# The Exact Solution of Vehicle Routing Problem by Mixed Integer Linear Programming in Matlab

Jaromír Zahrádka[1]

**Abstract.** This contribution comes up with a specific solution of the vehicle routing problem. The driver has to deliver the goods from the central warehouse to n customers as efficiently as possible. Each customer has ordered goods that fill a certain number of containers. Each customer point of delivery is given by GPS coordinates. The objective of the solution is to select the number of vehicles and their routes between customers in such a way that the total travel time, including the time for unloading the goods, is as short as possible. Each delivery point is visited only once by one of the vehicles. All used vehicles have a pre-limited capacity of containers. All vehicles return to the central warehouse. In this contribution, the algorithm of the exact solution of the vehicle routing problem was created, which can be used in general for any number n of customers. The algorithm is implemented in Matlab code.

**Keywords:** Matlab code, mixed integer linear programming, optimization, point of delivery, vehicle routing problem

**JEL Classification:** C64
**AMS Classification:** 68W04, 90C11, 05C20

# 1 The Vehicle Routing Problem

The vehicle routing problem (VRP) is described in [1]. Our solution came from the principles of integer programming and exact algorithms which are listed in [4, 5]. The basic general principles of operating intelligent transport systems published in [2] are respected. The VRP solution concept used in this paper is similar to that of the traveling salesman problem in [6].

## 1.1 Mathematical Formulation

The vehicle routing problem can be presented as the subsequent graph problem. Let $G = (V, E)$ be a complete directed graph where $V = \{0,1,...,n\}$ is the nodes set and $E$ is the set of all oriented arcs. The truck depot is marked by $0$. Nodes $i = 1, 2, ..., n$ correspond to the customers, each with a number $q_i$ of demand containers, which form the row vector $\mathbf{q} = (q_1, q_2, \dots, q_n)$. Each oriented $arc(i, j)$ is associated with non-negative values $d_{ij}$ travel distance (in meters), and $c_{ij}$ travel time (in sec) from node $i$ to node $j$. For easier references, let $I = \{1, \dots, n\}$, and $I_0 = \{0\} \cup I$. The distance matrix $\mathbf{D} = \left(d_{ij}\right)_{i,j \in I_0}$ and time distance matrix $\mathbf{C} = \left(c_{ij}\right)_{i,j \in I_0}$ are the non-negative and asymmetric.

The VRP consists of finding a collection of $k$ simple cycles, each corresponding to a vehicle route with minimal sum of the distances or time distances of the cycle arcs, such that:

a) each cycle visits the depot - node 0;
b) each vertex $j \in I$ is visited by exactly one cycle;
c) the sum of delivered containers during a cycle does not exceed the vehicle capacity $Q$.

The order of the customers visited is not limited. For each customer $i \in I$ let $m_i$ be the service time associated with the unloading of goods and dealing with the customer. The service times form a row vector $\mathbf{m} = (m_1, m_2, ..., m_n)$.

[1] University of Pardubice, Department Mathematics and Physics, Studentská 95, 53210 Pardubice, jaromir.zahradka@upce.cz.

## 1.2 Mathematical Solution

The main method for the exact solution of the vehicle routing problem is optimization implemented by using mixed integer linear programming, which is generally described by

$$\min_{\mathbf{V}} (\mathbf{f} \cdot \mathbf{V}) \; subject\ to \begin{cases} \mathbf{V}_{intcon}\ are\ integers \\ \mathbf{A} \cdot \mathbf{V} \le \mathbf{b} \\ \mathbf{A}_{eq} \cdot \mathbf{V} = \mathbf{b}_{eq} \\ \mathbf{l}_b \le \mathbf{V} \le \mathbf{u}_b \ . \end{cases} \tag{1}$$

The vector $\mathbf{V}$ is the column vector of all flow variables; $\mathbf{f} \cdot \mathbf{V}$ is the objective function with the coefficients contained in the row vector $\mathbf{f}$ ; $\mathbf{V}_{intcon}$ is the list of variable indices of the vector $\mathbf{V}$ that takes only the integer values; $\mathbf{A} \cdot \mathbf{V} \le \mathbf{b}$ denotes the system of inequality constrains; $\mathbf{A}_{eq} \cdot \mathbf{V} = \mathbf{b}_{eq}$ denotes of the system of linear equations; and $\mathbf{l}_b \le \mathbf{V} \le \mathbf{u}_b$ indicates the lower and upper limits of flow variables.

The linear inequality matrix $\mathbf{A}$ is specified as a matrix of real numbers, which are the linear coefficients in the system of inequality constraints. The right sides of the inequality constraints are included in column vector $\mathbf{b}$ . The linear equality constraint matrix $\mathbf{A}_{eq}$ is specified as the matrix of real numbers, which are the linear coefficients in the system of equations. The right sides of the system of equalities are included in column vector $\mathbf{b}_{eq}$ .

The lower and upper bounds of flow variables (components of $\mathbf{V}$ ) are specified as real numbers - elements of column vectors $\mathbf{l}_b$ , and $\mathbf{u}_b$ .

The core of the practical solution of VRP is to find the appropriate number of $k$ cycles in the graph $G$ which includes all nodes of the graph and which gives the shortest total length of all cycles (or the shortest total driving time of all vehicles). For this purpose, integer flow variables $x_{ij}$ for $i, j \in I_0$ are introduced, which can only take the values 0 or 1 (binary values). The value $x_{ij} = 1$ means that the arc from node $i$ to $j$ is included in one cycle and the value $x_{ij} = 0$ means that the corresponding arc is not included. For systemic reason variables, $x_{ii}$ are used but all are fixed $x_{ii} = 0$, for each $i \in I_0$. Variables $x_{ij}$ are elements of a matrix $\mathbf{X} = (x_{ij})_{i,j \in I_0}$, and the number of $x_{ij}$ variables is $(n+1)^2$.

In our work we use other specific integer flow variables, $y_i$, for each $i \in I$, which indicate the number of containers that were unloaded to customers during the journey from the depot up to and including the node $i$ . The variables $y_i$ are $n$ elements of the vector $\mathbf{y} = (y_1, y_2, \ldots, y_n)$. The sum of all flow variables is $(n+1)^2 + n$.

Elements of matrix $\mathbf{X} = (x_{ij})_{i,j \in I_0}$ and vector $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ are arranged in row vector $\mathbf{V}$ so that, the transposed vector $\mathbf{V}$ is $\mathbf{V}^T = (\mathbf{X}, \mathbf{y}) = (x_{00}, x_{01}, \ldots, x_{0n}, x_{10}, x_{11}, \ldots x_{1n}, \ldots, x_{n0}, x_{n1}, \ldots, x_{nn}, y_1, y_2, \ldots, y_n)$.

The solution of VRP is realized like the optimal solution of a mixed-integer linear programming problem:

$$\min_{(\mathbf{X},\mathbf{y})} \left\{ \sum_{i,j=0}^{n} d_{ij} \cdot x_{ij} + \sum_{i=1}^{n} y_i / 100 \right\} \text{ subject to} \tag{2}$$

$$x_{ij},\ i,j \in I_0 \text{ are binary, } x_{ij} \in \{0,1\} \tag{3}$$

$$y_i,\ i \in I \text{ are integer} \tag{4}$$

$$Q\, x_{ij} + y_i - y_j \le Q - q_j, \quad i,j \in I,\ i \ne j \tag{5}$$

$$\sum_{j \in I_0} x_{ij} = 1, \quad i \in I \tag{6}$$

$$\sum_{i \in I_0} x_{ij} = 1, \quad j \in I \tag{7}$$

$$\sum_{j \in I} x_{0j} - \sum_{i \in I} x_{i0} = 0 \tag{8}$$

$$x_{ii} = 0, \quad i \in I_0 \tag{9}$$

$$0 \le x_{ij} \le 1, \quad i, j \in I_0 \tag{10}$$

$$q_j \le y_j \le Q, \quad j \in I \tag{11}$$

In the expressed model (2) is minimized the linear optimization function

$$\mathbf{f} = \sum_{i,j=0}^{n} d_{ij} x_{ij} + \sum_{i=0}^{n} y_i / 100 \,. \tag{12}$$

The main part $\sum_{i,j=0}^{n} d_{ij} x_{ij}$ (the sum of distance in meters) of the optimized function guarantees finding a collection of cycles such that the sum of their lengths is minimal. The second part $\sum_{i=1}^{n} y_i / 100$ of the optimization function (12) is about four orders of magnitude smaller and does not affect the optimization according to the smallest length, but it guarantees that the generated flow values of the $y_i$ is indeed the smallest as possible. Constraint (5) defines $n(n-1)$ conditions between flow variables $x_{ij}$ and numbers $y_i$ and $y_j$ of unloaded containers at nodes $i$, and $j$. In the case $x_{ij} = 1$, the inequality (5) expresses the relationship $y_j \ge y_i + q_j$. Thanks to the inclusion of the variables $y_1, y_2, \ldots, y_n$ in the optimized function (12), it is ensured that the values of $y_j$ will always be minimal, i.e. the equation $y_j = y_i + q_j$ will be applied instead of an inequality. In the case $x_{ij} = 0$, for $i \ne j$, the inequality (5) expresses the relationship $y_j \ge y_i + q_j - Q$.

Statements (6) and (7) declare $2n$ equations, which express that only one arc leads from each node $i \in I$ and only one arc leads to each node $j \in I$. Statement (8) declares that the number of arcs leaving the node number 0 is equal to the number of arcs entering the node 0. Statement (9) declares that each $x_{ii} = 0$ (no loops).

The inequalities in (10) declare that the lower and upper bounds of flow variables $x_{ij}$ are 0 and 1. The inequalities in (11) express that the each flow variable $y_j$ is greater than or equal to $q_j$, and each $y_j$ can't be greater than the capacity $Q$ of vehicle.

## 1.3 Transformation into Matlab

The distance matrix $\mathbf{D} = \left(d_{ij}\right)_{i,j \in I_0}$ is transformed into Matlab environment as matrix `D`, with the row and column indices *i, j = 1,2, ... ,n+1*. The components `D(i,j)` correspond to the distances $d_{i-1\,j-1}$ of the nodes $i-1$ and $j-1$. Similarly time distance matrix $\mathbf{C} = \left(c_{ij}\right)_{i,j \in I_0}$, is transformed into matrix `C`, i.e. each component `C(i,j)` corresponds to the driving time distance $c_{i-1\,j-1}$.

| Variable | n | $Q$ | **A** | **b** | $\mathbf{A}_{eq}$ | $\mathbf{b}_{eq}$ | $\mathbf{l}_b$ | $\mathbf{u}_b$ | **m** | **q** | **v** | $\mathbf{V}_{intcon}$ | **D** | **C** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Matlab identifier | `n` | `Q` | `A` | `b` | `Aeq` | `beq` | `lb` | `ub` | `m` | `q` | `V` | `intcon` | `D` | `C` |

| Variable | $d_{ij}$ | $c_{ij}$ | **X** | **X** | $x_{ij}$ |
|---|---|---|---|---|---|
| Matlab identifier | `D(i+1,j+1)` | `C(i+1,j+1)` | `X` | `V((1:(n+1)^2,1)` | `V((n+1)*i+j+1,1)` |

| Variable | **y** | **y** | $y_i$ | $d_{TotLgth}$ | $t_{TotDur}$ |
|---|---|---|---|---|---|
| Matlab identifier | `y` | `V((n+1)^2+1:(n+1)^2+n,1)` | `V((n+1)^2+i)` | `TotLgth` | `TotDur` |

**Table 1** The transformations of variables to Matlab identifiers

The created procedure for VRP solving in the Matlab code is included in the M-function `VRP_SOLVER_Za.m` and it is fully listed as an Appendix at the end of the article. The key to transforming used variables into Matlab identifiers can be found in Table 1. The main output variable is the column vector *V* of flow variables, which is obtained

as an output of the command `V=intlinprog(f,intcon,A,b,Aeq,beq,lb,ub,[],options)` (App. row No. 16). A more detailed explanation of command `intlinprog` can be found in the User's Guide [3].

For the solution of VRP via the `intlinprog` command, all flow variables are arranged in a column vector `V` with $(n+1)^2+n$ components. The first $(n+1)^2$ flow variables are integer variables $x_{ij}$, and each variable $x_{ij}$, $i, j \in I_0$ is represented by the Matlab flow variable `V(i*(n+1)+j+1,1)`. The last $n$ flow variables of `V` are the values $y_1, y_2, \ldots, y_n$, and each variable $y_i$, $i \in I$ is represented by `V((n+1)^2+i,1)`.

The objective function of the mixed-integer linear programming problem (12) is, in the Matlab code, expressed like `f'*V`, where `f` is a column vector of coefficients with $(n+1)^2+n$ components. The first $(n+1)^2$ components are elements of the distance matrix `D` so that `f((i-1)*(n+1)+j, 1)=D(i, j)`, $i,j \in \{1,2,\ldots,n+1\}$ (the Appendix, row No. 14).

Alternatively we can use the elements of the time distance matrix `C` so that `f((i-1)*(n+1)+j, 1)=C(i, j)`, $i,j \in \{1,2,\ldots,n+1\}$ (the Appendix, row No. 14, listed as a note after the `%` sign). For the last $n$ components of `f` we use the value 0.01 (the Appendix, row No. 15). It is important to guarantee the optimization of the first $(n+1)^2$ terms. The optimization of the last n-flow variables is of secondary concern.

The vector `intcon` in the command `intlinprog` specifies indices of flow variables, which are taken integers (listend at the end of row No. 15) i.e. command `intcon = 1:(n+1)^2+n`. This means in this case that all flow variables are integers. The constraints (5) give the system of $n^2-n$ linear inequalities with $(n+1)^2+n$ variables. The matrix `A` of system inequalities and the column vector *b* of right sides are created for any $n$ in Matlab code statements on lines No. 3 to 6 in the Appendix. The constraints (6), (7) and (8) give the system of $n^2$ linear equalities with $(n+1)^2+n$ variables. The matrix `Aeq` of system equalities and the column vector `beq` of right sides are created for any $n$ in the Matlab code statements on lines No. 3 and 7 to 11 of the Appendix.

The other two input variables of the `intlinprog` command (row No. 2 in the Appendix) are the column vectors `lb` and `ub` of lower and upper bounds of the flow variables. With respect to the relations (9), (10), (11) the components of vectors `lb` and `ub` are filled by commands on rows No. 11, 12, 13 in the Appendix. By installation of input variables `f`, `intcon`, `A`, `b`, `Aeq`, `beq`, `lb`, `ub`, and `options` (row No. 2 of the Appendix) in the command `intlinprog`, and running it (the line No. 16), Matlab gives the optimal VRP solution, i.e. the vector of flow variables `V`. The first $(n+1)^2$ components of `V` can be reshaped to square matrix `X` (you can see it on row No. 17 in the Appendix). The last $n$ components of `V` are the components of the vector `y`.

The variables `X(i,j)` which take the value 1, determine the arcs of cycles which make an optimized solution. The two for-cycle commands on lines No. 17, 18, 19, 20 allow calculation of the total length of cycles `TotLgth` and the total driving time `TotDur` of all used vehicles, which is increased by customer service times (command on row No. 17 of the Appendix). The input data for the M-function `SOLVER_VRP_Za.m`, the execution, output variable processing, and drawing output cycles are have to be done using a startup M-script that is not listed in this article.

# 2 Applied Task

The created M-function `SOLVER_VRP_Za.m` was applied to the delivery of food products from a central warehouse (depot) in the east region of Bohemia to other parts of the Czech Republic. Nineteen customers were

| | Depot | Customers | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $i$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $E_i$ (°) | 15.90839 | 16.24160 | 16.08722 | 14.69055 | 15.27025 | 16.66231 | 16.98885 | 14.90985 | 17.11870 | 15.58698 |
| $N_i$ (°) | 50.02656 | 50.57513 | 50.27706 | 49.41570 | 49.95616 | 49.72949 | 49.95986 | 50.42491 | 49.46988 | 49.61639 |
| $q_i$ | - | 7 | 6 | 5 | 4 | 4 | 3 | 3 | 3 | 3 |
| | Customers | | | | | | | | | |
| $i$ | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| $E_i$ (°) | 14.85888 | 14.74052 | 15.63217 | 15.36320 | 15.598478 | 15.30779 | 15.04006 | 15.22476 | 16.65998 | 15.99651 |
| $N_i$ (°) | 49.99393 | 50.15575 | 50.35722 | 50.43471 | 49.42432 | 50.73916 | 50.76432 | 49.42156 | 49.49256 | 49.84431 |
| $q_i$ | 3 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 3 |

**Table 2** The deport and customers GPS coordinates $E_i$, $N_i$ and numbers of delivered containers $q_i$

selected for the application task. Each customer ordered a certain number of containers filled with goods. The location of the depot and the location of the customers are indicated using GPS coordinates in Table 2, along with the number of delivered containers. Figure 1 shows the locations of the depot and customers. The distances (in meters) and time distances (in seconds) between the locations of the depot and customers, i.e. the elements of the distance matrix **D** and time distance matrix **C** , were taken from the logistics company operating the transport. They are used in the setup program. Distances and time distances can be realistically obtained based on GPS coordinates using geographic navigation applications specialized for trucks.

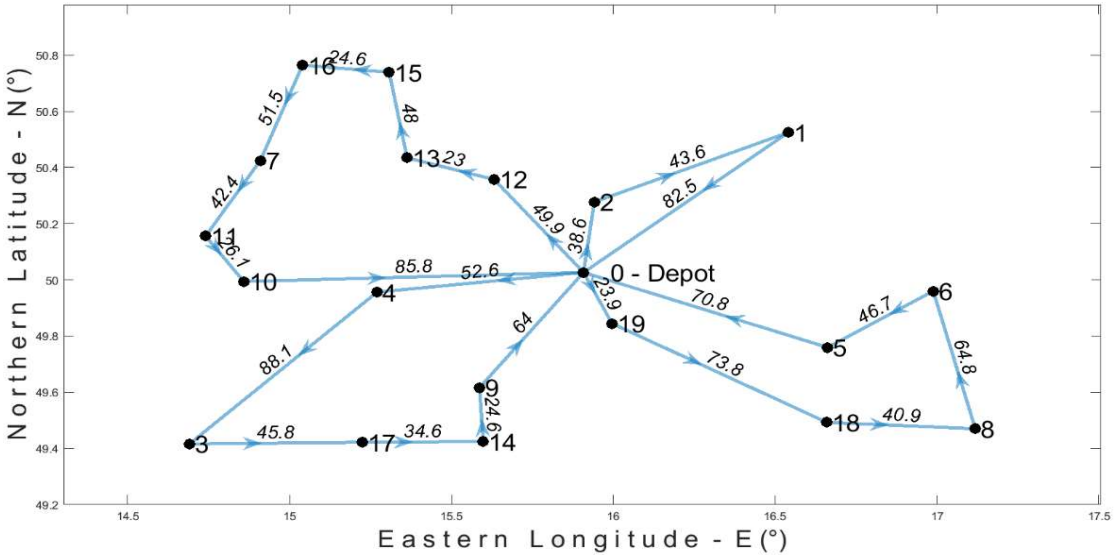


**Figure 1** The optimal solution VRP for minimal traveling distance

By running the *VRP_SOLVER_Za.m* function with above-given input parameters, four cycles were found to be the optimal solution. This means that the sum of all four travel distances for trucks is minimal. All four cycles are shown in Figure 1. The list of customers on the cycle route with times of arrival and departure to and from

| **Cycle 1** | **Start** | Customers | | | | | | | Return | Cycle lengths (km) | Duration of Cycles (hh:mm:ss) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $i$ | 0 | 2 | 1 | - | - | - | - | - | 0 | | |
| $q_i$ | - | 6 | 7 | - | - | - | - | - | - | | |
| $t_{arr_i}$ | - | 8:00:00 | 9:22:24 | - | - | - | - | - | 11:34:32 | 164.716 | 4:25:22 |
| $t_{dep_i}$ | 7:09:10 | 8:26:00 | 9:49:24 | - | - | - | - | - | - | | |
| **Cycle 2** | **Start** | Customers | | | | | | | **Return** | | |
| $i$ | 0 | 4 | 3 | 17 | 14 | 9 | - | - | 0 | | |
| $q_i$ | - | 4 | 5 | 1 | 2 | 3 | - | - | - | | |
| $t_{arr_i}$ | - | 8:00:00 | 10:16:35 | 11:36:17 | 12:37:44 | 13:29:29 | - | - | 15:12:00 | 309.575 | 8:16:02 |
| $t_{dep_i}$ | 6:55:58 | 8:24:00 | 10:41:35 | 11:57:17 | 12:59:44 | 13:52:29 | - | - | - | | |
| **Cycle 3** | **Start** | Customers | | | | | | | **Return** | | |
| $i$ | 0 | 12 | 13 | 15 | 16 | 7 | 11 | 10 | 0 | | |
| $q_i$ | - | 2 | 1 | 1 | 2 | 3 | 1 | 3 | - | | |
| $t_{arr_i}$ | - | 8:00:00 | 8:47:14 | 10:04:12 | 10:54:39 | 12:19:57 | 13:41:02 | 14:37:21 | 16:42:36 | 351.270 | 9:38:38 |
| $t_{dep_i}$ | 7:03:58 | 8:22:00 | 9:08:14 | 10:25:12 | 11:16:39 | 12:42:57 | 14:02:02 | 15:00:21 | - | | |
| **Cycle 4** | **Start** | Customers | | | | | | | **Return** | | |
| $i$ | 0 | 19 | 18 | 8 | 6 | 5 | - | - | 0 | | |
| $q_i$ | - | 3 | 1 | 3 | 3 | 4 | - | - | - | | |
| $t_{arr_i}$ | - | 8:00:00 | 9:55:50 | 11:05:07 | 12:53:25 | 14:09:34 | - | - | 15:52:44 | 351.442 | 8:25:28 |
| $t_{dep_i}$ | 7:27:16 | 8:23:00 | 10:16:50 | 11:28:07 | 13:16:25 | 14:33:34 | - | - | - | | |
| | | | | | | | | | Total | $d_{TotLgth}$ | $t_{TotDur}$ |
| | | | | | | | | | | 1146.319 | 30:45:30 |

**Table 3** The customers cycles, arrive and departure times, length and duration of cycles.

customers, numbers of delivered containers, cycle lengths, and duration of cycles, can be found in Table 3. The delivery of goods to customers can be ensured by four trucks, the total distance is 1146.919 km. The sum of driving times (hh:mm:ss) of all four trucks, including the service time for unloading goods, is 30:45:30.

# 3 Conclusion

The main result of this contribution is the construction of a linear programming model (2) for minimizing travel distance or duration travel time of trucks delivering ordered containers to customers. An integral part is the implementation of the created model in Matlab, i.e. the creation of an M-function *VRP_SOLVER_Za.m* that realizes the solution of vehicle routing problem for any number of n customers. The created function was applied to 19 customers, and the solution took 17 minutes on a common PC. The function is practically usable for up to 30 customers, but the necessary processing time increases. The optimal solution of VRP ensures the shortest travel length or shortest duration of a business trip, and thus the best solution in terms of cost price.


# Acknowledgements

The paper was supported by the grant No. CZ.01.1.02/0.0/0.0/21_374/0027244 "Technology development for intelligent traffic flow management - 2nd part - optimization and extension" of Czech Ministry of Industry and Trade.


# References


[1] Eksioglu, B., Vural, A.V. & Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering,* 57, 1472-1483.

[2] Jonak, R., Smutný, Z., Simunek, M. & Dolezel, M. (2020). Rout and Travel Time Optimization for Delivery and Utility Services. *Acta Informatica Pragensia,* (2) 9, 200-209.

[3] Math Works. Inc. (2023). *Optimalization Toolbox™. User´s* Guide. Natick.

[4] Winston, W. L. (1994). *Operations Research. Applications and Algorithms.* Duxbury: Duxbury Press.

[5] Toth, P. & Vigo, D. (1998). *Exact algorithms for vehicle routing.* Boston: Kluwer Academic Publisher

[6] Zahrádka, J. (2022) The Traveling Salesman Problem Solution by Mixed Integer Lin. Programming in Matlab Code. *Journal of Applied and Computational Sciences,* 1(1), 45–52. https://doi.org/10.528/zenodo.6880928


# Appendix

```
function [X, y, TotLgth, TotDur] = VRP_SOLVER_Za(n, D, C, Q, q, m, PvInd)
options = optimoptions('intlinprog', 'MaxTime', 3600, 'MaxNodes', 3000000);
p=(n+1)*(n+1); A=zeros(n*n-n, p+n); Aeq=zeros(3*n+2,p+n); TotLgth=0;
k=0; for i=1:n for j=1:n if i~=j k=k+1;A(k, p+i)=1;A(k, p+j)=-1; end; end; end
k=0; for i=1:n  for j=1:n  if i~=j k=k+1; A(k, (n+1)*i+1+j)=Q; end; end; end
for i=1:n*n-n  for j=1:n  if A(i, p+j)==-1 b(i, 1)=Q-q(j); end; end; end
for i=1:n  for j=1:n+1 Aeq(i, (n+1)*i+j)=1; end; beq(i, 1)=1; end
for i=1:n  for j=1:n+1 Aeq(n+i, (j-1)*(n+1)+i+1)=1; end; beq(n+i, 1)=1; end
for i=1:n+1  Aeq(i+2*n, (i-1)*(n+1)+i)=1; beq(i+2*n, 1)=0; end
for i=2:n+1  Aeq(3*n+2, i)=1; end
for i=1:n  Aeq(3*n+2, (n+1)*i+1)=-1; end; beq(3*n+2, 1)=0; lb=zeros(p,1); k=0;
for i=1:n+1 for j=1:n+1 k=k+1;if i==j ub(k, 1)=0;else ub(k, 1)=1;end; end; end
for i=1:n  lb(p+i, 1)=q(i); end; for i=1:n ub(p+i, 1)=Q; end
DT=D'; f=DT(:);  %CT=C'; f=CT(:);
f(p+1:p+n)=ones(1, n)/100; intcon=1:(n+1)^2+n;
V=intlinprog(f, intcon, A, b, Aeq, beq, lb, ub, [], options); V=round(V);
X=V(1:p); X=reshape(X, [n+1, n+1]); X=X'; y=V(p+1: end) ; TotDur=sum(m);
  for i=1:(n+1) for j=1:(n+1)
    if X(i, j)==1 TotLgth=TotLgth+D(i, j); TotDur=TotDur+C(i, j); end
  end; end
end
```