determines not only the vehicle routes, but also for each route how many compartments and what the size of each compartment the vehicle capacity should be divided. Results for random generated and real case instances are presented in the paper.

A complete study about the urban solid waste collection system is found in (LU et al.,). The paper highlight the deficiencies of the state-of-the-art studies on algorithms for this problem, they focus on U.S. and Europe waste collection system. In densely populated cities, like in China, the waste collection needs to be done daily. (LU et al.,) propose a heuristic for multi-constrained and multi-compartment roll-on and roll-off waste collection and use it to solve a real case in a city in China.

The Table 3.1 summarizes some of the papers cited in this chapter. The columns represent the main features, the mark • means that the paper implements the feature in the respective column.

Table 3.1: Summary of the main features implemented by the papers.

	Split demands	Split compartments	Split tank	Homogeneous fleet	Fixed number of compartments	Fixed compartment capacity	Dedicated compartments	Assign demands to compartments	Stochastic demands	Time windows	Multi-period	Route duration limited	Limited number of vehicles	Test with real-world instances	Solution method
This work Chap. 4		•		•	•	•	•					•	•		Tabu search
This work Chap. 5		•		•	•	•	•					•	•		Iterated tabu search
This work Chap. 5	•	•		•	•	•	•					•	•		Iterated tabu search
(El Fallahi; PRINS; Wolfler Calvo, 2008)		•		•	•	•	•					•			Memetic and Tabu search
(El Fallahi; PRINS; Wolfler Calvo, 2008)	•	•		•	•	•	•					•			Memetic and Tabu search
(MUYLDERMANS; PANG, 2010)	•	•		•	•	•	•								Metaheuristic
(DERIGS et al., 2010)	•	•		•	•	•	•						•		General heuristic
(REED; YIANNAKOU; EVERING, 2014)		•		•	•	•	•								Ant colony algorithm
(ABDULKADER; GAJPAL; ELMEKKAWY, 2015)		•		•	•	•	•					•	•		Hybrizided ant colony algorithm
(SETHANAN; PITAKASO, 2016)	•		•		•	•		•				•	•		Exact algorithm and metaheuristic
(MENDOZA et al., 2010)		•		•	•	•	•		•			•	•		Memetic algorithm
(OPPEN; KKETANGEN, 2008)						•		•				•		•	Tabu Search
(LAHYANI et al., 2015)	•	•			•	•		•			•			•	Branch-and-cut algorithm
(COELHO; LAPORTE, 2015)	•	•	•	•	•	•		•			•		•		Exact algorithms
(COELHO; LAPORTE, 2015)	•	•		•	•	•		•			•		•		Exact algorithms
(COELHO; LAPORTE, 2015)	•		•	•	•	•		•			•		•		Exact algorithms
(COELHO; LAPORTE, 2015)	•			•	•	•		•			•		•		Exact algorithms
(DERIGS et al., 2010)	•	•		•	•			•							General heuristic
(CORNILLIER et al., 2008)	•	•	•	•	•			•		•	•				Exact algorithm
(AVELLA; BOCCIA; SFORZA, 2004)					•	•		•		•	•	•	•	•	Heuristic and exact approach
(ELBEK; WØHLK, 2016)	•	•			•	•	•		•		•			•	Variable neighborhood search
(HENKE; SPERANZA; WÄSCHER, 2015)	•	•		•				•							Variable neighborhood search

3.2 Problem Definition

The MCVRP is a variation of the VRP where the fleet consists of identical vehicles with multiple compartments and the customers have demands for different products. We are given a set of locations $V = \{V_0\} \cup V_+$, where V_0 is the *depot*, and $V_+ = \{V_1, \ldots, V_n\}$ is the set of *customers*. Each pair of locations $i, j \in V$ has a travel time d_{ij}. We assume symmetric travel times ($d_{ij} = d_{ji}$). Each customer may have additionally a drop time t_i, i.e. the time needed to load or unload the demand. There are m different types of products $P = [m]$[1] , and a fleet of identical vehicles with m compartments, dedicated to the different products, with capacity $C \in \mathbb{R}^m$. Each customer $i \in V_+$ has a demand $c_i \in \mathbb{R}^m$, and we assume $c_i \leq C$. A valid route of a vehicle starts and ends at the depot and visits a number of customers. There is no constraint on the number of visited customers per route, but a customer may be visited several times in different routes. Formally, a visit is a pair $v = (V(v), P(v)) \in V_+ \times 2^P$ of a customer $V(v)$ and a set of product types $P(v)$, and a route is represented by an ordered subset $R = \{v_1, \ldots, v_{l(R)}\}$ of visits of length $l(R)$. The set of visited customers of a route is $V(R) = \{V(v) \mid v \in R\}$, the set of attended client-demand pairs $P(R) = \{(V_i, p) \mid V_i \in V, p \in P, (V_i, P) \in R\}$. The total

[1] We use the notation $[n] = \{1, 2, \ldots, n\}$.

time of a route is

$$d(R) = d_{V_0,V(v_1)} + \sum_{1 \leq i < l(R)} d_{V(v_i),V(v_{i+1})} + d_{V(v_{l(R)}),V_0} + \sum_{i \in V(R)} t_i,$$

and its demand is $c(R)$, where $c(R)_p = \sum_{i \in l(R) | P(r_i) = p} c_{r_i,p}$.

We want to find a set of valid routes $s = \{R_1, \ldots, R_r\}$ that partition the set of client-demand pairs ($P(R_i) \cap P(R_j) = \emptyset$ for all $i, j \in [r]$, and $\cup_{i \in [r]} P(R_i) = V_+ \times P$) satisfying the capacity constraints $c(R_i) \leq C$, and such that the total time $d(S) = \sum_{i \in [r]} d(R_i)$ is minimized. The total time travelled by each vehicle must not exceed a maximum time D. There is no limit on the number of routes. For $m = 1$ the problem reduces to the standard CVRP.

In the MCVRP a customer may be visited up to m times, one for each product type, but the demand of a product must be attended in one visit. Different from the classical VRP, the fleet size is not upper bounded. In the taxonomoy of (COELHO; LAPORTE, 2015) for fuel delivery problems, the MCVRP would be classified as as split compartments and unsplit tanks (i.e. product demands).

This is the same problem definition as used in (El Fallahi; PRINS; Wolfler Calvo, 2008). (MUYLDERMANS; PANG, 2010) and (DERIGS et al., 2010) have only one difference which is that they do not consider the total travel time constraint (D), but this is not a significant difference because most of the instances do not have this restriction. In (DERIGS et al., 2010) the authors consider also with other variants where the vehicle compartments are not dedicated to a single product, thus they define incompatibilities between products (products that must not be transported together) and between products and compartments (products that must not be transported in some specific compartment).

3.3 Mathematical Model

The MCVRP can be modeled as an integer linear program as follows. The decision variable x_{ijk} indicates that vehicle k visits arc (i, j), and the decision variable y_{jkp}

indicates that the demand for product p of client j is attended by vehicle k.

$$\textbf{minimize} \quad \sum_{i,j \in V} \sum_{k \in [r]} d_{ij} x_{ijk}, \tag{3.1}$$

$$\textbf{subject to} \quad \sum_{i \in V} x_{ijk} \leq 1, \qquad \forall j \in V_+, \forall k \in [r], \tag{3.2}$$

$$\sum_{i \in V} x_{ijk} = \sum_{i \in V} x_{jik}, \qquad \forall j \in V, k \in [r], \tag{3.3}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1, \qquad \forall S \subseteq V_+, |S| \geq 2, k \in [r], \tag{3.4}$$

$$y_{jkp} \leq \sum_{i \in V} x_{jik}, \qquad \forall j \in V_+, k \in [r], p \in P, \tag{3.5}$$

$$\sum_{k \in [r]} y_{jkp} = 1, \qquad \forall j \in V_+, p \in P, \tag{3.6}$$

$$\sum_{j \in V_+} c_{jp} y_{jkp} \leq C_p, \qquad \forall k \in [r], \forall p \in P, \tag{3.7}$$

$$\sum_{i,j \in V} (d_{ij} + t_j) x_{ijk} \leq D, \qquad \forall k \in [r], \tag{3.8}$$

$$x_{ijk} \in \{0, 1\}, \qquad \forall i, j \in V, k \in [r], \tag{3.9}$$

$$y_{jkp} \in \{0, 1\}, \qquad \forall j \in V_+, k \in [r], p \in P. \tag{3.10}$$

In this formulation we minimize the total travel time (3.11). By constraint (3.12) every customer can be visited at most once per route. Constraint (3.13) establishes route flow conservation, and constraint (3.14) eliminates sub-routes that do not include the depot. Constraint (3.5) couples routing variables x to demand variables y. Constraints (3.6) guarantee that client's demand for a product is attended by a single visit. The capacity and total length constraints are guaranteed by (3.15) and (3.16). Solving this model directly is unpractical due to the exponential number of constraints (3.14), but it can be solved by branch-and-cut methods.

3.3.1 Problem definition for the MCVRP-WS

In this work we also study a less general variant of the MCVRP that is the single visit MCVRP, also called as the MCVRP without splitting (MCVRP-WS) by (El Fallahi; PRINS; Wolfler Calvo, 2008). The MCVRP-WS is a variation of the MCVRP where the client demand for all product types must be attended in one single visit. The MCVRP-WS

can be formulated as follows. Let x_{ijk} indicate that vehicle k travels from $i \in V$ to $j \in V$. Then we want to

$$\textbf{minimize} \quad \sum_{i,j \in V} \sum_{k \in [r]} (d_{ij} + t_j) x_{ijk}, \tag{3.11}$$

$$\textbf{subject to} \quad \sum_{i \in V} \sum_{k \in [r]} x_{ijk} = 1, \qquad \forall j \in V \setminus \{V_0\}, \tag{3.12}$$

$$\sum_{i \in V} x_{ijk} = \sum_{i \in V} x_{jik}, \qquad \forall j \in V, k \in [r], \tag{3.13}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1, \qquad \forall S \subseteq V \setminus \{V_0\}, |S| \geq 2, k \in [r], \tag{3.14}$$

$$\sum_{i,j \in V} c_j x_{ijk} \leq C, \qquad \forall k \in [r], \tag{3.15}$$

$$\sum_{i,j \in V} d_{ij} x_{ijk} \leq D, \qquad \forall k \in [r], \tag{3.16}$$

$$x_{ijk} \in \{0, 1\}, \qquad \forall i, j \in V, k \in [r]. \tag{3.17}$$

In this formulation we minimize the total travel time (3.11). By constraint (3.12) every customer has to be attended exactly once in some route. Constraint (3.13) establishes flow conservation, and constraint (3.14) eliminates subroutes that do not include the depot. The capacity and total length constraints are guaranteed by (3.15) and (3.16). Note that constraint (3.15) is vector-valued and will be expanded into m separate constraints, one for each product type. Solving this model directly is unpractical due to the exponential number of contraints (3.14).

This problem was studied in the literature by three papers (El Fallahi; PRINS; Wolfler Calvo, 2008), (REED; YIANNAKOU; EVERING, 2014) and (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) that we will explain in the following sections.

3.4 Instances

The test instances are based on instances proposed by (CHRISTOFIDES; MINGOZZI; TOTH, 1979) and (GOLDEN et al., 1998), which are two of the most common data sets for the VRP (LAPORTE; ROPKE; VIDAL, 2014). For further details about the CVRP instances see section 2.2.

An instance for the CVRP is transformed into an instance for the MCVRP by a strategy that divides the vehicle's capacity into compartments, and the demand of the

customers into demands for different products. We use four division strategies S1–S4 in our tests. Strategies S1 and S2 have been proposed by (El Fallahi; PRINS; Wolfler Calvo, 2008). Strategies S3 and S4 apply only to the CMT instances, and have been proposed by (REED; YIANNAKOU; EVERING, 2014). Strategy S1 divides the capacity of the vehicle and the demand of each customer into m equal parts where m is the number of compartments. For these instances a solution of the corresponding VRP is also a solution for the MCVRP instance. If we do not allow split demands, the optimal solutions are the same. Therefore, these instances are mainly useful for evaluating the advantage of allowing to visit a customer multiple times, and for testing the scalability of the algorithms with respect to the number of compartments.

Strategy S2 divides the demands of CVRP instances unevenly. For each customer $i \in V_+$ with a demand of c_i in the corresponding VRP, the demand for the first product is $c_{i1} = c_i/k$, for a random $k \in \{3, 4, 5\}$, and the demand for the second product is the remainder $c_{i2} = c_i - c_{i1}$. To define the capacity of the compartments of the vehicles, let $\overline{D}_1$ be the average demand for the first product, and $\overline{D}_2$ the average demand for the second product. Then the capacity of compartment $p \in \{1, 2\}$ is set to

$$C_p = C\overline{D}_p/(\overline{D}_1 + \overline{D}_2).$$

Strategy S3 divides the vehicle's capacity into two compartments in a ratio of 3:1. The customer demands are divided using a 3:1 ratio, except the demands of the sub-region $0 < x, y < 35$, which are divided using a 2:1 ratio. Strategy S4 is similar, but divides vehicle compartments and customer demands using a 4:1 ratio, except for region mentioned above, which maintains a 2:1 ratio. Strategies S3 and S4 apply only to the CMT instances, since they assume that the client's coordinates lie within the square $[0, 100]^2$.

(MUYLDERMANS; PANG, 2010) have proposed another set of instances to explore the benefits of co-collection. The instances are randomly generated varying the number of customers (n=30, 100 and 300), number of product types (k=2, 3, and 4), the compartments capacity and the customers demands. The instances are Euclidean 100x100 square. The location of the depot can be in the lower left corner or in the middle of the square, while the customers can be uniformly distributed or concentrated in the upper right 50x50 area.

3.5 State-of-the-art Heuristics

Although the MCVRP is practically relevant, the problem variant defined above is not widely studied. The previous examples are all motivated by multiple compartments, but, as shown in Table 3.1, have varying characteristics, such as free assignment of products to compartments (OPPEN; KKETANGEN, 2008; CORNILLIER et al., 2008; COELHO; LAPORTE, 2015), time windows (CORNILLIER et al., 2008), stochastic demands (MENDOZA et al., 2010; ELBEK; WØHLK, 2016), or multiple planning periods (AVELLA; BOCCIA; SFORZA, 2004; LAHYANI et al., 2015; COELHO; LAPORTE, 2015). To the best of our knowledge, only (El Fallahi; PRINS; Wolfler Calvo, 2008), (DERIGS et al., 2010), (MUYLDERMANS; PANG, 2010), (REED; YIANNAKOU; EVERING, 2014), and (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) study the MCVRP.

(El Fallahi; PRINS; Wolfler Calvo, 2008) is the first study in the literature about the MCVRP. They were motivated by the distribution of cattle food to farms, where the different feeds are kept separate to avoid contamination. They have proposed a Memetic algorithm, a Tabu search and a set of instances that will be used in further studies. In (MUYLDERMANS; PANG, 2010) the authors have studied the MCVRP and benefits of co-collection presenting several cases where it saves costs compared to separate collection. They have proposed a guided local search that present very good results in instances with only one product type, but for instances with two or more products the solution quality decrease quickly. For this reason, they conclude that or the MCVRP is more difficult or there is good room for improvements in their algorithm. This question is other motivation for our work. (DERIGS et al., 2010) present a set of generic heuristics for the MCVRP and variants and they compare their results against the other two papers. (REED; YIANNAKOU; EVERING, 2014), and (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) are two complementary work about a ant colony algorithm for the MCVRP-WS.

In this section we present in more details these heuristics that are the state-of-the-art for the MCVRP.

3.5.1 Memetic Algorithm

The memetic algorithm (MA) for the MCVRP described here was proposed by (El Fallahi; PRINS; Wolfler Calvo, 2008). The MA is a genetic algorithm hybridized with a

local search procedure used to intensify the search. They use a population of constant size so each offspring obtained by crossover immediately replaces one existing solution. The GA use the *elitism* property it means that the best solution is either preserved or improved.

The chromosome is represented as a string of demands, then for a instance of n customers and m products the chromosome will have nm genes. In this representation the chromosome can be seen as *giant tours* (BEASLEY, 1983) performed by a vehicle of infinite capacity. To obtain a complete solution this representation is split into tours and product demands are aggregated such that each client is visited only once.

The initial population is generated by two techniques. The first half is generated by the random permutation of customers. The other half is generated by a constructive heuristic that generates a VRP solution for each product using the savings algorithm of (CLARKE; WRIGHT, 1964), randomizes these solutions and combines them to solutions for the MCVRP.

They use a classical TSP crossover, where two chromosomes P_1 P_2 and two cutting points i and j are chosen by random. The new solution is formed by customers from index i to j from solution P_1 and the remain customers are from solution P_2 swept circularly from index $j + 1$.

The local search is applied with a fixed probability to improve the new solution. It operates in a complete solution with delimiters obtained by the splitting procedure. Adjacent demands from the same customer in a route are called *aggregate*. In the local search is not allowed to split *aggregates*, it only happens in the crossover. The neighborhood combines 2-opt moves and the relocation of a single demand or all demands of a client to another tour. The first-improvement strategy is used. The resulting solution replaces a random solution of the worse half of the population, if it is fitter than the worst individual.

3.5.2 Tabu Search

In (El Fallahi; PRINS; Wolfler Calvo, 2008) the authors also have proposed a tabu search for the MCVRP. The tabu search starts from a initial solution using the same constructive algorithm used in the MA, as well the same local search to improve the solution. Different from the MA, it relaxes the capacity and length constraints during the search allowing infeasible solutions. Those infeasible solutions are penalized in the objective function using a classical criteria for the VRP proposed by (GENDREAU; HERTZ; LAPORTE, 1994).

To avoid loop back to already visited solutions they use the short-term memory strategy as tabu list. A move is tabu, if it inserts an arc back into the solution that has been removed less than the tabu tenure Θ iterations ago. The tabu tenure is decreased or increased dynamically according to the solution quality to intensify or diversify the search. For diversification their algorithm uses also dynamic restarts, and splits the route of maximum capacity and length excess if no progress to feasibility is made. The latter mechanism is also the only way to split demands of a customer, which were not allocated to different routes in the initial solution. In average the tabu search performs better then the MA.

3.5.3 Guided Local Search

(MUYLDERMANS; PANG, 2010) propose a guided local search for the MCVRP and demonstrate the cost savings of co-collection and -delivery compared to single-product vehicles in several applications.

The algorithm starts with an initial feasible solution obtained by applying the savings algorithm of (CLARKE; WRIGHT, 1964), this solution in subsequently improved with a local search phase combined with a guided local search (GLS) to improve the solution quality. The GLS is a penalty based heuristic, the distance matrix is modified by penalising long distances between two consecutive stops in the current solution (VOUDORIS, 1997). Each combination of customer-product is defined as *stop* in the solution representation. Thus, a route is composed by stops and in a valid solution all stops must be visited.

The local search use the first improvement strategy and explores four well-known neighborhood moves: 2-opt, relocate, exchange and cross.

To speed up the search procedure their algorithm do not examine the full neighbourhood. For each stop s_i a list of the nearest other stops is created in a non-decreasing order. Then, only moves containing these stops are evaluated. The local search also allow the search in infeasible solutions with a penalization procedure.

They find very good results (0.79% above best known values) using classical VRP set of instances with only one product type and single compartment. But for instances with 2 compartments their results are 3 times worse (2.7% above best known values). The authors conclude that MCVRP is more difficult or their algorithm still has some room for improvements.

To demonstrate the benefits of the co-collection they compute the routing cost for co-collection and for separate collection using the last set of instances described in section 3.4. To evaluate the separate collection a CVRP is solved for each commodity using the vehicle total capacity. Their results show that the co-collection take advantages over the separate collection specially in cases when the number of commodities is higher and in large vehicle capacity.

3.5.4 General Heuristic for the MCVRP

(DERIGS et al., 2010) study the vehicle routing problem with compartments (VRPC), which generalizes the MCVRP to flexible compartments, and the allocation of several products to a compartment, subject to product-product and product-compartment incompatibilities. They present a suite of heuristic algorithms including constructive methods, local searches, solution modification by destruction and reconstruction, and metaheuristics. Individual components and seven global configurations were tested on 200 instances. The best combination uses the savings method of (CLARKE; WRIGHT, 1964) to create an initial solution, and several neighborhoods based on k-opt and the removal and re-insertion of product demands. The best performing metaheuristic was record-to-record-travel accepting solutions with a relative deviation from the incumbent of up to 3%.

3.5.5 Ant Colony Algorithm

(REED; YIANNAKOU; EVERING, 2014) present an ant colony system (ACS) for the MCVRP-WS. In each phase a number of ants equal to the number of clients construct feasible tours. Each time no client can be added to the current tour, a new tour starts from a random unvisited client. At the end of each phase the dynamic preferences for a transition from a client to a successor are updated according to standard ACS rules. The tours are improved a a 2-opt local search. The algorithm preprocesses instances by clustering them using a modified k-means algorithm, that maintains the clusters balanced with respect to the capacity of the vehicles. The ACS then is applied to each cluster independently. The authors evaluate the algorithm on five exemplary instances.

3.5.6 Hybridized ant colony algorithm

(ABDULKADER; GAJPAL; ELMEKKAWY, 2015) build on the work of (REED; YIANNAKOU; EVERING, 2014) to create a hybridized ant colony algorithm (HAC). The authors adds a different initialization that sets the dynamic preferences to D^{-1} for the total length D of a random initial solution, and add two neighborhoods to the improvement by local search after the construction via the ACS. The first inserts a customer into a different position in the same or another route, the second swaps to customers from the same or different routes. Experiments suggest that these changes improve the solutions of (REED; YIANNAKOU; EVERING, 2014) by about $5\,\%$, in average.

3.6 Conclusion

In this chapter we have presented the multi-compartment vehicle routing problem (MCVRP) that introduces the co-collection or co-delivery of different types of products to the classical CVRP. This problem have a large range of applications for logistics in different industries. The petrol distribution and waste collection systems are the applications with more studies in the literature, but these problems have specific constraints. The MCVRP that we presented here with the mathematical model still is not well studied in the literature. We have given an overview of the state-of-the-art heuristics and we conclude that this problem have space for improvements.

4 A TABU SEARCH FOR THE MCVRP-WS

In this chapter we propose a tabu search for the MCVRP-WS. The following subsection presents our constructive heuristic used to generate de initial solution and in section 4.2 we present the tabu search in detail.

4.1 A savings method for the MCVRP-WS

To generate an initial solution we modified the savings heuristic proposed by (CLARKE; WRIGHT, 1964).

The generalization to the multi-compartment and time-restricted case is straightforward. We consider a join only feasible if the combined route still satisfies the time and each compartment capacity constraints. A initial solution of good quality has shown experimentally to be important to get better final results for the problem.

4.2 Tabu search

The Tabu search meta-heuristic has been proposed by Glover and is a heuristic based on modification of a solution (see (GLOVER; LAGUNA, 1997)). For a search space S and a neighborhood function $N : S \rightarrow 2^S$ it starts from some initial solution, repeatedly passes from the current solution $s \in S$ to a neighboring solution $s' \in N(s)$ until some stopping criterion is satisfied. Similar to local search, Tabu search chooses a neighbor of better objective function value, until no such neighbor exists. In standard Tabu search, one of the best such neighbors is chosen. Otherwise, the best neighbor which has not been declared tabu is chosen. The tabu mechanism is a short-term memory designed to avoid cycling in local minima and to diversify the search. Commonly, some attributes of recently visited solutions are declared tabu for a number of steps, called the *tabu tenure*, and a solution is considered tabu if it has some tabu attribute. Attributes may be elements of solutions, e.g. an arc visited by some vehicle in a solution of a VRP, or complete solutions. Tabu search also frequently includes so-called *aspiration criteria*, i.e. rules that allow neighboring solutions to be chosen even if they are tabu. After stopping, Tabu search returns the best found solution during the search.

4.2.1 Neighborhoods and tabu mechanism

We use four different neighborhoods in our Tabu search. They are defined in terms of *moves types*, i.e. modifications of the current solution to obtain some neighboring solution. A *shift move* removes some customer from his current route, and inserts him into an arbitrary position in some other route; a *swap move* selects two customers in different routes, and exchanges their positions, i.e. the first customer is inserted into the second route in place of the second customer and *vice versa*. A *crossover move* selects two customers in two different routes, and combines the initial and final parts of the routes to obtain two new routes. For routes $R = \{r_1, r_2, \ldots, r_{l(R)}\}$ and $S = \{s_1, s_2, \ldots, s_{l(S)}\}$ selecting customers r_i and s_j produces new routes $R' = \{r_1, \ldots, r_{i-1}, s_j, \ldots, s_{l(S)}\}$ and $S' = \{s_1, \ldots, s_{j-1}, r_i, \ldots, r_{l(R)}\}$. Finally a *route swap move* selects two customers in a route and swaps their positions.

The Tabu search examines all moves in the presented order (shift, swap, crossover, and route swap). Within a move category, routes are always visited in a random order, and customers always in order of the route. We consider only feasible solutions that respect the capacity and length constraints. The number of examined route swap moves has been limited to $\min\{n^2/4, 250\}$. The search adopts a first improvement strategy, accepting the first non-tabu neighbor which is better than the current solution, or the best non-tabu neighbor, if no better one exists. Ties among several best neighbors are broken in favour of the first best neighbor. The only aspiration criterion is to accept tabu solutions that improve the incumbent.

To define the tabu rules, we consider a given customer being part of some route as a solution attribute. For any of the move types, a client that has been moved from some source route is prohibited to return to that route during the tabu tenure. In some preliminary experiments we have fixed the tabu tenure at 15 steps.

Algorithm 2 shows a pseudo code of the proposed Tabu search.

4.3 Computational Results

The Tabu search has been implemented in C++ and tested on a PC with an AMD FX-8150 Eight-Core processor running at 3.4 GHz, and with 32 GB of main memory. For the tests only one core has been used. The algorithms were tested with classical VRP instances and multiple compartments generated from existing VRP instances since

Algorithm 2 Tabu search pseudocode

Input: Current solution s.
Output: A local minium s^*.

1: $s^* \leftarrow s$ ▷ initialize the best solution with current
2: **while** timeout **do**
3: $N' \leftarrow worstpossiblesolution$ ▷ initialize best neighbor as worst possible solution
4: $improved \leftarrow false$ ▷ flag to stop neighbor search when a neighbor better then s is found
5: **if** !$improved$ **then**
6: $improved \leftarrow$ try All $ShiftMoves(s^*, s, N')$
7: **end if**
8: **if** !$improved$ **then**
9: $improved \leftarrow$ try All $SwapMoves(s^*, s, N')$
10: **end if**
11: **if** !$improved$ **then**
12: $improved \leftarrow$ try All $CrossOverMoves(s^*, s, N')$
13: **end if**
14: **if** !$improved$ **then**
15: $maxMoves \leftarrow min(n^2/4, 250)$
16: **while** !$maxMoves$ & !$improved$ **do**
17: $improved = RouteSwapMove(s^*, s, N')$
18: **end while**
19: **end if**
20: update $TabuList$ with most recent move
21: **if** $N' < s^*$ **then**
22: $s^* \leftarrow N'$
23: **end if**
24: $s \leftarrow N'$
25: **end while**

we were not able to find publicly available MC-VRP instances. This section shows the obtained results and compares them against (El Fallahi; PRINS; Wolfler Calvo, 2008) and (REED; YIANNAKOU; EVERING, 2014), which are, to our knowledge, the only publications which address the same problem.

4.3.1 Analysis of the results

The results obtained in our tests are reported in Tables 4.1 and 4.2. Table 4.1 shows the results obtained on instances sets S1 (one comparment), S3, and S4. Each instance of the set was executed ten times with the same parameters and a different random seed. We present for each instance the best known value of the VRP case (column "BKV") and the solution obtained by the constructive method of Clarke and Wright (column "C/W"). For the Tabu search we report the average relative deviation from the best known value (column "TS"), the average time in seconds to find the best solution (column "T (s)") and the relative deviation of the best solution in all ten replications from the best known value (column "Best"). The results have been obtained with a time limit of $n^2/100$ seconds, where n is the number of customers of the instance.

Table 4.1: Results of the constructive heuristic and the Tabu search on instance sets S1, S3, and S4 compared to best known values of the VRP.

		S1				S3				S4			
Name	BKV	CW	TS	T (s)	Best	C/W	TS	T (s)	Best	C/W	TS	T (s)	Best
CMT1	524.6	11.4	2.1	10.0	0.6	18.8	5.6	9.7	5.0	17.8	6.2	13.3	4.8
CMT2	835.3	8.6	6.8	24.4	5.4	10.2	7.6	27.7	5.7	13.2	8.8	25.8	8.0
CMT3	826.1	7.6	4.0	68.6	2.0	10.8	8.8	58.5	6.9	13.0	7.1	71.4	6.6
CMT4	1028.4	10.9	6.1	144.9	4.8	17.4	12.9	169.5	10.9	18.6	13.5	182.9	11.6
CMT5	1291.4	8.1	6.8	265.8	6.2	13.2	12.4	300.1	12.1	16.2	14.0	325.9	13.6
CMT6	555.4	11.3	1.3	14.6	0.6	10.9	5.5	14.6	4.0	10.9	4.1	10.5	1.2
CMT7	909.7	7.2	4.4	35.1	3.4	7.0	4.8	33.2	3.0	7.4	5.7	21.8	4.5
CMT8	865.9	12.5	5.2	75.8	3.6	15.0	8.8	70.1	6.0	15.0	8.0	70.6	5.2
CMT9	1162.5	10.8	7.0	191.0	5.6	14.0	10.9	121.7	9.3	12.7	8.9	167.1	6.6
CMT10	1395.8	10.2	7.8	266.4	6.7	13.9	10.6	280.4	9.0	13.9	11.3	318.2	10.5
CMT11	1042.1	2.5	2.5	0.0	2.5	7.0	6.4	72.8	6.0	23.0	20.3	108.3	16.0
CMT12	819.6	1.7	0.9	2.2	0.9	12.2	8.4	70.2	6.7	19.7	17.2	62.7	16.6
CMT13	1541.1	3.3	1.5	75.7	1.0	3.3	1.4	84.6	1.1	3.3	1.5	68.6	1.4
CMT14	866.4	1.1	0.9	10.7	0.8	6.4	4.7	30.7	4.5	16.6	12.8	52.2	12.4
E072-04f	241.9	5.9	4.2	15.6	2.2	11.5	9.4	40.4	8.5	9.6	9.6	2.6	9.2
E076-07u	690.8	6.9	2.9	20.4	2.2	6.3	4.0	34.3	2.9	11.0	5.7	36.5	4.6
E076-08s	742.6	7.0	2.9	28.6	1.8	9.6	7.1	20.9	5.5	12.3	8.4	36.6	5.6
E135-07f	1162.9	4.8	2.7	101.6	2.5	6.0	5.0	82.9	5.0	14.7	13.8	100.3	13.8
E241-22k	666.8	14.8	13.4	421.0	12.9	23.1	22.0	449.7	21.5	26.7	24.4	485.8	23.7
E484-19k	1137.2	11.8	8.9	2056.6	8.6	17.6	16.1	2117.0	15.9	17.4	13.8	2023.4	13.3
Average	915.3	7.9	4.6	191.5	3.7	11.7	8.6	204.5	7.5	14.7	10.8	209.2	9.5

In the results of set S1 we can see that our algorithm is not far from the classical VRP solutions with results 4.6% worse in average, although it has not been designed for this problem. The results obtained for set S3 show that splitting the vehicle capacity and the customers demands in different ratios makes different routes necessary to attend all customers. This happens since one of the compartments can get full and forces the vehicle go back to depot even when other compartment still has a residual capacity. The solution of set S3 are in average 8.6% above the best-known values for the VRP. (Note that the optimal values in this case are probably higher than the best known values for the VRP.)

Looking at the results of instance set S4 we can see that when splitting the compartments in a more unbalanced way cause the total time of the routes tends to increase, which results in solutions with more routes. In this instance set, the solutions are on average 10.7% worse than the best-known values for the VRP. We can also notice a slight increase in the average time to find the best value from 204.45 to 209.23 seconds.

(REED; YIANNAKOU; EVERING, 2014) present results only for the instance *CMT1* with vehicle capacity and customers demands split in the same way as instance sets S3 and S4. They have obtained a total route length of 560.74 and 564.04 for splitting methods S3 and S4, respectively. For these instances we were able to improve their results. We obtain a total length of 553.76 in average for splitting method S3, and a total length of 556.91 in average for splitting method S4. The best found solutions were with total length of 550.62 for splitting method S3 and 549.51 for splitting method S4.

In Table 4.2 we report the results for instance set S2 and compare them with the results of (El Fallahi; PRINS; Wolfler Calvo, 2008). For each instance the table reports the best known value obtained by (El Fallahi; PRINS; Wolfler Calvo, 2008), and the relative deviations from this best known values in percent (columns "Cost") and the time to find them (columns "Time") for their Memetic Algorithm (MA) as well as their Tabu search Algorithm (TS). These are the only known results for this set of MC-VRP instances. The last two columns give the same results obtained by running our Tabu search algorithm ten times for each VRP instance with demands and capacities randomly generated as described above for instance set S2. The times reported are total execution times. In our case the execution time has been limited to $n^2/300$ seconds, for an instance with n customers. This time has been chosen to provide a fair comparison, considering that the results of (El Fallahi; PRINS; Wolfler Calvo, 2008) have been obtained on a Pentium 4 processor running at 2.4GHz. This processor is about a factor two slower than the processor of our machine. The comparison is further complicated by the fact that (El

Fallahi; PRINS; Wolfler Calvo, 2008) report the result of only a single random instance. In our experiments we have found a considerable variation of the results for different demand splittings of the same instance.

Table 4.2: Results of Tabu search on instances S2 compared to (El Fallahi; PRINS; Wolfler Calvo, 2008).

		(El Fallahi; PRINS; Wolfler Calvo, 2008)				This work	
		MA		TS			
Name	BKV	Cost	Time(s)	Cost	Time(s)	Cost	Time(s)
CMT1	556.1	0.5	17.4	0.0	15.3	-1.8	8.3
CMT2	863.6	2.9	25.5	0.0	13.9	0.9	18.5
CMT3	837.6	4.9	21.8	0.0	39.8	1.4	33.0
CMT4	1070.7	1.7	93.9	0.0	109.7	2.0	74.3
CMT5	1361.4	3.5	115.9	0.0	208.4	2.3	130.7
CMT6	563.4	1.1	16.5	0.0	10.2	-0.8	8.3
CMT7	949.0	0.6	39.2	0.0	22.0	-0.5	18.5
CMT8	916.2	4.7	18.7	0.0	18.3	-1.6	33.0
CMT9	1262.7	0.0	98.7	2.2	8.6	-4.2	74.3
CMT10	1490.2	1.3	140.2	0.0	190.3	0.2	130.8
CMT11	1122.9	0.0	47.8	7.0	27.9	1.8	47.5
CMT12	926.5	0.0	18.2	0.8	15.8	-4.4	33.0
CMT13	1542.4	0.0	76.4	2.6	21.9	1.1	47.5
CMT14	966.5	0.0	23.3	18.1	35.7	-3.5	33.0
E072-04f	262.3	0.5	11.7	0.0	5.6	-1.2	17.0
E076-07u	697.8	0.6	15.1	0.0	16.5	0.0	19.1
E076-08s	772.2	2.8	15.4	0.0	13.9	1.6	19.1
E135-07f	1233.2	0.0	47.3	0.2	51.9	1.6	59.1
E241-22k	787.8	1.1	504.5	0.0	202.9	-1.5	190.2
E484-19k	1177.3	5.4	1643.6	0.0	2122.5	5.6	770.6
Average	968.0	1.6	149.6	1.5	157.6	-0.05	88.3

On average, our tabu search is able to find results that are about 1.5% better than those of (El Fallahi; PRINS; Wolfler Calvo, 2008) in a comparable time. The actual differences in solution quality may vary for the instances used in the experiments of (El Fallahi; PRINS; Wolfler Calvo, 2008), but we observe that in 10 of the 20 instances our method consistently obtains equal or better solution values, so we expect this result to be robust. Our results show that a much simpler Tabu search can obtain comparable results, but also show that there is still a potential for an improvement. Another interesting observation is that the overall gain of about 1.5% is of the same order of the improvement that (El Fallahi; PRINS; Wolfler Calvo, 2008) obtain by allowing the splitting of routes, i.e. the demand of a customer for different product types can be satisfied by multiple vehicles.

4.4 Conclusion

We have proposed a constructive heuristic based on the savings method of (CLARKE; WRIGHT, 1964) and a Tabu search to the MCVRP-WS. We have presented results for twenty different VRP instances on four different sets of MC-VRP with instances of two compartments. Our algorithm has generated good results compared to existing algorithms, but still has potential for improvement in performance and neighborhood exploration. It would be interesting, in particular, to find a heuristic which combines the advantages of our approach and that of (El Fallahi; PRINS; Wolfler Calvo, 2008) and to study the potential gain of our method by allowing the satisfaction of customer demand for different product types in separate routes.

5 AN ITERATED TABU SEARCH FOR THE MCVRP

The heuristic we propose to solve the MCVRP is inspired by the iterated tabu search (ITS) algorithm for vehicle routing problems of (CORDEAU; MAISCHBERGER, 2012), which is considered one of the best performing metaheuristics for the CVRP (VIDAL et al., 2013). (CORDEAU; MAISCHBERGER, 2012) introduce a general heuristic that is applicable to periodic, multi-depot, or side-dependent VRPs and their variants with time windows.

An iterated tabu search (ITS) starts from some local minimum and repeatedly applies a perturbation to escape from it followed by a tabu search to find another local minimum, until some stopping criterion is satisfied. An acceptance criterion decides if the new local minimum is accepted, or the search continues from some previously visited solution. The initial local minimum can be obtained by applying any local search to an initial solution. An ITS can be seen as a generalization of an iterated local search, replacing the local search by a tabu search.

Our ITS is detailed in Algorithm 3. It constructs an initial solution, and applies the tabu search to it. Then, for I iterations, the ITS perturbs the current solution and applies the tabu search. At the end of iteration i the new solution is accepted with probability $1 - (i/I)^2$. Otherwise the search continues from the incumbent s^*, which is updated during the search (not shown in the algorithm), and returned at the end. The acceptance criterion was chosen to diversify the search at the beginning and intensify it around the best solution towards the end.

We represent a route by a sequence of visits, where in each visit a vehicle attends the demand of one or more products of a client. A client can be visited several times in different routes, but at most once per route.

The initial solution is the better of two constructions. The first is the angular sweep algorithm of (WREN, 1971; GILLETT; MILLER, 1974). We use a backward-sweep and insert single demands into the current route. If an insertion violates the length or capacity constraints a new route is created, unless the limit of the number of routes is reached. In this case all the remaining customers are inserted into the last route. The second is the savings method of (CLARKE; WRIGHT, 1964) which has been extended to handle multiple compartments. For the case of split demands, Clarke and Wright's algorithm works on each demand separately.

To perturb a solution a random client is chosen and removed from its route, to-

Algorithm 3 Iterated tabu search

```
1: function ITS( )
2:     s ← better of sweepConstruction(), ClarkeWrightConstruction()
3:     s ← tabuSearch(s)
4:     for i = 1, ..., I iterations do
5:         s' ← perturb(s)
6:         s ← tabuSearch(s')
7:         with probability (i/I)^2: s ← s*
8:     end for
9:     return the best solution s* found during search
10: end function
```

gether with its π nearest neighbors, where for each perturbation π is randomly chosen in $[0, \lceil\sqrt{n}\rceil]$. The removed clients are reinserted in a random order into the solution. Each client is inserted into the route and position within the route which minimizes the increase in the total routing cost.

We use the generalized insertion procedure (GENI) proposed by (GENDREAU; HERTZ; LAPORTE, 1992) for the traveling salesman problem and widely used in VRP heuristics to insert visits into routes or remove visits from routes. Together with the insertion or removal of a vertex, GENI applies a subset of 3-opt and 4-opt moves to the route. The selection of the edges in these moves is limited to contain one of the q-nearest vertices of the vertex to be inserted, and has time complexity $O(nq^4 + n^2)$. We apply GENI only when a complete visit is inserted or removed from a route. If a single product demand is added to an existing visit or removed from a visit which attends other demands GENI is not applied, since the route does not change.

We explain the tabu search algorithm next.

5.1 Tabu search algorithm

Tabu search is a meta-heuristic which guides a local search through the search space. It allows non-improving moves and avoids cycling by storing recent moves in a short-term memory and declaring moves that return to previous solutions tabu (GLOVER; LAGUNA, 1997). It is one of the most successful heuristics for vehicle routing problems (CORDEAU et al., 2002). The performance of a tabu search depends on the neighborhood structure, the handling of unfeasible solutions, and the design of the short-term memory.

The proposed tabu search starts from some initial solution and repeatedly moves

Algorithm 4 Tabu search.

Input: A solution s.
Output: A local minimum s^*.

```
1: α ← 1; β ← 1
2: while the incumbent improved in the last √((I − i)π) iterations do
3:     s ← bestShiftMove(s)
4:     Every n_r iterations: s ← refinement(s)
5:     updatePenalties(α, β)
6:     updateTabuList()
7: end while
8: return s*
```

to the best non-tabu neighbor. Every n_r iterations a refinement procedure is applied to the current solution. The current solution may exceed the capacity or length constraints. For a set of routes $s = \{R_1, \ldots, R_r\}$ we define its time (or distance) excess $D^+(s) = \sum_{R \in s} \max\{d(R) - D, 0\}$, and its capacity excess $C^+ = \sum_{R \in s} \max\{\max_{i \in [m]} \Delta c_i, 0\}$, for $\Delta c = c(R) - C$. The objective value of solution s then is

$$F(s) = d(s) + \alpha C^+(s) + \beta D^+(s)$$

where α and β are penalties for each unit of capacity and time excess. Initially, $\alpha = \beta = 1$. Every time the current solution exceeds the capacity or time constraints, the corresponding penalty is increased by a factor $1 + \gamma$; otherwise it is decreased by a factor $1 - \gamma$. The value of γ is chosen uniformly randomly in $[0, 1]$ at the start of the tabu search. If the value of the incumbent s^* does not improve for $\sqrt{(I - i)\pi}$ iterations the search stops. Algorithm 4 summarizes the main steps of the tabu search.

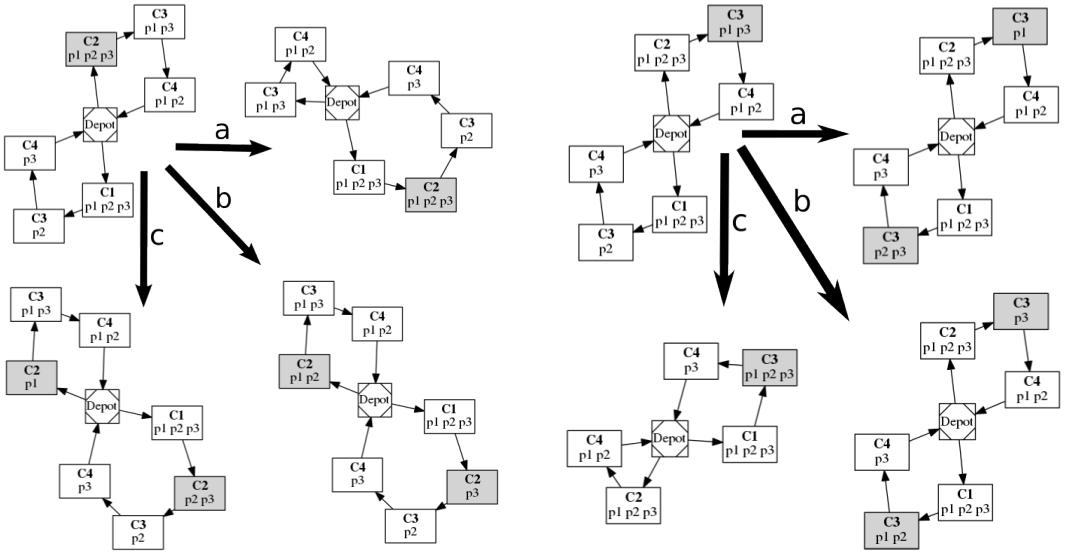
In the following sections we present the neighbourhood and the tabu list, the diversification strategy, and the intra-route refinement procedure.

5.1.1 Neighbourhood and tabu list

The neighborhood consists of a single kind of move: relocating a non-empty subset of the demands of a visit to another route. A move is defined by a tuple (s, d, v, p), where s is the source route, d is the destination route, $v \in s$ is a visit in the source route, and $p \subseteq P(v)$ is a subset of the demands attended by visit v.

If the subset p contains all demands, visit v is removed entirely from the source route s, and the GENI procedure is applied to optimize the source route. Otherwise,

Figure 5.1: Example of a move which creates a new visit in the destination route (left) and a move where the visits already exists in the destination route.



the visit is split into two demands. Demands $P(v) \setminus p$ remain in visit v in the source route s, which does not need to be optimized. The selected demands p are inserted into the destination route d. If client $V(v)$ does exist in the destination route, demands $p(v)$ are simply added to the existing visits, and the route does not need to be optimized. Otherwise, a new visit $(v(V), p)$ is created and inserted into the destination route. In this case the GENI procedure is applied to optimize the destination route. Figure 5.1 illustrates a move of all demands of a visit, and a move of a proper subset. GENI insertion procedure is applied in the moves a, b, c of the example in the left side. GENI removal procedure is applied in the move a of the example on the left side and in the move c in the example on the right.

A move is limited to insert the demands into a few nearest routes. A list of nearest routes is maintained for each client i. Let v be the total number of visits of the current solution, and $\overline{d} = d(s)/v$ the average distance between visits. Demands of client i are relocated only to routes which have a visit at a distance at most $2d$ from i. If the list of nearest routes empty, then the distance is increased by 20% until it contains at least one destination route.

When at move is applied to the current solution all solutions that have one of the demands p of customer $V(v)$ in source route s are tabu for τ iterations. The tabu tenure τ is chosen randomly in $[1, \sqrt{n\,I}]$ at the beginning of each tabu search. If a move is able to improve the incumbent it satisfies the aspiration criterion and is performed even when tabu.

Diversification

To diversify the search, we penalize frequent moves each time a local minimum is reached. If the current solution is a local minimum for F, we choose the best move according to the modified objective function

$$F'(s, M) = F(s)(1 + \zeta \sum_{p' \in p} f(V(v), p')/i) \tag{5.1}$$

for a move $M = (s, d, v, p)$ of demands p of visit v into destination route d, and current iteration i. Here $f(V(v), p')$ is the number of times demand $p' \in P$ of client $V(v)$ entered route d before, and ζ is randomly chosen from $[0, 1]$ at the beginning of each tabu search.

5.1.2 Route refinement

The tabu search algorithm also has a route refinement that is applied to each route every n_r iterations. The procedure is based on the US algorithm proposed by (GENDREAU; HERTZ; LAPORTE, 1994) for the traveling salesman problem. For each route the algorithm removes and reinserts the visit which leads to the largest reduction in route length, if any. For removals and insertions the GENI procedure with a larger neighborhood size of $q' = 2q$ is used.

5.2 Computational Experiments

In this section we report the results of computational experiments with the ITS. We first test our code without compartments on well-known instances of the VRP problem to establish a baseline compared to state-of-the-art heuristics. We next analyze the performance of ITS on instances with 2 to 5 compartments, and compare them to the results of (El Fallahi; PRINS; Wolfler Calvo, 2008), (MUYLDERMANS; PANG, 2010) and (DERIGS et al., 2010). We finally study the impact of split demands.

Table 5.1: Computational environments.

Reference	Instances	Strat.	Environment
(El Fallahi; PRINS; Wolfler Calvo, 2008)	CMT	S1,S2	Pentium 4, 2.4 GHz
(DERIGS et al., 2010)	CMT_u, GWKC_u	S1	Pentium D 3.0 GHz processor
(MUYLDERMANS; PANG, 2010)	CMT_u, GWKC_u	S1	Pentium M740, 1.73 GHz
(LAPORTE; ROPKE; VIDAL, 2014)	-	-	Core i7 2.93 GHz
(ABDULKADER; GAJPAL; ELMEKKAWY, 2015)	CMT	S3,S4	2.1 GHz processor
This work	CMT, GWKC	S1–S4	AMD FX 8150, 3.6 GHz

5.2.1 Experimental methodology

The ITS has been implemented in C++ and tested on a PC with an AMD FX-8150 processor with 8 cores running at 3.4 GHz, and with 32 GB of main memory. For the tests only one core has been used. The instances and detailed results reported in this work can be found at <www.inf.ufrgs.br/algopt/MCVRP>.

Table 5.1 summarizes the instances and division strategies used in the literature and the computing environment in which the corresponding results have been obtained. Instance set CMT_u and GWKC_u are the subsets of instances without route length constraints, which are used by (DERIGS et al., 2010) and (MUYLDERMANS; PANG, 2010). For comparing running times, we consider the environment of (El Fallahi; PRINS; Wolfler Calvo, 2008) four times slower, that of (DERIGS et al., 2010) three times slower, and that of (MUYLDERMANS; PANG, 2010) two times slower, and that of (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) about the same. These are conservative estimates, based on the PassMark benchmark.

The quality of a solution of value v is measured by its relative deviation $100(v - v^*)/v^*$ in percent from the best known value v^*. Since our algorithm is stochastic, all experiments have been replicated 10 times with different random seeds, and in the tables below we report average relative deviations over the 10 replications ("Avg.") and the relative deviation of the best solution found ("Best"), if not stated otherwise. The refinement period n_r has been set to 200. For the neighborhood size GENI values $q \in \{3, 4, 5\}$ have been tested, and we have found $q = 4$ to perform best.

5.2.2 Experiment 1: Performance on VRP instances

In our first experiment we evaluate the performance of our algorithm on standard VRP instances, and its scaling behaviour by applying division strategy S1 with $m \in$

Table 5.2: Computational results of the best metaheuristics for the VRP according to (LAPORTE; ROPKE; VIDAL, 2014).

		CMT		GWKC	
Reference	Configuration	Avg.	t(s)	Avg.	t(s)
(NAGATA; BRÄYSY, 2009)	Best of 10 runs	0.0	361	0.2	9290
(SUBRAMANIAN; UCHOA; OCHI, 2013)	Best of 10 runs	0.0	720	0.4	28130
(GROËR; GOLDEN; WASIL, 2011)	129 threads, best of 5 runs	0.0	99214	0.1	99214
(VIDAL et al., 2012)	Average of 10 runs, 50K it.	0.0	254	0.2	2419
(NAGATA; BRÄYSY, 2009)	Average of 10 runs	0.0	36	0.3	929
(MESTER; BRÄYSY, 2007)	Best configuration	0.0	62	0.3	559
(GROËR; GOLDEN; WASIL, 2011)	8 threads, best of 5 runs	0.0	2930	0.3	6050
(VIDAL et al., 2012)	Average of 10 runs, 10K it.	0.1	58	0.3	744
(GROËR; GOLDEN; WASIL, 2011)	4 threads, best of 5 runs	0.1	1465	0.4	3025
(PRINS, 2009)	-	0.1	6	0.6	166
(MESTER; BRÄYSY, 2007)	Fast configuration	0.1	1	1.2	5
(SUBRAMANIAN; UCHOA; OCHI, 2013)	Average of 10 runs	0.1	72	0.4	2813
(PISINGER; ROPKE, 2007)	Best of 10 runs, 50K it.	0.1	424	0.8	2616
(TARANTILIS, 2005)	Standard configuration	0.2	15	0.9	121
(GROËR; GOLDEN; WASIL, 2010)	Set partitioning	0.3	5	1.3	31
(GROËR; GOLDEN; WASIL, 2010)	Ejection – random	0.3	12	1.2	7
(PISINGER; ROPKE, 2007)	Average of 10 runs, 50K it.	0.3	42	1.4	261
(CORDEAU et al., 2001)	-	0.6	378	1.8	862
(TOTH; VIGO, 2003)	-	0.6	4	3.2	15

Table 5.3: Results of the ITS with 10^3, 10^4 , 10^5 and 10^6 iterations on instances with one compartment.

	10^3			10^4			10^5			10^6		
Inst.	Avg.	Best	t(s)	Avg.	Best	t(s)	Avg.	Best	t(s)	Avg.	Best	t(s)
CMT	1.9	1.1	0.2	0.9	0.4	6.8	0.4	0.2	70.1	0.2	0.1	698.9
GWKC	5.0	4.3	3.1	3.2	2.5	32.1	1.9	1.2	306.7	1.2	0.7	3035.9
Avg.	3.7	3.0	1.9	2.3	1.6	21.7	1.3	0.8	209.3	0.8	0.4	2073.6

$\{2, 3, 4, 5\}$ compartments. Table 5.2 gives an overview over the most successful heuristics for the VRP. The table comes from the recent survey of (LAPORTE; ROPKE; VIDAL, 2014) that compares more than 18 metaheuristics in terms of solution quality, speed, simplicity, flexibility and parameter sensitivity. (LAPORTE; ROPKE; VIDAL, 2014) have normalized the running times to an Intel Core i7 CPU running at 2.93 GHz. The times in our table are normalized to be comparable to our computational environment (see Table 5.1).

Table 5.3 shows the results of running our algorithm for 10^3, 10^4, 10^5, 10^6 iterations, and with only one compartment on instances CMT and GWKC. We can see that the solution quality improves with an increasing number of iterations and the running time is, as expected, proportional to the number of iterations. The solution quality obtained with 10^5 to 10^6 iterations is comparable to that of the lower third of the best heuristics for the VRP, but our implementation is somewhat slower than most of them. This may be due

Table 5.4: Results of the ITS with 10^3, 10^4 , 10^5 and 10^6 iterations on instances with two compartments and division strategy S1. Split demands are allowed.

	10^3				10^4				10^5				10^6			
Inst.	Avg.	Best	Vis.	t(s)	Avg.	Best	Vis.	t(s)	Avg.	Best	Vis.	t(s)	Avg.	Best	Vis.	t(s)
CMT	2.6	1.5	0.2	0.8	1.0	0.5	0.1	11.5	0.5	0.2	0.0	116.7	0.2	0.1	0.0	1195.0
GWKC	5.2	4.2	0.1	5.2	3.5	2.6	0.1	51.7	2.1	1.5	0.0	489.5	1.2	0.8	0.0	4681.1
Avg.	4.1	3.1	0.1	3.4	2.5	1.7	0.1	35.2	1.5	1.0	0.0	336.0	0.8	0.5	0.0	3245.7

Table 5.5: Results of the ITS with 10^5 iterations and two to five compartments and splitting strategy S1. Split demands are allowed.

	$m = 2$				$m = 3$				$m = 4$				$m = 5$			
Inst.	Avg.	Best	Vis.	t(s)	Avg.	Best	Vis.	t(s)	Avg.	Best	Vis.	t(s)	Avg.	Best	Vis.	t(s)
CMT	0.6	0.2	0.0	137.7	0.6	0.2	0.0	253.2	0.9	0.3	0.1	497.6	0.9	0.3	0.1	999.8
GWKC	1.7	1.1	0.1	538.2	2.0	1.2	0.4	940.3	2.0	1.3	0.5	1777.8	2.2	1.4	0.8	3475.9
Avg.	1.2	0.8	0.0	373.3	1.4	0.8	0.2	657.4	1.5	0.9	0.3	1250.7	1.7	1.0	0.5	2456.3

to some missing optimizations, and is partly due to the overhead for considering multiple compartments. Since our algorithm was not designed for the standard VRP, and must handle multiple compartments and multiple visits, we consider these results demonstrate a reasonable baseline performance.

5.2.3 Experiment 2: Performance on MCVRP instances

In our second experiment, we evaluate the performance our algorithm and its scaling behaviour on the MCVRP instances generated from instances CMT and GWKC using strategy S1, with two to five compartments. For the most common case of two compartments, we have run experiments with 10^3, 10^4, 10^5, and 10^6 iterations. For the remaining cases we have fixed the number of iterations at 10^5. The results are shown in Tables 5.4 and 5.5. Since split demands are allowed, we additionally present the average relative deviation of the number visits from the number of customers.

Comparing the results for one compartment in Table 5.3 to those with two compartments we find that the performance of the algorithm is about the same for the same number of iterations. The running time however, increases by about 70%. This comes from the additional handling of the multiple compartments. We can also see that allowing multiple visits seems to have no or little advantage for instances generated by strategy S1, since the average percentage of additional visits is small and tends to zero, for an increasing number of iterations.

These observations continue to hold for the results with more than two compartments in Table 5.5. For each additional compartment, the average and best solution quality increases slightly, and the running approximately doubles. The increase in running can be explained by the additional combinations of subsets of demands that must be considered. For a practical number of compartments, the running times still remain acceptable. However, for a large number of compartments, or a tight time budget, heuristic methods for selecting demands of compartments when splitting would be necessary. As in the case of two compartments, the results show that for the instances generated with strategy S1 there seems to be very little advantage of split demands.

5.2.3.1 Comparison to the results from the literature

In Table 5.6 we compare our results to those of (El Fallahi; PRINS; Wolfler Calvo, 2008), (MUYLDERMANS; PANG, 2010), and (DERIGS et al., 2010) for the instances with two products obtained by division strategy S1. The table shows the average relative deviation and the average time for each instance, which was solved by at least one approach. For the missing entries ("–") no results were reported. In the case of (MUYLDERMANS; PANG, 2010) and (DERIGS et al., 2010) the missing entries are the instances with route length restrictions. The table also provides averages for different groups. Group 1 contains the instances for which all authors report results, Group 2 the instances used by (El Fallahi; PRINS; Wolfler Calvo, 2008), and Group 3 those used by (MUYLDERMANS; PANG, 2010) and (DERIGS et al., 2010). All approaches, except that of (El Fallahi; PRINS; Wolfler Calvo, 2008) respect the fleet size, so we do not report the number of vehicles in the solution. For completeness we also provide overall averages.

We find that ITS with 10^4 iterations performs better than both the tabu search and the memetic algorithm of (El Fallahi; PRINS; Wolfler Calvo, 2008) and is about a factor 4 faster. This holds overall and for all 16 instances individually. Exceptions are CMT5, GWKC12, which take more iterations, but are still faster with a mean time of $69\,\mathrm{s}$ and $185\,\mathrm{s}$, respectively, to reach the same relative deviation, and CMT13, where ITS produces solution with a relative deviation of 0.2% higher in the same time.

Algorithm GLS of (MUYLDERMANS; PANG, 2010) quickly finds good solutions, but improves little over time: the mean relative deviation of 3.6% for 300K iterations improves with 1200K iterations only to 3.0%. We can also see that the time per iteration increases only slightly with the number of clients, which leads to higher relative

Table 5.6: Comparison to the results of (El Fallahi; PRINS; Wolfler Calvo, 2008), (MUYLDERMANS; PANG, 2010) and (DERIGS et al., 2010).

		MA		TS		GLS 300K		GLS 600K		GLS 1200K		5min	10min	20min	60min	10^3		10^4		10^5		10^6	
Inst.	BKV	Avg.	t(s)	Avg.	t(s)	Avg.	t(s)	Avg.	t(s)	Avg.	t(s)	Avg.	Avg.	Avg.	Avg.	Avg.	t(s)	Avg.	t(s)	Avg.	t(s)	Avg.	t(s)
CMT1	524.6	0.0	5.9	0.0	4.0	0.0	61.8	0.0	123.6	0.0	269.1	0.2	0.2	0.0	0.0	0.0	0.0	0.0	5.8	0.0	61.5	0.0	626.9
CMT2	835.3	0.9	13.8	2.0	5.4	0.9	63.9	0.3	126.9	0.3	258.3	0.5	0.4	0.3	0.2	2.9	0.1	0.5	8.2	0.1	83.0	0.0	831.7
CMT3	826.1	3.3	15.6	1.1	23.8	0.4	69.9	0.4	139.5	0.4	282.3	0.5	0.4	0.3	0.2	1.7	1.0	0.5	9.7	0.3	99.3	0.2	986.7
CMT4	1028.4	4.1	51.8	2.6	102.8	1.1	71.4	1.1	147.9	1.1	303.3	1.2	0.9	0.7	0.4	4.3	1.0	2.3	14.1	1.4	146.5	0.4	1449.2
CMT5	1291.3	3.0	102.6	4.5	202.3	3.0	75.3	2.1	148.8	1.8	306.0	2.7	2.5	2.1	1.7	5.6	2.0	3.7	23.0	2.1	228.0	1.2	2265.3
CMT6	555.4	0.7	4.8	0.8	3.5	–	–	–	–	–	–	–	–	–	–	0.3	0.0	0.0	4.3	0.0	49.8	0.0	488.0
CMT7	909.7	2.0	11.5	3.0	13.5	–	–	–	–	–	–	–	–	–	–	0.8	0.3	0.3	8.5	0.0	86.0	0.0	869.6
CMT8	866.0	3.0	14.2	3.3	26.3	–	–	–	–	–	–	–	–	–	–	1.3	0.3	1.0	8.8	0.5	91.3	-0.0	907.5
CMT9	1162.5	3.2	39.3	1.9	98.9	–	–	–	–	–	–	–	–	–	–	5.3	1.0	1.6	14.8	0.5	152.0	0.2	1515.3
CMT10	1395.8	4.7	125.2	5.1	98.9	–	–	–	–	–	–	–	–	–	–	4.8	2.0	2.2	21.2	1.6	212.2	1.1	2305.4
CMT11	1042.1	0.2	31.6	0.2	16.2	0.6	62.7	0.6	121.2	0.6	249.9	0.1	0.1	0.1	0.0	0.6	1.0	0.2	13.0	0.1	130.6	0.0	1317.0
CMT12	819.6	0.0	19.2	0.3	8.7	0.0	64.8	0.0	136.2	0.0	279.3	0.0	0.0	0.0	0.0	0.1	0.0	0.0	6.3	0.0	65.8	0.0	730.7
CMT13	1541.1	0.0	34.9	0.1	41.9	–	–	–	–	–	–	–	–	–	–	7.6	1.0	2.2	10.6	0.1	97.7	0.1	1038.1
CMT14	866.4	0.0	13.9	0.1	10.5	–	–	–	–	–	–	–	–	–	–	0.6	1.0	0.0	12.4	0.0	130.1	0.0	1399.1
GWKC9	579.7	–	–	–	–	6.0	99.9	5.2	193.2	4.5	391.8	5.2	4.1	3.2	2.6	6.9	2.9	4.7	32.4	2.4	318.6	1.1	2698.7
GWKC10	735.7	–	–	–	–	5.8	107.4	4.8	207.3	4.8	424.2	5.9	5.1	4.0	3.0	8.3	3.6	5.2	39.0	3.0	403.2	1.3	3548.5
GWKC11	912.0	–	–	–	–	6.1	118.5	5.9	229.8	5.9	470.1	7.6	6.1	5.2	4.3	9.2	4.7	5.5	50.7	3.7	524.9	1.5	4682.3
GWKC12	1101.5	9.3	514.0	4.3	1614.9	9.2	137.4	7.3	269.1	6.5	543.9	8.4	7.5	6.2	4.9	8.2	6.7	5.2	62.1	3.4	622.5	1.8	5938.3
GWKC13	857.2	–	–	–	–	4.4	73.2	4.2	147.6	3.2	302.7	4.2	3.5	3.3	2.8	5.9	3.0	3.6	32.7	2.0	309.4	1.4	3077.2
GWKC14	1080.5	–	–	–	–	5.1	76.2	5.0	151.8	4.2	309.6	4.7	4.1	3.5	3.2	6.0	4.0	4.0	42.6	2.4	397.5	1.3	3914.6
GWKC15	1337.9	–	–	–	–	5.1	81.3	5.1	158.7	5.1	323.4	5.7	4.5	4.1	3.8	7.2	5.4	4.1	52.5	2.3	499.2	1.6	4861.9
GWKC16	1611.6	–	–	–	–	5.5	86.1	5.2	169.8	5.0	340.5	6.9	6.1	5.1	4.6	5.3	6.8	4.2	67.8	3.1	652.3	1.6	6227.9
GWKC17	707.8	5.1	76.6	4.7	201.4	2.2	79.8	1.7	159.3	1.7	329.1	2.7	2.2	1.9	1.7	6.0	3.6	2.8	35.4	0.8	299.0	0.3	2796.1
GWKC18	995.1	–	–	–	–	4.9	87.6	4.7	168.9	4.1	351.0	3.1	2.2	2.0	1.3	5.7	6.7	4.6	59.9	2.3	442.9	1.6	4030.3
GWKC19	1365.6	–	–	–	–	3.3	90.6	3.3	180.9	2.9	369.0	3.8	3.4	3.2	2.3	6.4	9.8	5.0	105.6	2.0	853.0	0.9	6768.6
GWKC20	1817.6	–	–	–	–	4.5	99.9	4.4	198.6	4.4	404.4	5.8	4.8	4.2	3.0	7.3	8.8	5.5	69.4	3.2	645.7	1.4	6313.2
Avg. grp. 1	844.4	1.9	35.9	1.7	63.2	0.9	67.9	0.7	136.3	0.7	282.9	0.9	0.8	0.6	0.5	2.3	1.0	1.1	13.5	0.5	130.6	0.2	1292.3
Avg. grp. 2	967.1	2.5	67.2	2.1	154.6	–	–	–	–	–	–	–	–	–	–	3.1	1.3	1.4	16.1	0.7	159.7	0.3	1591.6
Avg. grp. 3	1024.7	–	–	–	–	3.6	84.6	3.2	167.3	3.0	342.5	3.6	3.1	2.6	2.1	5.1	3.7	3.3	38.4	1.8	357.0	0.9	3319.2
Avg.	1029.5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	4.5	3.0	2.7	31.2	1.4	292.4	0.7	2753.4

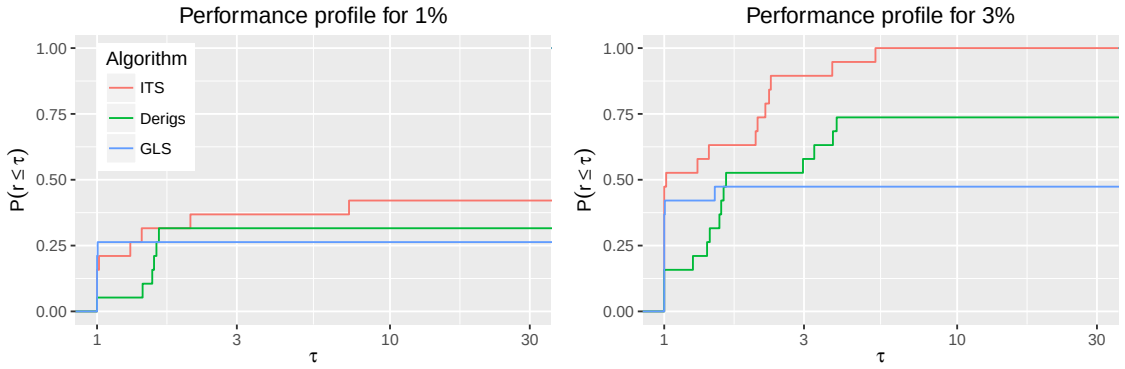
deviations in larger instances. ITS with 10^4 iterations has a smaller average relative deviation compared to GLS with 300K iterations, in about half the time, and finds solutions of comparable quality to GLS with 600K iterations, a factor four faster. GLS with 1200K and ITS with 10^5 iterations run in a comparable time, but ITS finds solutions with an average relative deviation of 1.8% compared to 3.0% of GLS. These observations hold also for most the of individual instances. Exceptions are instances CMT4,5, and GWCK19,20, where ITS takes in some of the time scales a factor of two to five longer to reach the same solution quality. Since (MUYLDERMANS; PANG, 2010) find solutions of average relative deviation of 0.79% for standard CVRP instances, the performance of their approach seems not to transfer well to multiple compartments.

(DERIGS et al., 2010) present results for 5, 10, 20, and 60 minutes, which correspond to 100, 200, 400 and 1200 seconds, respectively, in our environment. Looking at the runs with 5 and 20 minutes, which can be safely compared to the runs of ITS with 10^4 and 10^5 iterations, we see that ITS finds for the majority of the instances better solutions in less time, especially on the large instances. The average time for ITS to reach the same solution quality as the runs with 5, 10, 20, and 60 minutes, is 60, 137, 171, and 302 seconds, respectively. Exceptions to the average are instances CMT4,5,11 and GWKC18,19, which need in some of the time scales up to a factor five more time. Also, even after 10^6 iterations ILS does not reach the solution quality of 1.3% for GWKC18 that the algorithm of (DERIGS et al., 2010) obtains after 60 minutes.

In summary, for each time scale and algorithm considered, ITS finds better values in less time in average, and also for the majority of the individual instances. Exceptions are some of the results of (MUYLDERMANS; PANG, 2010) for 300K iterations, where GLS finds solutions faster, and particularly instances CMT4,5 and GWKC17-19, for which other algorithms are frequently faster.

Finally, we offer a summarized view of the results using performance profiles (DOLAN; MORE, 2002) in Figure 5.2. The plots show the probability $P(r \leq \tau)$ of reaching a fixed quality in a time at most a factor τ slower than the fastest algorithm. In particular, for $\tau = 1$ we obtain the probability that the algorithm is the fastest, and for $\tau \to \infty$ the probability of reaching the target quality at all. We have excluded the algorithm of (El Fallahi; PRINS; Wolfler Calvo, 2008), since they report only one time scale, and the shorter runs of ITS with 10^3 and 10^4 iterations, since we have no data for the other algorithms on these short time scales. Consistent with our observations, we can see that ITS, for each τ has a higher probability of finding a solution of a given quality, except for a relative deviation

Figure 5.2: Performance profiles for GLS of (MUYLDERMANS; PANG, 2010), the algorithm of (DERIGS et al., 2010), and ITS, for reaching a relative deviation of 1% (left) and 3% (right).



of $1\,\%$ and a short time scale, where GLS has a slightly higher probability.

5.2.4 Experiment 3: The single-visit MCVRP

In this section we focus on the single-visit MCVRP (or MCVRP without splitting, MCVRP-WS), a variant of the MCVRP where the demand of the clients for all product types must be attended in a single visit. We compare our algorithm in this setting to the two current best approaches from the literature: the MA and TS of (El Fallahi; PRINS; Wolfler Calvo, 2008), and the hybridized ant colony algorithm HCA of (ABDULKADER; GAJPAL; ELMEKKAWY, 2015). Following the literature, in this section limit on the fleet size have been ignored.

The comparison to the results of (El Fallahi; PRINS; Wolfler Calvo, 2008) on instances with division strategy S2 is shown in Table 5.7. For their MA and TS it reports the relative deviation from the best known values ("R.d.") of the single run provided by the authors and the run time in seconds ("t(s)"). The best known values are those reported by (El Fallahi; PRINS; Wolfler Calvo, 2008). Thus, negative relative deviations indicate new best solutions.

(El Fallahi; PRINS; Wolfler Calvo, 2008) find significant gains for multiple visits compared to single ones. For ITS such differences exist, but are much smaller. Gains are strongly correlated with more visits, and for a subset of instances, e.g. CMT11, gains remain significant. It is evident from the negative relative deviations that ITS for both single and multiple visits, finds almost always better solutions, in comparable, often shorter running time. (El Fallahi; PRINS; Wolfler Calvo, 2008) report the value of a single run only, however even the worst of our 10 replications is better than the best known value

Table 5.7: Comparison to the results of (El Fallahi; PRINS; Wolfler Calvo, 2008) on instances with division strategy S2.

		MA					TS					ITS						
		Single		Mult.			Single		Mult.			Single			Mult.			
Inst.	BKV	R.d.	t(s)	R.d.	t(s)	Vis.	Avg.	t(s)	Avg.	t(s)	Vis.	Avg.	Best	t(s)	Avg.	Best	Vis.	t(s)
CMT1	548.4	1.9	4.3	0.0	5.6	2.0	1.4	3.8	0.4	3.2	0.0	-2.5	-4.1	5.0	-2.3	-4.1	2.6	6.0
CMT2	863.6	2.9	6.4	1.3	11.3	0.0	0.0	3.5	1.2	14.0	0.0	-0.6	-1.4	6.8	-0.5	-1.9	4.8	9.8
CMT3	832.9	5.5	5.5	1.3	23.0	0.0	0.6	9.9	0.0	39.7	0.0	-0.0	-0.4	11.2	0.2	-0.4	1.1	15.3
CMT4	1070.7	1.7	23.5	0.7	39.7	0.0	0.0	27.4	0.5	86.7	0.7	-0.8	-2.0	16.0	-0.6	-1.8	2.3	23.3
CMT5	1361.4	3.5	29.0	0.5	121.8	0.0	0.0	52.1	0.1	97.7	0.0	-0.6	-1.4	22.8	-0.3	-1.2	3.3	32.9
CMT6	558.6	1.9	4.1	0.0	6.8	0.0	0.9	2.5	0.0	7.5	0.0	-0.5	-0.6	5.8	-0.5	-0.6	0.0	5.0
CMT7	949.0	0.6	9.8	1.1	4.5	0.0	0.0	5.5	0.4	8.4	2.7	-2.7	-3.7	7.0	-2.8	-3.6	1.1	10.5
CMT8	890.1	7.7	4.7	0.0	25.3	1.0	2.9	4.6	0.5	28.3	0.0	-1.5	-2.7	9.7	-1.8	-2.7	0.2	10.3
CMT9	1186.2	6.5	24.7	3.3	38.5	0.0	8.8	2.1	0.0	77.6	0.7	-0.2	-1.0	12.5	-0.2	-1.1	0.1	18.9
CMT10	1475.8	2.3	35.0	0.0	103.1	0.0	1.0	47.6	1.7	92.0	1.0	-2.4	-3.3	16.6	-2.5	-3.4	0.3	26.2
CMT11	1113.4	0.9	11.9	0.0	23.7	1.7	7.9	7.0	3.8	19.9	0.8	-0.2	-5.4	11.0	-2.0	-5.8	3.8	13.6
CMT12	906.9	2.2	4.5	0.0	8.4	1.0	3.0	4.0	0.0	22.8	0.0	-1.9	-4.4	12.1	-2.1	-4.4	2.4	16.2
CMT13	1541.2	0.1	19.1	0.0	35.3	0.8	2.7	5.5	0.1	41.1	1.7	0.3	0.1	7.8	1.8	0.1	0.0	12.6
CMT14	934.7	3.4	5.8	0.0	15.4	1.0	22.1	8.9	0.7	13.3	0.0	-0.6	-2.2	13.0	-0.3	-2.1	0.3	9.1
GWKC12	1175.5	5.6	410.9	4.1	386.5	0.4	0.2	530.6	0.0	969.6	0.2	-0.4	-1.1	49.8	1.0	-0.2	2.8	77.9
GWKC17	771.2	3.3	126.1	2.8	147.1	0.4	2.2	50.7	0.0	229.8	0.4	-2.5	-3.6	23.0	-3.2	-4.3	4.9	33.2
E072-04f	262.3	0.5	2.9	0.1	3.1	0.0	0.0	1.4	0.2	3.4	0.0	-3.5	-5.1	7.2	-5.0	-6.8	3.2	7.3
E076-07u	697.8	0.6	3.8	1.1	12.7	0.0	0.0	4.1	0.2	5.3	0.0	-0.3	-0.6	7.3	-0.2	-0.7	0.4	10.1
E076-08s	748.4	6.0	3.9	1.1	10.7	1.3	3.2	3.5	0.0	13.1	0.0	0.7	-0.1	7.2	0.2	-0.4	2.3	9.4
E135-07f	1233.2	0.0	11.8	1.9	17.9	1.5	0.2	13.0	1.2	65.1	0.0	-0.1	-2.2	18.5	-1.0	-2.6	3.3	22.7
Avg.	956.1	2.8	37.4	1.0	52.0	0.6	2.8	39.4	0.5	91.9	0.4	-1.0	-2.3	13.5	-1.1	-2.4	2.0	18.5

in 25 of the 38 instances, so it is unlikely that the better performance of ITS is due to a particularly bad run of the MA or the TS.

The comparison to the results of (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) on instances with division strategies S3 and S4 is shown in Tables 5.8 and 5.9, which report the same values as the previous table. (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) also report only the value of a single run.

The relative deviations of HAC for both strategies are 0%, since they represent the current best known values, except instance CMT12 with division strategy S3, where the best known value has been found by (REED; YIANNAKOU; EVERING, 2014). In both cases ITS finds in nine instances better solutions. For some instances, e.g. CMT7,8,14 the relative deviations are slightly higher, and CMT12 for strategy S4 has the highest deviation of 1.5%. ITS reaches optimal values for these instances with a factor of at most 3 more time, and often faster. For example, the average time to find the best known value for CMT12 and strategy S4 is 74s. In the remaining instances, the running times of ITS are always shorter, some markedly, e.g. for instances CMT9,10, and in particular on the larger instances. This suggests that ITS scales better. The results for ITS with multiple

Table 5.8: Comparison of the results of (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) on instances with division strategy S3 to ITS with single and multiple visits and 10^4 iterations.

		HAC		ITS						
				Single			Multiple			
Inst.	BKV	R.d.	t(s)	Avg.	Best	t(s)	Avg.	Best	Vis.	t(s)
CMT1	550.7	0.0	5	0.1	0.0	4.7	0.0	0.0	0.0	6.4
CMT2	890.7	0.0	15	-1.8	-2.1	6.5	-1.0	-2.1	1.2	9.2
CMT3	874.1	0.0	40	-0.5	-0.8	9.9	-0.1	-0.6	0.3	15.0
CMT4	1126.1	0.0	146	-2.2	-3.0	15.5	-1.7	-2.4	0.7	22.8
CMT5	1444.3	0.0	257	-3.3	-3.9	23.0	-2.9	-3.5	1.4	32.6
CMT6	557.5	0.0	11	0.0	0.0	3.8	0.0	0.0	0.0	4.8
CMT7	928.2	0.0	28	0.5	-0.2	6.8	0.0	-0.2	0.0	9.9
CMT8	883.0	0.0	93	0.1	-0.7	7.9	1.4	-0.7	0.4	10.8
CMT9	1228.9	0.0	326	-1.7	-3.0	14.6	-1.2	-2.1	0.5	17.0
CMT10	1511.7	0.0	624	-2.1	-3.1	15.8	-2.5	-3.8	0.9	25.0
CMT11	1110.5	0.0	75	-0.2	-0.4	11.8	-0.1	-0.4	1.1	14.7
CMT12	911.9	0.1	15	-0.5	-0.7	12.0	-0.2	-0.6	0.7	15.4
CMT13	1556.5	0.0	117	-0.5	-0.8	7.6	1.5	-0.7	0.0	11.4
CMT14	911.4	0.0	34	0.1	-0.0	11.9	0.1	0.0	0.7	8.4
Avg.	1034.7	0.0	128	-0.9	-1.3	10.8	-0.5	-1.2	0.6	14.5

visits are about the same. Instances with division strategy S3 seem to gain nothing from multiple visits, while the average for instances with division strategy S4 is slightly better when multiple visits are allowed. Indeed, for runs with 10^4 the improvement with multiple runs reaches 0.5% for strategy S4.

5.3 Conclusion

Vehicle routing problems with multiple compartments have important practical applications but have been little studied in the literature. In this work we have proposed an iterated tabu search to solve this problem with single and multiple visits and have compared it to existing heuristic algorithms. Different from previous approaches, demands for different product types are can be attended in multiple visits without the overhead of representing each demand separately.

We have shown that the ITS has a reasonable baseline performance on the VRP, and dominates the existing approaches in average and in the majority of the instances in solution quality and time on different sets of instances. Although the approaches of (MUYLDERMANS; PANG, 2010) and (DERIGS et al., 2010) may have some overhead for being general frameworks for different variants of the VRP, additional time does not help to improve the results significantly, while ITS makes a good progress over time. The

Table 5.9: Comparison of the results of (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) on instances with division strategy S4 to ITS with single and multiple visits and 10^4 iterations.

		HAC		ITS						
				Single			Multiple			
Inst.	BKV	R.d.	t(s)	Avg.	Best	t(s)	Avg.	Best	Vis.	t(s)
CMT1	551.9	0.0	5	-0.7	-0.7	4.8	-0.7	-0.7	0.8	5.9
CMT2	919.0	0.0	14	-3.4	-4.2	6.6	-4.5	-5.2	7.5	8.4
CMT3	895.3	0.0	44	-2.7	-3.3	9.5	-4.2	-5.4	5.6	10.1
CMT4	1159.5	0.0	151	-3.7	-5.0	16.0	-3.3	-4.5	7.7	17.6
CMT5	1525.9	0.0	236	-6.6	-7.5	19.1	-7.6	-8.5	6.3	22.0
CMT6	559.4	0.0	10	-0.7	-0.7	4.8	-0.7	-0.7	0.0	4.6
CMT7	932.7	0.0	26	0.3	-0.0	6.7	0.1	-0.1	1.5	10.3
CMT8	884.9	0.0	95	0.8	-1.1	8.4	0.3	-1.1	1.0	10.7
CMT9	1226.6	0.0	333	-1.5	-2.9	15.2	-1.3	-2.2	0.5	17.0
CMT10	1526.0	0.0	620	-2.7	-3.3	15.9	-2.6	-3.4	1.2	25.4
CMT11	1221.7	0.0	87	0.8	-1.5	11.8	-0.9	-2.0	14.9	14.6
CMT12	950.8	0.0	30	1.5	1.3	13.3	1.1	0.2	5.1	15.1
CMT13	1550.1	0.0	123	-0.1	-0.4	7.8	3.7	-0.3	0.0	11.6
CMT14	965.8	0.0	38	0.4	0.1	13.4	0.6	0.2	0.6	10.3
Avg.	1062.1	0.0	129	-1.3	-2.1	10.9	-1.4	-2.4	3.8	13.1

results of the ITS also suggest that the value of multiple visits is rather limited, especially when considering the additional algorithmic effort needed to permit them. It is open what kind of instance structure makes multiple visits attractive.

6 CONCLUSION

In this dissertation we have studied the multi-compartment vehicle routing problem (MCVRP), that is a variant of the classical vehicle routing problem (VRP), where the customers have demands for different product types that must be kept separated during the transportation. The VRP is a well studied problem due to its strong practical application and many variants of this problem has been proposed in over 50 years of research. We have presented a study of the best performing metaheuristics for the VRP. Based on this existing methods for the VRP we were able to design an algorithm for the MCVRP.

The MCVRP introduces the co-collection or co-delivery of different types of products to the classical VRP. This problem has a large applicability in different industries such as: petrol distribution, waste collection, livestock collection and groceries delivery. In our study of the heuristics available in the literature, we have concluded that they do not explore well the multiple visit attribute of this problem which motivates this work.

We have proposed an iterated tabu search for the MCVRP with single and multiple visits and we have tested it against instances from 50 to 483 customers with demands for up to 5 different product types. The results show that our algorithm is able find better solutions in less computational time compared with the existing approaches. The principal contribution of the proposed heuristic is that different product types can be attended in multiple visits without the overhead of representing each demand separately.

6.1 Future work

A question that remain open after this work is what kind of instance structure makes multiple visits attractive compared to single visit. This is interesting for real applications purposes because if we know such instances we do not need to spend computational effort searching for multiple visits solutions and instead focus in find better single visit solutions.

Current studies on the MCVRP cannot be immediately applied to waste collection system because of the lack of realistic constraints (LU et al., 2015). Then, a future work is to understand these constraints and implement in the algorithm proposed to contribute to this real world problem.

In this study we consider the vehicles where each compartment is dedicated for one product type. A suggestion of work is to extend the proposed algorithm to allow

the products to be loaded in any compartment respecting some loading constraint. Other possible extensions are allow compartments with variable sizes (DERIGS et al., 2010) and stochastic customer demands (MENDOZA et al., 2010).

REFERENCES

ABDULKADER, M. M.; GAJPAL, Y.; ELMEKKAWY, T. Y. Hybridized ant colony algorithm for the Multi Compartment Vehicle Routing Problem. **Applied Soft Computing**, Elsevier B.V., v. 37, p. 196–203, 2015. ISSN 15684946.

ARCHETTI N. BIANCHESSI, M. G. S. C. A Column Generation Approach for the Split Delivery Vehicle Routing Problem. **European Journal of Operational Research**, v. 47, n. 1, p. 26–36, 2014.

AVELLA, P.; BOCCIA, M.; SFORZA, A. Solving a fuel delivery problem by heuristic and exact approaches. **European Journal of Operational Research**, v. 152, n. 1, p. 170–179, jan. 2004. ISSN 03772217.

BEASLEY, J. E. Route-first cluster-second methods for vehicle routing. **Omega**, v. 11, p. 403–408, 1983.

BENANTAR, A.; OUAFI, R. Optimization of vehicle routes: an application to logistic and transport of the fuel distribution. In: **Proc. 9th Int. Conf. Modeling, Optim., and Simulation**. Bordeaux, France: [s.n.], 2012.

CHAJAKIS, E. D.; GUIGNARD, M. Scheduling deliveries in vehicles with multiple compartments. **Journal of Global Optimization**, v. 26, n. 1, p. 43–78, 2003. ISSN 09255001.

CHEN, P.; HUANG, H.-k.; DONG, X.-Y. Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem. **Expert Systems with Applications**, v. 37, n. 2, p. 1620–1627, mar 2010. ISSN 09574174.

CHRISTOFIDES, N.; MINGOZZI, A.; TOTH, P. Combinatorial optimization. In: ____. [S.l.]: John Wiley, Chichester, 1979. cap. The vehicle routing problem, p. 315–338.

CLARKE, G. u.; WRIGHT, J. Scheduling of vehicles from a central depot to a number of delivery points. **Operations research**, INFORMS, v. 12, n. 4, p. 568–581, 1964.

COELHO, L. C.; LAPORTE, G. Classification, models and exact algorithms for multi-compartment delivery problems. **European Journal of Operational Research**, v. 242, n. 3, p. 854–864, maio 2015. ISSN 03772217.

CORDEAU, J.-F. et al. A guide to vehicle routing heuristics. **Journal of the Operational Research Society**, v. 53, n. 5, p. 512–522, maio 2002. ISSN 0160-5682.

CORDEAU, J.-F. et al. A unified tabu search heuristic for vehicle routing problems with time windows. **Journal of the Operational research society**, Palgrave Macmillan, v. 52, n. 8, p. 928–936, 2001.

CORDEAU, J.-F.; MAISCHBERGER, M. A parallel iterated tabu search heuristic for vehicle routing problems. **Computers & Operations Research**, v. 39, n. 9, p. 2033–2050, set. 2012. ISSN 03050548.

CORNILLIER, F.; BOCTOR, F.; RENAUD, J. Heuristics for the multi-depot petrol station replenishment problem with time windows. **European Journal of Operational Research**, Elsevier B.V., v. 220, n. 2, p. 361–369, jul. 2012. ISSN 03772217.

CORNILLIER, F. et al. A heuristic for the multi-period petrol station replenishment problem. **European Journal of Operational Research**, v. 191, n. 2, p. 295–305, dez. 2008. ISSN 03772217.

CVRPLIB. **Capacitated Vehicle Routing Problem Library**. 2016. http://vrp.atd-lab.inf.puc-rio.br/index.php/en/plotted-instances. July 16, 2016.

DANTZIG, G. B.; RAMSER, J. H. The Truck Dispatching Problem. **Management Science**, v. 6, n. 1, p. 80–91, 1959.

DERIGS, U. et al. Vehicle routing with compartments: applications, modelling and heuristics. **OR Spectrum**, v. 33, n. 4, p. 885–914, fev. 2010. ISSN 0171-6468.

DOERNER, K. F. et al. Parallel Ant Systems for the Capacitated Vehicle Routing Problem. In: GOTTLIEB, J.; RAIDL, G. R. (Ed.). **Evolutionary Computation in Combinatorial Optimization: 4th European Conference, EvoCOP 2004, Coimbra, Portugal, April 5-7, 2004. Proceedings**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 72–83. ISBN 978-3-540-24652-7.

DOLAN, E. D.; MORE, J. J. Benchmarking optimization software with performance profiles. **Math. Prog. A**, v. 91, p. 201–213, 2002.

DORIGO, M.; CARO, G. D.; GAMBARDELLA, L. M. Ant algorithms for discrete optimization. **Artificial life**, MIT Press, v. 5, n. 2, p. 137–172, 1999.

El Fallahi, A.; PRINS, C.; Wolfler Calvo, R. A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. **Comput. Oper. Res.**, v. 35, n. 5, p. 1725–1741, maio 2008. ISSN 03050548.

ELBEK, M.; WØHLK, S. A variable neighborhood search for the multi-period collection of recyclable materials. **European Journal of Operational Research**, v. 249, n. 2, p. 540–550, 2016. ISSN 03772217.

GENDREAU, M.; HERTZ, A.; LAPORTE, G. New insertion and postoptimizaton procedures for the traveling salesman problem. **Manag. Sci.**, v. 40, n. 10, p. 1086–1094, 1992.

GENDREAU, M.; HERTZ, A.; LAPORTE, G. A tabu search heuristic for the vehicle routing problem. **Management Science**, v. 40, n. 10, p. 1276–1290, 1994.

Gendreau, Michel and Hertz, Alain and Laporte, G. **NEW INSERTION AND POSTOPTIMIZATION PROCEDURES FOR THE TRAVELING SALESMAN PROBLEM**. [S.l.]: Operations Research, 1992. 6 p.

GILLETT, B. E.; MILLER, L. R. A heuristic algorithm for the vehicle-dispatch problem. **Oper. Res.**, v. 22, n. 2, p. 340–349, 1974.

GLOVER, F.; LAGUNA, M. **Tabu Search**. [S.l.]: Kluwer, 1997.

GOLDEN, B. L. et al. The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In: **Fleet management and logistics**. [S.l.]: Springer, 1998. p. 33–56.

GROËR, C.; GOLDEN, B.; WASIL, E. A library of local search heuristics for the vehicle routing problem. **Mathematical Programming Computation**, Springer, v. 2, n. 2, p. 79–101, 2010.

GROËR, C.; GOLDEN, B.; WASIL, E. A parallel algorithm for the vehicle routing problem. **INFORMS Journal on Computing**, INFORMS, v. 23, n. 2, p. 315–330, 2011.

HENKE, T.; SPERANZA, M. G.; WÄSCHER, G. The multi-compartment vehicle routing problem with flexible compartment sizes. **European Journal of Operational Research**, v. 246, n. 3, p. 730–743, nov 2015. ISSN 03772217.

KYTÖJOKI, J. et al. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. **Computers & Operations Research**, v. 34, n. 9, p. 2743–2757, sep 2007. ISSN 03050548.

LAHYANI, R. et al. A multi-compartment vehicle routing problem arising in the collection of olive oil in tunisia. **Omega**, v. 51, n. 0, p. 1 – 10, 2015. ISSN 0305-0483.

LAPORTE, G.; ROPKE, S.; VIDAL, T. Heuristics for the Vehicle Routing Problem. In: TOTH, P.; VIGO, D. (Ed.). **Vehicle Routing: Problems, Methods, and Applications**. [S.l.]: SIAM, 2014. cap. 4, p. 87–112.

LIN, S. Computer solutions of the traveling salesman problem. **Bell System Technical Journal, The**, Alcatel-Lucent, v. 44, n. 10, p. 2245–2269, 1965.

LU, J.-W. et al. Smart and green urban solid waste collection systems: Advances, challenges, and perspectives. IEEE.

LU, J.-w. et al. Smart and Green Urban Solid Waste Collection Systems: Advances, Challenges, and Perspectives. **IEEE**, p. 1–14, 2015.

MENDOZA, J. E. et al. A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. **Computers & Operations Research**, v. 37, n. 11, p. 1886–1898, nov. 2010. ISSN 03050548.

MESTER, D.; BRÄYSY, O. Active-guided evolution strategies for large-scale capacitated vehicle routing problems. **Computers & Operations Research**, Elsevier, v. 34, n. 10, p. 2964–2975, 2007.

MOSCATO, P.; COTTA, C. A modern introduction to memetic algorithms. In: **Handbook of metaheuristics**. [S.l.]: Springer, 2010. p. 141–183.

MUYLDERMANS, L.; PANG, G. On the benefits of co-collection: Experiments with a multi-compartment vehicle routing algorithm. **European Journal of Operational Research**, Elsevier B.V., v. 206, n. 1, p. 93–103, out. 2010. ISSN 03772217.

NAGATA, Y.; BRÄYSY, O. Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. **Networks**, Wiley Online Library, v. 54, n. 4, p. 205–215, 2009.

OPPEN, J.; KKETANGEN, A. L. A tabu search approach for the livestock collection problem. **Computers & Operations Research**, v. 35, n. 10, p. 3213–3229, out. 2008. ISSN 03050548.

OSMAN, I. H. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. **Annals of Operations Research**, v. 41, n. 4, p. 421–451, 1993. ISSN 1572-9338.

PISINGER, D.; ROPKE, S. A general heuristic for vehicle routing problems. **Computers & Operations Research**, v. 34, n. 8, p. 2403–2435, 2007. ISSN 0305-0548.

POPOVIC, D.; VIDOVIC, M.; RADIVOJEVIC, G. Variable Neighborhood Search heuristic for the Inventory Routing Problem in fuel delivery. **Expert Systems with Applications**, v. 39, n. 18, p. 13390–13398, dez. 2012. ISSN 09574174.

PRINS, C. A simple and effective evolutionary algorithm for the vehicle routing problem. **Computers & Operations Research**, v. 31, n. 12, p. 1985–2002, 2004. ISSN 03050548.

PRINS, C. A grasp× evolutionary local search hybrid for the vehicle routing problem. In: **Bio-inspired algorithms for the vehicle routing problem**. [S.l.]: Springer, 2009. p. 35–53.

REED, M.; YIANNAKOU, A.; EVERING, R. An ant colony algorithm for the multi-compartment vehicle routing problem. **Applied Soft Computing**, Elsevier B.V., v. 15, p. 169–176, fev. 2014. ISSN 15684946.

SETHANAN, K.; PITAKASO, R. Differential evolution algorithms for scheduling raw milk transportation. **Computers and Electronics in Agriculture**, v. 121, p. 245–259, feb 2016. ISSN 01681699.

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.

TARANTILIS, C. D. Solving the vehicle routing problem with adaptive memory programming methodology. **Computers & Operations Research**, Elsevier, v. 32, n. 9, p. 2309–2327, 2005.

TOTH, P.; VIGO, D. The granular tabu search and its application to the vehicle-routing problem. **Informs Journal on computing**, INFORMS, v. 15, n. 4, p. 333–346, 2003.

VIDAL, T. et al. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. **Operations Research**, INFORMS, v. 60, n. 3, p. 611–624, 2012.

VIDAL, T. et al. Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. **European Journal of Operational Research**, Elsevier B.V., v. 231, n. 1, p. 1–21, nov. 2013. ISSN 03772217.

VOUDORIS, C. **Guided local search for combinatorial optimisation problems**. Tese (Doutorado) — Department of Computer Science, University of Essex, 1997.

WREN, A. **Computers in Transport and Planning**. [S.l.]: Ian Allan, London, 1971.

APPENDIX: RESUMO EM PORTUGUÊS

O problema de roteamento de veículos com múltiplos compartimentos (MCVRP) estende o problema de roteamento de veículos (VRP) clássico permitindo a entrega ou coleta de produtos que devem ser mantidos separados em diferentes compartimentos. Os clientes tem demandas diferentes por cada tipo de produto, e os veículos possuem compartimentos dedicados a um tipo de produto. Os clientes podem ser visitados mais de uma vez, porém toda a demanda por um tipo de produto deve ser atendida em uma única visita. Como no VRP classico, no MCVRP temos um único depósito de onde parte uma frota de veículos idênticos. O objetivo é atender a demanda de todos os clientes minimizando o tempo total das rotas necessárias. O MCVRP é $\mathcal{NP}$-difícil já que é uma generalização do VRP. Métodos exatos conseguem resolver instâncias com até 50 clientes. Portanto, instâncias maiores ou com mais restrições normalmente são resolvidas com métodos heurísticos.

O uso de veículos com múltiplos compartimentos se da em casos onde produtos precisam ser mantidos separados. Alguns exemplos de uso de veículos com múltiplos compartimentos são: coleta de leite de tipo, qualidade ou data de ordenha (MENDOZA et al., 2010); coleta seletiva de lixo (REED; YIANNAKOU; EVERING, 2014), (MUYLDERMANS; PANG, 2010), (ELBEK; WØHLK, 2016); distribuição de mercadorias que necessitam níveis de refrigeramento diferentes (CHAJAKIS; GUIGNARD, 2003); transporte de animais vivos (OPPEN; KKETANGEN, 2008).

Apesar de ter uma grande aplicabilidade na indústria o MCVRP não foi amplamente estudado pela literatura. Em nossa busca encontramos penas 5 trabalhos que tratam deste problema. (El Fallahi; PRINS; Wolfler Calvo, 2008) foi o primeiro trabalho estudar o MCVRP. Eles propuseram um conjunto de instâncias e duas heurísticas: algoritmo memético e uma busca tabu. (MUYLDERMANS; PANG, 2010) propos uma busca local guiada para o MCVRP e demonstrou os benefícios do uso de veículos com compartimentos para fazer a co-coleta comparado com o uso de veículos com apenas um compartimento. (DERIGS et al., 2010) estudou o MCVRP e algumas variações como compartimentos flexíveis e alocação de produtos a compartimentos sujeito a restrições. (REED; YIANNAKOU; EVERING, 2014) e (ABDULKADER; GAJPAL; ELMEKKAWY, 2015) estudaram outra variação do MCVRP que exige que todas as demandas do cliente sejam atendidas em uma única visita. Esta variação é conhecida como MCVRP-WS e também é abordada neste trabalho.

A heuristica que nós propomos para resolver o MCVRP foi inspirada na busca tabu iterada (ITS) para VRP proposta por (CORDEAU; MAISCHBERGER, 2012), que é considerada uma das melhores heurísticas para VRP. Uma busca local iterada começa num mínimo local e repetidamente aplica uma perturbação seguida de uma busca tabu para encontrar outro mínimo local. Um critério de aceite é utilizado para decidir se a busca continua com a melhor solução encontrada ou da última solução retornada pela busca tabu. O algoritmo inicia construindo uma solução inicial e aplica a busca tabu nesta solução. Depois por I iterações o ITS perturba a solução atual e aplica a busca tabu. A probabilidade de uma nova solução ser aceite para a próxima iteração é $1 - (i/I)^2$, caso contrário a melhor solução já encontrada é utilizada. Este critério de aceite é importante para diversificar a busca no início e intensificar a busca no final do algoritmo.

Como solução inicial selecionamos a melhor entre dois algoritmos construtivos bem conhecidos do VRP que são: varredura angular proposto por (WREN, 1971) e (GILLETT; MILLER, 1974); método das economias proposto em (CLARKE; WRIGHT, 1964). A perturbação da solução é feita removendo uma quantidade de clientes e reinserindo na solução de tal forma que minimize o custo. Nós utilizamos para remoção e inserção de clientes um heurística conhecida como GENI proposta em (Gendreau, Michel and Hertz, Alain and Laporte, 1992).

A busca tabu é uma metaheurística que guia uma busca local através do espaço de busca. Ela permite movimentos para soluções piores e evita ciclos através de uma memória de curta duração e declarando movimentos que retornam a soluções já visitadas como tabu. A busca tabu proposta inicia em uma solução inicial e move para a melhor solução não tabu. Durante essa busca soluções não factíveis também são visitadas. A violação das restrições de capacidade dos compartimentos e distância máxima da rota são penalizadas na função objetivo. A busca tabu para depois de $\sqrt{(I-i)\pi}$ iterações sem melhorar a solução incumbente. A vizinhança consiste em apenas um tipo de movimento: realocar um subconjunto de demandas de uma rota para outra. O tabu tenure é definido randomicamente no inicia da busca tabu entre $[1, \sqrt{n\,m}]$ onde n é o número de clientes e m é a quantidade máxima de rotas.

Os experimentos computacionais foram feitos usando instâncias com quantidade de clientes entre 50 e 483. As instâncias para MCVRP são geradas a partir de instâncias bem conhecidas do VRP. O algoritmo implementado foi avaliado usando 3 diferentes experimentos. No primeiro experimento comparamos nossos resultados com as melhores heurísticas para o VRP afim de verificar o quão longe estamos do estado da arte para

este problema. O resultado é que nosso algoritmo encontra soluções com qualidade comparável ao estado da arte do VRP, porém necessita de mais tempo para isso devido a falta de alguma otimização ou a quantidade extra de processamento que temos por considerar multiplos compartimentos.

Nosso experimento dois avalia o algoritmo em comparação com o estado da arte do MCVRP. Nós apresentamos resultados para todas as instâncias com 2 até 5 compartimentos. Os resultados mostram que nosso algoritmo supera os resultados existentes na literatura tanto em termos de qualidade da solução quanto em termos de tempo computacional de processamento. No experimento três avaliamos nosso algoritmo para a variação do MCVRP que exige que toda demanda de um cliente seja atendida em uma única visita. Para esta variação nosso algoritmo também supera os resultados existentes na literatura conseguindo soluções melhores com até 4 vezes menos tempo de processamento.

Este trabalho propôs uma busca tabu iterada para o problema de roteamento de veiculos com multiplos compartimentos que supera todas as heuristicas encontradas na literatura para este problema. O diferencial do algoritmo proposto é principalmente a maneira como a vizinhança é explorada. O resultados apresentados também mostram que permitir múltiplas visitas não oferece uma economia muito significativa no custo total da rota.