Exact and approximate formulations for the Close-Enough TSP

Extended abstract submitted for presentation at the 12th Triennial Symposium on Transportation Analysis conference (TRISTAN XII)
June 22-27, 2025, Okinawa, Japan

October 31, 2024

Keywords: close-enough traveling salesman problem, second-order cone programming formulation, polyhedral approximation

1 INTRODUCTION

The Traveling Salesman Problem (TSP) aims to find the shortest Hamiltonian cycle over a set of vertices, with corresponding distances between them. The Close-Enough TSP (CETSP) is a generalization where it is sufficient to visit a predefined compact region around each vertex. This problem has several applications, such as radio frequency identification-based meter reading (Di Placido *et al.*, 2022) and unmanned aerial vehicle routing (Coutinho *et al.*, 2016). A common assumption in CETSP literature is that neighborhoods are disks centered at vertices (Coutinho *et al.*, 2016). Under this assumption, the problem can be formally defined as follows: there is a set I of n vertices in a two-dimensional space, with corresponding coordinates $c_i = (c_{ix}, c_{iy})$ $\forall i \in I$. Each vertex i is covered by a closed ball centered at c_i with radius r_i. It is assumed $r_0 = 0$ so that vertex $i = 0$ serves as the depot. The problem consists of finding the shortest way to, starting from the depot, visit each neighborhood and then return to the depot.

While there are multiple heuristics for finding quality solutions to the problem (Di Placido *et al.*, 2022), there is a notable lack of exact algorithms that find optimal solutions. Mennell (2009) introduced the overlap ratio as a measure of problem instance difficulty, which is the ratio between the average radius and the longest side of the smallest rectangle that contains all neighborhoods. This metric represents how much the objective value of the problem can decrease compared to the TSP objective value on the same vertices without neighborhoods.

This work proposes two exact formulations for CETSP, along with arbitrarily close linear approximations. Computational results for all mentioned formulations are presented.

2 METHODOLOGY

In the arc-based formulation (ABF), the variables x_{ij} represent the transitions between neighborhoods, taking the value 1 if the route moves from neighborhood i to j. The variables d_{ij} represent the distance between representative points of the neighborhoods, and da_{ij} is an auxiliary variable to define the objective function. Finally, the variables p_i represent the coordinates of the representative points for each neighborhood. Given this setup, the objective function (1) minimizes the total distance traveled, constraints (2) and (3) ensure a cycle is formed, constraints (4) to (6) with variable u prevent subtour formation (Gavish & Graves, 1978), constraint (7) ensures representative points are within neighborhoods, constraint (8) defines neighborhood distances based on representative points, and finally, constraint (9) activates da_{ij}, causing it to equal $x_{ij}d_{ij}$. The parameter M_{ij} is the maximum distance between neighborhoods, ensuring the

constraint is trivially satisfied if $x_{ij} = 0$.

$$\min \sum_{i \in I} \sum_{j \in I} da_{ij} \tag{1}$$

$$\text{s.t.} \sum_{j \in I} x_{ij} = 1, \forall i \in I \tag{2}$$

$$\sum_{i \in I} x_{ij} = 1, \forall j \in I \tag{3}$$

$$\sum_{j \in I} u_{ij} - \sum_{j \in I} u_{ji} = -1, \forall i \in I \setminus \{0\} \tag{4}$$

$$\sum_{j \in I} u_{0j} - \sum_{j \in I} u_{j0} = n - 1 \tag{5}$$

$$u_{ij} \leq (n-1)x_{ij}, \forall i, j \in I \times I \tag{6}$$

$$\|p_i - c_i\|_2 \leq r_i, \forall i \in I \tag{7}$$

$$\|p_i - p_j\|_2 \leq d_{ij}, \forall i, j \in I \times I \tag{8}$$

$$d_{ij} - M_{ij}(1 - x_{ij}) \leq da_{ij}, \forall i, j \in I \times I \tag{9}$$

$$x_{ij} \in \{0, 1\}, \forall i, j \in I \times I \tag{10}$$

$$d_{ij}, da_{ij} \in \mathbb{R}, \forall i, j \in I \times I \tag{11}$$

$$p_i \in \mathbb{R}^2, \forall i \in I \tag{12}$$

$$u_{ij} \in \mathbb{R}_{\geq 0}, \forall i, j \in I \times I \tag{13}$$

In the sequence-based formulation (SBF), the variables z_{ik} indicate whether neighborhood i is visited in stage k, q_k represents the coordinates of the point visited in stage k, and t_k indicates the distance traveled in each stage. The objective function (14) minimizes the total distance, constraint (15) ensures each neighborhood is visited in exactly one stage, constraint (16) ensures a neighborhood is visited in each stage, constraint (17) fixes the first neighborhood as the depot to eliminate symmetry in solutions, constraint (18) defines distances between points, and finally, constraint (19) ensures that the representative point of stage k is within the neighborhood visited at that stage.

$$\min \sum_{k \in N} t_k \tag{14}$$

$$\text{s.t.} \sum_{k \in N} z_{ik} = 1, \forall i \in I \tag{15}$$

$$\sum_{i \in I} z_{ik} = 1, \forall k \in N = \{1, ..., |I|\} \tag{16}$$

$$z_{00} = 1 \tag{17}$$

$$\|q_k - q_{k+1}\|_2 \leq t_k, \forall k \in N \tag{18}$$

$$\left\| \sum_{i \in I} z_{ik} \cdot c_i - q_k \right\|_2 \leq \sum_{i \in I} z_{ik} \cdot r_i, \forall k \in N \tag{19}$$

$$z_{ik} \in \{0, 1\}, \forall i \in I, \forall k \in N \tag{20}$$

$$q_k \in \mathbb{R}^2, \forall k \in N \tag{21}$$

$$t_k \in \mathbb{R}_{\geq 0}, \forall k \in N \tag{22}$$

Due to the high complexity associated with solving the previous problems, it is helpful to approximate them using linear programming. For this purpose, Ben-Tal & Nemirovski (2001) proposed a linear approximation of cone constraints such that, given a precision ϵ and denoting

(QCP) as the feasible region of the original problem, (LP_ϵ) as the feasible region of the linear approximation, and (QCP_ϵ) as the feasible region of the problem by relaxing the original cone constraints by a factor of $(1+\epsilon)$, it holds that $(QCP) \subseteq (LP_\epsilon) \subseteq (QCP_\epsilon)$. This means the approximation is ϵ-close to the original problem. This precision is controlled by a parameter ν, where $\epsilon(\nu) = \left(\cos\left(\frac{\pi}{2\nu+1}\right)\right)^{-1} - 1$. We denote LP-ABF and LP-SBF the resulting formulations. Both approximations provide lower bounds on the objective value of the original formulations. To obtain a feasible solution to the original problem, and thus an upper bound, one can take the visit sequence given by the solution of LP-ABF (resp. LP-SBF), fix the binary variables in ABF (resp. SBF), and solve for the remaining variables with second-order cone programming.

3 RESULTS

To compare the previous formulations, random instances were generated with different numbers of vertices placed randomly within a square with a side length of 10. If an instance does not have a uniform radius for all neighborhoods, these are also assigned randomly. Before solving, preprocessing is performed: if one neighborhood is entirely contained within another, the latter is eliminated, as visiting the first neighborhood trivially covers it, thus preserving optimality. All presented instances were solved using Gurobi 10.0.1 in Python, with a time limit of 600 seconds.

Table 1 presents the results for ABF and SBF in instances of up to 15 vertices, with varied radii. The column n indicates the number of vertices, with the number in parentheses representing the count of vertices remaining after preprocessing if vertices are eliminated. The Gap represents the relative difference between the best upper and lower bounds, and Time denotes the time required to solve the instance. The table shows that overlap has a significant relationship with computational time, though this effect can be offset by the higher likelihood of vertex elimination in preprocessing with increased overlap. It is also evident that, for 10-vertex instances, optimality is mostly reached, but as the number increases to 15 vertices, the time limit is reached. Finally, SBF generally outperforms ABF in most instances.

Table 2 presents the results for LP-ABF and LP-SBF on the same instances, using $\nu = 3$, which provides an accuracy of $\epsilon = 0.02$. This table shows that both LP-ABF and LP-SBF yield solutions with smaller optimality gaps in less time. However, LP-ABF is outperformed by SBF in these instances. This could be due to ABF having n^2+n conic constraints, whereas SBF has only $2n$. This quadratic increase in constraints makes ABF significantly more complex to solve, while the linear programming approximation results in additional variables and constraints. Notably, LP-SBF produces solutions with a gap of at most 1% from optimality, taking on average less than half a minute, without requiring a high value for the approximation's construction parameter.

Finally, we point out that Benders-like decompositions did not significantly improve performance of any formulation, and thus we do not report results on them.

4 CONCLUDING REMARKS

The results suggest that the difference in the growth rate of the number of constraints with respect to the number of vertices between ABF and SBF makes the second formulation and its derivatives more computationally manageable. Promising lines for further research include finding better decompositions to simplify ABF and SBF, methods to close the gaps in the linear approximations to guarantee optimality, and possibly decomposing these approximations to handle larger instances.

References

Ben-Tal, A., & Nemirovski, A. 2001. On Polyhedral Approximations of the Second-Order Cone. *Mathematics of Operations Research*, **26**(2), 193–205.

Table 1 – *Results of ABF and SBF*

n	**Radii**	**Overlap ratio**	**ABF**		**SBF**	
			Gap (%)	Time (s)	Gap (%)	Time (s)
10	0.05-0.5	0.0275	0	11.3	0	9
10	0.25-1	0.0625	0	35.4	0	17.5
10(6)	0.5-2	0.125	0	0.5	0	0.9
10	0.25	0.025	0	4.3	0	12.5
10	0.5	0.05	0	16.8	0	13.6
10	1	0.1	0	82.3	0	45.9
10	1.5	0.15	54.5	600	0	80.2
10(7)	2	0.2	0	2.8	0	9.1
Average			6.8	94.2	0	23.6
15	0.05-0.5	0.0275	58.6	600	30.4	600
15	0.25-1	0.0625	75.6	600	16.4	600
15(12)	0.5-2	0.125	49.2	600	0	508.1
15	0.25	0.025	41.2	600	17.8	600
15	0.5	0.05	52	600	17.6	600
15	1	0.1	81.9	600	18.7	600
15	1.5	0.15	80.4	600	16.7	600
15(11)	2	0.2	79.6	600	0	53.2
Average			64.8	600	14.7	520.2

Table 2 – *Results of LP-ABF and LP-SBF*

n	**Radii**	**Overlap ratio**	**LP-ABF**		**LP-SBF**	
			Gap (%)	Time (s)	Gap (%)	Time (s)
15	0.05-0.5	0.0275	0.34	37.49	0.34	35.85
15	0.25-1	0.0625	26.43	600	0.5	13.52
15(12)	0.5-2	0.125	29.6	600	0.44	2.1
15	0.25	0.025	0.72	60	0.72	22.3
15	0.5	0.05	5	600	0.97	71.9
15	1	0.1	31.5	600	1.19	46.6
15	1.5	0.15	74	600	0.8	8.6
15(11)	2	0.2	27.94	600	0.94	2
Average			24.4	529.7	0.73	25.4

Coutinho, W. P., Nascimento, R. Q., Pessoa, A. A., & Subramanian, A. 2016. A Branch-and-Bound Algorithm for the Close-Enough Traveling Salesman Problem. *INFORMS Journal on Computing*, **28**(4), 752–765.

Di Placido, A., Archetti, C., & Cerrone, C. 2022. A genetic algorithm for the close-enough traveling salesman problem with application to solar panels diagnostic reconnaissance. *Computers Operations Research*, **145**, 105831.

Gavish, B., & Graves, S. C. 1978. The Travelling Salesman Problem and Related Problems. *Operations Research*. Corpus ID: 14353795.

Mennell, W. K. 2009. *Heuristics for Solving Three Routing Problems: Close-Enough Traveling Salesman Problem, Close-Enough Vehicle Routing Problem, Sequence-Dependent Team Orienteering Problem*. Ph.D. thesis, The Robert H. Smith School of Business, University of Maryland.