Metaheuristic Lesson: Genetic Algorithm – VRP (Vehicle Routing Problem)

Method · August 2021

DOI: 10.13140/RG.2.2.35098.98248

1 author:

Valdecy Pereira
Universidade Federal Fluminense

UNIVERSIDADE FEDERAL FLUMINENSE

Metaheuristics

Lesson: Genetic Algorithm – VRP (Vehicle Routing Problem)

Professor: Valdecy Pereira, D. Sc.

email: valdecy.pereira@gmail.com

Outline

1. Genetic Algorithm
2. Initialization
3. Selection
4. Crossover
5. Mutation
6. Elitism
7. Github

The **GA** (**Genetic Algorithm**) is a metaheuristic inspired by the process of natural selection that is used for solving optimization problems. The **GA** selects the **fittest** **individuals** (solutions) that are used for reproduction in order to produce **offspring** (usually a better solution) for the next **generation**. John Henry Holland (February 2, 1929 – August 9, 2015) was the father of the **GA** and popularized this term in his paper published in 1973, entitled "*Genetic Algorithms and the Optimal Allocation of Trials*".

MH – *Genetic Algorithm*

The **GA** (**Genetic Algorithm**) can be understood as a search and optimization technique, inspired by the Darwinian principle of the evolution of species and genetics. It can find the global optimum solution or local optimum solutions depending on the initial setup and parameter calibration.

The **GA** creates an abstract representations (**chromosome** or **genome**) of a candidate solution (**individual**). Traditionally, solutions are encoded in a binary form, but other encodings are also possible. To solve an optimization problem, initially a random set of candidate solutions are created (**population**), and them it evolves towards a better solution using genetic operators called **crossover** and **mutation** to create a **new population**.

The **GA** represents an iterative process, where each iteration is called a **generation**. Then, evaluating the **fitness** of each **individual**, we can select a pair for breeding that will generate an **offspring**. To create the **offspring**, the **crossover** operator exchanges parts of the pair of **chromosomes**, and the **mutation** operator changes the **gene** value in some randomly chosen location of the **offspring chromosome**. This process is repeated until the new **population** has enough members, ending the current **generation**. A common practice is to terminate a **GA** after a specified number of **generations**.

The best solution (**elite**) of a **generation** can be stored and then given to next **generation**, this optional process is called **elitism**.

MH – *Genetic Algorithm*

GA Pseudocode

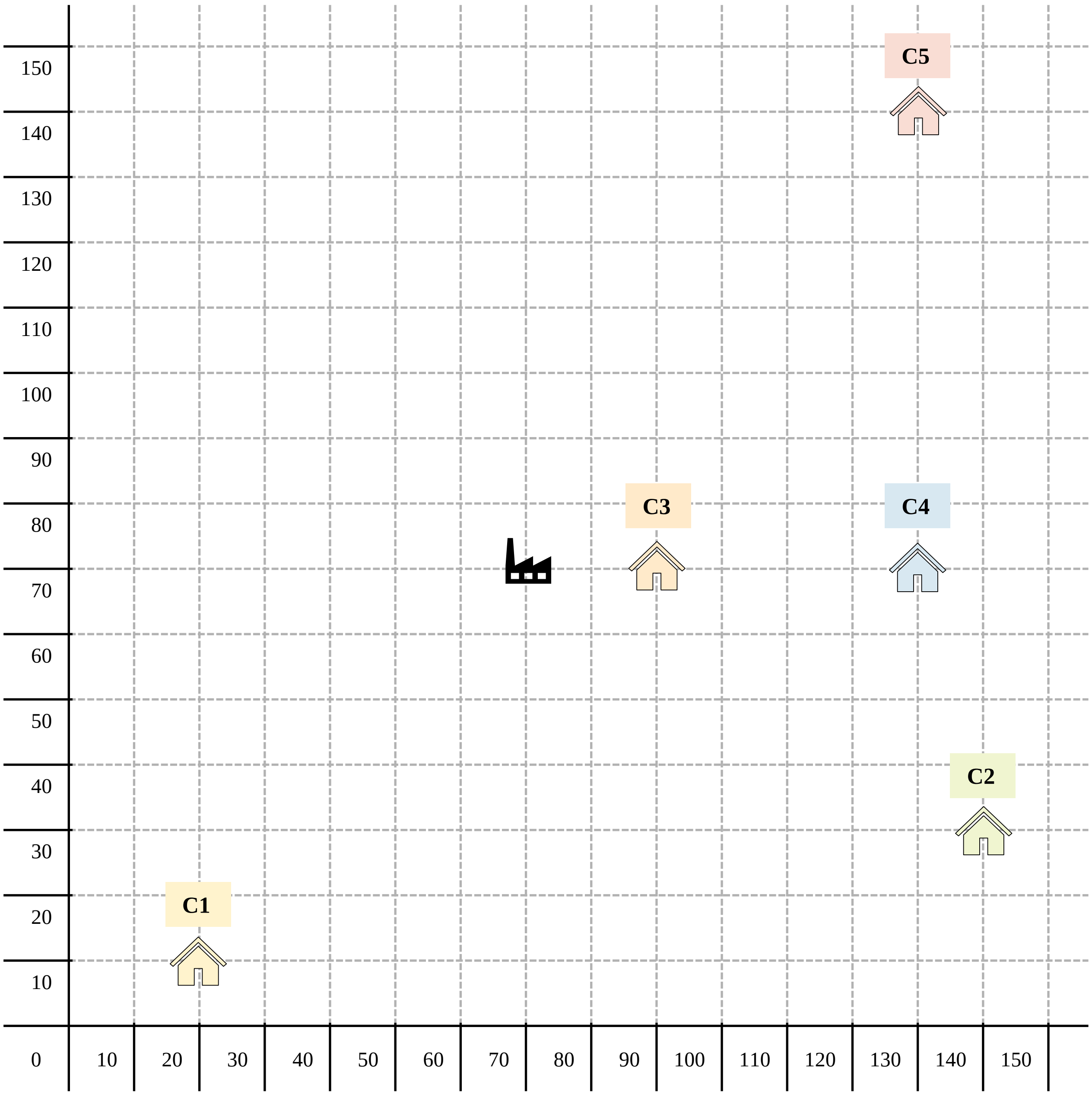
Randomly initialize the first **Population** $(i = 0)$
Set $\boldsymbol{\eta}$ *(***Population** *size) and* $\boldsymbol{k}$ *(number of* **generations***)*
for $(i = 1\ ; i = k; i{+}{+})$ **do**
  Add the **Elite** *as a member of the* **Population** i *(Optional)*
  while card(Population i**)** $< \boldsymbol{\eta}$ **do**
    Evaluate the **Fitness** *of all* **Individuals** *the* **Population** $i - 1$
    Using the **Fitness** *from the* **Population** $i - 1$, *select any two* **Individuals** *to breed*
    Create one **Offspring** *using the genetic operators* **Crossover** *and* **Mutation**
    Add the **Offspring** *as a member of the* **Population** i
next $\boldsymbol{i}$

MH – *Genetic Algorithm*

Suppose that we want to minimize the following **CVRP** (Capacitated Vehicle Routing Problem), where each vehicle capacity is equal to 250. Infeasible solutions are penalized with a value equal to 1000 for each violation.

Id	x	y	Demand
Depot	70	70	-//-
Client 1	20	10	50
Client 2	140	30	60
Client 3	90	70	50
Client 4	130	70	40
Client 5	130	140	250

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

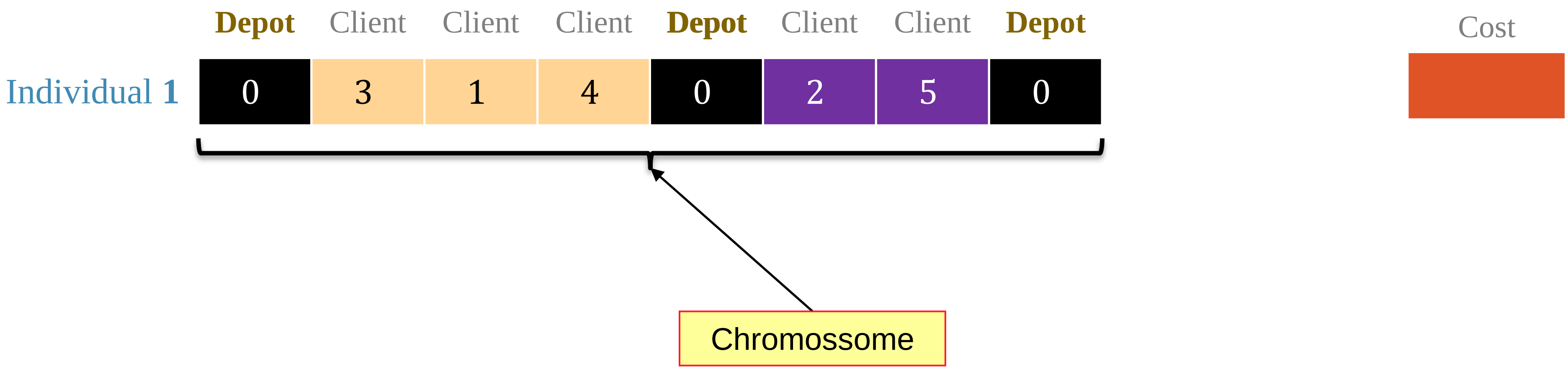

C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

MH – *Genetic Algorithm*

Individual **1**

Encoding a Solution

MH – *Genetic Algorithm*



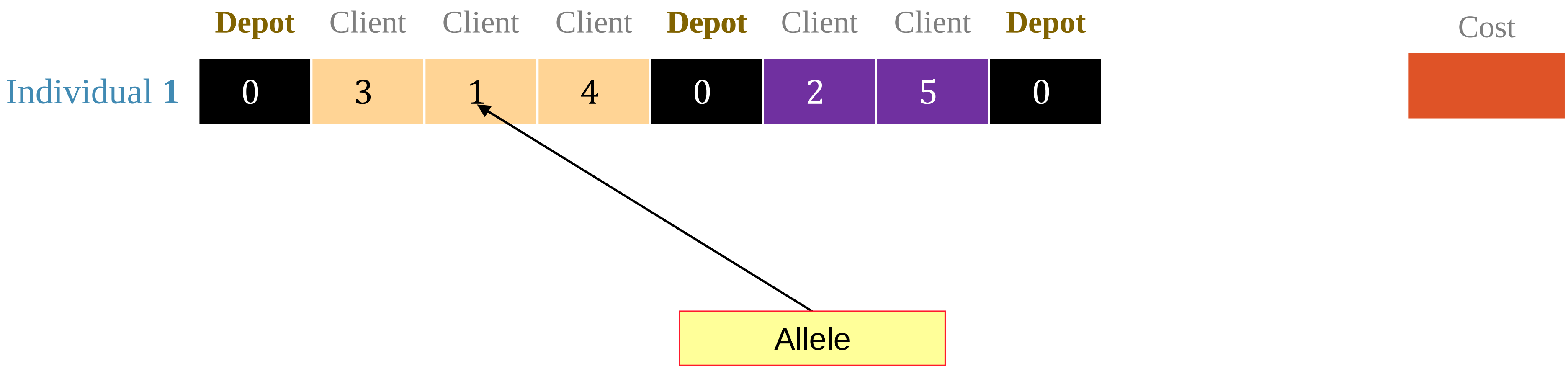
MH – *Genetic Algorithm*

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**
Individual **1**	0	3	1	4	0	2	5	0

Gene

Cost

MH – *Genetic Algorithm*



MH – *Genetic Algorithm*

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	

Fitness

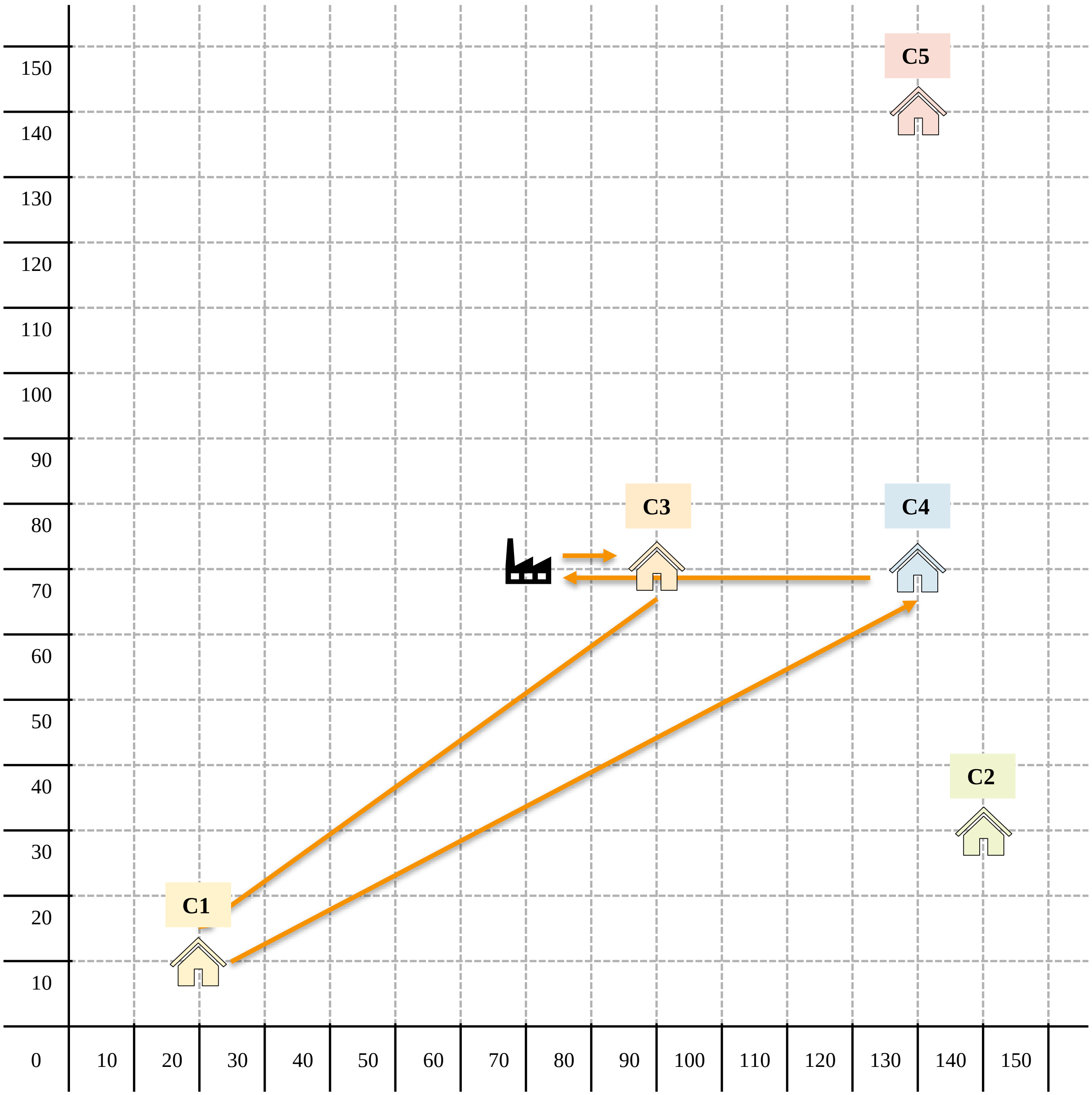
MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot
Individual **1**	0	3	1	4	0	2	5	0

Cost

Route 1:

$0 - 3 - 1 - 4 - 0$



C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

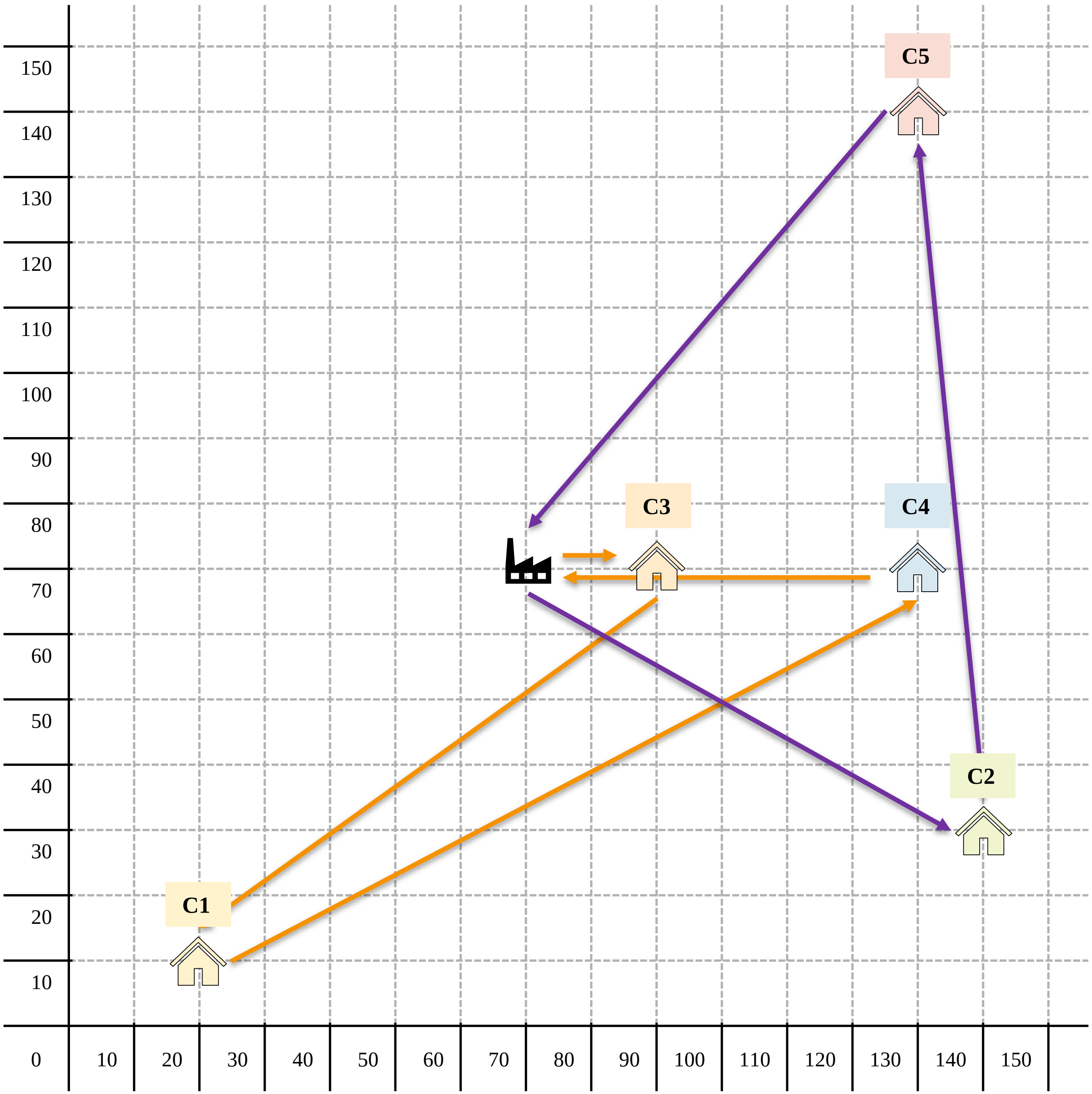
MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot
Individual **1**	0	3	1	4	0	2	5	0

Cost

Route 2:

$0 - 2 - 5 - 0$



C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

Route 1: $0 - 3 - 1 - 4 - 0$

Route 2: $0 - 2 - 5 - 0$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$$\text{Load} = 50 + 50 + 40 = 140$$

Route 1: $0 - \mathbf{3} - \mathbf{1} - \mathbf{4} - 0$

Route 2: $0 - 2 - 5 - 0$

Id	Demand
Client 1	**50**
Client 2	60
Client 3	**50**
Client 4	**40**
Client 5	250

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $\mathbf{0-3}-1-4-0$

$\text{Cost} = 20$

Route 2: $0-2-5-0$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - \mathbf{3} - \mathbf{1} - 4 - 0$

$\text{Cost} = 20 + 92.20$

Route 2: $0 - 2 - 5 - 0$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - \mathbf{1} - \mathbf{4} - 0$

$\text{Cost} = 20 + 92.20 + 125.30$

Route 2: $0 - 2 - 5 - 0$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - \mathbf{4} - \mathbf{0}$

$\text{Cost} = 20 + 92.20 + 125.30 + 60$

Route 2: $0 - 2 - 5 - 0$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - 4 - 0$

$\text{Cost} = 20 + 92.20 + 125.30 + 60 = 297.50$

Route 2: $0 - 2 - 5 - 0$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - 4 - 0$

$\text{Cost} = 297.50$

Route 2: $0 - \mathbf{2} - \mathbf{5} - 0$

$\text{Load} = 60 + 250 = 310$

Id	Demand
Client 1	50
Client 2	**60**
Client 3	50
Client 4	40
Client 5	**250**

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - 4 - 0$ $\quad \text{Cost} = 297.50$

Route 2: $0 - 2 - 5 - 0$ $\quad \text{Cost} = 1000$

$\text{Load} = 310$

Penalty Value for violating the vehicle maximum capacity constraint (250)

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - 4 - 0$

$\text{Cost} = 297.50$

Route 2: $0 - 2 - 5 - 0$

$\text{Cost} = 1000$

$\text{Load} = 310$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - 4 - 0$ $\quad$ $\text{Cost} = 297.50$

Route 2: $\mathbf{0 - 2} - 5 - 0$ $\quad$ $\text{Cost} = 1000 + 80.62$

$\text{Load} = 310$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - 4 - 0$

$\text{Cost} = 297.50$

Route 2: $0 - \mathbf{2} - \mathbf{5} - 0$

$\text{Cost} = 1000 + 80.62 + 110.42$

$\text{Load} = 310$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

$\text{Load} = 140$

Route 1: $0 - 3 - 1 - 4 - 0$

$\text{Cost} = 297.50$

Route 2: $0 - 2 - \mathbf{5} - \mathbf{0}$

$\text{Cost} = 1000 + 80.62 + 110.42 + 92.20$

$\text{Load} = 310$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

Load $= 140$

Route 1: $0 - 3 - 1 - 4 - 0$ Cost $= 297.50$

Route 2: $0 - 2 - \mathbf{5 - 0}$ Cost $= 1000 + 80.62 + 110.42 + 92.20 = 1283.24$

Load $= 310$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

Route 1: $0-3-1-4-0$ $\quad$ $\text{Cost} = 297.50$

Route 2: $0-2-5-0$ $\quad$ $\text{Cost} = 1283.24$

$$\text{Total Cost} = 297.50 + 1283.24 = 1580.74 \text{ km}$$

	Depot	Client 1	Client 2	Client 3	Client 4	Client 5
Depot	0	78.10	80.62	20.00	60.00	92.20
Client 1	78.10	0	121.66	92.20	125.30	170.29
Client 2	80.62	121.66	0	64.03	41.23	110.45
Client 3	20.00	92.20	64.03	0	40.00	80.62
Client 4	60.00	125.30	41.23	40.00	0	70.00
Client 5	92.20	170.29	110.45	80.62	70.00	0

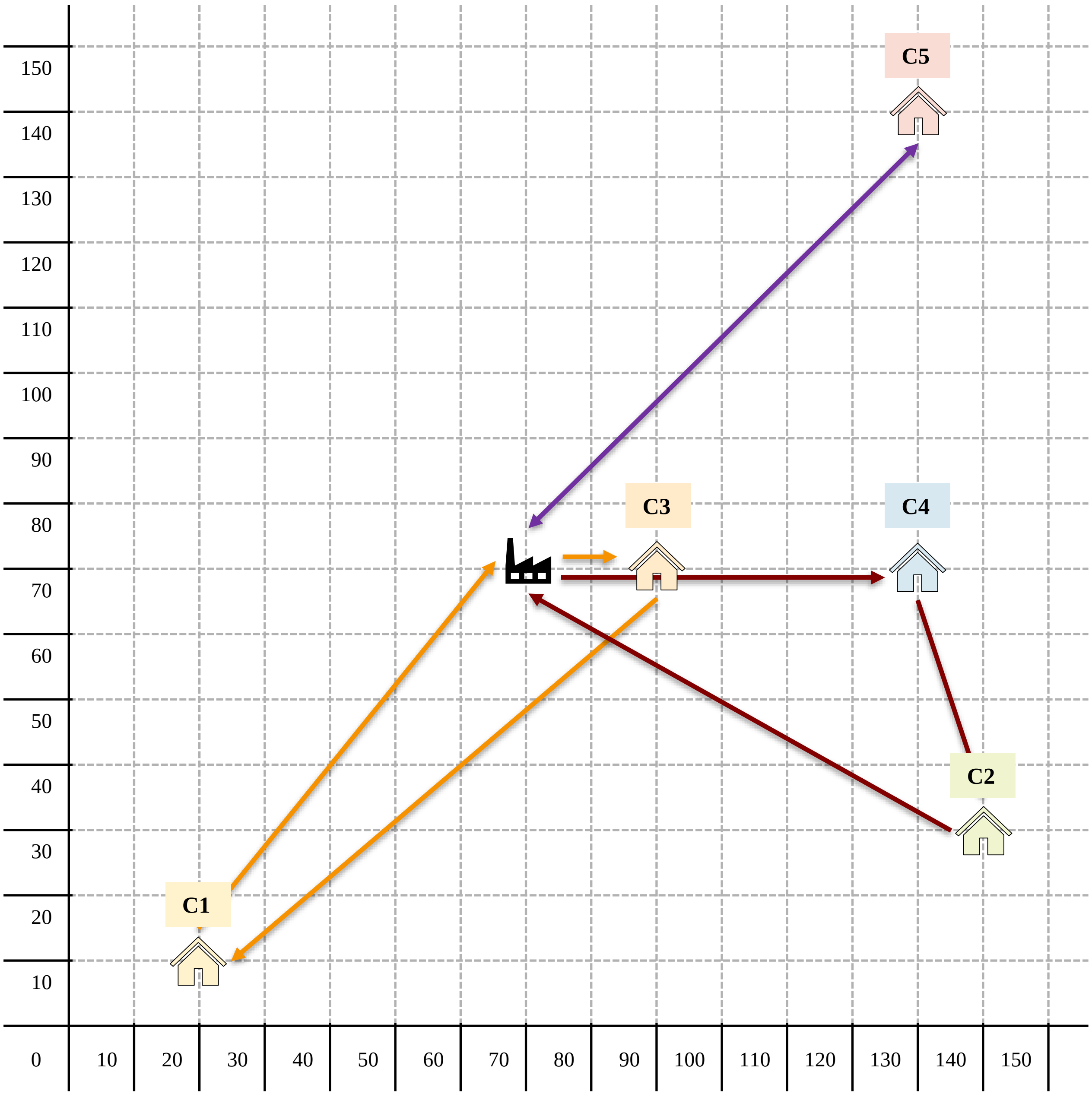
MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	



C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

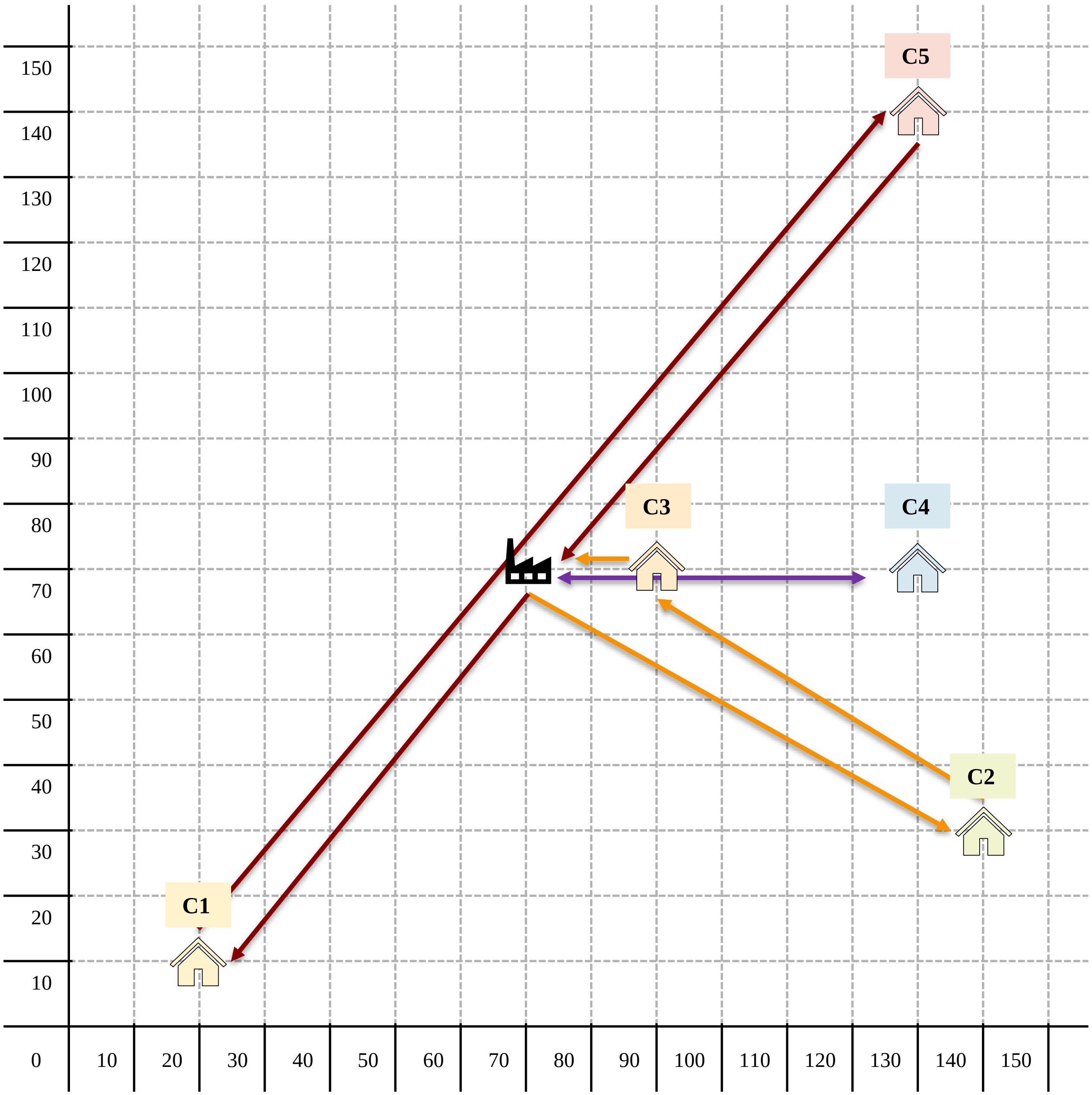
No Penalty

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **3**	0	2	3	0	4	0	1	5	0	


C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **3**	0	2	3	0	4	0	1	5	0	1625.25

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot		Cost
Individual **1**	0	3	1	4	0	2	5	0		1580.74

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **3**	0	2	3	0	4	0	1	5	0	1625.25

Penalty Value

MH – *Genetic Algorithm*

Individual **1**

Depot	Client	Client	Client	Depot	Client	Client	Depot
0	3	1	4	0	2	5	0

Cost: 1580.74

Individual **2**

Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot
0	3	1	0	5	0	4	2	0

Cost: 556.54

Individual **3**

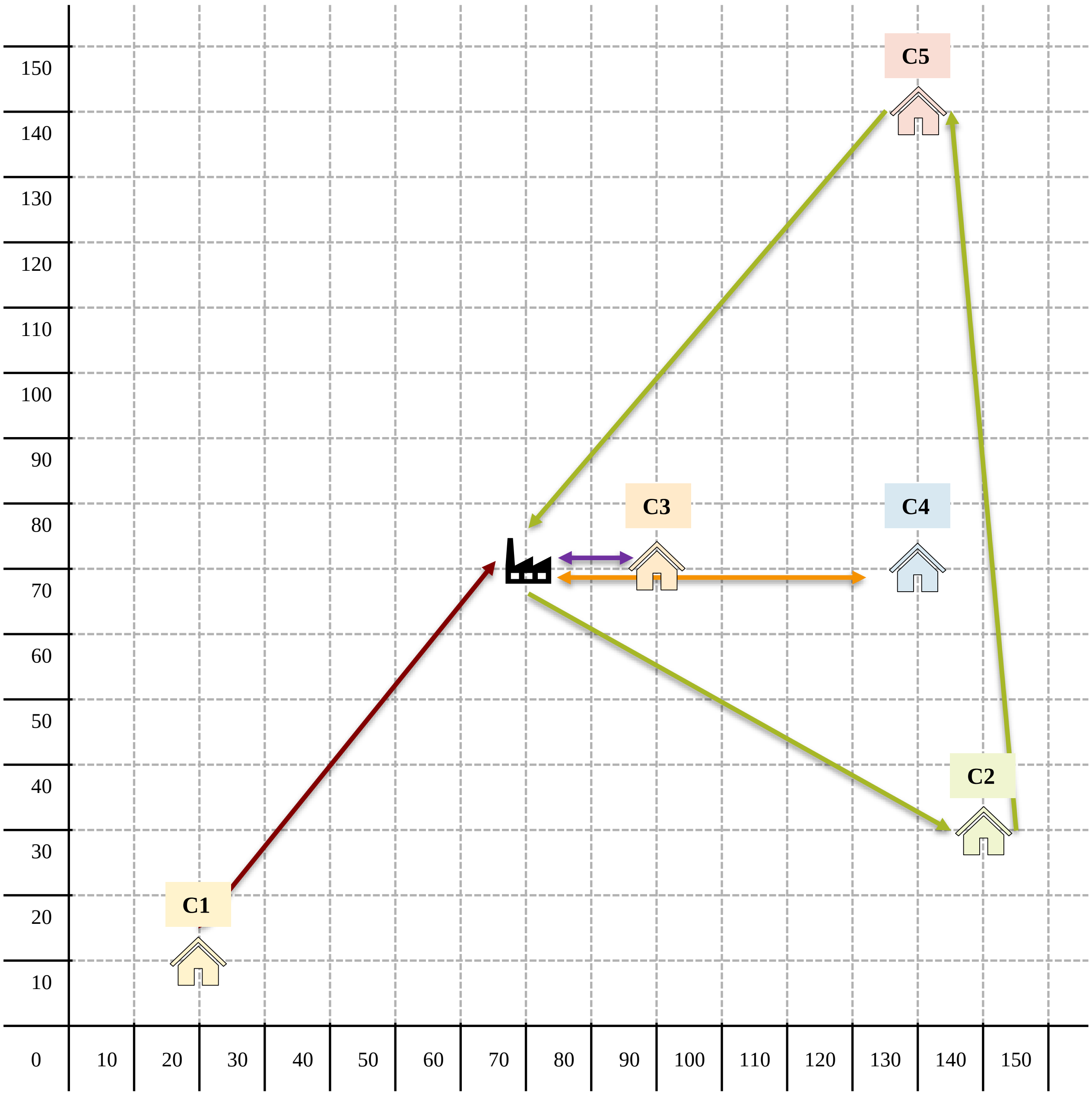
Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot
0	2	3	0	4	0	1	5	0

Cost: 1625.25

Individual **4**

Depot	Client	Depot	Client	Depot	Client	Depot	Client	Client	Depot
0	4	0	3	0	1	0	2	5	0

Cost


C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **3**	0	2	3	0	4	0	1	5	0	1625.25

	Depot	Client	Depot	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **4**	0	4	0	3	0	1	0	2	5	0	1599.48

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot	Cost
Individual 1	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual 2	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual 3	0	2	3	0	4	0	1	5	0	1625.25

	Depot	Client	Depot	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual 4	0	4	0	3	0	1	0	2	5	0	1599.48

Penalty Value

Initialization

MH – *Genetic Algorithm*

In the first **generation**, the **population** size depends on the nature of the problem, but a common approach is to have a **population** size with hundreds of **individuals**. Traditionally, this **population** is generated randomly, but it can start with any set of solutions.

MH – *Genetic Algorithm*

	Depot	Client	Client	Client	Depot	Client	Client	Depot	Cost
Individual 1	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual 2	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual 3	0	2	3	0	4	0	1	5	0	1625.25

	Depot	Client	Depot	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual 4	0	4	0	3	0	1	0	2	5	0	1599.48

Population

Selection

MH – *Genetic Algorithm*

For each successive **generation**, the fitter **individuals** (solutions with the best **fitness** values) are typically more likely to be selected to breed to form the **new population**. A small proportion of less fit **individuals** can also be selected to breed, helping keep the diversity of the **population**, preventing premature convergence on poor solutions.

A popular and well-studied selection method, called **Roulette Wheel Selection**, can be performed to make selections of **individuals** in a way that is directly proportionate to their **fitness**.

Each pair can generate one or more **offspring**, however as a good practice, different pairs should be used to generate the **new population**.

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1. Take from each **individual** the **fitness** value. For maximization problems, use the **fitness** values without modifications, for minimization problems go to Step 1b.

	Cost
Individual **1**	1580.74
Individual **2**	556.54
Individual **3**	1625.25
Individual **4**	1599.48

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1b. Use the inverse formula to transform a minimization problem into a maximization problem.

$$inverse = \frac{1}{1 + Cost + |\min(Cost)|}$$

	Cost
Individual **1**	1580.74
Individual **2**	556.54
Individual **3**	1625.25
Individual **4**	1599.48

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1b. Use the inverse formula to transform a minimization problem into a maximization problem.

$$inverse = \frac{1}{1 + Cost + |\min(Cost)|}$$

	Cost	$inverse$
Individual **1**	1580.74	
Individual **2**	556.54	
Individual **3**	1625.25	
Individual **4**	1599.48	

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1b. Use the inverse formula to transform a minimization problem into a maximization problem.

$$inverse = \frac{1}{1 + Cost + |\min(Cost)|}$$

	Cost	*inverse*
Individual **1**	1580.74	4.68
Individual **2**	556.54	
Individual **3**	1625.25	
Individual **4**	1599.48	

$$\frac{1}{1 + 1580.74 + |556.54|}$$

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1b. Use the inverse formula to transform a minimization problem into a maximization problem.

$$inverse = \frac{1}{1 + Cost + |\min(Cost)|}$$

	Cost	$inverse$
Individual **1**	1580.74	4.68
Individual **2**	556.54	8.98
Individual **3**	1625.25	
Individual **4**	1599.48	

$$\frac{1}{1 + 556.54 + |556.54|}$$

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1b. Use the inverse formula to transform a minimization problem into a maximization problem.

$$inverse = \frac{1}{1 + Cost + |\min(Cost)|}$$

	Cost	*inverse*
Individual **1**	1580.74	4.68
Individual **2**	556.54	8.98
Individual **3**	1625.25	4.58
Individual **4**	1599.48	

$$\frac{1}{1 + 1625.25 \; + |556.54|}$$

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1b. Use the inverse formula to transform a minimization problem into a maximization problem.

$$inverse = \frac{1}{1 + Cost + |\min(Cost)|}$$

	Cost	$inverse$
Individual **1**	1580.74	4.68
Individual **2**	556.54	8.98
Individual **3**	1625.25	4.58
Individual **4**	1599.48	4.64

$$\frac{1}{1 + 1599.48 \ + |556.54|}$$

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 1b. Use the inverse formula to transform a minimization problem into a maximization problem.

$$inverse = \frac{1}{1 + Cost + |\min(Cost)|}$$

	Cost	$inverse$
Individual **1**	1580.74	4.68
Individual **2**	556.54	8.98
Individual **3**	1625.25	4.58
Individual **4**	1599.48	4.64

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 2. Accumulate the values sequentially.

	Cost	*inverse*	*acc.*
Individual **1**	1580.74	4.68	
Individual **2**	556.54	8.98	
Individual **3**	1625.25	4.58	
Individual **4**	1599.48	4.64	

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 2. Accumulate the values sequentially.

	Cost	*inverse*	*acc.*
Individual **1**	1580.74	4.68	
Individual **2**	556.54	8.98	
Individual **3**	1625.25	4.58	
Individual **4**	1599.48	4.64	

4.68

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 2. Accumulate the values sequentially.

	Cost	*inverse*	*acc.*
Individual **1**	1580.74	4.68	4.68
Individual **2**	556.54	8.98	
Individual **3**	1625.25	4.58	
Individual **4**	1599.48	4.64	

4.68

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 2. Accumulate the values sequentially.

	Cost	*inverse*	*acc.*
Individual **1**	1580.74	4.68	4.68
Individual **2**	556.54	8.98	13.66
Individual **3**	1625.25	4.58	
Individual **4**	1599.48	4.64	

4.68 + 8.98

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 2. Accumulate the values sequentially.

	Cost	*inverse*	*acc.*
Individual **1**	1580.74	4.68	4.68
Individual **2**	556.54	8.98	13.66
Individual **3**	1625.25	4.58	18.24
Individual **4**	1599.48	4.64	

4.58 + 13.66

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 2. Accumulate the values sequentially.

	Cost	$inverse$	$acc.$
Individual **1**	1580.74	4.68	4.68
Individual **2**	556.54	8.98	13.66
Individual **3**	1625.25	4.58	18.24
Individual **4**	1599.48	4.64	22.88

4.64 + 18.24

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 2. Accumulate the values sequentially.

	Cost	*inverse*	*acc.*
Individual **1**	1580.74	4.68	4.68
Individual **2**	556.54	8.98	13.66
Individual **3**	1625.25	4.58	18.24
Individual **4**	1599.48	4.64	22.88

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 3. Divide each value of the acc. column by the sum of the inverse column.

	Cost	*inverse*	*acc.*
Individual **1**	1580.74	4.68	4.68
Individual **2**	556.54	8.98	13.66
Individual **3**	1625.25	4.58	18.24
Individual **4**	1599.48	4.64	22.88
	Sum	22.88	

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 3. Divide each value of the acc. column by the sum of the inverse column.

	Cost	*inverse*	*acc.*	%
Individual **1**	1580.74	4.68	4.68	
Individual **2**	556.54	8.98	13.66	
Individual **3**	1625.25	4.58	18.24	
Individual **4**	1599.48	4.64	22.88	
	Sum	22.88		

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 3. Divide each value of the acc. column by the sum of the inverse column.

	Cost	*inverse*	*acc.*	%
Individual **1**	1580.74	4.68	4.68	20.45%
Individual **2**	556.54	8.98	13.66	
Individual **3**	1625.25	4.58	18.24	
Individual **4**	1599.48	4.64	22.88	
	Sum	22.88		

$$\frac{4.68}{22.88}$$

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 3. Divide each value of the acc. column by the sum of the inverse column.

	Cost	*inverse*	*acc.*	%
Individual **1**	1580.74	4.68	4.68	20.45%
Individual **2**	556.54	8.98	13.66	59.70%
Individual **3**	1625.25	4.58	18.24	
Individual **4**	1599.48	4.64	22.88	
Sum		22.88		

$$\frac{13.66}{22.88}$$

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 3. Divide each value of the acc. column by the sum of the inverse column.

	Cost	*inverse*	*acc.*	%
Individual **1**	1580.74	4.68	4.68	20.45%
Individual **2**	556.54	8.98	13.66	59.70%
Individual **3**	1625.25	4.58	18.24	79.72%
Individual **4**	1599.48	4.64	22.88	
	Sum	22.88		

$$\frac{18.24}{22.88}$$

MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 3. Divide each value of the acc. column by the sum of the inverse column.

	Cost	*inverse*	*acc.*	%
Individual **1**	1580.74	4.68	4.68	20.45%
Individual **2**	556.54	8.98	13.66	59.70%
Individual **3**	1625.25	4.58	18.24	79.72%
Individual **4**	1599.48	4.64	22.88	100%
	Sum	22.88		

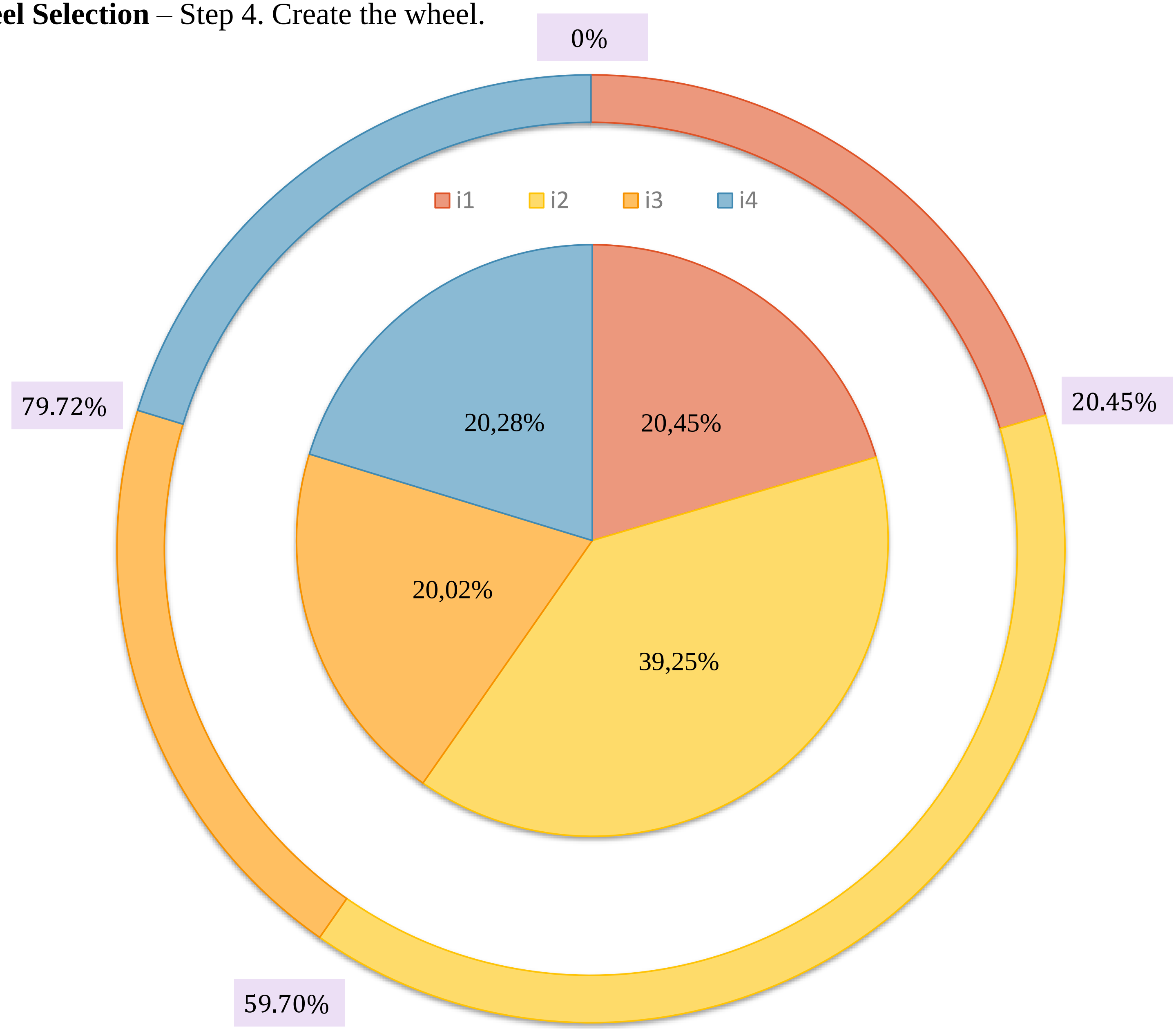
$$\frac{22.88}{22.88}$$

MH – *Genetic Algorithm*

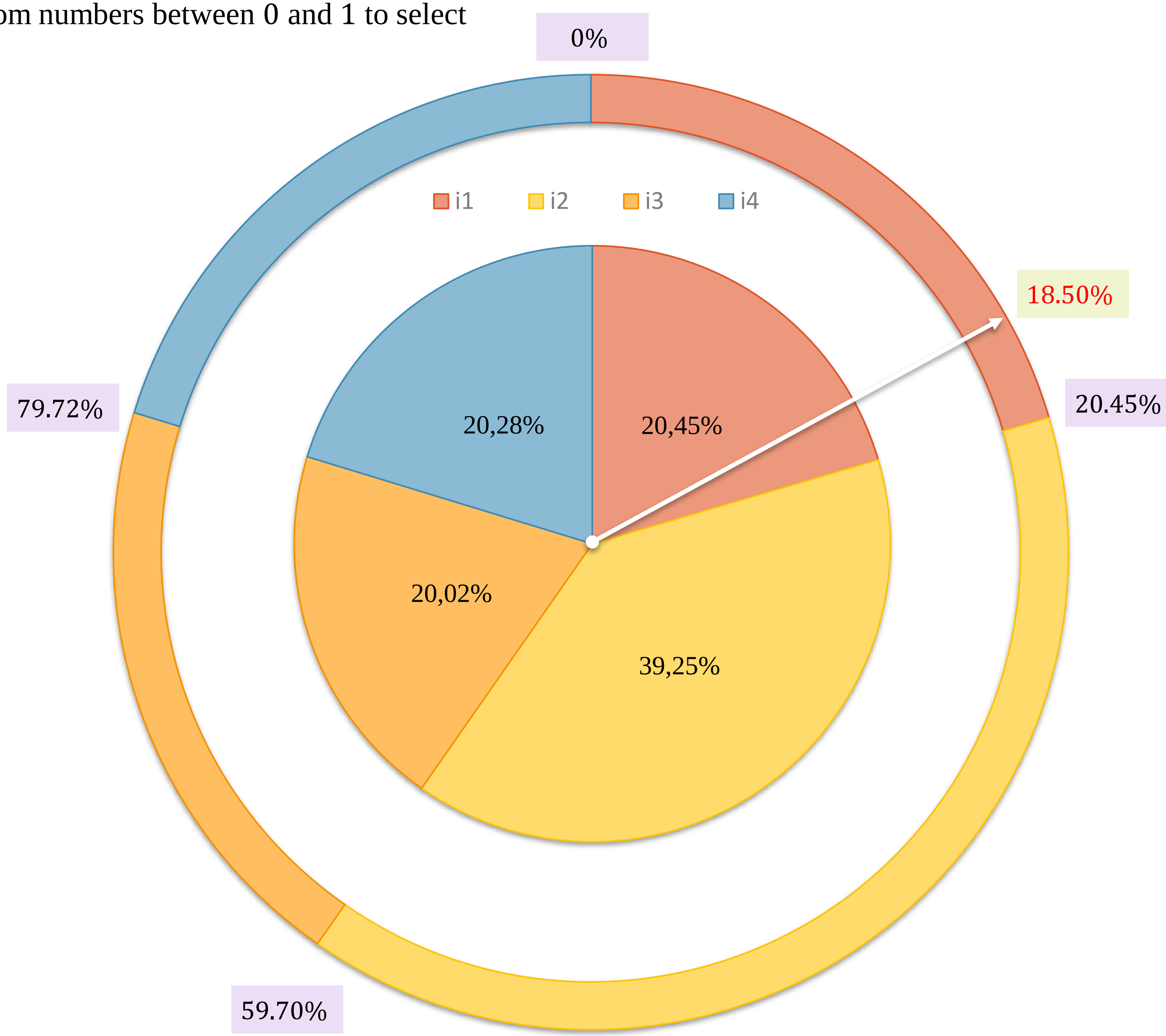
Roulette Wheel Selection – Step 3. Divide each value of the acc. column by the sum of the inverse column.

	Cost	*inverse*	*acc.*	%
Individual **1**	1580.74	4.68	4.68	20.45%
Individual **2**	556.54	8.98	13.66	59.70%
Individual **3**	1625.25	4.58	18.24	79.72%
Individual **4**	1599.48	4.64	22.88	100%
	Sum	22.88		

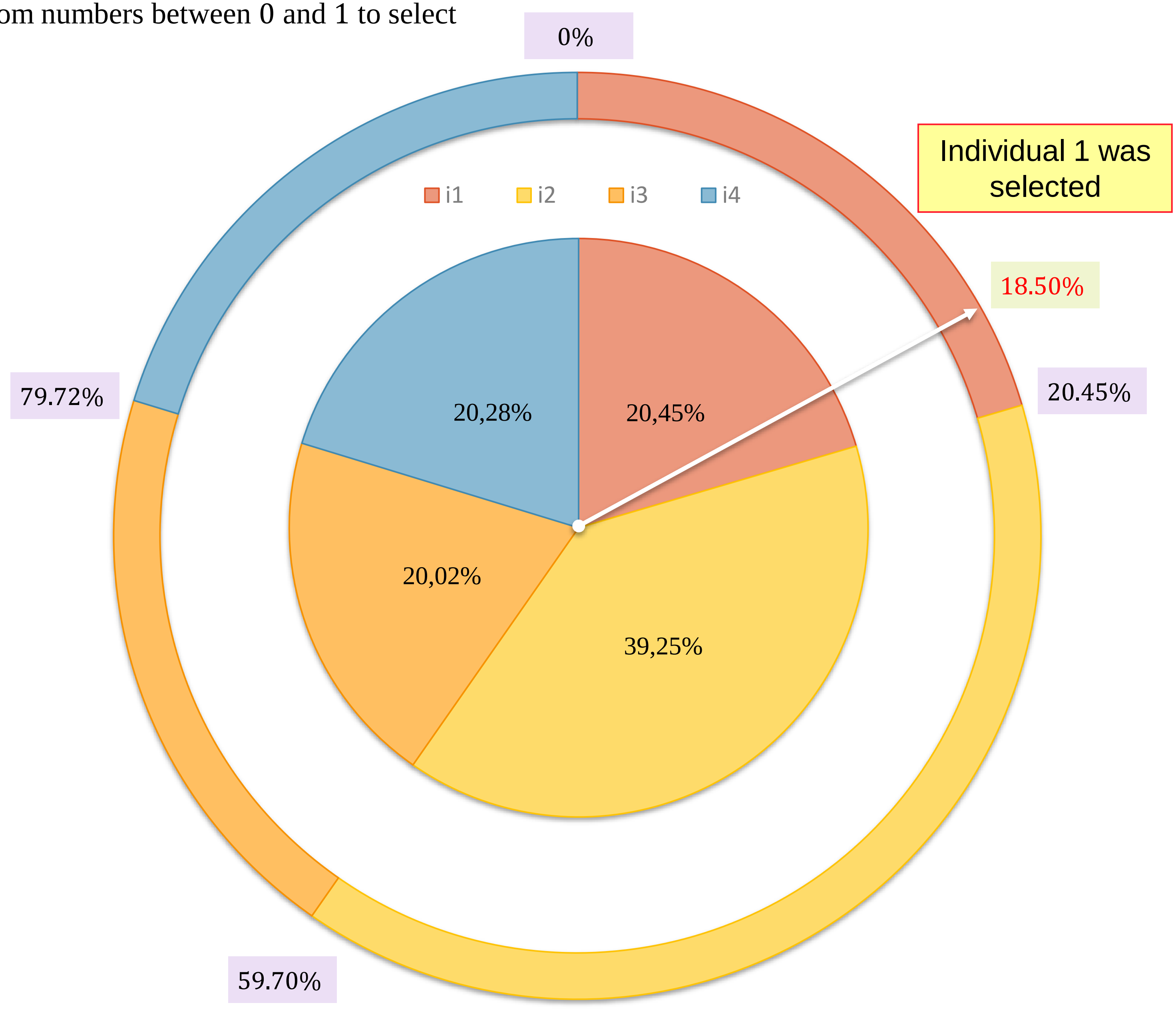
Roulette Wheel Selection – Step 4. Create the wheel.



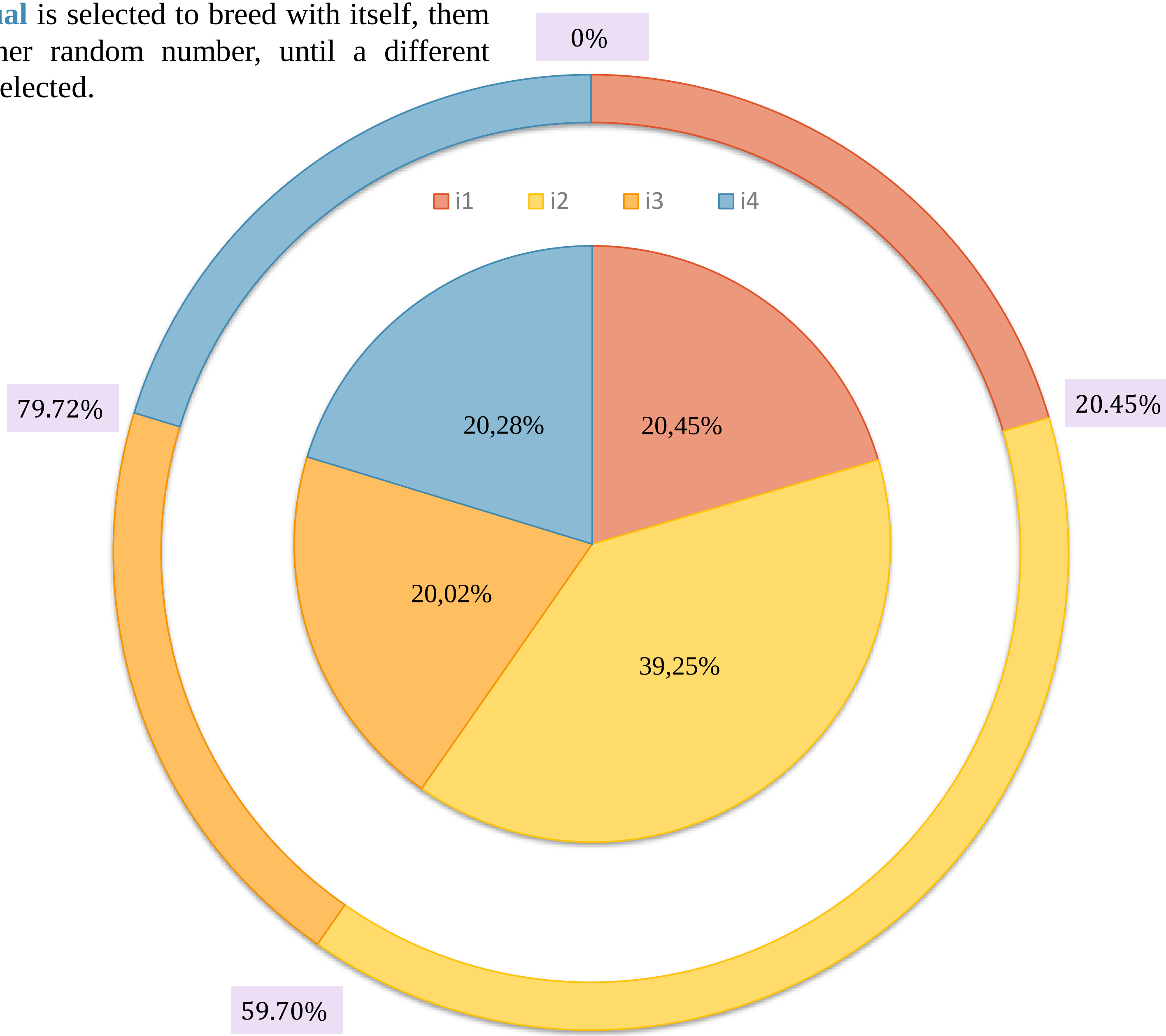
Generate random numbers between 0 and 1 to select an **individual**.



Generate random numbers between 0 and 1 to select an **individual**.

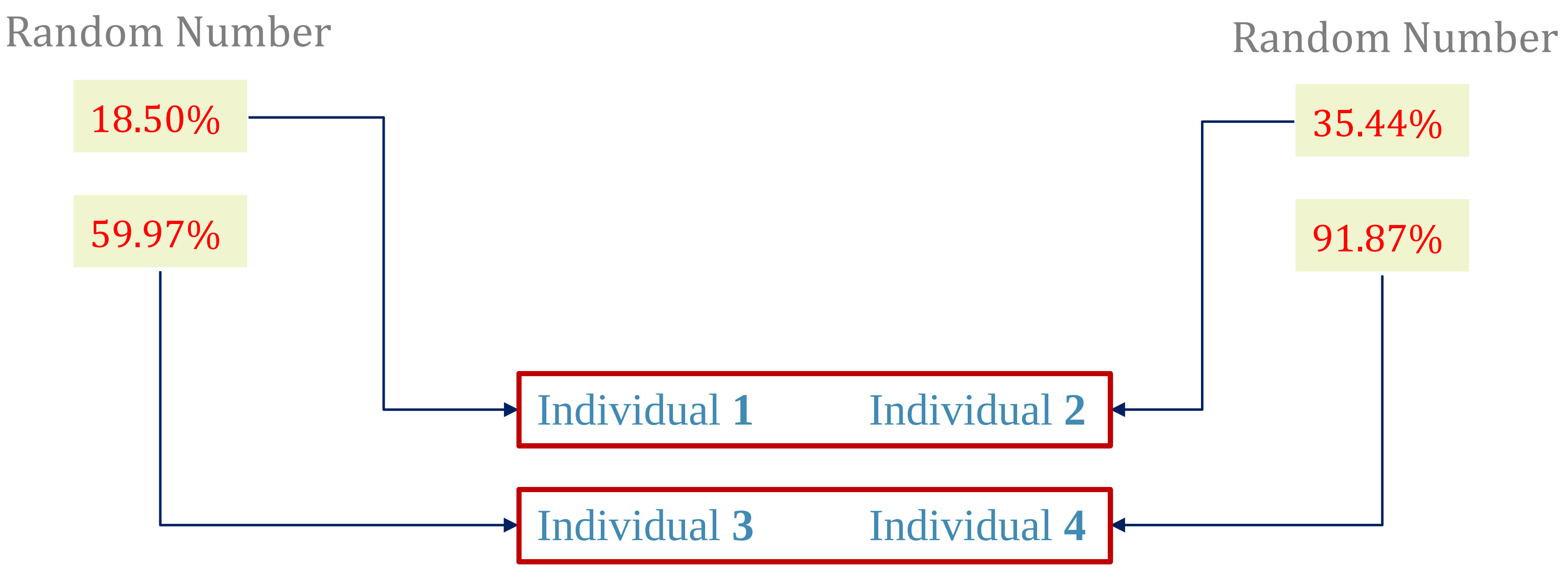


If the **individual** is selected to breed with itself, them generate another random number, until a different **individual** is selected.



MH – *Genetic Algorithm*

Roulette Wheel Selection – Step 5. Select the pairs.



Crossover

MH – *Genetic Algorithm*

The **crossover** may happen always (100% chance) or have a probability to occur, for example, a **crossover** for a selected pair may have a chance of 70% to breed, and if no **crossover** occurs then the selected pair is copied directly to the **new population**. The most common type o **crossover** operator, called **OX** (**Order Based Crossover**), is performed by copying a route from the first element of a pair ($\mathbf{parent_1}$ & $\mathbf{parent_2}$) and the remaining routes from the second element of the same pair. Adjusts may be needed to avoid repeated nodes in the **offspring**.

Optionally a local search method (**2-opt**, **k-opt**, **best insertion**, etc.) can be performed to improve the **offspring fitness**.

MH – *Genetic Algorithm*

Let's suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

MH – *Genetic Algorithm*

Let's suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

Route Randomly Selected

MH – *Genetic Algorithm*

Let's suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	Depot	Client	Client	Depot	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

Offspring

Cost

MH – *Genetic Algorithm*

Let's suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**		Cost
Individual **1**	0	3	1	4	0	2	5	0		1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Client	**Depot**	Cost
Offspring	0	3	1	4	0	

MH – *Genetic Algorithm*

Let's suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	~~3~~	~~1~~	0	5	0	~~4~~	2	0	556.54

	Depot	Client	Client	Client	**Depot**	Cost
Offspring	0	3	1	4	0	

MH – *Genetic Algorithm*

Let’s suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	Depot	Client	Client	Depot	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Individual **2**	0	~~3~~	~~1~~	0	5	0	~~4~~	2	0	556.54

	Depot	Client	Client	Client	Depot	Client	Depot	Client	Depot	Cost
Offspring	0	3	1	4	0	5	0	2	0	

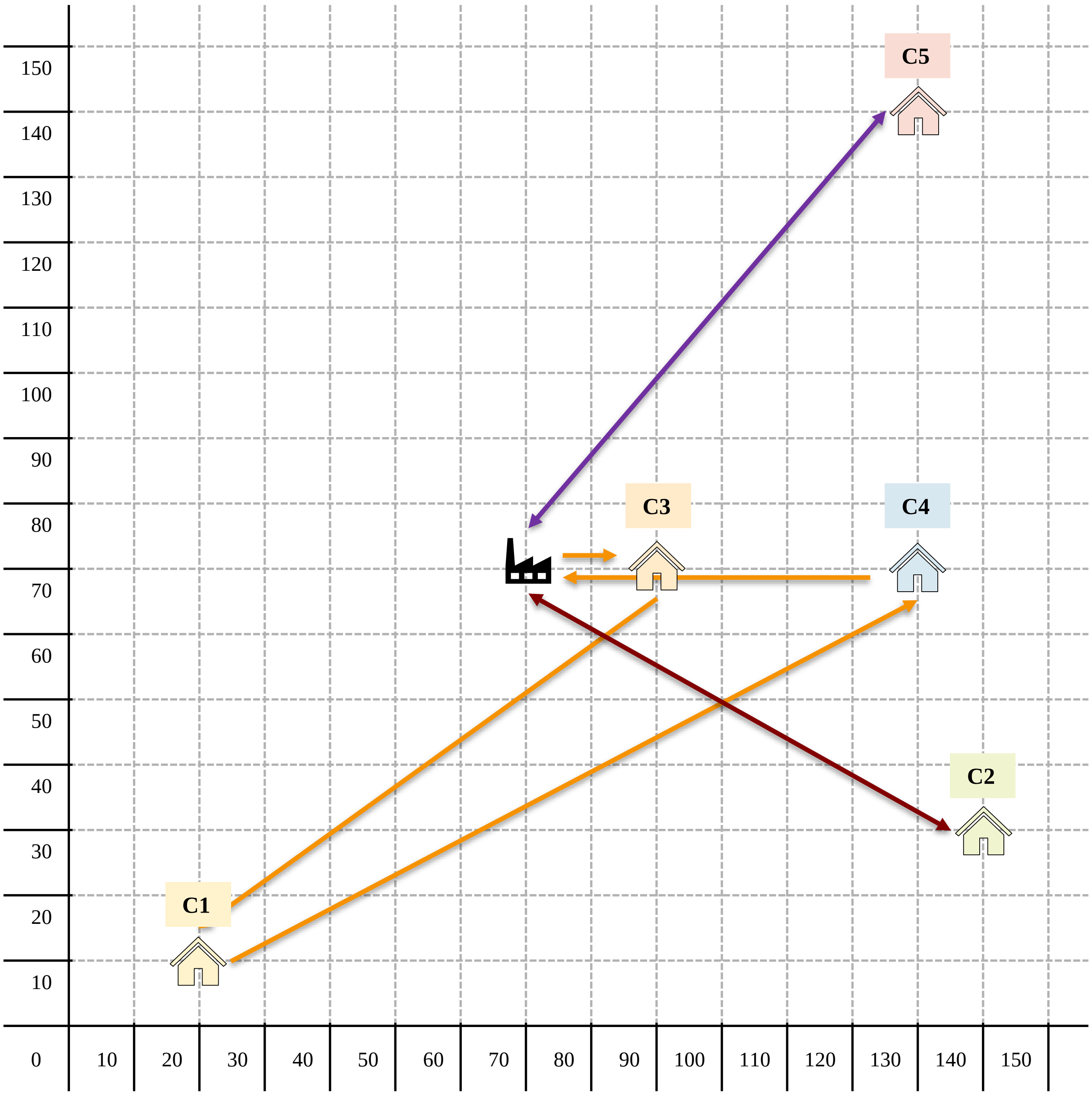
MH – *Genetic Algorithm*

Let's suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Client	**Depot**	Client	**Depot**	Client	**Depot**	Cost
Offspring	0	3	1	4	0	5	0	2	0	


C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

MH – *Genetic Algorithm*

Let's suppose that the pair (**individual 1** & **individual 2**) will breed, the pair (**individual 3** & **individual 4**) will be copied to directly to the **new population**. Therefore:

	Depot	Client	Client	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **1**	0	3	1	4	0	2	5	0	1580.74

	Depot	Client	Client	**Depot**	Client	**Depot**	Client	Client	**Depot**	Cost
Individual **2**	0	3	1	0	5	0	4	2	0	556.54

	Depot	Client	Client	Client	**Depot**	Client	**Depot**	Client	**Depot**	Cost
Offspring	0	3	1	4	0	5	0	2	0	643.13

Mutation

MH – *Genetic Algorithm*

The **mutation** is a genetic operator that, for a small chance (usually between 10% and 15%), can modify a **gene** of the **offspring**. This process ensures the genetic diversity and avoids solutions to be trapped in local maxima/minima. The most common type of **mutation** operator is called **Insertion** and it randomly selects a **gene** and inserts it in another randomly selected place of the same **individual**.

MH – *Genetic Algorithm*

Let’s suppose that the following **gene** is mutated:

	Depot	Client	Client	Client	Depot	Client	Depot	Client	Depot	Cost
Offspring	0	3	1	4	0	5	0	2	0	643.13

MH – *Genetic Algorithm*

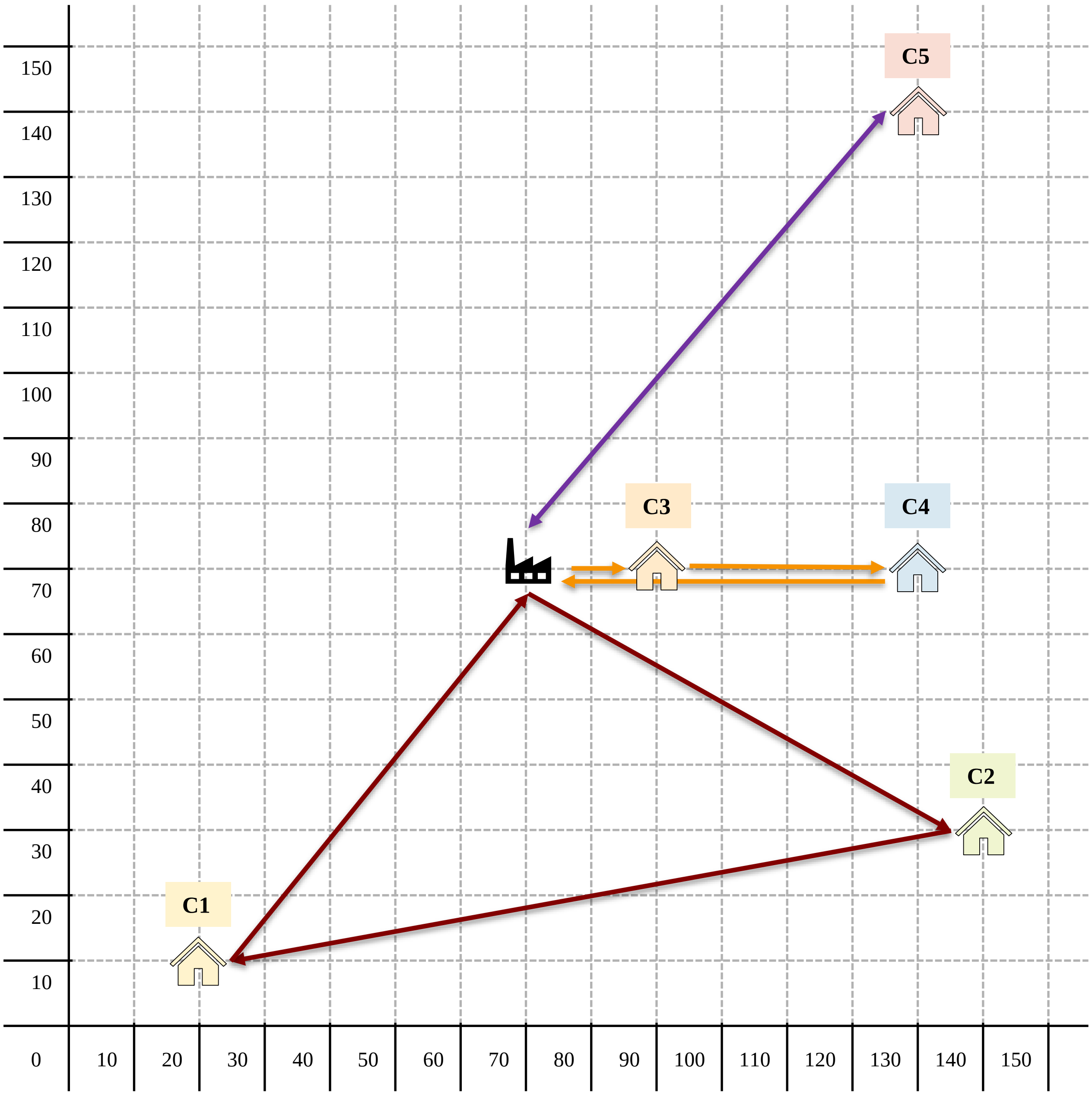
Let’s suppose that the following **gene** is mutated:

	Depot	Client	Client	Client	Depot	Client	Depot	Client	Depot	Cost
Offspring	0	3	1	4	0	5	0	2	0	643.13

MH – *Genetic Algorithm*

Let's suppose that the following **gene** is mutated:

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot	Cost
Offspring	0	3	4	0	5	0	2	1	0	



C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

MH – *Genetic Algorithm*

Let's suppose that the following **gene** is mutated:

	Depot	Client	Client	Depot	Client	Depot	Client	Client	Depot		Cost
Offspring	0	3	4	0	5	0	2	1	0		584.77

Elitism

The **elitism** is an optional process, where the **best individual** of the last the **population** is included in the **new population**. Hence:

Individual **1**	0	3	1	4	0	2	5	0			1580.74
Individual **2**	0	3	1	0	5	0	4	2	0		556.54
Individual **3**	0	2	3	0	4	0	1	5	0		1625.25
Individual **4**	0	4	0	3	0	1	0	2	5	0	1599.48

The **elitism** is an optional process, where the **best individual** of the last the **population** is included in the **new population**. Hence:

Individual **1**	0	3	1	4	0	2	5	0			1580.74
Individual **2**	0	3	1	0	5	0	4	2	0		556.54
Individual **3**	0	2	3	0	4	0	1	5	0		1625.25
Individual **4**	0	4	0	3	0	1	0	2	5	0	1599.48

The **elitism** is an optional process, where the **best individual** of the last the **population** is included in the **new population**. Hence:

Individual **1**	0	3	1	4	0	2	5	0			1580.74
Individual **2**	0	3	1	0	5	0	4	2	0		556.54
Individual **3**	0	2	3	0	4	0	1	5	0		1625.25
Individual **4**	0	4	0	3	0	1	0	2	5	0	1599.48

Elite

New Ind. **1**	0	3	1	0	5	0	4	2	0	556.54

The **elitism** is an optional process, where the **best individual** of the last the **population** is included in the **new population**. Hence:

Individual **1**	0	3	1	4	0	2	5	0			1580.74
Individual **2**	0	3	1	0	5	0	4	2	0		556.54
Individual **3**	0	2	3	0	4	0	1	5	0		1625.25
Individual **4**	0	4	0	3	0	1	0	2	5	0	1599.48

Copied

New Ind. **1**	0	3	1	0	5	0	4	2	0		556.54
New Ind. **2**	0	2	3	0	4	0	1	5	0		1625.25
New Ind. **3**	0	4	0	3	0	1	0	2	5	0	1599.48

The **elitism** is an optional process, where the **best individual** of the last the **population** is included in the **new population**. Hence:

Individual **1**	0	3	1	4	0	2	5	0			1580.74
Individual **2**	0	3	1	0	5	0	4	2	0		556.54
Individual **3**	0	2	3	0	4	0	1	5	0		1625.25
Individual **4**	0	4	0	3	0	1	0	2	5	0	1599.48

Crossover & Mutation

New Ind. **1**	0	3	1	0	5	0	4	2	0		556.54
New Ind. **2**	0	2	3	0	4	0	1	5	0		1625.25
New Ind. **3**	0	4	0	3	0	1	0	2	5	0	1599.48
New Ind. **4**	0	3	4	0	5	0	2	1	0		584.77

We discard the **old population** and use the **new population** to repeat this process all over again, or in another words, to create a new **generation**.

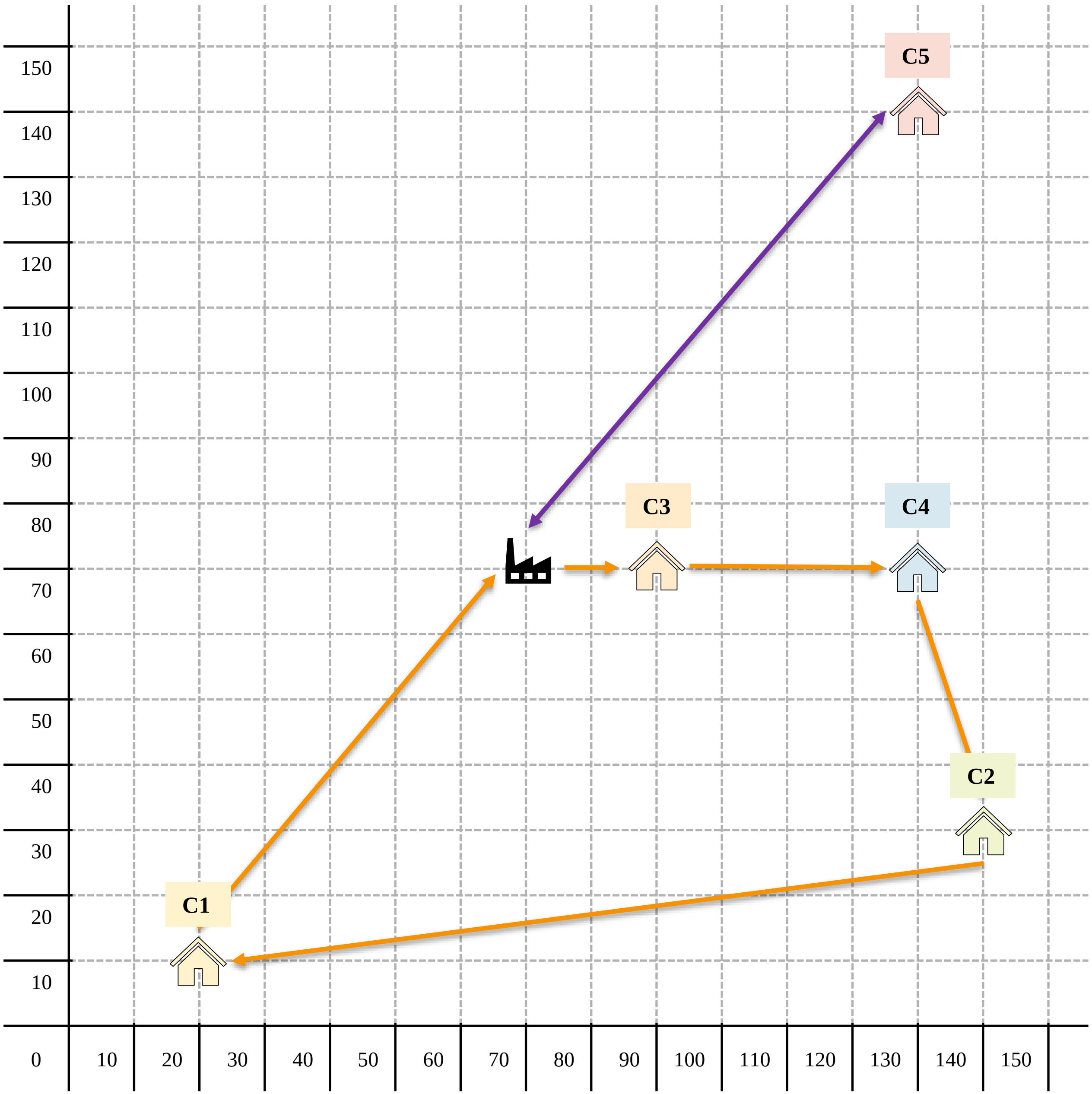
New Ind. **1**	0	3	1	0	5	0	4	2	0		556.54
New Ind. **2**	0	2	3	0	4	0	1	5	0		1625.25
New Ind. **3**	0	4	0	3	0	1	0	2	5	0	1599.48
New Ind. **4**	0	3	4	0	5	0	2	1	0		584.77

After 2,500 generations

MH – *Genetic Algorithm*

The **best individual** found was:

									$Cost$
0	3	4	2	1	0	5	0		485.38


C5
C3
C4
C2
C1
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150

MH – *Genetic Algorithm*

https://github.com/Valdecy/pyVRP

The **pyVRP** is python library that solves (using **Genetic Algorithms**): Capacitated VRP, Multiple Depot VRP, VRP with Time Windows, VRP with Homogeneous or Heterogeneous Fleet, VRP with Finite or Infinite Fleet, Open or Closed Routes, TSP, mTSP and various combination of these types.

Follow the link to try it onine.

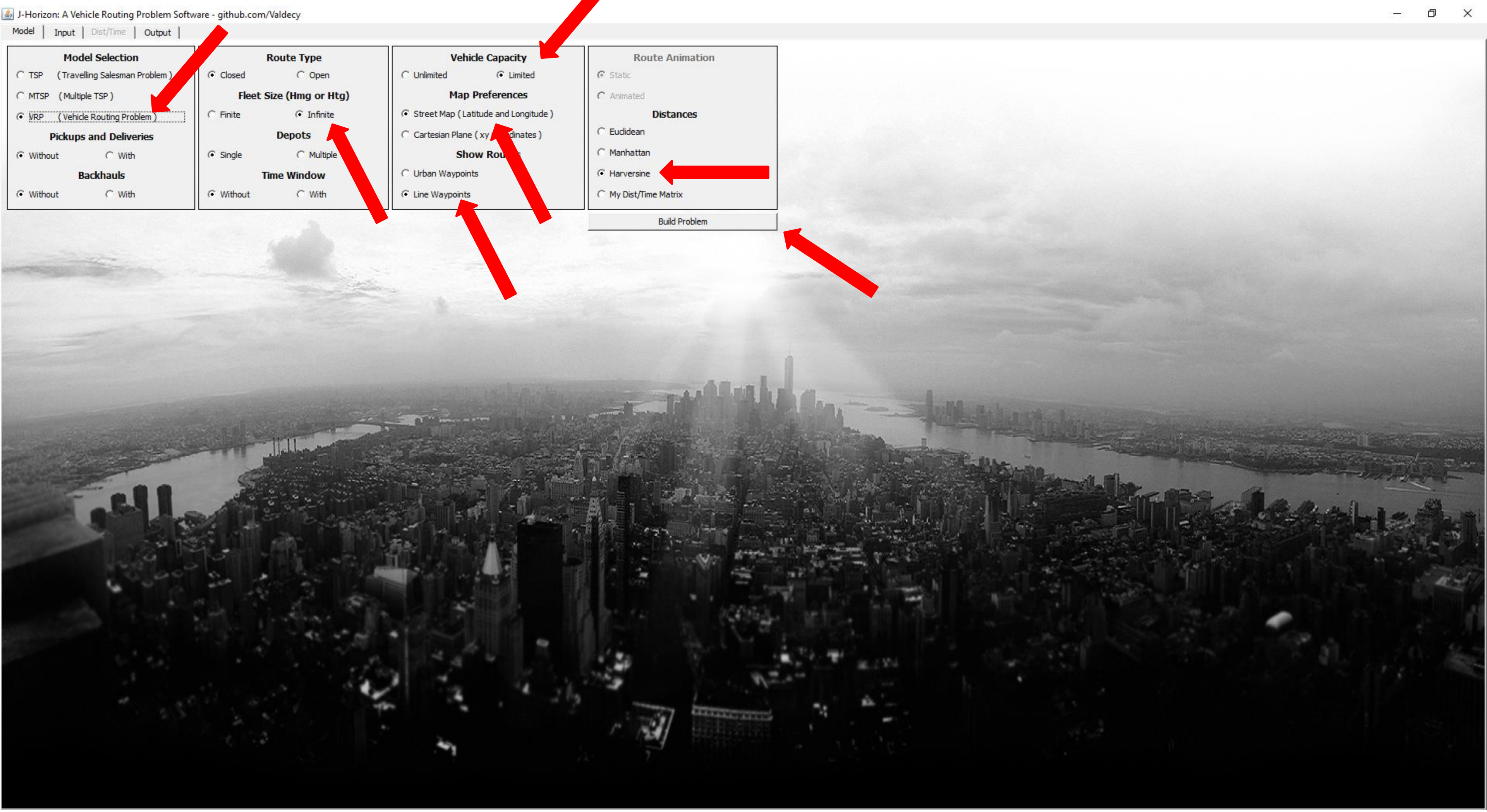
MH – *Genetic Algorithm*

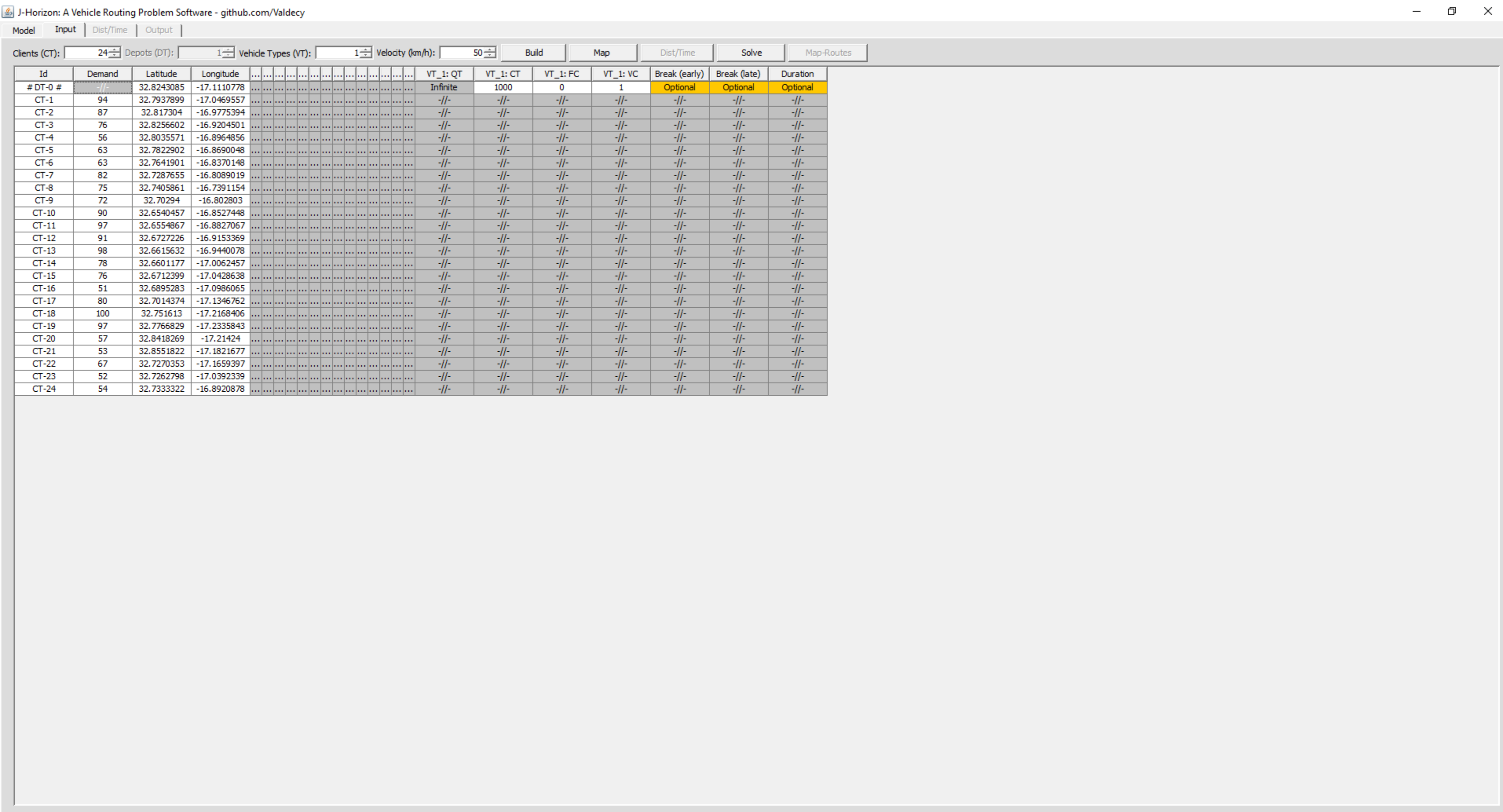
https://github.com/Valdecy/J-Horizon

The **J-Horizon** is java based vehicle problem software that uses the jsprit library to solve: Capacitated VRP, Multiple Depot VRP, VRP with Time Windows, VRP with Backhauls, VRP with Pickups and Deliveries, VRP with Homogeneous or Heterogeneous Fleet, Finite or Infinite Fleet, TSP, mTSP and various combination of these types.

Accepted inputs: Cartesian Coordinates, Latitude and Longitude, Distance Matrix, Time Matrix, Distance and Time Matrices.

J-Horizon: CVRP with LatLong Example


J-Horizon: A Vehicle Routing Problem Software - github.com/Valdecy
Model
Input
Dist/Time
Output
Model Selection
TSP (Travelling Salesman Problem)
MTSP (Multiple TSP)
VRP (Vehicle Routing Problem)
Pickups and Deliveries
Without
With
Backhauls
Without
With
Route Type
Closed
Open
Fleet Size (Hmg or Htg)
Finite
Infinite
Depots
Single
Multiple
Time Window
Without
With
Vehicle Capacity
Unlimited
Limited
Map Preferences
Street Map (Latitude and Longitude)
Cartesian Plane (xy
dinates)
Urban Waypoints
Line Waypoints
Route Animation
Static
Animated
Distances
Euclidean
Manhattan
Harversine
My Dist/Time Matrix
Build Problem



Id	Demand	Latitude	Longitude	...	...	...	...	...	...	...	...	...	...	...	...	...	...	VT_1: QT	VT_1: CT	VT_1: FC	VT_1: VC	Break (early)	Break (late)	Duration
# DT-0 #	-//-	32.8243085	-17.1110778	...	...	...	...	...	...	...	...	...	...	...	...	...	...	Infinite	1000	0	1	Optional	Optional	Optional
CT-1	94	32.7937899	-17.0469557	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-2	87	32.817304	-16.9775394	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-3	76	32.8256602	-16.9204501	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-4	56	32.8035571	-16.8964856	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-5	63	32.7822902	-16.8690048	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-6	63	32.7641901	-16.8370148	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-7	82	32.7287655	-16.8089019	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-8	75	32.7405861	-16.7391154	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-9	72	32.70294	-16.802803	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-10	90	32.6540457	-16.8527448	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-11	97	32.6554867	-16.8827067	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-12	91	32.6727226	-16.9153369	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-13	98	32.6615632	-16.9440078	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-14	78	32.6601177	-17.0062457	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-15	76	32.6712399	-17.0428638	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-16	51	32.6895283	-17.0986065	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-17	80	32.7014374	-17.1346762	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-18	100	32.751613	-17.2168406	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-19	97	32.7766829	-17.2335843	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-20	57	32.8418269	-17.21424	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-21	53	32.8551822	-17.1821677	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-22	67	32.7270353	-17.1659397	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-23	52	32.7262798	-17.0392339	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-
CT-24	54	32.7333322	-16.8920878	...	...	...	...	...	...	...	...	...	...	...	...	...	...	-//-	-//-	-//-	-//-	-//-	-//-	-//-

J-Horizon: A Vehicle Routing Problem Software - github.com/Valdecy

Model | Input | Dist/Time | Output

Problem	
Indicator	**Value**
Clients	24
Services	24
Shipments	0
Breaks	0
Fleet Size	INFINITE

Solution	
Indicator	**Value**
Costs	164.59
Vehicles	2
Unassig. Jobs	0

Detailed Solution

Route	Vehicle	Activity	Job	ArrTime (minutes)	EndTime (minutes)	Costs
1	DT_0_VT-1#1	start	-	***	0	0
1	DT_0_VT-1#1	service	2	15.008	15.008	12.506
1	DT_0_VT-1#1	service	3	21.507	21.507	17.923
1	DT_0_VT-1#1	service	4	25.499	25.499	21.249
1	DT_0_VT-1#1	service	5	29.69	29.69	24.741
1	DT_0_VT-1#1	service	6	34.017	34.017	28.347
1	DT_0_VT-1#1	service	7	39.701	39.701	33.085
1	DT_0_VT-1#1	service	8	47.694	47.694	39.745
1	DT_0_VT-1#1	service	9	56.434	56.434	47.028
1	DT_0_VT-1#1	service	10	65.04	65.04	54.2
1	DT_0_VT-1#1	service	11	68.413	68.413	57.011
1	DT_0_VT-1#1	service	12	72.741	72.741	60.618
1	DT_0_VT-1#1	service	24	81.242	81.242	67.702
1	DT_0_VT-1#1	service	1	100.406	100.406	83.671
1	DT_0_VT-1#1	end	-	108.672	***	90.56
2	DT_0_VT-1#1	start	-	***	0	0
2	DT_0_VT-1#1	service	23	15.369	15.369	12.807
2	DT_0_VT-1#1	service	13	29.117	29.117	24.264
2	DT_0_VT-1#1	service	14	36.113	36.113	30.095
2	DT_0_VT-1#1	service	15	40.487	40.487	33.74
2	DT_0_VT-1#1	service	16	47.209	47.209	39.341
2	DT_0_VT-1#1	service	17	51.561	51.561	42.967
2	DT_0_VT-1#1	service	22	56.46	56.46	47.05
2	DT_0_VT-1#1	service	18	63.049	63.049	52.541
2	DT_0_VT-1#1	service	19	66.887	66.887	55.739
2	DT_0_VT-1#1	service	20	75.848	75.848	63.207
2	DT_0_VT-1#1	service	21	79.862	79.862	66.552
2	DT_0_VT-1#1	end	-	88.836	***	74.03


Street Map
Porto Moniz
Santa
Ribeira da Janela
Ponta do Pargo
Prazeres
Jardim do Mar
Estreito da Calheta
Calheta
Arco da Calheta
Canhas
Ponta do Sol
Ribeira Brava
Campanário
Estreito de Câmara de Lobos
Jardim da Serra
Câmara de Lobos
Área Protegida do Cabo Girão
São Martinho
Curral das Freiras
Ponta Delgada
Arco de São Jorge
São Jorge
Reserva Natural do Sítio da Rocha do Navio
Ilha
Santana
Faial
São Roque do Faial
Santo António da Serra
Machico
Água de Pena
Santa Cruz
Pedras do Rochão
Achada da Camacha
Parque Ecológico do Funchal
Monte
São Roque
Imaculado Coração de Maria
Choupana
Funchal
Bica de Pau
Serralhal
Pedra Mole
Caniço
Portinho
Porto Novo
Gaula
Cancela
Caniço de Baixo
Reserva Natural Parcial do Garajau
VE 2
VE 4
ER 105
ER 103
ER 107
ER 110
ER 214
VE 1
Leaflet | © OpenStreetMap Contributors