OPSEARCH
https://doi.org/10.1007/s12597-024-00758-0

THEORETICAL ARTICLE

New strategy for anti-loop formulations

Jose Manuel García[1]

Accepted: 11 March 2024

Abstract
This paper presents a strategy based on binary labelling of nodes for the creation of anti-loop formulations from existing strategies. This strategy prevents by default the formation of odd cycles, therefore it can have important role in iterative procedures based on generating subtour elimination constraints. It can also be used to modify the classic strategies used in problems associated to graphs. In this paper we focus on this last application. The behavior of this strategy is analyzed with two problems associated with graphs, the Asymmetric Traveling Salesman Problem (ATSP) and the Steiner Problem, where two configurations that modify the Miller-Tucking-Zemlig proposal to avoid cycles are compared. The experimental analysis shows that this strategy keep a good convergence, highlighting its use for the Steiner problem.

Keywords Mathematical programming · Anti-loops formulations · Subtour elimination constraints · Travelling salesman problem · Steiner problem

1 Introduction

In this paper, a new strategy for the creation of anti-loop formulations in optimization problems associated with graphs, based on binary labelling of nodes, is presented. In the new formulations presented in this work, the formation of cycles with an odd number of nodes is not allowed. This strategy can lead to an improvement in the iterative methods that allow the formation of cycles, in addition to allowing the existing anti-loop strategies to be reformulated.

When we propose arborescence formulations based on level or sequence relationships, avoiding the formation of cycles is one of the constraints for obtaining a tree from a graph. Also, to obtain a path that visits all the nodes of the graph (Hamiltonian circuit) in the case of the traveling salesman problem. In the case of trees, the solution must be a connected subgraph where each node connects to

✉ Jose Manuel García
jmgs@us.es

[1] School of Engineering, University of Seville, Camino de los Descubrimientos, s/n, 41092 Seville, Spain

Published online: 05 April 2024

a higher level node or parent node, except the node that is defined as root node of the tree. In the traveling salesman case, the solution must present a connected subgraph with as many edges as nodes, and each node is connected to two nodes (one preceding node and one posterior node in the sequence).

The most relevant optimization problems where it is necessary to obtain a tree of a graph are the MST problem, the VRP problem and the Steiner problem, and all those derived from them.

In general, the outstanding formulations that have been used on graphs to create trees or for the traveling salesman problem have been the following:

- Obtain all the cycles [1]: Calculate all the cycles and avoids that all the nodes of any cycle are part of the tree (DFJ).
- Cuts: Derived from the previous, it defines that there is at least one edge between each pair of complementary node sets.
- Flow sending: Flow sent from a source node to the rest of the nodes of the graph imposing the flow conservation restrictions. Different versions: Single commodity flow [2], two-commodity flow [3] and multiple-commodity flow [4].
- Based on decisions of precedence. Proposed by Sarin et al. [5] for the ATSP problem. Formulation also applied in scheduling problems.
- Time Staged Formulations: Arcs are added to the tree or path at each stage or period [6].
- Setting depth levels at nodes or sequential formulation. Proposed by Miller et al. [7], MTZ from now, and later lifted by Desrochers and Laporte [8], DL from now. Other versions such as Sherali and Driscoll [9] and Sawik [10] also stand out.

In addition, there are some versions that consider mixed variants of the previous ones, as appears in Sherali et al. [11] and Gouveia and Pires [12], where the identification of some cycles is incorporated into sequence formulations. There are also formulations like Martin [13], in this case for obtaining trees, which uses decisions about which part of the path each node is in with respect to each selected arc. It is inefficient because it makes use of $O(n^3)$ variables and constraints.

The Danzig and Cuts strategies turn out to be formulations with little practical scope. The exponential growth in the number of constraints, that is, in the number of cycles, makes it difficult for us to experimentally analyse these models.

The strategies that have received more practical attention have been the flow formulation and the sequential formulation, the latter being more efficient when solving larger problems. Both in one and in the other, each edge $(i,j) \in E$ becomes in two directed arcs, (i,j) and (j,i), to express a concept of direction for the case of flow formulations, and a concept of successor/predecessor relationship for sequential formulations.

Formulation MTZ is based on introducing as a variable the position of each node in the sequence (or depth level of each node in the tree):

$\forall i : u_i =$ Position in the sequence (or depth level in the tree) of node i

$$1 \leq u_i \leq n-1$$

Position 0 is assumed for node 1 (root node) and for the rest of it is imposed as a constraint that if we select the arc (i,j), the position of j in the sequence is greater than the position of i (at least one more unit). We define the boolean variable x_{ij} to collect the selection of arcs $(i,j) \in A$.

Mathematically this is expressed by modelling the following logical proposition:

$$\forall i,j = 2...n, i \neq j : IF\ x_{ij} = 1\ THEN\ u_i \geq u_j + 1$$

Whose modelling results in:

$$\forall i,j = 2...n, i \neq j : u_i - u_j + (n-1)x_{ij} \leq n-2 \quad (1)$$

The proposed modification DL, proposes a formulation based on the modeling of the following logical statement: $\forall i,j = 2...n, i \neq j : IF\ x_{ij} + x_{ji} = 1\ THEN\ u_i + x_{ji} = u_j + x_{ij}$.

which is developed in García [14] and generates constraints:

$$\forall i,j = 2...n, i \neq j : \ u_i - u_j + (n-3)x_{ij} + (n-1)x_{ji} \leq (n-2) \quad (2)$$

DL improves the results of the MTZ formulation because the DL formulation integrates into the anti-loop constraint that $x_{ij} + x_{ji}$ is a binary expression.

On the other hand, solving problems associated with graphs to integer optimality, strategies based on obtaining solutions for the previous models have also been used, eliminating the anti-loop restrictions. These are iterative processes, where for each solution obtained it is calculated whether it has subtours, incorporating the specific subtour elimination constraints into the model to avoid these cycles, and solving the model again in the next iteration. Model resolutions in this way are quite fast. For some applications of these strategies see Aguayo et al. [15], Miliotis [16], Pferschy and Stanek [17], Crowder et al. [18] or Bosch [19]. With the technique that we are going to introduce in this work, these iterative methods will always avoid the formation of odd subtours, without the need to introduce new integer variables into the model, as will be explained in Sect. 2. These methods are not undertaken in this work, but we focus on the study of analysing variations in the MTZ and DL strategies derived from introducing our odd anti-loop constraints.

2 Binary labelling of nodes

Binary labelling of nodes consists of incorporating a boolean variable δ_i for each node i = 2…n, and imposing constraints (3) and (4) whose logical meaning expresses that.

"IF $x_{ij} = 1$ THEN $\delta_i + \delta_j = 1$ "

$$\forall (i,j) : \delta_i + \delta_j \leq 2 - x_{ij} \quad (3)$$

$$\forall(i,j) : \delta_i + \delta_j \geq x_{ij} \tag{4}$$

Therefore, when we select the arc $(i,j) \in A$:

$x_{ij} = 1 \Rightarrow \delta_i + \delta_j = 1$. The labelling of the two nodes connected by the arc must be different.

$x_{ij} = 0 \Rightarrow \begin{matrix} \delta_i + \delta_j \leq 2 \\ \delta_i + \delta_j \geq 0 \end{matrix}$ No constraint for δ_i and δ_j.

The consequence of this labelling is that the model cannot allow solutions with odd loops, as shown in Fig. 1.

The labelling of the root node or initial node (node 1) can be assigned freely. This has the consequence that if the constraints of the problem ensure a connected graph, we can consider the labelling variables as continuous, although in that case they would not form part of the decision process in the branch&bound resolution. Both configurations will be analysed experimentally in the computational results section.

From previous experiments, it is more efficient to group the constraint associated with x_{ij} and x_{ji} in a single formulation:

$\forall(i,j) \in E$: IF $x_{ij} + x_{ji} = 1$ THEN $\delta_i + \delta_j = 1$.

Generating:

$$\delta_i + \delta_j \leq 2 - x_{ij} - x_{ji} \tag{5}$$

$$\delta_i + \delta_j \geq x_{ij} + x_{ji} \tag{6}$$

For the case of the TSP problem, if the number of nodes is odd, since the constraints do not apply to node 1, the model would allow a solution that creates the Hamiltonian path.

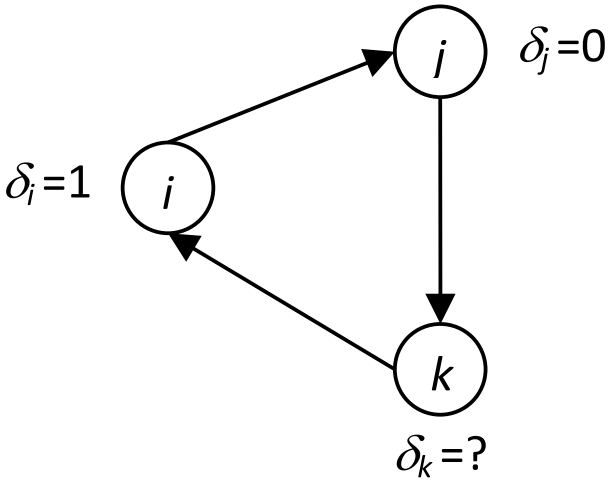

Fig. 1 Nodes loop

3 Application to classical formulations

To apply binary labelling, we are going to modify the TSP formulations proposed by Danzig, Fulkerson, and Johnson, the TSP MTZ formulation, and for tree generation, an arborescent formulation of the Steiner problem.

3.1 TSP-Danzig, Fulkerson, and Johnson

The classic formulation DFJ would go on to consider only the calculation of even cycles. Odd cycles cannot occur.

For a directed graph $G(N,A)$ from the undirected graph $G(N,E)$ and given the set of decision variables,

$x_{ij} = 1$ if the directed arc (i,j) is selected; 0 otherwise. $(i,j) \in A$.

The TSP would be as follows:

$$Min \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{7}$$

s.t.

$$\forall j \in N : \sum_{i/(i,j)\in A} x_{ij} = 1 \tag{8}$$

$$\forall i \in N : \sum_{j/(i,j)\in A} x_{ij} = 1 \tag{9}$$

$$\forall S/S \subset N, |S| \geq 2, |S| even : \sum_{i,j\in S \& (i,j)\in A} x_{ij} \leq |S| - 1 \tag{10}$$

$$\forall (i,j) \in E, i \neq 1 : \delta_i + \delta_j \leq 2 - x_{ij} - x_{ji} \tag{11}$$

$$\forall (i,j) \in E, i \neq 1 : \delta_i + \delta_j \geq x_{ij} + x_{ji} \tag{12}$$

$$\delta_1 = 1 \tag{13}$$

$$\forall (i,j) \in A : x_{ij} \in \{0,1\}$$

$$\forall i \in N, i > 1 : \delta_i \leq 1$$

With (10) partial even cycles are not allowed. (11) and (12) express the binary labelling. (13) assigns the label to the first node, hence we would allow the variables of the binary label to be relaxed, that is, we do not need to use binary variables to prevent odd cycles.

3.2 TSP-Miller, Tucker, and Zemlin

When we introduce binary labelling in the MTZ formulation, the number of positions can be halved. We transform the MTZ proposition into the following:

$$\forall i,j = 2...n, i \neq j : IF\ x_{ij} = 1\ THEN\ u_i \geq u_j + \delta_j \tag{14}$$

$$\forall i = 2...n :\ u_i \geq 0$$
$$\forall i = 2...n :\ u_i \leq \left\lceil \frac{n}{2} \right\rceil - 1$$

Hence, the positions are sequenced as shown in Fig. 2.

Therefore the upper bound of variables u_i becomes $\left\lceil \frac{n}{2} \right\rceil - 1$, and the modeling of (14) is as:

$$\forall i,j \in N, i > 1, j > 1, i \neq j : u_i - u_j + (\left\lceil \frac{n}{2} \right\rceil - 1)x_{ij} \leq \left\lceil \frac{n}{2} \right\rceil - 1 - \delta_j$$

The complete resulting model would be:

$$\begin{aligned} &Min \sum_{(i,j)\in A} c_{ij}x_{ij} \\ &s.t.\ \forall j \in N : \sum_{i/(i,j)\in A} x_{ij} = 1 \\ &\forall i \in N : \sum_{j/(i,j)\in A} x_{ij} = 1 \\ &\forall i,j \in N, i > 1, j > 1, i \neq j : u_i - u_j + (\left\lceil \frac{n}{2} \right\rceil - 1)x_{ij} \leq \left\lceil \frac{n}{2} \right\rceil - 1 - \delta_j \end{aligned} \tag{15}$$

$$\begin{aligned} &\forall i,j \in N, i > 1, j > 1, i \neq j : \delta_i + \delta_j \leq 2 - x_{ij} - x_{ji} \\ &\forall i,j \in N, i > 1, j > 1, i \neq j : \delta_i + \delta_j \geq x_{ij} + x_{ji} \end{aligned} \tag{16}$$

$$\delta_1 = 1 \tag{17}$$

$$\forall i \in N, i > 1 : 0 \leq n_i \leq \left\lceil \frac{n}{2} \right\rceil - 1 \tag{18}$$

$$\forall\ i,j \in N, i \neq j : x_{ij} \in \{0,1\}$$
$$\forall\ i \in N, i > 1 : \delta_i \leq 1$$

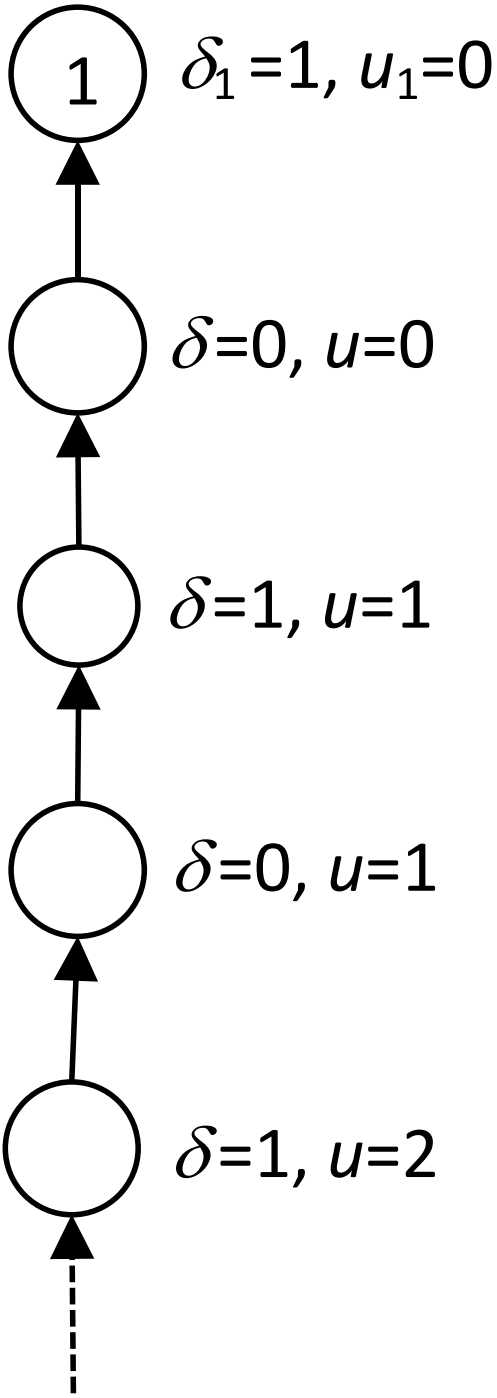

Fig. 2 Sequence of positions with binary labelling

3.3 Steiner problem in graphs

In an equivalent way, we are going to analyse a tree formulation of Steiner's problem, collected in Khoury, Pardalos, and Du [20]. The model includes the MTZ constraints.

Given a directed graph G(N,A) where are defined as data:
$\forall j \in N : T_j = 1$ if i is Terminal node; 0 is Steiner node.
$\forall (i,j) \in A : c_{ij} =$ Cost of selecting arc (i,j).
And the following variables:
$\forall (i,j) \in A : x_{ij} = 1$ if arc (i,j) is selected; 0 otherwise;
The common formulation without introducing anti-loop constraints is as follows:

$$s.t.\ Min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\forall i/T_i = 1 \& i \neq R : \sum_{j/(i,j)\in A} x_{ij} = 1 \tag{19}$$

$$\forall i/T_i = 0, j/(j,i) \in A : x_{ji} \leq \sum_{k/(i,k)\in A} x_{ik} \tag{20}$$

(19) Every terminal node, except the root node, has a parent node.
(20) Connectivity of the Steiner nodes used. If a Steiner node acts as a parent of a node, then it must be connected to another node.

This formulation would only need to prohibit the formation of subtours. The original model incorporates the MTZ constraints. In our case we incorporate the same constraints incorporated to the TSP problem, constraints (15), (16), (17) and (18).

4 Computational results

For the analysis of the formulations we have executed the following formulations both for the ATSP to a battery of problems collected in TSLIB, as well as for a set of existing Steiner problems in the same library:

- MTZ: Incorporates anti-loop constraints (1)
- MTZBL: Binary labelling. Incorporates constraints (10).
- MTZRBL: Relaxing binary labelling. Incorporates constraints (10), relaxing variables δ_i.

Experiments were performed on an Intel(R) Core(TM) i7-10700 K CPU @ 3.80GH and 16 Gb RAM. The optimization library used was CPLEX v22.1.0, widely used in computational analysis of mathematical models [21, 22] and [21, 22]).

Table 1 Comparison of the LP relaxation bound for the ATSP

Problem	Cities	Opt	MTZ	MTZBL
br17	17	39	2,25	18
ftv33	34	1286	1187,73	1215.3
ftv35	36	1473	1382,89	1411,5
ftv38	39	1530	1440,21	1476.10
p43	43	5620	149,714	184
ftv44	45	1613	1523,4	1569,75
ftv47	48	1776	1655,82	1725,44
ry48p	48	14,422	12,564,70	13,809,07
ft53	53	6905	5935,38	6010,51
ftv55	56	1608	1438,29	1510,73
ftv64	65	1839	1722,89	1761,00
ft70	70	38,673	37,987,40	38,107,46

4.1 Results for ATSP

Table 1 shows a comparison of the solutions of the relaxed linear problem of each model. It is shown that MTZ presents the worst values. MTZBL presents an average lower bound improvement of 5.9%.

Table 2 presents the results of the execution of the models up to a maximum of 300 s. CPU time presents the time until the completion of the branch&bound resolution (LB = UB) or the best solution found (Z_{IP}) in the case of not completing after

Table 2 CPU time for branch & bound (aborted after 300 s run) for the ATSP

Problem	Cities	Opt	MTZ		MTZBL		MTZRBL	
			Z_{IP}	CPU time (sec)	Z_{IP}	CPU time (sec)	Z_{IP}	CPU time (sec)
br17	17	39		**0.06**		0.36		0.89
ftv33	34	1286		**0.34**		0.36		0.39
ftv35	36	1473		**0.36**		0.67		0.55
ftv38	39	1530		0.47		**0.44**		0.73
p43	43	5620	5622	300	5622	300	**5620**	300
ftv44	45	1613		0.92		1.22		**0.8**
ftv47	48	1776		**1.42**		4.34		3.47
ry48p	48	14,422		**5.31**		14.09		9.33
ft53	53	6905		**0.89**		2.3		2.69
ftv55	56	1608		**1.63**		2.45		3.81
ftv64	65	1839		**5.33**		13.56		13.83
ft70	70	38,673	**38,673**	300	**38,673**	300	**38,673**	300
ftv70	70	1950		**6.34**		15.67		20.02
kro124p	124	36,230		**7.64**		52.14		31.55

Bold values indicate the best result

Table 3 CPU time for branch & bound (until 300 s run) for the Steiner problem (SteinC problems)

					MTZ		MTZBL		MTZRBL	
Problem	Nodes	Edges	Terminals	Opt	Z_{IP}	CPU time (sec)	Z_{IP}	CPU time (sec)	Z_{IP}	CPU time (sec)
Steinc1	500	625	5	85		0,44		0,97		**0,33**
Steinc2	500	625	10	144		0,83		1,77		**0,5**
Steinc3	500	625	83	754		**0,81**		4,64		1,36
Steinc4	500	625	125	1079		**0,72**		4,55		1,7
Steinc5	500	625	250	1579		**0,38**		0,81		0,58
Steinc6	500	1000	5	55		1,27		5,16		**0,81**
Steinc7	500	1000	10	102		0,53		3,34		**0,3**
Steinc8	500	1000	83	509		**0,98**		6,81		1,84
Steinc9	500	1000	125	707		**0,89**		8,22		2,16
Steinc10	500	1000	250	1093		**4,89**		176,75		6,28
Steinc11	500	2500	5	32		**39,5**		203		95,06
Steinc12	500	2500	10	46		**1,89**		13,92		3,13
Steinc13	500	2500	83	258		**8,06**		196,81		23,55
Steinc14	500	2500	125	323		**0,91**		10,5		1,19
Steinc15	500	2500	250	556		**39,36**		196,44		81,44
Steinc16	500	12,500	5	11		27,73		17,26		**17,22**
Steinc17	500	12,500	10	18		**12,63**		14,76		14,33
Steinc18	500	12,500	83	113	**113**	300	114	300	**113**	300
Steinc19	500	12,500	125	146		5,69		3,75		**2,56**
Steinc20	500	12,500	250	267		3,88		6,2		**3,69**

Bold values indicate the best result

Table 4 CPU time for branch & bound (until 300 s run) for the Steiner problem (SteinD problems)

					MTZ		MTZBL		MTZRBL	
Problem	Nodes	Edges	Terminals	Opt	Z_{IP}	CPU time (sec)	Z_{IP}	CPU time (sec)	Z_{IP}	CPU time (sec)
Steind1	1000	1250	5	106		1,8		6,17		**1,56**
Steind2	1000	1250	10	220		1,78		8,61		**1,61**
Steind3	1000	1250	167	1565		0,44		5,39		**0,42**
Steind4	1000	1250	250	1935		**0,91**		6,26		1,13
Steind5	1000	1250	500	3250		**0,45**		1,38		0,5
Steind6	1000	2000	5	67		**17,31**		192,22		35,83
Steind7	1000	2000	10	103		2,94		15,05		**2,75**
Steind8	1000	2000	167	1072	**1072**	300	1074	300	**1072**	300
Steind9	1000	2000	250	1448		**5,06**		48,19		11,33
Steind10	1000	2000	500	2110		**2,42**		18,69		4,83
Steind11	1000	5000	5	29		**10,48**		85,38		20,86
Steind12	1000	5000	10	42		9,74		46,5		**4,69**
Steind13	1000	5000	167	499		**14,91**		248,13		26,83
Steind14	1000	5000	250	663	**663**	300	664	300	**663**	300
Steind15	1000	5000	500	1116		**3**		40,53		12,75

Bold values indicate the best result

300 s. The MTZBL strategy has a slightly longer time than MTZ strategies. In general, all formulations have difficulty as the problem size increases (increase in the computational time). And this is emphasized more in the binary labeling strategies, since the size of the problem increases with respect to the MTZ strategy in n variables and $2n$ constraints, with n the number of nodes.

The three strategies do not end the B & B on 2 occasions out of 14 problems. MTZRBL was the only strategy achieving all the optimal solutions.

Regarding the comparison between MTZBL and MTZRBL, the results are very similar, on 7 test cases MTZBL was better and on 6 it was MTZRBL.

4.2 Results for the Steiner problem

In the case of the Steiner problem, the LP relaxation follows a trend similar to the ATSP problem. MTZ gets a slightly lower value. The results obtained are presented in Tables 3 and 4. In Table 3 we show the SteinC problems (problems with 500 nodes), whereas in Table 4 are shown the results for the SteinD problems (1000 nodes). We test each problem with a maximum of 300 s.

In the case of completing 300 s without finishing the brach&bound, we show in Z_{IP} the best solution found.

In the case of the Steiner problem, the results of the continuous labelling are always better than those of the integer label. MTZRBL was also better than the MTZ strategy on seven SteinC problems and five SteinD problems. In general, the binary labelling strategy shows better behaviour when the percentage of terminal nodes is low.

5 Conclusions

We have proposed in this paper a new strategy to formulate anti-loop constraints, avoiding odd loops. The experimentation carried out shows the best performance for the binary labelling is its relaxation, which keeps integrity of the solution. Although better global results are achieved for the MTZ strategy, the relaxing binary labelling does have a good convergence and in some problems it improves the results of the MTZ. For the Steiner problem the results are better than the MTZ formulation in 12 problems over 35.

On other hand, this strategy can provide new ideas for the design of algorithms and models in problems related to graphs when iterative strategies are proposed based on generating subtour elimination constraints.

Acknowledgements Not applicable.

Author contributions Not applicable.

Funding Funding for open access publishing: Universidad de Sevilla/CBUA. No funds, grants, or other support was received.

Data availability All the mathematical programming models analysed in this paper are available from the corresponding author upon request.

Declarations

Conflict of interest The authors declare that they have no conflicts of interest to this work.

Ethical approval Not applicable.

Consent for publication Not applicable.

References

1. Dantzig, G.B., Fulkerson, D.R., Johnson, S.M.: Solution of a large scale traveling salesman problem. Oper. Res. Res. **2**, 393–410 (1954)
2. Gavish, B. & Graves, S.C. The travelling salesman problem and related problems. Working paper GR-078–78, Operations Research Center, Massachusetts Institute of Technology (1978)
3. Finke, G., Claus, A., Gunn, E.: A two-commodity network flow approach to the travelling salesman problem. Congr. Numer. Numer. **41**, 167–178 (1984)
4. Wong, R.T. Integer programming formulations of the travelling salesman problem. Proceedings of the IEEE International Conference of Orcuits and Computers, 149–152 (1980)
5. Sarin, S.C., Sherali, H.D., Bhootra, A.: New tighter polynomial length formulations for the asymmetric traveling salesman problem with and without precedence constraints. Oper. Res. Lett. Res. Lett. **33**, 62–70 (2005)
6. Fox, K.R., Gavish, B., Graves, S.C.: An *n*-constraint formulation of the (time-dependent) travelling salesman problem. Ops. Res. **28**, 1018–1021 (1980)
7. Miller, C.E., Tucker, A.W., Zemlin, R.A.: Integer programming formulations and traveling salesman problems. J. ACM **7**, 326–329 (1960)
8. Desrochers, M., Laporte, G.: Improvements and extensions to the Miller-Tucker-Zem-lin subtour elimination constraints. Oper. Res. Lett. Res. Lett. **10**(1), 27–36 (1991)
9. Sherali, H.D., Driscoll, P.J.: On tightening the relaxations of Miller-Tucker-Zemlin formulations for asymmetric traveling salesman problems. Oper. Res. Res. **50**, 656–669 (2002)
10. Sawik, T.: A note on the Miller-Tucker-Zemlin model for the asymmetric traveling salesman problem. Bull. Pol. Acad. Sci. Tech. Sci. **64**(3), 517–520 (2016). https://doi.org/10.1515/bpasts-2016-0057
11. Sherali, H.D., Sarin, S.C., Tsai, P.: A class of lifted path and flow-based formulations for the asymmetric traveling salesman problem with and without precedence constraints. Discret. Optim.. Optim. **3**, 20–32 (2006)
12. Gouveia, L., Pires, J.M.: The asymmetric travelling salesman problem and a reformulation of the Miller-Tucker-Zemlin constraints. Eur. J. Oper. Res. **112**, 134–146 (1999)
13. Martin, R.K.: Using separation algorithms to generate mixed integer model reformulations. Oper. Res. Lett. Res. Lett. **10**, 119–128 (1991)

14. García, J.M. Medium-High Level Problems. In: Building and Solving Mathematical Programming Models. International Series in Operations Research & Management Science, vol 329. Springer, Cham. (2022) https://doi.org/10.1007/978-3-030-97626-2_6
15. Aguayo, M., Sarin, S.C., Sherali, H.D.: Solving the single and multiple asymmetric traveling salesmen problems by generating subtour elimination constraints from integer solutions. IISE Trans. **50**(1), 45–53 (2018). https://doi.org/10.1080/24725854.2017.1374580
16. Miliotis, P.: Integer programming approaches to the travelling salesman problem. Math. Program. **10**, 367–378 (1976)
17. Pferschy, U., Stanek, R.: Generating subtour elimination constraints for the TSP from pure integer solutions. CEJOR **25**, 231–260 (2017). https://doi.org/10.1007/s10100-016-0437-8
18. Crowder, H., Johnson, E.L., Padberg, M.: Solving large-scale zero-one linear programming problems. Oper. Res. **31**(5), 803–834 (1983)
19. Bosch, R.: Connecting the dots: the ins and outs of TSP art. In: Séquin, C.H. (ed.) Sarhangi R. Bridges Leeuwarden: mathematics, music, art, architecture, culture, pp. 235–242. Southwestern College, Winfield (2008)
20. Khoury, B.N., Pardalos, P.M., Du, D.Z.: A test problem generator for the Steiner problem in graphs. ACM Trans. Math. Softw. **19**(4), 509–522 (1993). https://doi.org/10.1145/168173.168420
21. Sawik, B., Serrano-Hernandez, A., Muro, A., Faulin, J.: Multi-criteria simulation-optimization analysis of usage of automated parcel lockers: a practical approach. Mathematics **10**(23), 4423 (2022). https://doi.org/10.3390/math10234423
22. Sawik, B., Faulin, J., Serrano-Hernandez, A., Ballano, A.: A simulation-optimization model for automated parcel lockers network design in urban scenarios in Pamplona (Spain), Zakopane, and Krakow (Poland). Winter Simul. Conf. (WSC) Singapore **2022**, 1648–1659 (2022). https://doi.org/10.1109/WSC57314.2022.10015458