ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

**Department of Computer Science**

**Exam of Decision Making & Constraint Programming**
**Project Report**

Academic Year 2020 - 2021

# Modelling and Solving the Vehicle Routing Problem (VRP)

**Mario Sessa**
0000983529

A project report of modelling and solving the Vehicle Routing Problem using the Constraint Programming approach

Alma Mater Studiorum, University of Bologna
Department of Computer Science
Bologna
Via Zamboni, 33, 40126
Italy

# Contents

**1 Introduction** **3**
1.1 Problem Presentation . . . . . 3
1.2 Formal Problem Definition . . . . . 4

**2 Modelling VRP in Constraint Programming** **5**
2.1 Input Data and any Potential Data Processing . . . . . 5
2.2 Decision Variables . . . . . 6
2.3 Problem & Additional Constraints . . . . . 7
2.3.1 Main Constraints . . . . . 7
2.3.2 Additional Constraints . . . . . 7
2.4 Objective Function . . . . . 8

**3 Experimental Study** **9**
3.1 Experimental Setup . . . . . 9
3.1.1 Software and Hardware . . . . . 9
3.1.2 Instances . . . . . 9
3.1.3 Search Strategy . . . . . 10
3.2 Experimental Results . . . . . 11
3.2.1 Tables of Results . . . . . 11

**4 Conclusions** **12**

# Chapter 1

# Introduction

## 1.1 Problem Presentation

The Vehicle Routing Problem (VRP) is an NP-optimization problem that has been of great interest for decades for both, science and industry. We got an example with Amazon that use it for the management of the deliveries from his orders around the world. The aim of our version of VRP is to find a set of minimum total cost routes for a fleet of capacitated vehicles (CVRP) based at a single depot, to serve a set of customers under the following constraints:

1. each route must begin and ends at the depot node
2. each customer/location is visited exactly once
3. the total demand of each routes does not exceed the capacity of vehicle assigned to the route

This type of problem is referred to a combination of two sub-problem:

1. Travelling Salesman Problem (TSP) which define an Hamiltonian routes similar to the combined routes of the Vehicle Routing Problem
2. Knapsack Problem which define constraints to manage the capacities of the vehicles in respect of the customers demands of their routes

The VRP belongs to the category of NP-hard problems that can be exactly solved only for small instances of the problem. Therefore, many researchers have concentrated on developing heuristics algorithm to solve this problem for hard instances or find a good solution (not necessarily the optimal one). In the following section, we will explain the strategy used to solve it using a Constraint Programming approach.

## 1.2 Formal Problem Definition

Let G = (V,H) be a complete directed graph with $V$ on domain $\{1, 2, 3, ...n, n+1\}$ as the set of nodes and H = $\{(i, j) : i, j \in V, i \neq j\}$ a set of arcs, where node n + 1 represents the depot for a fleet of v vehicles with capacities $C_w > 0, w \in \{1, 2..v\}$. The remaining n nodes represents the geographically position of customers. Each node of V has a location with coordinates $(x_i, y_i)$ for a node $i \in \{1, 2, 3, .., n + 1\}$. Each customer $c \in V - \{n - 1\}$ has a demand $0 < de_c \leq \sum_{w=1}^{v} C_w$. We also have a set of distances $D = \{d_{ij} | i, j \in V\}$ represents the weight of the edge between nodes i and j.

Each vehicle needs to start and end his route to the depot, every customer can be visited just from a vehicle. Moreover, given $de_{cw}$ the demand $d_c$ satisfied by a vehicle $w \in \{1, 2...v\}, \sum_{i,j \in V} d_{cw} \leq C_w, \forall w \in \{1, 2...v\}$.

Defined $d_{ijw}$ the distance traveled by a vehicle w and $b_w \in \{0, 1\}$ a boolean representing if a vehicle is used or not, the aim of the problem is to minimize $\sum_{w \in \{1,2..v\}} d_{ijw} + (\sum_{w \in \{1,2..v\}} b_w) * weight_v$ with $weight_v$ is a constant value to balance the optimal result between the use of vehicles and the total distance traveled. [1]

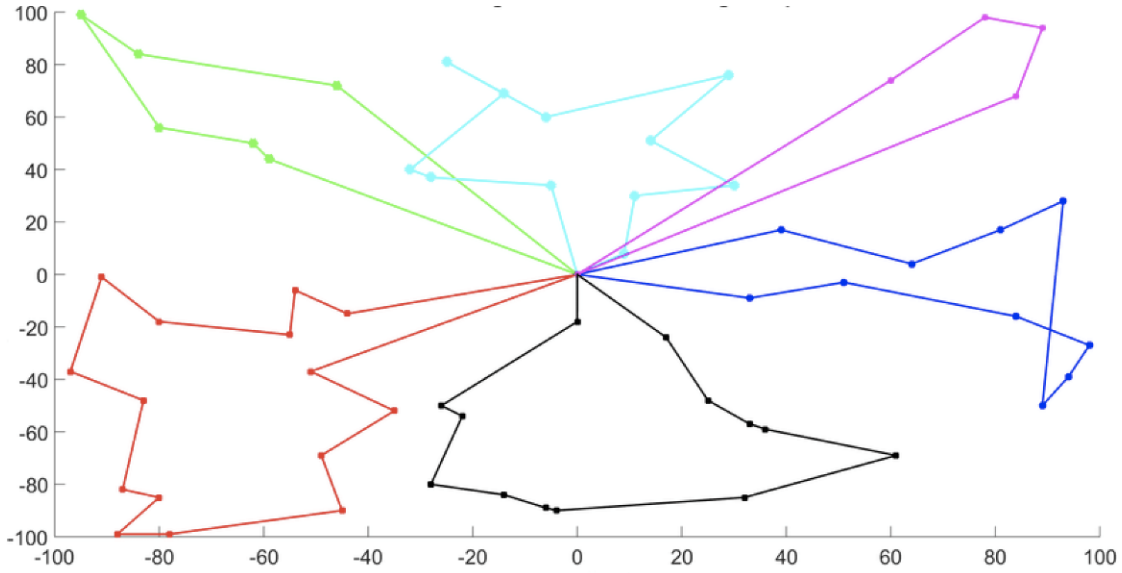


Figure 1.1: Graph of a VRP solution based on the use of 6 vehicles

# Chapter 2

# Modelling VRP in Constraint Programming

## 2.1 Input Data and any Potential Data Processing

Let N the number of customers, input data are as follow:

1. $locX_i$ for each i in {1..N+1} with a float domain defines the coordinate X of a location $L_i$, the last one is a reference of the depot coordinate
2. $locY_i$ for each i in {1..N+1} with a float domain defines the coordinate Y of a location, the last one is a reference of the depot coordinate
3. a list $DEM$ with $de_i \in DEM, i \in \{1..N\}$ and an integer domain; the demand of the customer i
4. V is the number of vehicles available
5. a list $C$ with $c_i \in C, i \in \{1..V\}$] and an integer domain; the capacity of the vehicle i

To solve the problem, each location needs to be visited. Therefore, it's important to take trace of visits in the routes of a solution. Each node represents a visit checked by only one vehicle. From the initial set of input data and observations taken from the problem, it's necessary to define the following processing data, useful to establish constraints in the model or to clarify some types of values:

6. $NumVisits$ is the total number of visits of the final combined route. It is equal to $N + V * 2$
7. a list $D$ with $d_{i,j} \in D$, $i, j \in \{1...(N+V*2)\}$ and an integer domain; represents distances between two nodes i and j. Values are calculated as follow:

   (a) $\forall i, j \in \{1, ..., N\} : d_{ij} = \sqrt{(locX_j - locX_i)^2 + (locY_j - locY_i)^2} * 1000$

   (b) $\forall i \in \{1, ..., N\}, j \in \{N+1, ..., N+V*2\} :$
   $d_{ij} = \sqrt{(locX_{N+1} - locX_i)^2 + (locY_{N+1} - locY_i)^2} * 1000$

(c) $\forall i, j \in \{N+1, ..., N+V*2\} : d_{i,j} = 0$

8. A weight of vehicle $w$ with integer domain represents the relevance of minimizing the number of vehicles compared to the minimal distance traveled during the optimal research.

## 2.2 Decision Variables

A solution of the problem is made up of routes, one route for vehicles. It is a sequence of nodes (visits) formed by customers nodes (visited in the route) and 2 depot visits (one start and one end node).

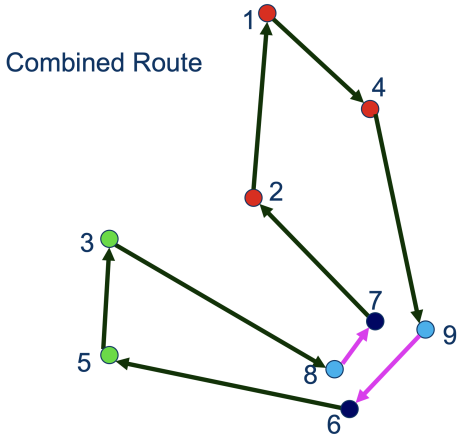


Figure 2.1: A solution of VRP with 5 customers and 2 vehicles with 2 routes. Nodes {6,7} are start visits nodes and {9,8}are end visits.

Starting from these observations, the model has the following decision variables:

1. list of vehicles assignments $VEH, ve_i \in VEH, i \in \{1...NumVisits\}$ with domains $\{1, ..., V\}$where for each $i \in \{1, ..., NumVisits\}, j \in 1, ..., V$,if $ve_i = j$ then the node i is visited by the vehicle j

2. list of successions $S, s_i \in S, i \in \{1...NumVisits\}$ with domains $\{1, ..., NumVisits\}$ where for each $i, j \in \{1, ..., NumVisits\}$ if $succ[i] = j$ then the successor of node i is j in the combined route environment.

3. list of values $U, u_i \in U, i \in \{1...V\}$ with domain $\{0, 1\}$ where for each $i \in \{1, ..., V\}$ if $u_i = 1$ then the vehicle visits at least one customer node, $u_i = 0$ otherwise

## 2.3 Problem & Additional Constraints

### 2.3.1 Main Constraints

As stated in paragraph 1.1, the Vehicle Routing Problem is seen as the combination of Travelling Salesman Problem (TSP) and Knapsack Problem. Therefore, VRP can inherit some TSP and Knapsack Problem constraints.

Using TSP definition, we can assume these following adaptive constraints for VRP model:

1. $circuit([s_1, ..., s_{NumVisits}])$ constraints the elements of the list $s$ to define the *Hamiltonian path* where $\forall i \in \{1, ..., NumVisits\}, s_i = j, j \in \{1, ., i-1, i+1, .., NumVisits\}$ means that j is the successor of i.

Using Knapsack Problem definition, we can add these following constraints for the $[c_1, ..., c_v]$ capacities and $[de_1, ..., de_N]$ customers demands:

2. $\forall v \in \{1, ..., V\}, \sum_{i \in \{1,..,N\}: ve_i = v} de_i \leq c_v,$

To maintain consistency with the problem definition, are defined a series of constraints to start and end every vehicles routes at the depot and linking constraints to manage the start and end nodes.
Linking and consistency constraints are as follow:

3. $\forall s \in \{N+1, ..., N+V\}, ve_s = s - N$
4. $\forall e \in \{N+V+1, ..., N+V*2\}, ve_e = e - N - V$
5. $\forall i \in \{N+V+1, ...N+(V*2)-1\}, s_{s_i} = s_i - V + 1$
6. $s_{N+2*V} = N + 1$
7. $\forall j \in \{1, ..., N\}, ve_{s_j} = ve_j$
8. $\forall i \in \{1, ..., N\}, u_{ve_i} = 1$

### 2.3.2 Additional Constraints

VRP additional constraints are used to increase the optimal solution with the reduction of the search space using symmetry breaking constraints or implied constraints. It is important to note that some breaking constraints caused a drop in search performance, so they were not considered within the final model.

9. $alldifferent([s_1, ..., s_{NumVisits}])$ successor implied constraint
10. $\forall i, j \in \{1, ..., V\} \wedge c_i = c_j \wedge i < j, u_j > 0 \Rightarrow u_i > 0$ breaks same-capacity vehicles symmetry

## 2.4 Objective Function

Let $w$ the weight of vehicles and let $td$ and $vu$ respectively the total distance traveled by vehicles and the number of vehicles used defined as follow:

- $td = \sum_{v \in \{1,...,NumVisits\}} d_{v,s_v}$
- $vu = \sum_{v \in \{1,...,V\}} u_v$

The objective function is:

$$f(td, vu) = td + (vu * w)$$

The objective of the problem is:

$$min(f(td, vu))$$

The value of w is the balance value of the reduction of vehicles number compared to the reduction of the total distance.The more the value of $w$ is huge and more the objective function tends to optimize the minimum values of vehicles than the total distances.

# Chapter 3

# Experimental Study

## 3.1 Experimental Setup

### 3.1.1 Software and Hardware

The model was implied using a constraint programming language: MiniZinc 2.5.3. The solver associated is Gecode 6.3.0. Experiments were done on a Macbook Pro 2018 with processor at 2,2 GHz, 6-Core on a Intel Core i7 of 9th generation and a memory of 16 GB, 2400 Mhz DDR4.

### 3.1.2 Instances

The following table represents instances with their features referred to the number of customers, the number of vehicles, the total capacity available to satisfy the customers demands and the total customers demands. Total capacity and total demands are distributed unevenly between customers and vehicles respectively.

| **Instance** | **Features** | | | |
|---|---|---|---|---|
| | **#Customers** | **#Vehicles** | **Tot. Capacity** | **Tot. Demand** |
| PR01 | 47 | 4 | 800 | 647 |
| PR02 | 95 | 20 | 3,900 | 1,210 |
| PR03 | 143 | 20 | 3,800 | 1,765 |
| PR04 | 191 | 20 | 3,700 | 2,472 |
| PR05 | 239 | 20 | 3,600 | 3,335 |
| PR06 | 287 | 20 | 3,700 | 3,665 |
| PR07 | 71 | 20 | 4,000 | 928 |
| PR08 | 143 | 20 | 3,800 | 1,985 |
| PR09 | 215 | 20 | 3,600 | 2,735 |
| PR10 | 287 | 20 | 3,900 | 3,825 |
| PR11 | 47 | 20 | 4,000 | 647 |

### 3.1.3 Search Strategy

[2] The following list represents the search strategy associated with their acronyms carried out on the model in Minizinc with Gecode solver:

1. **SEQDR**: identify the sequential search structured in the following ordered sequence:

   (a) Take a decision for each element of the vehicles list *ve* using domWdeg and random value heuristics

   (b) Take a decision for each element of successors list *s* using domWdeg and random value heuristics

   with the Luby Strategy on L = 250

2. **SEQSM**: identify the sequential search structured in the following ordered sequence:

   (a) Take a decision for each element of the vehicles list *ve* using smallest domain and minimum value heuristics

   (b) Take a decision for each element of the successors list *s* using domWdeg and random value heuristics

   with the Luby Strategy on L = 300.

3. **DRLSN**: identify the search structured with domWdeg and random domain heuristic on *s* variables list using Luby Strategy with L = 350 and the Large Neighbourhood Strategy fixing 85% of the *s* variables list.

4. **SMLNS**: identify the SEQ heuristics using Large Neighbourhood Strategy fixing 85% of the *s* variables list.

## 3.2 Experimental Results

### 3.2.1 Tables of Results

- Table of Failures

| Instance | SEQDR | SEQSM | DRLNS | SMLNS |
|---|---|---|---|---|
| PR01 | 1,617,823 | 1,846,879 | 1,758,637 | 1,790,778 |
| PR02 | 261,731 | 276,031 | 304,075 | 307,405 |
| PR03 | 107,615 | 111,227 | 108,265 | 153,923 |
| PR04 | 54,289 | 59,572 | 53,443 | 86,818 |
| PR05 | 25,276 | 35,984 | 30,839 | 52,607 |
| PR06 | 14,075 | 24,078 | 15,912 | 37,785 |
| PR07 | 365,984 | 431,475 | 397,412 | 523,065 |
| PR08 | 101,392 | 104,439 | 107,554 | 168,311 |
| PR09 | 32,350 | 31,886 | 33,942 | 69,721 |
| PR10 | 13,015 | 15,580 | 14,493 | 36,780 |
| PR11 | 738,268 | 491,894 | 650,122 | 723,032 |

- Table of Optimal Solutions

| Instance | SEQDR | | SEQSM | | DRLNS | | SMLNS | |
|---|---|---|---|---|---|---|---|---|
| | Dist. | Veh. | Dist. | Veh. | Dist. | Veh. | Dist. | Veh. |
| PR01 | 2,395,540 | **4** | 2,300,229 | **4** | 940,728 | **3** | 959,341 | **3** |
| PR02 | 5,759,370 | **20** | 5,171,574 | **7** | 2,481,414 | **9** | 2,280,093 | **9** |
| PR03 | 1,1090,848 | **20** | 10,433,132 | **10** | - | **-** | 6,768,854 | **10** |
| PR04 | 12,834,190 | **20** | 12,666,579 | **14** | - | **-** | 8,538,335 | **14** |
| PR05 | 13,939,739 | **20** | 13,457,287 | **19** | - | **-** | 9,538,436 | **19** |
| PR06 | 18,746,215 | **20** | 19,188,412 | **20** | - | **-** | 14,869,127 | **20** |
| PR07 | 5,040,291 | **20** | 4,238,722 | **5** | 1,227,767 | **7** | 1,265,506 | **4** |
| PR08 | 10,159,608 | **20** | 9,736,174 | **11** | - | **-** | 5,649,437 | **11** |
| PR09 | 15,267,652 | **20** | 15,056,711 | **16** | - | **-** | 10,050,036 | **16** |
| PR10 | 19,773,277 | **20** | 19,778,614 | **20** | - | **-** | 13,794,840 | **20** |
| PR11 | 3,171,559 | **18** | 2,413,564 | **4** | 920,830 | **3** | 913,719 | **3** |

The previous table have reported the number of the total distance traveled on the combined route ($Dist. \equiv td$) and the number of vehicles used ($Veh. \equiv vu$). The objective value $min(f(td, vu))$ have been calculated using the objective function, $tu$ and $vu$ defined in the paragraph 2.4. and $w = 1000$.

# Chapter 4

# Conclusions

We have described a version of the Capacitated Vehicle Routing Problem with Constraint Programming method. The nature of the problem is reducible to the combination between Travelling Salesman and Knapsack Problem. We have considered the *circuit* global constraint to define the Hamiltonian property given by the Travelling Salesman modelling and the capacity symmetry breaking constraint from the Knapsack model. Furthermore, the model has been improved by an implied $alldifferent$ constraint but, adding some symmetry breaking constraint, the model decreased performance in terms of search. Finally, the search strategies have had problems in finding optimal solutions for instances with more than 100 customers. Moreover, they have been efficient in sequential cases and adding the Large Neighbourhood Strategy has given better results than the complete approach.

# References

[1] Bruno De Backer, Vincent Furnon, Paul Shaw, Philip Kilby and Patrick Prosser *Solving Vehicle Routing Problems Using Constraint Programming and Metaheuristics*. J. Heuristics, 2000.

[2] Gilbert Laporte, Stefan Ropke, Thibaut Vidal *Chapter 4: Heuristics for the Vehicle Routing Problem* In book Vehicle Routing (pg. 87-116), 2014