J. Appl. Comp. Sci.
Received: 02/02/2022
Accepted: 19/04/2022
Published online: 30/06/2022
Volume 1, Issue 1, Pages 45-52
doi: 10.5281/zenodo.6880928
Research Article

THE TRAVELLING SALESMAN PROBLEM SOLUTION BY MIXED INTEGER LINEAR PROGRAMMING IN MATLAB CODE

JAROMÍR ZAHRÁDKA*

*Department of Mathematics and Physics, Faculty of Electrical Engineering and Informatics, University of Pardubice, Pardubice, Czech Republic

1. INTRODUCTION

ABSTRACT: This article introduces one more specific solution of the travelling salesman problem. The seller has to distribute, using his truck, goods from the depot (warehouse) to n customers. Each customer point of delivery is given by GPS coordinates. This problem can be called the travelling salesman problem. The objective of the solution is to select the sequence of delivery points so that firstly the travel distance and subsequently the total travel time are minimal. The seller visits all delivery points and returns to the depot. In this article, one general solution is presented using the branch-and-bound method and by using mixed integer linear programming implemented in Matlab code. The created algorithm can be used for any number n of customers.

Keywords : Travelling salesman problem; branch-and-bound method; mixed-integer linear programming; Matlab.

AMS Subject Classification : 68W04; 90C11, 05C20.

2. MAIN RESULTS

The travelling salesman problem (TSP) and its classical solutions are described e.g. in [1, 2, 4, 5]. Our solution was inspired by the use of integer programming published in [7]. One implementation of the TSP solution with Matlab programming can be found in [3].

The TSP can be defined as follows. Let $G_0 = (V.E)$ be a connected complete oriented graph consisting of a set of n nodes (customer locations) indexed by $i = 0,1,...,n$, and a set E of non-negatively weighted arcs between each pairs of corresponding nodes of the graph G_0. The index $i = 0$ is for the seller, and the indexes $i = 1,...,n$ are for all customers. For easier references, let $I = \{1,...,n\}$ be the set of n customers, and $I_0 = I \dot{E} \{0\}$. The constant t_0 means the time-moment when the dealer vehicle leaves the depot. Each customer can be visited only once at any time greater than t_0. The order of the customers visited is not limited, other than by the requirement that the duration of the seller's journey through all customers (terminated by return to the depot) be as short as possible. For each customer $i \hat{I} I$ let m_i be the assumed service time associated with the unloading of goods and dealing with the customer.

Let d_{ij} be the length of the path from i - node to j - node for all $i, j \hat{I} I_0$. Therefore $\mathbf{D} = \left(d_{ij}\right)_{i,j \hat{I} I_0}$ is the non-negative distance matrix. The matrix $\mathbf{D}$ can be, in general, an asymmetric one with zeros on the places of the main diagonal, i.e. $d_{ii} = 0$ for each $i \hat{I} I_0$. It is necessary that the triangular inequalities be satisfied for distances among nodes of graph G_0.

Instead of the distance matrix $\mathbf{D}$ we will use, for the solution TSP, the time matrix $\mathbf{C} = \left(c_{ij}\right)_{i,j \hat{I} I_0}$. Each element c_{ij} represents the pure travelling time of the seller from i - node to j - one. It is assumed that if the average speed v of the vehicle among each two nodes is used, then the driving time c_{ij} can be expressed

E-mail address: jaromir.zahradka@upce.cz
* Corresponding Author.

$c_{ij} = \frac{d_{ij}}{v}$. In this case the travel time c_{ij} is proportional to the distance d_{ij}. We assume that it is given the moment t_0 when the seller's vehicle leaves the depot.

The core of the practical TSP solution is to find the one cycle in the graph G_0 which includes all nodes of the graph and which gives the shortest total driving time. For this purpose, integer variables x_{ij} for $i, j \hat{I} I_0$ are introduced, which can only take the values 0 or 1. The variables x_{ij} are called binary variables. Value $x_{ij} = 1$ means that the arc from node i to j is included in the cycle and value $x_{ij} = 0$ means that the corresponding arc is not included. For systemic reason variables, x_{ii} are used but all are fixed by the value zero, i.e. $x_{ii} = 0$, for each $i \in I$. Variables x_{ij} are elements of a matrix $\mathbf{X} = (x_{ij})_{i,j \hat{I} I_0}$. The number of flow variables x_{ij} is $(n+1)^2$.

In our work we use other specific non-integer variables t_i, for each $i \in I$. Each t_i indicates the moment when the seller leaves the i 's customer location. By using variables t_i, it is guaranteed that the solution will be correct with all nodes during only one cycle in the graph G_0. The variables t_i are included as n elements of the vector $\mathbf{t} = (t_1, t_2, \ldots, t_n)$. The number of all flow variables is $(n+1)^2 + n$.

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

$$\min_{(\mathbf{X},\mathbf{t})} \left\{ \sum_{i,j=0}^{n} c_{ij} \cdot x_{ij} + \sum_{i=1}^{n} \frac{1}{n \cdot u} \cdot t_i \right\} \text{ subject to} \tag{1}$$

$$x_{ij},\ i, j \in I_0 \text{ are binary, } x_{ij} \in \{0,1\} \tag{2}$$

$$(c_{ij} + u - t_0 - c_{0j})\, x_{ij} + t_i - t_j \le u - t_0 - c_{0j} - m_j, \quad i, j \in I,\ i \neq j \tag{3}$$

$$c_{0j}\, x_{0j} - t_j \le -t_0 - m_j, \qquad j \in I \tag{4}$$

$$\sum_{j \in I_0} x_{ij} = 1, \qquad i \in I_0 \tag{5}$$

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

$$x_{ii} = 0, \qquad i \in I_0 \tag{7}$$

$$0 \le x_{ij} \le 1, \qquad i, j \in I_0 \tag{8}$$

$$t_0 + c_{0j} + m_j \le t_j \le u, \qquad j \in I. \tag{9}$$

In the expressed model (1) with flow variables x_{ij} and t_i, the linear optimization function

$$\sum_{i,j=0}^{n} c_{ij} \cdot x_{ij} + \sum_{i=1}^{n} \frac{1}{n \cdot u} \cdot t_i \tag{10}$$

is minimized.

The first (main) part $\sum_{i,j=0}^{n} c_{ij} \cdot x_{ij}$ of the minimized optimization function guarantees finding the cycle which takes the minimum amount of time. Due to the assumed constant average speed v, the total travel length is also minimal. In the second (marginal) part $\sum_{i=1}^{n} \frac{1}{n \cdot u} \cdot t_i$ of the optimization function, the coefficients of the variables t_i are very small due to the very large value of the constant u which is defined as

$$u = t_0 + \sum_{i=1}^{n} m_i + \sum_{j=0}^{n} \max_{i \in I_0} c_{ij} \,. \tag{11}$$

The second part of the optimization function does not change the optimal solution for the variables x_{ij}, while the sum of time variables t_i are minimized. This means that the optimization process only provides solutions for variables t_i so that no waiting times are included when the seller travels between any two customers.

The constraint (3) defines $n(n-1)$ conditions between flow variables x_{ij} and departure times t_i, t_j, for $i, j \in I$. In the case $x_{ij} = 0$, the inequality (3) expresses the relationship $t_j \geq t_0 + c_{0j} + m_j + t_i - u$. Due to the large enough value of u, the right side of inequality (3) can be only non-positive and the relationship is satisfied.

In the case $x_{ij} = 1$, the inequality (3) is reduced to $t_i + c_{ij} + m_j \leq t_j, \quad i, j \in I$. This expresses that the departure time from the node j has to be greater than or equal to the sum of the departure time t_i (from node i), the travelling time c_{ij} (from node i to node j) and the service time m_j in the node j. The created optimization process ensures that, in the case of $x_{ij} = 1$, the condition (3) is satisfied only by the equation $t_i + c_{ij} + m_j = t_j$.

The constraint (4) defines relations between flow variables x_{0j} and t_j, $j \in I$. In the case $x_{0j} = 0$ the inequality expresses the relationship $t_0 + m_j \leq t_j$, $j \in I$. Departure time from the node j is greater than or equal to the sum of departure time t_0 and service time m_j. In the case $x_{0j} = 1$ the inequality (4) expresses the relationship $t_0 + c_{0j} + m_j \leq t_j$. Departure time from the node j is greater than or equal to sum of departure time t_0 from the depot, travelling time c_{0j} from depot to node j and service time m_j.

Statements (5) and (6) declare $2(n+1)$ equation constrains, which express that only one arc leads from each node and only one arc leads to each node. Statement (7) declares that each $x_{ii} = 0$.

The inequalities in (10) declare that the lower and upper bounds of variables x_{ij} are 0 and 1. The inequalities in (11) express the bounds of flow variables (departure times) $t_j, j \in I$.

In MATLAB system the index 0 can not to be used, therefore all vector and matrix variables use the smallest index 1. The distance matrix is transferred to the Matlab environment as matrix `D`, with the row and column indices `i,j = 1,2, … , n+1`, where each component `D(i,j)` corresponds to the distance $d_{i-1\,j-1}$ of the nodes $i-1$ and $j-1$. Similarly each component `C(i,j)` of the time matrix corresponds to the driving time $c_{i-1\,j-1}$ from the node $i-1$ to the $j-1$ one.

Our procedure for solving TSP in the Matlab code is contained in the M-function `TSP_SOLVER.m`, which has 83 rows and is fully listed as an APPENDIX at the end of the article. In the first line the mentioned M-function is declared. The input variables are `n` - number of customers, `D` – distance matrix, `v` – velocity of the vehicle, `t0` – the moment when the seller leaves the depot and `m` – row vector with customer service duration times. The main output variable is the column vector `X` of flow variables, which is obtained as an output of the optimization via the mixed-integer linear programming by the command `intlinprog`.

The mixed-integer linear programming problem is generally expressed by

$$\min_X f^T \cdot X \; subject\; to \begin{cases} X\,(intcon)\; are\; integers \\ A \cdot X \leq b \\ A_{eq} \cdot X = b_{eq} \\ l_b \leq X \leq u_b \,. \end{cases} \tag{12}$$

The solver for this problem is the command `X=intlinprog(f,intcon,A,b,Aeq,beq,lb,ub)` in Matlab code (you can see it on the APPENDIX row No. 55). A more detailed explanation of the specified command is available in the User's Guide [6].

For the solution of TSP via the `intlinprog` command, all flow variables are arranged in a column vector `X` with $(n+1)^2 + n$ components. First $(n+1)^2$ flow variables are integer variables x_{ij}, and each variable x_{ij}, $i, j \in I_0$ is represented by Matlab flow variable `X(i*(n+1)+j+1,1)`. The last n flow variables of `X` are the seller's departure times $t_1, t_2, ..., t_n$, and each variable t_i, $i \in I$ is represented by `X((n+1)^2+i,1)`.

The objective function of the mixed-integer linear programming problem (12) is, in the Matlab code, expressed like `f'*X`, where `f` is a column vector of coefficients with $(n+1)^2 + n$ components. The first $(n+1)^2$ components are elements of the time matrix `C` so that `f((i-1)*(n+1)+j,1)=C(i,j)`, `i,j` Î $\{1,2,...,n+1\}$. For the last n components of `f` we use the value $\frac{1}{n \times u}$ according to relation (10) (the APPENDIX, row No. 3).

The vector `intcon` in the command `intlinprog` specifies the indexes of flow variables, which are taken integers, i.e. `intcon=1:(n+1)^2` (row No. 53). They are first $(n+1)^2$ flow variables, i.e. x_{ij}, $i, j \in I_0$.

The constraints (3) and (4) give the system of n^2 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 number of customers n in Matlab code, statements on lines No. 4 to 16 in the APPENDIX.

The constraints (5), (6) and (7) 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 number of customers n in the Matlab code, statements on lines No. 17 to 33 in the APPENDIX.

Another two input variables of the `intlinprog` command (line No. 54) are the column vectors `lb` and `ub` of lower and upper bounds of all flow variables. With respect to the relations (7), (8), (9) the components of vectors `lb` and `ub` are filled by commands on lines No. 34 to 51 in the APPENDIX.

The maximum optimization time in the `intlinprog` command can be set with the `options` command by specifying the number of seconds after the `'MaxTime'` parameter (see line 53 of the APPENDIX). If the `intlinprog` program terminates before the optimization is complete, the program offers a solution that may not be optimal, but that is close to the optimal solution. Other configurable parameters of the `intlinprog` command can be set similarly, you can see in the User's Guide [6].

After optimization via the `intlinprog` command, the TSP solution is stored in the flow variables. The variables `X(k,1)`, $k \in \{1,2,...,(n+1)^2\}$, which take the value 1, determine the arcs of the shortest travel cycle. The last n values of flow variables indicate times when the seller leaves individual customers.

The commands of function TSP_SOLVER.m from line No. 56 to 68 allow the creation of a sequence of cycle nodes, i.e. the `CYCLE` vector. The first item of the `CYCLE` vector is the number 0 – depot, and the other n items are the sequence of customer numbers, and the last item is supplemented by the number 0 with regard to the fact that the seller returns to the depot.

The Matlab variables `X(k,1)`, $k \in \{(n+1)^2+1, (n+1)^2+2, ..., (n+1)^2+n\}$ are the departure times of the seller from the customer (node) number `k` at the optimal cycle. The vector of the departure times `t` is created by a for-cycle command on lines No. 69, 70 and 71. The time `tRet` of the seller's arrival back to the depot after the visit to all customers is calculated by command on the line No. 72, and the total duration of the seller's business trip `AllWorkTime` is calculated on line No. 73.

For practical use, commands on lines No. 74 and 75 create a vector `tArr` of arrival times to the nodes in the order given by items of the vector `CYCLE`. The exception is the first item of the `tArr`, which means the departure time from the depot, i.e. time `t0`. The last item of the vector `tArr` means the time when the seller returns to the depot, i.e. `tRet`. The total distance `TotDist` travelled by the seller on his most advantageous business trip is calculated on lines No. 76 to 83 by using the distance matrix elements.

The input variables for the M-function `TSP_SOLVER` and its execution must be done using a special startup M-script that contains commands for drawing the output circle (the example you can see in Figure 1). The startup script in Matlab code is not listed in this article.

3. APPLICATION AND ILLUSTRATIVE EXAMPLE

To illustrate the program we have created, we assume a seller and twelve customers. The GPS coordinates of the seller's depot are $E_0 = 14.288^0$ (the eastern longitude) and $N_0 = 49.447^0$ (the northern latitude). The GPS coordinates E_i, N_i and the service times m_i of customers you can find in TABLE 1.

TABLE 1 The GPS coordinates and the service times of the customers

	Customer											
i	1	2	3	4	5	6	7	8	9	10	11	12
E_i (°)	14.415	14.465	14.764	14.818	14.100	14.178	14.360	14.057	14.522	14.336	14.176	14.209
N_i (°)	49.036	49.121	49.027	49.221	49.007	49.449	49.098	49.047	49.154	49.228	49.051	49.498
m_i (min)	11	13	9	13	13	18	18	17	9	9	18	20

The distance between two customer locations (nodes) is their orthonormal distance on the sphere multiplied by a factor of 1.25. The orthonormal distance is calculated with a sphere radius R = 6371 km (mean radius of the Earth). All taken distances are included in the symmetric distance matrix **D** in TABLE 2.

TABLE 2 The distance matrix **D**

Distance (km)		j												
		0	1	2	3	4	5	6	7	8	9	10	11	12
i	0	0	58.09	50.32	87.01	79.69	64.79	15.29	48.05	62.74	51.12	30.24	55.58	12.95
	1	58.09	0	13.39	48.52	61.29	43.96	64.65	11.32	49.78	21.76	28.09	33.28	68.49
	2	50.32	13.39	0	43.44	50.87	53.01	59.52	14.92	57.58	9.08	23.00	41.26	62.00
	3	87.01	48.52	43.44	0	27.13	92.33	99.30	56.96	98.31	37.72	65.35	81.79	99.84
	4	79.69	61.29	50.87	27.13	0	103.87	94.10	65.77	108.34	42.12	67.00	92.12	92.49
	5	64.79	43.96	53.01	92.33	103.87	0	60.55	38.16	8.05	61.91	44.30	12.11	67.89
	6	15.29	64.65	59.52	99.30	94.10	60.55	0	53.62	56.74	62.16	36.99	53.64	7.88
	7	48.05	11.32	14.92	56.96	65.77	38.16	53.62	0	42.67	23.75	17.82	26.35	57.82
	8	62.74	49.78	57.58	98.31	108.34	8.05	56.74	42.67	0	66.22	45.81	16.55	64.36
	9	51.12	21.76	9.08	37.72	42.12	61.91	62.16	23.75	66.22	0	27.71	50.05	63.55
	10	30.24	28.09	23.00	65.35	67.00	44.30	36.99	17.82	45.81	27.71	0	32.61	40.43
	11	55.58	33..28	41.26	81.79	92.12	12.11	53.64	26.35	16.55	50.05	32.61	0	60.41
	12	12.95	68.49	62.00	99.84	92.49	67.89	7.88	57.82	64.36	63.55	40.43	60.41	0

By running the function `TSP_SOLVER_Z.m` with the above chosen parameters, the optimal solution was found. The shortest cycle is given with a node sequence 0-12-6-8-5-11-7-1-3-4-9-2-10-0. The calculated seller's departure times t_{dep_i} from customers, and arrival times t_{arr_i} to customers are in TABLE 3.

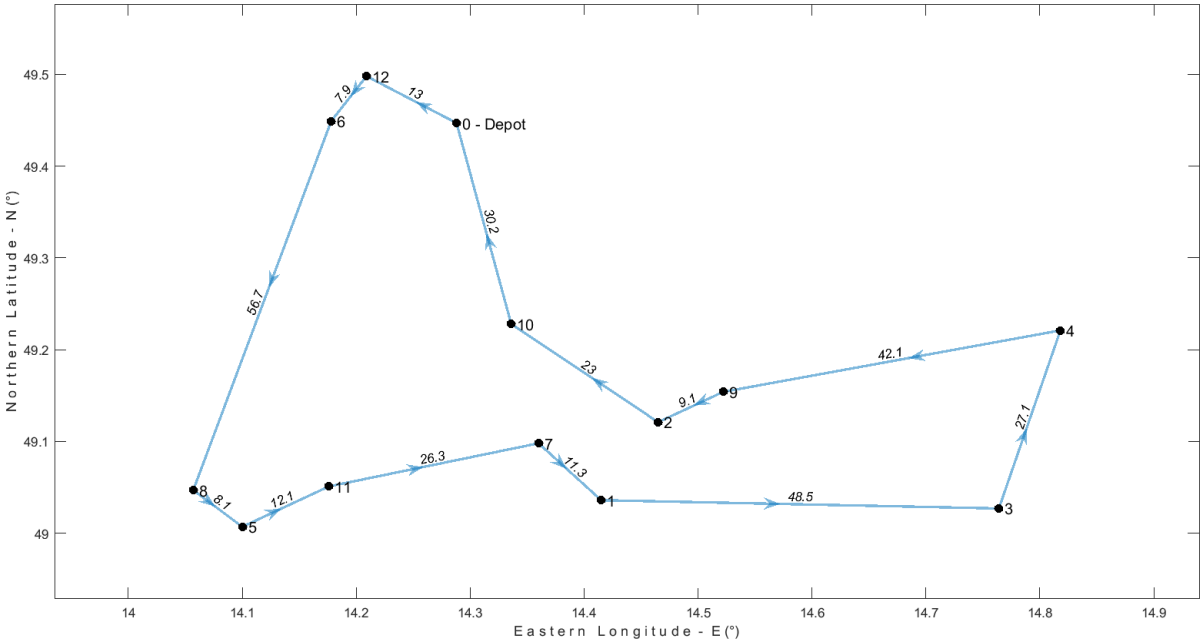
TABLE 3 The arrive and departure times of the seller

Times	Depot	Customers ranking in minimal cycle												Depot
i	0	12	6	8	5	11	7	1	3	4	9	2	10	0
t_{arr_i}	-	4:12	4:40	5:55	6:20	6:45	7:30	7:59	8:58	9:35	10:30	10:48	11:24	12:03
t_{dep_i}	4:00	4:32	4:51	6:12	6:33	7:03	7:48	8:10	9:07	9:48	10:39	14:42	11:33	-

The found cycle of minimal length is drawn in Figure 1. The total travelled distance by the seller vehicle is 315.50 km and the total time of a business trip is 8 hours and 3 minutes.

However, due to the symmetry of the distance matrix **D**, there is another equivalent solution that gives the same minimum travel distance and minimum driving time. This is the cycle with the opposite orientation of all arcs, i.e. the cycle 0-10-2-9-4-3-1-7-11-5-8-6-12-0.

FIGURE 1 The minimal length cycle of the seller around all customers



4. CONCLUSION

This paper proposes a practical solution of the travelling salesman problem for any number n-customers in Matlab code. The TSP is formulated as a mixed-integer linear programming problem with a new approach, which respects the given matrix of distances and service duration times of customers, as well as the constant speed of the seller's movement. The solution lies in minimizing of the seller's trip duration that leads across all nodes (customers). The constant speed of seller's movement is assumed, therefore the total distance travelled is also the minimum. The created objective function guarantees that the total travelled distance and the total travelled time of the seller are minimal.

The main result of this article is the created M-script, which allows the TSP to be solved generally for any number of n customers. The created M-script is practically usable on a common personal computer for up to 30 customers. For 30 customers, the calculation takes less than 85 minutes, and for up to 19 customers, the calculation takes less then 20 seconds. With the growing number of customers, the time to find the optimal solution increases.

On the line No. 53 of the M-function it is possible to set the maximum running time limit of the command `intlinprog` by specifying the number of seconds in the parameter `'MaxTime'`. If the optimal solution is not found within the given time limit, the Matlab offers an ongoing solution that was found during the optimization process. The created M-function was successfully tested for a maximum of 37 customers on a standard PC. In this case the optimal solution was found after 8 hours.

Acknowledgement. The author would like to thank the anonymous reviewers for their valuable comments and suggestions for improving the paper.

Declaration of Competing Interests. The author declare that he has no known financial interests or personal relationships that conflict with each other affecting the study reported in this article.

REFERENCES

[1] J. J. Bentley, Fast Algorithms for Geometric Travelling Salesman Problems, ORSA Journal on Computing, (4) 4,1992.
[2] G. Davendra et al, Travelling Salesman Problem, Theory and Applications, Rieaka: IN Tech, 2010.
[3] K. P. Gradle and Y. U. Muley. Travelling Salesman with MATLAB programming, International Journal of Advaces in Applied Mathematics and Mechanics, (3) 2 , 258-266, 2015.
[4] G. Gutin and A. P. Punnen, The Travelling Salesman Problem and Its Variations, Springer Science+Business Media, LLC, New York, 2007.
[5] R. Jonak, Z. Smutny, M. Simunek and M. Dolezel, Rout and Travel Time Optimization for Delivery and Utility Services, Acta Informatica Pragensia, (2) 9 , 200-209, Prague, 2020.
[6] MathWorks. Inc., Optimization Toolbox™. User's Guide, Natick, 2020.
[7] W. L. Winston, Operations Research, Applications and Algorithms, Duxbury Press, Duxbury, 1994.

APPENDIX

```
function [X, CYCLE, TotDist, AllWorkTime, tArr] = TSP_SOLVER(n, D, v, t0, m)
C=D/60; CT=C'; u=t0+sum(max(CT))+sum(m(1:n));
f=[CT(:); ones(n,1)/u/n];
p=(n+1)*(n+1); A=zeros(n^2,p+n); k=0;
for i=1:n
for j=1:n
if i~=j; k=k+1;
A(k,(n+1)*i+1+j)=C(i+1,j+1)+u-t0-C(1,j+1);
A(k,p+i)=1; A(k,p+j)=-1;
b(k,1)=u-t0-C(1,j+1)-m(j);
end
end
end
for i=1:n
k=k+1; A(k,1+i)=C(1,1+i); A(k,p+i)=-1; b(k,1)=t0-m(i);
end
Aeq=zeros(3*n+3,(n+1)^2+n);
for i=1:n+1
for j=1:n+1
Aeq(i,(i-1)*(n+1)+j)=1;
end
Aeq(i,(i-1)*(n+1)+i)=0;
beq(i,1)=1;
end
for i=1:n+1
for j=1:n+1
Aeq(n+1+i,(j-1)*(n+1)+i)=1;
end
Aeq(n+1+i,(i-1)*(n+1)+i)=0; beq(n+1+i,1)=1;
end
for i=1:n+1
Aeq(2*n+2+i,(i-1)*(n+1)+i)=1; beq(2*n+2+i,1)=0;
end
lb = zeros(p,1);
for i=1:n
lb(p+i,1)=t0+C(1,1+i)+m(i);
end
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
ub(p+i,1)=u;
end
intcon=1:(n+1)^2;
options=optimoptions('intlinprog', 'MaxTime', 300, 'MaxNodes', 3000000);
```

```
X=intlinprog(f, intcon, A, b, Aeq, beq, lb, ub, [], options);
X(1:p)=round(X(1:p))
for i=2:n+1
if X(i)==1
CYCLE=0;
Nok=2; CYCLE(Nok)=i-1; TEST=i; break
end
end
while TEST~=1
for j=1:n+1
if X((CYCLE(Nok))*(n+1)+j)==1
Nok=Nok+1; CYCLE(Nok)=j-1; TEST=j; break
end
end
end
for i=1:n
t(i)=X(p+i);
end
tRet=t(CYCLE(end-1))+C(CYCLE(end-1)+1,1)
AllWorkTime=tRet-t0;
tArr=[t0, t(CYCLE(2:end-1))-(m(CYCLE(2:end-1))), tRet],
tArr=hours(tArr), tArr.Format='hh:mm'
TotDist=0;
for i=1:(n+1)
for j=1:(n+1)
if X((n+1)*(i-1)+j)==1
TotDist=TotDist+D(i,j); break
end
end
end
```