A modified ACO with K-Opt for restricted covering salesman problems in different environments

Prasanta Dutta1 · Indadul Khan2 · Krishnendu Basuli3 · Manas Kumar Maiti4

Accepted: 3 March 2022 / Published online: 8 April 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract
In this study, the ant colony optimization (ACO) algorithm is modified with the K-Opt operation to solve the covering salesman problem (CSP) under one restriction in crisp and imprecise (fuzzy, rough) environments. A CSP involves two phases—the division of cities into groups with the selection of the visiting cities and searching of the Hamiltonian circuit through the visiting cities. But, none of the studies in the literature is made following the direct approach. Also, none of the studies in the literature gives attention to reduce the total travel distance of the unvisited cities from the visited city of a group. Moreover, there is no algorithm in the literature which provides the solution of a CSP with the specified coverage range \( r \). Also, none has introduced any algorithm to solve CSPs in imprecise environments. Though algorithms are available to solve the Traveling Salesman Problems in the imprecise environments, the approach cannot deal with the problems involving fuzzy data with nonlinear membership functions or the problems involving rough data where the rough estimation can not be done using Lebesgue measure. The well-established algorithm for any routing problem is the ACO, but not much attention has been paid to solve the CSP using ACOs. To overcome these limitations on the studies of the ACO on the CSPs, here, an algorithm is proposed for the division of groups of the set of cities depending upon the maximum number of cities in a group and the total number of groups. Then, ACO is used to find the shortest/minimum-cost path of the problem by selecting only one visiting the city from each group without violating the restriction of the specified coverage range \( r \) of the location of the unvisited cities. K-Opt operation is applied periodically at the end of ACO operation to improve the quality of the best found solution so far by the ACO algorithm and to arrest any premature convergence. For the restricted problems, paths are searched in such a manner that the total distance/travel cost of different unvisited cities of a group from the visited city of the group should not exceed a predefined upper limit. To solve the problem in an imprecise environment, some approach is followed so that the tour is searched without transferring the imprecise optimisation problem into an equivalent crisp optimisation problem. Also, the simulation approaches in fuzzy and rough environments are proposed to deal with the CSPs with any type of estimation of the imprecise data set. Algorithm is tested with the standard benchmark crisp problems available in the literature. To test the algorithm in the imprecise environments, the imprecise instances are derived randomly from the standard crisp instances using a specified rule. Test results imply that the proposed algorithm is efficient enough in solving the CSPs in the crisp as well as in the imprecise environments.

Keywords Covering salesmen problem · Ant colony optimization · K-Opt operation · Fuzzy simulation · Rough simulation

1 Introduction

A covering salesman problem (CSP) is a complete weighted graph, consists of a set of vertices, called cities, and a weight matrix, called distance (cost) matrix which consists of the distance (travelling cost) between any two cities. The goal of the problem is to find the different clusters (groups) of the cities and then find a minimum cost Hamiltonian circuit visiting only one city in each cluster so that each unvisited city in a cluster should be located in a predefined cover-
age range of the visiting city. A maximum number of cities in a group is provided in the problem. The clustering and the searching of Hamiltonian circuit is made in such a manner that the travelling distance (cost) through the circuit is a minimum. The CSP has various real-life applications, like, telecommunication Kotenko and Saenko (2019), health care Kergosien et al. (2009), disaster management Maurya and Kumar (2017), humanitarian relief Campbell et al. (2008), transportation Ungureanu (2006), etc. For the relief of affected areas of different natural disasters, like, flood, earthquake, tsunami, etc., the locations of the relief camps are selected depending upon the human resources in such a manner that all the affected peoples can get different services with the minimum effort and expenditure. Mobile communication tower points are selected in such a way that covers all the customers’ area with the minimum expenditure. So, the CSP has a wide range of applications in the real-life problems.

As CSP belongs to the class of NP-hard problems, exact (analytical approach) algorithms are not suitable to find the solution of a CSP in a usable time window. So heuristic approaches are applicable to find a compromise solution in a specific time window. The initial study in this direction was made by current and schilling [1] in 1989. They proposed the problem, and the problem was solved by the authors using a two phase heuristics approach. In the first phase of the algorithm, the set covering approach is followed to find the clusters of the vertex set. Then, one node from each cluster is selected as the facility node. Then, a traveling salesman problem (TSP) is formed through the selected nodes. In the second phase, this TSP is solved using any suitable heuristics approach. These two phases are repeated iteratively to search the best path.

Using Lin–Kernighan procedure Lin (1973), Golden et al. Golden et al. (2012) proposed two heuristics search algorithms, \( L_1 \) and \( L_2 \) for solving different CSPs in 2012. Both the algorithms start with a randomly generated set of feasible solutions of the problem. Then, \( L_1 \) uses some stochastic approach to replace some nodes from the path by some new nodes to improve the quality of the solution in such a manner that the feasibility of the solution persists. The authors used mutation operations to prevent any local optima and used some uphill moves to introduce diversity in the algorithm. On the other hand, \( L_2 \) uses two iterative procedures, namely, the Improvement Procedure and the Perturbation Procedure to find the best feasible solution.

Using the features of integer linear programming (ILP), Salari and Naji-Azimi Salari and Naji-Azimi (2012) proposed a Heuristics method for solving CSP in 2012. The algorithm starts with a randomly generated set of feasible solutions of the problem, and the tour is improved iteratively using two procedures, namely, Heuristic-based improvement and ILP-based improvement. An Extraction-Reassignment procedure is used to decrease the tour length in the heuristic phase. At first, a subset of the vertices visited by the tour is extracted and then, this vertices are reassigned into a new tour by solving the ILP model for the improvement of the objective function. Another heuristic algorithm by Lin–Kernighan Lin (1973) is also used for the possible improvement of the tour. Also, a perturbation phase is introduced in the algorithm to escape from the local optima.

Similar to the CSP, another problem, named, Covering Tour Problem (CTP), was proposed by Gendreau et al. Gendreau et al. (1992) in 1992. In this problem, the set of vertices, \( V \), is partitioned into to two subsets \( V_1 \) and \( V_2 \) and the goal is to search a minimum cost Hamiltonian circuit through \( V_1 \) so that the distance of each of the vertices of \( V_2 \) from the circuit is located within a prescribed distance. They first formulate the problem as an ILP problem, and then, an exact branch and cut algorithm is developed to solve the problem. They have also proposed a heuristics approach for it. In a recent study, Singh et al. Pandiri et al. (2020) proposed a mixed integer programming model for a bi-objective generalised covering salesman problem, where simultaneously, the total covered demand is maximised and the total tour length is minimised.

Combining the features of ant colony optimization (ACO) algorithm and the dynamic programming technique, Salari et al. Salari et al. (2015) proposed a heuristic approach to solve CSPs. Together with 3-opt, they introduced two perturbation approaches vertex-removal and vertex-addition for searching quality solutions. Venkatesh and Singh Venkatesh et al. (2019) modified the Artificial Bee Colony (ABC) algorithm with different perturbation techniques to solve the CSPs in 2019. Venkatesh et al. Venkatesh et al. (2019) proposed a local search heuristic for the CSPs using multiple perturbation strategy. In another study, Pandiri et al. Pandiri et al. (2020) modified two heuristic approaches ABC and Genetic Algorithm (GA) to solve CSPs in 2020. They proposed four approaches for selecting neighbour solutions in both the algorithms—GA and ABC, namely, Subset Neighbour, Local search with two neighbour structures, General Exchange and Permutation neighbour. Zang et al. Zang et al. (2020) defined Bilevel CSP(BCSP) and proposed two algorithms based on, parallel variable neighbourhood search (PVNS), namely, synchronous master slave PVNS and asynchronous cooperative PVNS, for the same. A crossover procedure using GPX was proposed by Tripathy et al. Tripathy et al. (2017) and using it a GA was developed by them for the CSPs in 2017. Though a significant amount of studies is made on the CSPs after its introduction by current and schilling [1], there are some lacunas in the existing literature, which are summarised below:

- The well-established algorithm for any routing problem is the ACO, but not much attention has been paid to solve the CSPs using ACO Salari et al. (2015).
A modified ACO with K-Opt for restricted covering...

- It is known that a CSP involves two phases—division of cities into groups with the selection of visiting cities and searching of the minimum cost Hamiltonian circuit through the visiting cities. But, none of the studies in the literature is made following this direct approach.
- Moreover, none of the studies gives attention to reduce the total travel distance of the unvisited cities from the visited city of a group. Though in real-life problem, it is crucial as different facilities are provided from the visited cities only.
- Also, there is no algorithm in the literature, which gives results of the CSP instances with the specified coverage range \( r \). All the studies have been made where groups are created with the nearest nodes from the selected centres of the groups.
- All the studies to date proposed different heuristics for CSPs for the betterment of the existing results, which in turn motivates us for the further betterment.
- In all the above studies, it is observed that the problems are considered in the crisp environment only, i.e. the cost of travel between any two specific cities is fixed, which is unrealistic in any real-life situation. In fact, the cost of travel between any two places varies with several factors, like, road conditions, type of vehicle used, the route used, etc. This cost is actually imprecise in nature and can be estimated as rough or fuzzy number using experts’ opinion. This type of estimation is less error prone as it is done using experts’ opinion. Though there are algorithms to solve basic TSPs and its variants in the imprecise environments Khan and Maiti (2018); Khan et al. (2018, 2019), none has introduced any algorithm to solve the CSPs in any imprecise environment.
- Though algorithms are available to solve TSPs in the imprecise environments Khan et al. (2018, 2019), the approach cannot deal with the problems involving fuzzy data with nonlinear membership functions (e.g. Parabolic Fuzzy Number(PFN)) or TSPs involving Rough data where the estimation cannot be done using Lebesgue measure.

To overcome the above mentioned lacunas, in this study an attempt has been made to introduce a better heuristic approach to solve the CSPs in the crisp and as well as in the imprecise environments. Simulation approaches in the imprecise environments are proposed to deal with the CSPs in the imprecise environments for any type of imprecise data sets. Moreover, a restricted problem is introduced and solved where the total travel distance of different unvisited cities from the visited city of the corresponding group should not exceed a prescribed limit.

The heuristic algorithm proposed here consists of three phases. In the first phase of the algorithm, the set of cities is partitioned into groups depending upon the maximum number of cities in a group and the total number of groups. A procedure is proposed for this purpose where initially a centre of each group is selected, and then, other cities of the group are selected in such a manner that all other cities are located within the specified coverage range \( r \). The centre of the next group is selected from the unselected cities which is nearest to this centre. In the second phase of the algorithm, the ACO Dorigo and Di Caro (1999) is used to find the shortest/minimum-cost path of the problem by visiting only the centre of each group. In the last phase, the K-Opt algorithm is used to improve the quality of the solution obtained by the ACO and to resist premature convergence. For the restricted problem, the path is searched in such a manner that the total distance/cost of different unvisited cities of a group from the visited city of the corresponding group should not exceed a predefined upper limit. To solve the problem in an imprecise environment, an approach is followed, where, the tour is searched without transferring the imprecise optimisation problem into any equivalent crisp optimisation problem. Moreover, fuzzy simulation and rough simulation approaches are proposed to deal with the problem with any general type imprecise data set.

The rest of the paper is organized as follows: in Sect. 2, the background and the main challenges of solving the CSPs in different environments are discussed. The required assumptions and different notations for the mathematical representation of the problem and the development of the proposed algorithm are listed in Sect. 3. The mathematical representation of the problem and some technical prerequisites for the proposed solution approach are presented in Sect. 4. Optimization in imprecise environments and the simulation approaches are discussed in Sect. 5. The ACO algorithm for the CSP is described in Sect. 6. In Sect. 7, the K-Opt algorithm is presented. Proposed approach is presented in Sect. 8. Experimental results are discussed in Sect. 9. A brief conclusion is drawn in Sect. 10. At length, the references are listed.

2 Background of the CSP and challenges

CSP was defined by Current and Schilling [1] in 1989. They formulated the problem as a zero-one linear programming problem (LPP) and proposed a heuristic approach to solve the same. In their approach, in the first step, the subset of cities to cover all the cities is determined using the approach of solving a set covering problem (SCP). For each of the solution of SCP, a TSP is generated and solved. The minimum cost tour is the solution of the CSP. As SCP and TSP, both are NP-hard problems, so it is not possible to solve a moderate size CSP in a reasonable time window following this approach.
Combining the features of dynamic programming technique, ACO and amalgamating different perturbation techniques, like, 3-opt, vertex removal and vertex addition techniques, etc., Salari et al. Salari et al. (2015) developed a novel heuristic to solve CSPs in 2015. In this study, paths are created by selecting successive cities using ACO until all the cities are covered. Then, the vertex removal and vertex addition techniques are used to improve the tour cost. Finally, 3-opt operation is applied on the path for further possible improvements. This study ignores the restriction of the upper limit, \( r \), of the coverage range of the unvisited cities and consider only the nearer nodes of any visiting city as the covered nodes for searching the path.

In a recent study, Pandiri et al. Pandiri et al. (2020) proposed two meta-heuristics for the CSPs. In the first approach, ABC algorithm is modified for the CSPs using four perturbation operations on a tour. One of these approach is the removal of a visited vertex and addition of an unvisited vertex which can cover the removed vertex and they named it LS2N. Another approach is the permutation of neighbour, where, some permutation is made on the positions of the visited cities in a tour to improve the quality. The third approach is the general exchange (GE) of an unvisited city with a visited city, such that the feasibility of the tour holds. The fourth approach is named subset neighbour (SSBN) and in this approach some cities of a tour are removed with some probability and some unvisited cities are added so the tour becomes feasible as well as the quality is improved. Using these perturbation approaches, they developed a tour improvement algorithm and is used in the employed bee phase and in the onlooker bee phase of the basic ABC to make it useful for the CSPs. For the generation of initial solution set for the ABC as well as for the GA approach, they followed the same approach as Salari et al. Salari et al. (2015). In GA, cyclic crossover operation is used, and for the mutation process, they proposed two new approaches. They have also used the perturbation approaches of the ABC algorithm in the GA for the further possible improvements. Performances of both these approaches are similar to the hybrid ACO proposed by Salari et al. Salari et al. (2015). So the approaches of Pandiri et al. Pandiri et al. (2020) did not give better solution than that of Salari et al. Salari et al. (2015). Moreover, this study also ignores the restriction of the upper limit, \( r \), of the coverage range of the unvisited cities and consider only the nearer nodes of any visiting city as the covered nodes for searching the path.

All the above studies as well the studies mentioned in the introduction section, the initial paths are created using successive addition of visiting cities to cover all the cities following the approaches of set covering problem, but none has mentioned the covering range in their test instances, i.e. the maximum distance of an unvisited city from its nearest visiting city, though the tour cost mostly depends on it and it is also the basic restriction of the problem. Also, none of these studies tries to reduce the total group covering distance of a visiting city, though it is realistic one for any real-life problem. Since the CSP is a routing problem, selection of the visiting cities and the determination of the optimal route through these cities using any suitable routing algorithm may produce better result (as proposed by Current and Schilling [1]). But the approach is overlooked by the researchers for developing any heuristic approach. Moreover, none have studied the CSPs in imprecise environments. There are some studies on the TSPs and GTSPs with fuzzy cost matrices and rough cost matrices Khan and Maiti (2018); Khan et al. (2018, 2019), using the credibility measure on fuzzy events and the trust measure on rough events, where TFNs are used as the fuzzy parameters and the Lebesgue measure is used for the estimation of rough parameters. Their approach cannot deal with such problems involving fuzzy cost matrices with nonlinear membership functions. In fact, when more than one fuzzy parameters with nonlinear membership function (e.g. PFN) are added, then it is not possible to determine the analytical form of the membership function of the resultant fuzzy number. So following their approach, it is not possible to determine the credibility measure of the fuzzy events involving the process when the membership functions of the fuzzy parameters are nonlinear type. The same problem has to be face in the rough environment also, if the Lebesgue measure is not possible to use for the rough estimation.

To overcome these limitations in this study, fuzzy simulation approach and rough simulation approaches are proposed to deal with such crucial situations. Moreover, the CSP is solved with an approach where initially the cities are divided into groups selecting one possible visiting city of each group. Then, the routing heuristic ACO is used to determine the optimal route through the visiting cities. K-Opt operation is also used periodically for the possible improvement of the best path found so far and to resist any premature convergence. In a particular case, the CSP is solved with the restriction limit of the total group coverage range as well as incorporating the normal restriction on the coverage range, \( r \).

3 Assumptions and notations

Following assumptions and notations are used to formulate the problem:

(i) \( n \) is the number of cities involved in the problem.
(ii) \( NC \) is the maximum number of unvisited cities covered by a visiting city.
(iii) \( NB \) is the maximum number of cluster of cities.
(iv) \( c_{ij} \) is the travelling distance/cost between city \( i \) and city \( j \).
(v) \( r \) is the maximum distance between a visiting city and the furthest unvisited city covered by that visiting city.

(vi) \((x_{ij})_{n \times n}\) is the decision variable matrix for the mathematical representation of the problem.

\[
\begin{align*}
\text{where } x_{ij} &= \begin{cases} 
1 & \text{if the salesman moves to city } j \text{ from city } i \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

(vii) Another array variable \((v_i)_{1 \times n}\) is also used for the mathematical representation of the problem, where,

\[
\begin{align*}
\text{where } v_i &= \begin{cases} 
1 & \text{if the salesman visits city } i \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

(viii) Also, the coverage matrix \((y_{ij})_{n \times n}\) is defined as,

\[
\begin{align*}
\text{where } y_{ij} &= \begin{cases} 
1 & \text{if } c_{ij} \leq r \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

(ix) \( Z \) is the total tour cost/travel distance.

(x) The ascent \( \tilde{\ } \) is used over a symbol to indicate fuzzy quantity.

(xi) The ascent \( \tilde{\ } \) is used over a symbol to indicate rough quantity.

4 Technical background

Covering Salesman Problem: The basic CSP consists of a set of cities \( V = \{1, 2, \ldots, n\} \) where the travelling distance/cost between any two cities \( i \) and \( j \) is known and fixed, say \( c_{ij} \). The goal of the problem is to search a minimum distance Hamiltonian circuit through a subset of cities so that the position of every unvisited city is located within a predefined coverage range \( r \) of a visited city. Then, the problem mathematically takes the following form

\[
\text{Minimize } Z = \sum_{i \in V} \sum_{j \in V} c_{ij}x_{ij} \tag{1}
\]

Subject to

\[
\begin{align*}
\sum_{j \in V} x_{ij} + \sum_{j \in V} x_{ji} &= 2v_i \quad \forall i \in V \tag{2} \\
\sum_{j \in V} y_{ji}v_j &\geq 1 \quad \forall i \in V \tag{3}
\end{align*}
\]

\[
\sum_{i \in S} \sum_{j \in V \setminus S} x_{ij} + \sum_{i \in V \setminus S} \sum_{j \in S} x_{ij} \geq 2(v_i + v_k - 1), \quad S \subseteq V, \ 2 \leq |S| \leq n - 2, \forall i \in V, \ k \in V \setminus S \tag{4}
\]

In the formulation, the objective function \( Z \) is the total travel distance of the tour, which is to be minimized. Constraints set (5) implies that if the salesman visits a city he must leave the city after his visit. Constraints set (6) implies that every vertex in the vertex set \( V \) is located in a covering distance \( r \) from a visited city. Finally, equation set (7) implies that the tour should not contain any sub-tour.

In the existing literature, every problem consists of a pre-defined number of clusters \((NB)\) and upper limit \((NC)\) of the number of cities in a cluster. A salesman should visit only one city of each cluster so that the problem constraints are satisfied. To solve the problem here, following steps are followed:

- At first, randomly select a city \( c_1 \) to be visited, and search \( NC - 1 \) cities nearest to the selected city. If these cities are located inside the covering distance \( r \), then this set of vertices is named as the first cluster. If it is not true, then another vertex is selected and the same process is continued until a cluster is created.
- After the formation of a cluster, the nodes in the clusters are eliminated from the node set and let it be \( V_1 \).
- Select next city \( c_2 \) to be visited from \( V_1 \) which is nearest to \( c_1 \) and form a cluster following the same procedure as first cluster. If fails, then the next nearest node of \( V_1 \) from \( c_1 \) is selected as \( c_2 \) and the process is continued.
- Following the same procedure, different clusters are created. In this formation, in any step if it fails to create a cluster, then the process again starts from the formation of first cluster.

A salesman at first partitioned the cities into some clusters according to the constraints. Then, the salesman visits one of the cities of each group which is located within the coverage range \( r \) of all the unvisited cities for the selling/canvassing/distribution of some products to its customers of the different cities of the groups. The goal of the problem is threefold:

- First partition the cities into clusters depending upon the maximum number of cities in a cluster and clusters limit.
- Search a minimum cost path for the traveller through all the clusters by visiting only one city of each cluster.
- Every unvisited city must lies within a predefined covering distance from a visited city.

Fuzzy Number: A fuzzy set \( \tilde{F} \) is defined on an universe \( U \) such that every \( u \in U \) belongs to the set \( \tilde{F} \) with some degree of membership \( \mu_F(u) \in [0, 1] \). The function \( \mu_F : U \to [0, 1] \subset \mathbb{R} \) is called the membership function of the fuzzy set \( \tilde{F} \). In other words, the function \( \mu_F \) completely defines the fuzzy set \( \tilde{F} \) on \( U \). A fuzzy set normally denoted by using an ascent over the name. A fuzzy number \( \tilde{A} \) is a fuzzy set on \( \mathbb{R} \) such that its membership function \( \mu_A \) satisfies the following two conditions Zadeh (1965):

- \( \exists a \in \mathbb{R} \) such that \( \mu_A(a) = 1 \).
• ∀a, b ∈ ℝ and λ ∈ (0, 1), \( μ_A(\lambda a + (1 - \lambda)b) \geq \min\{μ_A(a), μ_A(b)\}\)

**Triangular Fuzzy Number (TFN):** A TFN \( \tilde{A} \) having membership function \( μ_A \) has positive membership in an interval [a, c] and has membership value 1 at only one point \( b \in [a, c] \). It is denoted by \( \tilde{A} = (a, b, c) \), and its membership function \( μ_{\tilde{A}} \) is given by

\[
μ_{\tilde{A}}(x) = \begin{cases} 
\frac{x - a}{b - a} & \text{for } a \leq x \leq b \\
\frac{c - x}{c - b} & \text{for } b \leq x \leq c \\
0 & \text{otherwise}
\end{cases}
\]  

(5)

**Parabolic fuzzy number (PFN):** A PFN \( \tilde{A} \) having membership function \( μ_A \) has positive membership in an interval [a, c] and has membership value 1 at only one point \( b \in [a, c] \). It is denoted by \( \tilde{A} = (a, b, c) \) and its membership function \( μ_{\tilde{A}} \) is given by

\[
μ_{\tilde{A}}(a) = \begin{cases} 
1 - \frac{(b - x)^2}{(b - a)^2} & \text{for } a \leq x \leq b \\
1 - \frac{(x - b)^2}{(c - b)^2} & \text{for } b \leq x \leq c \\
0 & \text{otherwise}
\end{cases}
\]  

(6)

**Fuzzy extension principle:** Let \( f : (X_1 \times X_2 \times \ldots \times X_n) \to Y \) be a function and \( \tilde{A}_1, \tilde{A}_2, \ldots, \tilde{A}_n \) are fuzzy sets defined on \( X_1, X_2, \ldots, X_n \) respectively. Then, \( \tilde{B} = f(\tilde{A}_1 \times \tilde{A}_2 \times \ldots \times \tilde{A}_n) \) is a fuzzy set defined on \( Y \) whose membership function is defined by the extension principle and the set is given by:

\[
\tilde{B} = \{(y, μ_B(y))|μ_B(y) = \max_{x_1 \in X_1, x_2 \in X_2, \ldots, x_n \in X_n} \{\min(μ_{A_1}(x_1), μ_{A_2}(x_2), \ldots, μ_{A_n}(x_n)), y = f(x_1, x_2, \ldots, x_n)\}\}
\]  

(7)

**α-cut of a fuzzy number:** Let \( \tilde{F} \) be a fuzzy number with membership function \( μ_F \). Then, \( \alpha \)-cut of \( \tilde{F} \) is a subset of \( \mathbb{R} \) denoted by \( \tilde{F}_\alpha \) and defined as

\[
\tilde{F}_\alpha = \{x \in \mathbb{R} : μ_F(x) ≥ \alpha\}
\]  

(8)

**Credibility Measure (Cr):** If \( \tilde{A} \) and \( \tilde{B} \) be two fuzzy numbers. Then, \( \tilde{A} \star \tilde{B} \) is a fuzzy event, where \( \star \in \{<, \leq, >, \geq\} \). Considering the level of uncertainty as the semantics of fuzzy numbers, credibility measure of the event \( \tilde{A} \star \tilde{B} \) is denoted by \( Cr(\tilde{A} \star \tilde{B}) \) and is defined as

\[
Cr(\tilde{A} \star \tilde{B}) = \frac{1}{2}[Pos(\tilde{A} \star \tilde{B}) + Nes(\tilde{A} \star \tilde{B})]
\]  

(9)

where \( Pos(\tilde{A} \star \tilde{B}) = \text{sup}\{μ_{\tilde{A}}(a), μ_{\tilde{B}}(b)) : a, b \in \mathbb{R}, \) such that \( a \star b \) holds \}

and \( Nes(\tilde{A} \star \tilde{B}) = 1 - Pos(\tilde{A} \star \tilde{B}) \), where \( \tilde{A} \star \tilde{B} \) denotes the complement of \( \tilde{A} \star \tilde{B} \).

**Lemma 1** Khanra et al. (2016): For any two TFNs \( \tilde{A} = (A_1, A_2, A_3) \) and \( \tilde{B} = (B_1, B_2, B_3) \)

\[
Pos(\tilde{A} > \tilde{B}) = \begin{cases} 
0 & A_3 \leq B_1 \\
\frac{A_1 - B_1}{B_2 - B_1 + A_3 - A_2} & A_3 \geq B_1 \text{ and } A_2 \leq B_2 \\
1 & A_2 \geq B_2
\end{cases}
\]  

(12)

and \( Nes(\tilde{A} > \tilde{B}) = \begin{cases} 
0 & A_2 \geq B_2 \\
\frac{A_3 - B_3}{B_1 - B_2 + A_3 - A_1} & B_2 \leq A_2 \text{ and } A_1 \leq B_3 \\
1 & A_1 \geq B_3
\end{cases}
\]  

(13)

**Lemma 2** Khanra et al. (2016): For any two PFNs \( \tilde{A} = (A_1, A_2, A_3) \) and \( \tilde{B} = (B_1, B_2, B_3) \)

\[
Pos(\tilde{A} > \tilde{B}) = \begin{cases} 
0 & A_3 \leq B_1 \\
1 - \frac{(A_3 - A_2 + B_2 - B_1)^2}{A_3 - A_2 + B_2 - B_1} & A_3 \geq B_1 \text{ and } A_2 \leq B_2
\end{cases}
\]  

(14)

and \( Nes(\tilde{A} > \tilde{B}) = \begin{cases} 
0 & B_2 \geq A_2 \\
1 - \frac{(A_3 - A_2 + B_2 - B_1)^2}{A_3 - A_2 + B_2 - B_1} & B_2 \leq A_2 \text{ and } A_1 \leq B_3
\end{cases}
\]  

(15)

**Lemma 3** If for any fuzzy event \( \tilde{A} \star \tilde{B} \), \( Nes(\tilde{A} \star \tilde{B}) > 0 \),

then \( Cr(\tilde{A} \star \tilde{B}) > 0.5 \).

**Proof**

\[
Nes(\tilde{A} \star \tilde{B}) > 0 \Rightarrow 1 - Pos(\tilde{A} \star \tilde{B}) > 0 \\
\Rightarrow Pos(\tilde{A} \star \tilde{B}) < 1 \Rightarrow Pos(\tilde{A} \star \tilde{B}) = 1 \\
\Rightarrow Cr(\tilde{A} \star \tilde{B}) = \frac{1}{2}[Pos(\tilde{A} \star \tilde{B}) + Nes(\tilde{A} \star \tilde{B})] \\
\Rightarrow Cr(\tilde{A} \star \tilde{B}) > 0.5
\]

**Lemma 4** Liu and Iwamurra (1998): For any fuzzy event \( \tilde{A} \star \tilde{B} \), \( Cr(\tilde{A} \star \tilde{B}) + Cr(\tilde{A} \star \tilde{B}) = 1 \).

**Lemma 5** Khanra et al. (2016): For any two TFNs \( \tilde{A} = (A_1, A_2, A_3) \) and \( \tilde{B} = (B_1, B_2, B_3) \)

\[
Cr(\tilde{A} < \tilde{B})
\]
A modified ACO with K-Opt for restricted covering

\[ \begin{align*}
&= \begin{cases} 
0 & A_3 \leq B_1 \\
\frac{1}{2} \left( 1 + \frac{B_2 - A_2}{A_3 - A_2 + B_2 - B_1} \right) & A_2 \leq B_2 \text{ and } B_1 < A_3 \\
\frac{1}{2} \left( \frac{B_3 - A_3}{B_3 - A_1} \right) & B_2 < A_2 \text{ and } B_3 > A_1 \\
0 & B_3 < A_1
\end{cases} \tag{16}
\end{align*} \]

**5 Optimization in imprecise environment**

Let us consider an optimization problem in the form:

\[
\begin{align*}
\text{Maximize } & f(x, y) \\
\text{subject to } & \phi_i(x, y) \leq 0, i = 1, 2, ..., p \tag{18}
\end{align*}
\]

Here, \( x = (x_1, x_2, ..., x_n) \) is a decision vector having \( n \) variables, \( y = (y_1, y_2, ..., y_m) \) is a vector representing, \( m \), parameters of the problem and \( \phi_i(x, y), i = 1, 2, ..., p \) are constraint functions. In any classical optimization problem, the parametric values are crisp in nature. When the parameters are imprecise in nature, the problem (18) reduces to optimization in imprecise environment.

**5.1 Optimization in fuzzy environment**

In the problem (18), if \( y \) is a vector of fuzzy numbers, \( \tilde{y} = (\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_m) \), the problem (18) reduces to an optimization problem in fuzzy environment having \( n \) crisp decision variables, \( x_1, x_2, ..., x_n \), and is presented below:

\[
\begin{align*}
\text{Maximize } & \tilde{f}(x, \tilde{y}) \\
\text{Subject to } & \tilde{\phi}_i(x, \tilde{y}) \leq 0, i = 1, 2, ..., p \tag{19}
\end{align*}
\]

clearly here objective function as well as the constraints are fuzzy in nature. Optimization in fuzzy environment is not properly defined till date. Moreover, there is no proper guideline to check whether the value of a decision vector is feasible with respect to the constraints involved in the problem. In problem(19), the value \( x_0 \) of the decision vector \( x \) may said to be feasible if the following condition holds

**Lemma 6** [23]: Let Lebesgue measure is used for trust measure and \( \tilde{R}_1 = ([p_1, q_1], [r_1, s_1]) \), \( \tilde{R}_2 = ([p_2, q_2], [r_2, s_2]) \) be two rough variables. Then, trust measure of the rough event \( \tilde{R}_1 < \tilde{R}_2 \) is denoted by \( \tilde{TR}(\tilde{R}_1 < \tilde{R}_2) \) and is given by

\[
\begin{align*}
\tilde{TR}(\tilde{R}_1 < \tilde{R}_2) = \begin{cases} 
0 & \text{for } s_2 - r_1 \leq 0 \\
\frac{q_2 - r_1}{2(q_2 - r_1 - s_2 + s_1)} & \text{for } q_2 - p_1 \leq 0 \leq s_2 - r_1 \\
\frac{1}{2} \left( \frac{s_2 - r_1}{s_2 - r_1 - s_2 + s_1} + \frac{q_2 - p_1}{q_2 - p_1 - q_2 - p_2 + q_1} \right) & \text{for } p_2 - q_1 \leq 0 \leq q_2 - p_1 \\
\frac{1}{2} \left( \frac{s_2 - r_1}{s_2 - r_1 - s_2 + s_1} + 1 \right) & \text{for } r_2 - s_1 \leq 0 \leq p_2 - q_1 \\
1 & \text{for } r_2 - s_1 \geq 0
\end{cases}
\end{align*}
\]

**CSP in imprecise environment**: In the problem (4) when some of the cost parameters are imprecise(Fuzzy/Rough, etc.) in nature, then the goal of the problem reduces to the determination of optimal path under minimization of imprecise objective. So the problem reduces to optimization in the imprecise environment(Fuzzy/Rough, etc.)
similar as, Lemma 5, one can easily determine the values of $\text{Cr}(\phi_i(x, \bar{y}) \leq 0)$, $i = 1, 2, ..., p$ for the determination of the feasibility of any value of the decision vector, $x$. Similarly if the analytical form of the membership function of the fuzzy objective $\tilde{f}(x, \bar{y})$ is available, then one can easily determine the values of $\text{Cr}(\tilde{f}(x', \bar{y}) > \tilde{f}(x', \bar{y}))$ for the determination of the better option between two feasible values, $x'$ and $x''$ of the decision vector, $x$. But if the analytical form of the membership function of any constraint or the analytical form of the membership function of the objective function is not available, then fuzzy simulation approach can be used for the validation of any constraint in (20) or (21). It can be done using the following algorithm.

**Algorithm 1.** Algorithm to verify $\text{Cr}(\bar{\phi}_i(x, \bar{y}) \leq 0) > 0.5$ for a particular value of the decision vector $x$. From Lemma-3, it can be stated that, for a decision vector $x$ if $\text{Nes}(\bar{\phi}_i(x, \bar{y}) \leq 0) > 0$ then $\text{Cr}(\bar{\phi}_i(x, \bar{y}) \leq 0) > 0.5$. Also, $\text{Nes}(\bar{\phi}_i(x, \bar{y}) \leq 0) > \alpha \Rightarrow \text{Pos}(\tilde{\phi}_i(x, \bar{y}) \geq 0) < 1 - \alpha$. So $\text{Cr}(\bar{\phi}_i(x, \bar{y}) \leq 0) > 0.5 \iff \text{Pos}(\tilde{\phi}_i(x, \bar{y}) \geq 0) < 1 - \alpha$ for any $\alpha > 0$. The following algorithm checks the feasibility of the statement $\text{Pos}(\tilde{\phi}_i(x, \bar{y}) \geq 0) < 1 - \alpha$, i.e. feasibility of the statement $\text{Cr}(\bar{\phi}_i(x, \bar{y}) \leq 0) > 0.5$.

1. Input $x$
2. Initialize $N$
3. $j \leftarrow 1$
4. Select randomly one vector $y_0$ from $[\bar{y}]_{1-\alpha}$
5. If $\phi_i(x, y_0) \geq 0$
6. \hspace{1em} Return Infeasible
7. End If
8. $j \leftarrow j + 1$
9. If $j \leq N$
10. \hspace{1em} Go to step 4.
11. End If
12. Return Feasible
13. End Algorithm

**5.2 Optimization in rough environment**

In the problem (18), if $y$ is a vector of rough numbers, $\bar{y} = (\bar{y}_1, \bar{y}_2, ..., \bar{y}_m)$, the problem (18) reduces to an optimization problem in rough environment having $n$ crisp decision variables, $x_1, x_2, ..., x_n$, and is presented below:

$$\begin{align*}
\text{Maximize} & \quad \hat{f}(x, \bar{y}) \\
\text{Subject to} & \quad \tilde{\phi}_i(x, \bar{y}) \leq 0, i = 1, 2, ..., p
\end{align*}$$

(22)

Clearly here, objective function as well as the constraints are rough in nature. Optimization in rough environment is not properly defined till date. Moreover, there is no proper guideline to check whether the value of a decision vector is feasible with respect to the rough constraints involve in the problem. In problem (22), the value $x_0$ of the decision vector $x$ may said to be feasible if the following conditions hold

$$\text{Tr}(\tilde{\phi}_i(x_0, \bar{y}) \leq 0) > 0.5, \forall i \in \{1, 2, ..., p\}$$

(23)

It is a valid approach as $\text{Tr}(\tilde{\phi}_i(x_0, \bar{y}) \leq 0) + \text{Tr}(\tilde{\phi}_i(x_0, \bar{y}) > 0) = 1$ (§Lemma 7). Similarly, a feasible solution $x'$ is better than another feasible solution $x''$ if

$$\text{Tr}(\hat{f}(x', \bar{y}) > \hat{f}(x'', \bar{y})) > 0.5$$

(24)

If the analytical form of the constraint functions $\tilde{\phi}_i(x, \bar{y})$, $i = 1, 2, ..., p$ is available, then similar as, Lemma 6, one can easily determine the values of $\text{Tr}(\tilde{\phi}_i(x, \bar{y}) \leq 0)$, $i = 1, 2, ..., p$ for the determination of the feasibility of any value of the decision vector, $x$. Similarly if the analytical form of the rough objective $\hat{f}(x, \bar{y})$ is available, then one can easily determine the values of $\text{Tr}(\hat{f}(x', \bar{y}) > \hat{f}(x'', \bar{y}))$ for the determination of the better option between two feasible values, $x'$ and $x''$ of the decision vector, $x$. But if the analytical form of any constraint or the analytical form of the objective function is not available, then rough simulation approach can be used for the validation of any constraint in (23) or (24). It can be done using the following algorithm.

**Algorithm 2.** Algorithm for the determination of $\text{Tr}(\hat{\phi}_i(x, \bar{y}) \leq 0)$ for a particular value of the decision vector $x$. Let $\bar{y}$ be an $m$ dimensional rough vector on the rough space $(\Lambda, \Delta, \kappa, \pi)$, then $\hat{\phi}(x, \bar{y})$ is also a rough vector on that space. To find the trust measure of the rough event $\hat{\phi}(x, \bar{y}) \leq 0$, the following algorithm can be used:

1. Let $n_1 = n_2 = 0$ and set value of $N$, sufficiently large.
2. Generate $\lambda$ uniformly from $\Delta$ according to the measure $\pi$
3. If $\phi(x, y(\lambda)) \leq 0$, then $n_1 = n_1 + 1$.
4. Generate $\bar{\lambda}$ uniformly from $\kappa$ according to the measure $\pi$
5. If $\phi(x, y(\bar{\lambda})) \leq 0$, then $n_2 = n_2 + 1$.
6. Repeat the steps 2-5 for $N$ times.
7. $\text{Tr}(\phi(x, \bar{y}) \leq 0) = (n_1 + n_2)/(2N)$
8. End algorithm

### 6 Ant colony optimization

Following different behaviours of ants for searching their food sources, the ACO algorithm was proposed by Dorigo et al. Dorigo and Di Caro (1999) in 1997 to solve TSPs. In the algorithm, the path of an ant from the nest to the food source is analogous to the path of a salesman in TSP. Ant algorithm is multi-agent system in which the behaviour for each single agent, called artificial ant or ant, follows real ants’s behaviour. Nowadays, a large number of algorithms...
on ant base have been available in the literature. The purpose of the algorithm is to find a minimum distance path from the source to the destination. Analogous to the real ant, in the algorithm, every artificial ant has a chemical called pheromone. When the ant travels from one place to another, it is left on the path. The path in which density of pheromone is maximum is the shortest path in reality. In the algorithm, also, this phenomenon is used to find the shortest path for the salesman. In the algorithm, in the iteration, \( t \), an ant \( k \), which is currently located at node \( i \), selects the next node \( j \), depending on a probability, \( P^k_{ij}(t) \), using some probabilistic selection process, e.g. for Roulette Wheel selection process Kumar (2012):

\[
P^k_{ij}(t) = \begin{cases} \frac{\tau_{ij}(t)^\alpha \eta_{ij}(t)^\beta}{\sum_{u \in N_i^k(t)} \tau_{ui}(t)^\alpha \eta_{ui}(t)^\beta} & \text{if } j \in N_i^k(t) \\ 0 & \text{if } j \notin N_i^k(t). \end{cases}
\]

(25)

where \( \tau_{ij} \) represents the pheromone value and \( \eta_{ij} \) represents the heuristics value of the move from node \( i \) to \( j \) at time step \( t \). \( N_i^k(t) \) represent the set of nodes which are not yet visited by ant \( k \) (when it is at node \( i \)). \( \alpha \) and \( \beta \) are positive real parameters whose values determine the relative importance of the pheromone versus the heuristics information. \( \eta_{ij} \) is calculated by following equation,

\[
\eta_{ij} = \frac{1}{d_{ij}}
\]

(26)

where \( d_{ij} \) is the distance (cost) between the node \( i \) and \( j \).

Pheromone Evaporation. At the time of visiting the nodes by an ant, some amount of pheromone are evaporated from each edge and some pheromone are deposited on the edges through which the ant move. For each edge \((i, j)\), evaporation takes place using the following rule:

\[
\tau_{ij}(t+1) = \rho \tau_{ij}(t)
\]

(27)

with \( \rho \in (0, 1) \), \( \rho \) is the constant, that specifies pheromone evaporate rate.

Pheromone update rule. After completion of a tour (path) from the source to the destination by all the ants, the pheromone on each edge \((i, j)\) through which any ant moves is updated (due to deposition of pheromone) as

\[
\tau_{ij}(t+1) = \tau_{ij}(t) + \sum_{k=1}^{n} \Delta \tau_{ij}^k(t)
\]

(28)

where \( \Delta \tau_{ij}^k(t) \) is the amount of pheromone deposited by the ant \( k \) on the edge \((i, j)\) at time step \( t \) and here \( \Delta \tau_{ij}^k(t) \) is taken as

\[
\Delta \tau_{ij}^k(t) = \begin{cases} \frac{1}{r(X_k)} & \text{if k-th ant passes through the edge } (i,j) \\ 0 & \text{otherwise} \end{cases}
\]

(29)

where \( X_k \) is the path of \( k \)-th ant and \( f(X_k) \) is the distance of the path. For detail algorithm of ACO please see Khanra et al. (2016).

7 K-Opt perturbation operation for CSP

K-Opt Sierksma (2014) is mainly used to improve the tour cost of a CSP. It is also called a tour improvement algorithm. The mechanism of K-Opt operation is to break a feasible tour into K-parts, rejoin different permutations of the parts to create different feasible tours and select the best among them for the improvement of the original tour. Here, K-Opt operation is applied on a complete tour of a CSP for its possible improvement. For detailed of K-Opt operation, one can refer Khan and Maiti (2018). 2-Opt algorithm removes one edge form the tour and reconnects the two sub-tours’ combination and reverses of their path to find the better movement. It is continued until no improvement is found using the process. Similarly in the case of 3-Opt, breaking 2 edges of a tour, eight possible new combinations can be found. In this way, we continue to break edges from tour, i.e. \( K = 1, 2, 3 \ldots, n \) and get new algorithm, like 2-Opt, 3-Opt, 4-Opt and so on. But the increase in \( K \) increases, the time complexity of the algorithm. For that reason, here, the 3-Opt operation is applied and it is found that it acts better than the 2-Opt operation for large size problems.

7.1 K-Opt Operation on a complete tour

The detailed algorithm of K-Opt operation for \( K = 3 \) is presented below. Here, a one-dimensional array bestpath\(_{ij}\) is used to represent \( i \)-th path/tour of the CSP obtained using ACO. Then, two cities of the path are randomly selected to divide the path into three sub-tour and let these be bestpath\(_{ij} j = 1, 2, 3 \). The reverse of these sub-tours are denoted by bestpath\(_{ij}^r j = 1, 2, 3 \), respectively. Then, eight new tours, nbestpath\(_{ij} j = 1, 2, 3 \) and bestpath\(_{ij}^r j = 1, 2, 3 \) as below:

- nbestpath\(_{12} = \text{bestpath}_{12} \rightarrow \text{bestpath}_{12} \rightarrow \text{bestpath}_{12} \)
- nbestpath\(_{23} = \text{bestpath}_{23} \rightarrow \text{bestpath}_{23} \rightarrow \text{bestpath}_{23} \)
- nbestpath\(_{31} = \text{bestpath}_{31} \rightarrow \text{bestpath}_{31} \rightarrow \text{bestpath}_{31} \)
- nbestpath\(_{12}^r = \text{bestpath}_{12}^r \rightarrow \text{bestpath}_{12}^r \rightarrow \text{bestpath}_{12}^r \)
- nbestpath\(_{23}^r = \text{bestpath}_{23}^r \rightarrow \text{bestpath}_{23}^r \rightarrow \text{bestpath}_{23}^r \)
- nbestpath\(_{31}^r = \text{bestpath}_{31}^r \rightarrow \text{bestpath}_{31}^r \rightarrow \text{bestpath}_{31}^r \)
- nbestpath\(_{12} = \text{bestpath}_{12} \rightarrow \text{bestpath}_{12} \rightarrow \text{bestpath}_{12} \)
- nbestpath\(_{23} = \text{bestpath}_{23} \rightarrow \text{bestpath}_{23} \rightarrow \text{bestpath}_{23} \)
- nbestpath\(_{31} = \text{bestpath}_{31} \rightarrow \text{bestpath}_{31} \rightarrow \text{bestpath}_{31} \)

Among these new tours, minimum cost tour is the best found movement using 3-Opt operation once. If cost of this tour is less than the parent tour, then it replaces the parent tour.
8 Proposed algorithm for covering salesman problem

As stated earlier, the proposed algorithm consists of two procedures. The first procedure divides the node set of the problem into groups in such a manner that all the nodes of any group located within the restricted covering distance \( r \) (of the problem) from the centre of the respective group. The second procedure determines the shortest route through the centres of the group. In the algorithm, \( N \) represents number of nodes/cities, \( NB \) represents number of groups, \( NC \) represents maximum number of cities in a group. At first, equal no of nodes are selected in all the groups. The remaining nodes are successively added to different groups depending upon the distance of the node (nearest to the centre element of the group) so that cardinality of a group does not exceed \( NC \). \( NE \) is the equal number of nodes in each group, i.e. \( NE = \lceil N/NB \rceil \) represents an integral part of \( N/NB \). \( G[i] \) represents \( i \)-th group, \( l \) is the cardinality of the group, i.e. \( G[i],l \) is the length of group \( i \), an array \( G[i].S \) represents the node set of group \( i \) and \( max\_iteration \) is the maximum number of iterations and \( N \) is the no of nodes. \( (d_{ij})_{N \times N} \) represents the distance matrix of the problem.

1. Start Algorithm
2. Set \( NC, \ NB, \ max\_iteration, \ max\_iteration2, N \).
3. Input \( (d_{ij})_{N \times N} \)
4. For \( i = 1 \) to \( max\_iteration \) do
5. Set \( G[1].l = 0 \) for \( k = 1, 2, ..., NB \)
6. Set \( NS = \{V_1, V_2, ..., V_N\} \)
7. Set \( G[1].S = \{\} \)
8. \( v_c = \) a randomly selected element from \( NS \)
9. \( G[1].S = G[1].S \cup \{v_c\} \)
10. \( NS = NS - \{v_c\} \)
11. \( G[1].l = G[1].l + 1 \)
12. End For
13. \( v = \) Nearest city of \( v_c \) in \( NS \)
14. \( G[1].S = G[1].S \cup \{v\} \)
15. \( G[1].l = G[1].l + 1 \)
16. \( NS = NS - \{v\} \)
17. End For
18. For \( k = 2 \) to \( NB \) do
19. \( v_c = \) Nearest city of \( v_c \) in \( NS \)
20. \( G[k].S = G[k].S \cup \{v_c\} \)
21. \( NS = NS - \{v_c\} \)
22. \( G[k].l = G[k].l + 1 \)
23. For \( j = 2 \) to \( NE \) do
24. \( v = \) Nearest city of \( v_c \) in \( NS \)
25. \( G[k].S = G[k].S \cup \{v\} \)
26. \( G[k].l = G[k].l + 1 \)
27. \( NS = NS - \{v\} \)
28. End For
29. End For
30. set \( count = 0 \)
31. While \( NS \neq \{} \) do
32. For \( k = 1 \) to \( NB \) do
33. If \( G[k].l \leq NC \) then
34. \( count = count + 1 \)
35. \( b_{count} = k \) where \( b \) is an array
36. End if
37. End For
38. \( v = \) first element from \( NS \)
39. \( NS = NS - \{v\} \)
40. \( min = d_{G[b_1].S[G[b_1].l]} \)
41. \( pos = b_1 \)
42. For \( k = 2 \) to \( count \) do
43. If \( (d_{G[b_1].S[G[b_1].l]} < min) \) then
44. \( min = d_{G[b_1].S[G[b_1].l]} \)
45. \( pos = b_k \)
46. End If
47. End For
48. \( G[pos].S = G[pos].S \cup \{v\} \)
49. \( G[pos].l = G[pos].l + 1 \)
50. End While
51. For \( k = 1 \) to \( max\_iteration2 \) do
52. set \( flag = 0 \)
53. For \( j = 1 \) to \( NB \) do
54. \( z = \) a randomly selected integer in the range \([1, G[j].l] \)
55. \( G[j].center = G[j].S[z] \)
56. For \( k = 1 \) to \( G[j].l \) do
57. If \( (d_{G[j].S[G[j].center]} > r) \) then
58. \( flag = 1 \)
59. break
60. End If
61. End For
62. If \( flag = 1 \) then
63. break
64. End If
65. End For
66. If \( (flag = 0) \) then
67. ACO algorithm is used to find the best path according the centres of the groups.
68. \( K\)-opt operation is applied on the best path for further possible improvement.
69. The best solution found so far, \( solution_b \), is updated.
70. End If
71. End for
72. End for
73. Output \( solution_b \)
74. End algorithm
8.1 Exploration and exploitation

For any CSP instance, a particular division of the groups of cities and the selection of visiting nodes may converge to local optima. To explore different possible paths, the ACO algorithm is applied on the different selections of the set of visiting nodes. In this study, K-opt operation is applied at the end of the ACO algorithm to avoid convergence of the path at any local optima. If the ACO converges to a local optima, then the application of 3-opt on the obtained path of ACO will find a better path and the repeated application of 3-opt on the improved paths may obtain the global optimal path. In this way, the exploration and exploitation is made in the proposed algorithm.

8.2 Implementation and testing

The algorithm is implemented in Dev C++ in a computer having Intel core-i3 first generation processor and 2 GB RAM. The algorithm is tested against a set of benchmark test problems from TSPLIB with significantly large sizes (size up to 654 nodes). The performance of the algorithm is compared with different existing successful algorithms for CSP in the literature. It is observed that the efficiency of the proposed algorithm is comparatively better with respect to the other existing algorithms used for the comparison.

9 Numerical illustration

So far the author’s knowledge go, in the literature, there is no algorithm that presents results of benchmark CSP instances with specified coverage range \( r \). All the studies have been made where groups are created with the nearest nodes from the selected centres of the groups. Due to this reason, here, to measure the efficiency of the proposed algorithm, some standard benchmark test instances are used that are proposed by Golden et al. Golden et al. (2012). These instances are generated form TSPLIB Gerhard (1999). The test problems have been divided into small, medium, and large size problems according to their sizes. The small size and medium size problems contain 51 to 200 number of nodes where each node can cover nearest 7, 9, 11 number of nodes and the large size problems contain 532 to 654 number of nodes where each node can cover 3, 5, 7 number of nearest nodes. Other notations and symbols are the same as previously stated.

The problems are solved using the proposed algorithm for CSPs, and the results are tabulated in Tables 1 and 2. The goal is to minimize the total tour cost of the Hamiltonian path through the visiting nodes. It is observed from the tables that in most of the considered instances tour cost increase with the number of groups, which agrees with reality. In very few cases, the algorithm gives higher tour cost for larger group size, due to the division of the groups. But in those cases also the proposed algorithm gives better results compared to the other well-established algorithms used for the comparison. The convergence graphs of the best found solutions for some instances are presented in Fig. 1.

Tables 3 and 4 present the comparative study of the computational results of the proposed algorithm with respect to seven other state-of-the-art algorithms for the CSPs. Table 3 represents the best found cost of the different instances obtained by the different well-established algorithms for CSPs in the literature along with the proposed algorithm. Table 4 represents the average cost and the standard deviation of the results obtained by the different algorithms in five different runs for the different instances. It is observed from the tables that the proposed algorithm obtains the best tour cost as well as the minimum average cost for each of the instances with different numbers of NCs. From these observations, it can be concluded that the efficiency of the proposed algorithm is better compared to the other algorithms in the literature for solving the CSPs.

Table 5 present the results due to the different values of the coverage range \( r \) for some large size test instances. It is observed from the tables that for each of the instances, the tour cost increases with the decrease of \( r \). In Table 6 also, the results due to the different values of the coverage range \( r \) are presented for all the considered test instances. But, here, at first result due to a fixed coverage range, \( r \), is obtained, which is presented in the column marking 100%. Then, results are obtained by decreasing the coverage range by taking its value 80%, 70%, 60% and 50% of \( r \). In some cases, the algorithm could not find any result. In those cases, result positions in the table are kept BLANK. It is observed from Table 6 that the tour costs increases gradually as the covering range decreases gradually. Clearly, this observation also agrees with reality.

Again results are obtained for the restricted problems where the Hamiltonian path of a CSP is searched under the restriction of a specific coverage range of different groups, i.e. total distance of different cities of a group from the visiting city should not exceed a predefined level. For this study, the group coverage range is considered as \( NC \ast r/3 \) for any problem, where the symbols are previously defined and the value of \( r \) is mentioned in the table. Results are obtained for different problems and are presented in Table 7. If, for any case, solution does not exists, then the solution position of such problems are kept BLANK in the table.

As this is the initial study on imprecise CSP, no standard test problems on fuzzy CSP exist in the literature. Due to this reason, test problems with fuzzy cost parameters are generated from the previously considered standard benchmark test problems of CSPs with crisp parameters. To test the proposed algorithm in a fuzzy environment, these fuzzy CSPs are used. In a crisp CSP, if travel cost between node \( i \) and node \( j \) is \( d_{ij} \), then following Khan et al. Khan et al.
Table 1 The computational results of the proposed algorithm

| Instance | NC | Best found tour cost |
|----------|----|----------------------|
| Berlin52 | 9  | 1112                 |
|          | NB | 5                    |
|          | 7  | 1029                 |
|          | NB | 10                   |
| st70     | 9  | 140                  |
|          | NB | 7                    |
|          | 11 | 108                  |
|          | NB | 11                   |
| eil76    | 9  | 90                   |
|          | NB | 7                    |
|          | 11 | 62                   |
|          | NB | 15                   |
| rat99    | 9  | 201                  |
|          | NB | 10                   |
|          | 11 | 269                  |
|          | NB | 15                   |
| kroA100  | 9  | 6641                 |
|          | NB | 10                   |
|          | 11 | 5651                 |
|          | NB | 15                   |
| kroB100  | 9  | 6816                 |
|          | NB | 10                   |
|          | 11 | 6135                 |
|          | NB | 15                   |
| kroC100  | 9  | 6349                 |
|          | NB | 10                   |
|          | 11 | 5851                 |
|          | NB | 15                   |
| kroD100  | 9  | 6595                 |
|          | NB | 10                   |
|          | 11 | 5575                 |


Table 2: The computational results of the proposed algorithm.

| Instance | NC  | Best found tour cost |
|----------|-----|----------------------|
| NB       | 15  | 8208 7862 8429 8699 9339 |
| NB       | 12  | 13 14 15 16 17 |
| NB       | 10  | 11 12 13 14 15 |
| NB       | 22  | 23 24 25 26 27 |
| NB       | 17  | 23 24 25 26 27 |
| NB       | 17  | 9568 9512 10044 9812 10262 |
| NB       | 17  | 18 19 20 21 22 |
| NB       | 15  | 18 19 20 21 22 |
| NB       | 29  | 30 33 34 35 36 |
| NB       | 23  | 25 26 27 28 29 |
| NB       | 19  | 26 27 28 29 30 |
| NB       | 19  | 26 27 28 29 30 |
| NB       | 178 | 179 180 181 182 183 |
| NB       | 34109 | 34750 35565 36559 43955 |
| NB       | 107 | 116 117 119 120 121 |
| NB       | 15  | 16 17 18 19 20 |
| NB       | 76  | 95 96 97 98 99 |
| NB       | 179 | 182 183 184 185 186 |
| NB       | 712 | 743 743 731 747 738 |
| NB       | 107 | 115 116 121 122 130 |
| NB       | 172 | 792 802 895 908 1103 |
| NB       | 77  | 91 92 94 95 105 |
| NB       | 664 | 698 716 738 779 1026 |
| NB       | 192 | 193 194 195 196 197 |
| NB       | 18311 | 18253 18558 18749 18840 18640 |
| NB       | 115 | 127 128 129 130 131 |
| u574     | 17960 | 18118 18332 18711 18840 18547 |
| NB       | 99  | 100 101 102 103 104 |
| u574     | 15598 | 15915 16062 16104 16296 17114 |
| NB       | 220 | 221 235 236 246 247 |
| 3        | 19921 | 19975 20145 20768 21173 21298 |
| NB       | 141 | 142 143 180 181 191 |
| p654     | 20250 | 20267 20302 20893 20872 21097 |
| NB       | 113 | 136 137 138 139 140 |
| 7        | 19381 | 20145 19883 20147 20221 20264 |
the corresponding fuzzy cost is generated as \( \tilde{d}_{ij} = (d_{ij1}, d_{ij2}, d_{ij3}) \), where \( d_{ij2} = d_{ij1} = d_{ij} - R1 \), \( d_{ij3} = d_{ij} + R2 \), where \( R1 \) and \( R2 \) are randomly generated between \( (0, R \times d_{ij}/100) \). Here, \( R \) is the percentage of fuzziness and its value is considered as 0.5, i.e. here, maximum 0.5% fuzziness is considered for the fuzzy costs. Table 8 presents the computational results of these problems using the proposed algorithm in fuzzy environment. As cost matrices are generated with 0.5% fuzziness the tour costs of the best found paths are near to the corresponding crisp problems and it implies that the proposed algorithm is efficient enough to solve CSPs with fuzzy cost matrices. Results are obtained following both the direct credibility measure approach and the simulation approach for the same, and the same results are obtained. As a result, only one result table is presented, as, the same results are obtained using two approaches.

Due to the same reason as the fuzzy CSPs, there are no standard test problems on rough CSPs in the literature. Due to this reason, test problems with rough cost parameters are generated from the previously considered standard benchmark test problems of CSPs with crisp parameters. To test the proposed algorithm in a rough environment, these generated CSPs with rough cost parameters are used. If in a crisp CSP, travel cost between node \( i \) and node \( j \) is \( d_{ij} \), then following Khan et al. Khan et al. (2019), the corresponding cost in the rough environment is generated as \( \tilde{d}_{ij} = (d_{ij1}, d_{ij2}, d_{ij3}, d_{ij4}) \), where \( d_{ij1} = d_{ij}, d_{ij2} = d_{ij} + R1, d_{ij3} = d_{ij} - R2, d_{ij3} = d_{ij} + R3 \) and \( R1, R2, R3 \) are randomly generated in \( (0, R \times d_{ij}/100) \). Here, \( R \) is the percentage of roughness and its value is considered as 0.5, i.e. here maximum 0.5% roughness is considered for the rough costs. Table 9 presents the computational results of these problems using the proposed algorithm in rough environment. As cost matrices are generated with 0.5% roughness the tour costs of the best found paths are near to the corresponding crisp problems and it implies that the proposed algorithm is efficient enough to solve CSPs with rough cost matrices. Results are obtained following both the direct trust measure approach and the simulation approach, and the same results are obtained. As a result, only one result table is presented, as, the same results are obtained using two approaches.

10 Conclusion

Here, for the first time, the ACO algorithm is modified with \( K \)-opt operation to develop an efficient and consistent algorithm for CSPs under one restriction in crisp and imprecise (fuzzy, rough) environments. An algorithm is proposed for the division of groups of the cities depending upon the maximum number of cities in a group and the total number of groups. The ACO is used to find the shortest/minimum-cost path of any problem by selecting only one city from each group in such a manner that all the unvisited cities are within a predefined coverage range \( r \). \( K \)-opt algorithm is used at the end of ACO operation for the possible improvement of the quality of the solution. Here, study is made only in two imprecise environments—fuzzy environment and rough.
Table 3. Performance study of the proposed algorithm with respect to different algorithms for CSP

| Instance | NC | CPLEX [1] | LS2 Golden et al. (2012) | ILP Salari and Naji-Azimi (2012) | Hybrid ACO Salari et al. (2015) | Hybrid GA Pandiri et al. (2020) | Hybrid ABC Pandiri et al. (2020) | PVNS Zang et al. (2020) | Hybrid ABC Venkatesh et al. (2019) | Proposed Algorithm |
|----------|----|-----------|--------------------------|---------------------------------|--------------------------------|--------------------------------|--------------------------------|---------------------|----------------------------------|--------------------|
| berlin52 | 7  | 4019      | 3878                     | 3878                            | 3887                           | 3887                           | 3887                           | 3887                | 3887                             | 1459               |
|          | 9  | 3430      | 3430                     | 3430                            | 3430                           | 3430                           | 3430                           | 3430                | 3430                             | 1112               |
| st70     | 7  | 297       | 288                      | 288                             | 288                            | 288                            | 288                            | 288                 | 288                              | 176                |
|          | 9  | 271       | 259                      | 259                             | 259                            | 259                            | 259                            | 259                 | 259                              | 140                |
| eil76    | 9  | 198       | 186                      | 185                             | 185                            | 186                            | 186                            | 186                 | 186                              | 90                 |
|          | 11 | 177       | 170                      | 170                             | 170                            | 170                            | 170                            | 170                 | 170                              | 61                 |
| rat99    | 7  | 572       | 486                      | 486                             | 486                            | 486                            | 486                            | 486                 | 486                              | 382                |
|          | 11 | 456       | 444                      | 444                             | 444                            | 444                            | 444                            | 444                 | 444                              | 269                |
| kroA100  | 7  | 10306     | 9674                     | 9674                            | 9674                           | 9674                           | 9674                           | 9674                | 9674                             | 7365               |
|          | 9  | 9573      | 9159                     | 9159                            | 9159                           | 9159                           | 9159                           | 9159                | 9159                             | 6641               |
| kroB100  | 7  | 10367     | 9723                     | 9723                            | 9723                           | 9723                           | 9723                           | 9723                | 9723                             | 7272               |
|          | 11 | 9150      | 8632                     | 8632                            | 8632                           | 8632                           | 8632                           | 8632                | 8632                             | 5821               |
|         | 7  | 11085     | 9626                     | 9626                            | 9626                           | 9626                           | 9626                           | 9626                | 9626                             | 6866               |
| Instance | NC | LS2 Golden et al. (2012) | ILP Salari and Naji-Azimi (2012) | Hybrid ACO Salari et al. (2015) | Hybrid GA Pandiri et al. (2020) | Hybrid ABC Pandiri et al. (2020) | PVNS Zang et al. (2020) | Hybrid ABC Venkatesh et al. (2019) | Proposed Algorithm |
|----------|----|--------------------------|----------------------------------|---------------------------------|---------------------------------|-------------------------------|---------------------|-------------------------------|-----------------|
| kroD100  | 9  | 10564                    | 8885                             | 8885                            | 8885                            | 8885                           | 8885                | 8885                           | 6595            |
|          | 11 | 9175                     | 8725                             | 8725                            | 8725                            | 8725                           | 8725                | 8725                           | 5575            |
|          | 7  | 9095                     | 10150                            | 10150                           | 10150                           | 10150                          | 10150              | 10150                          | 7407            |
| kroE100  | 9  | 9095                     | 8991                             | 8991                            | 8991                            | 8992                           | 8992                | 8992                           | 6709            |
|          | 11 | 8936                     | 8450                             | 8450                            | 8450                            | 8450                           | 8450                | 8450                           | 6287            |
|          | 7  | 4105                     | 3461                             | 3461                            | 3461                            | 3461                           | 3461                | 3461                           | 2424            |
| rd100    | 9  | 3414                     | 3194                             | 3194                            | 3194                            | 3194                           | 3194                | 3194                           | 2171            |
|          | 11 | 3453                     | 2922                             | 2922                            | 2922                            | 2922                           | 2922                | 2922                           | 1933            |
|          | 7  | 12367                    | 11423                            | 11423                           | 11423                           | 11423                          | 11423              | 11423                          | 9220            |
| kroA150  | 9  | 11955                    | 10056                            | 10056                           | 10056                           | 10056                          | 10056              | 10056                          | 8106            |
|          | 11 | 10564                    | 9439                             | 9439                            | 9439                            | 9439                           | 9439                | 9439                           | 7408            |
|          | 7  | 14667                    | 13285                            | 13285                           | 13286                           | 13285                          | 13285              | 13285                          | 10783           |
| kroA200  | 9  | 12683                    | 11708                            | 11708                           | 11708                           | 11708                          | 11708              | 11708                          | 8994            |
|          | 11 | 12736                    | 10748                            | 10748                           | 10748                           | 10748                          | 10748              | 10748                          | 8432            |
| att532   | 5  | 52399                    | 52412                            | 51616                           | –                               | –                              | 51457              | –                             | 50684           |
|          | 7  | 42634                    | 42387                            | 42212                           | –                               | –                              | 42148              | –                             | 40950           |
| ali535   | 5  | 38186                    | 38016                            | 37741                           | 37608                           | 37506                          | 37621              | –                             | 35928           |
|          | 7  | 38186                    | 38016                            | 37741                           | 37608                           | 37506                          | 37621              | –                             | 1034            |
| p654     | 5  | 25158                    | 25155                            | 25166                           | 25186                           | 25119                          | 25181              | –                             | 23537           |
|          | 7  | 23226                    | 23211                            | 23242                           | 23285                           | 23205                          | 23224              | –                             | 20669           |
|          | 7  | 22121                    | 22126                            | 22125                           | 22121                           | 22118                          | 22135              | –                             | 20444           |
Table 4  Performance study with respect to the average cost and the standard deviation of the results in 5 runs

|                         | LS2 Golden et al. (2012) | ILP Salari and Naji-Azimi (2012) | Hybrid ACO Salari et al. (2015) | Hybrid GA Pandiri et al. (2020) | Hybrid ABC Pandiri et al. (2020) | PVNS Zang et al. (2020) | Proposed Algorithm |
|-------------------------|--------------------------|----------------------------------|----------------------------------|----------------------------------|----------------------------------|-----------------------|-------------------|
| Berlin52                | 3887 0.0                 | 3887 0.0                         | 3887.4 0.01                      | 3887 0.0                         | 3887.0 0.0                      | 3887 0.0             | 1459 0.0          |
| st70                    | 259 0.0                  | 259 0.0                          | 259 0.0                          | 259 0.0                          | 259 0.0                          | 259 0.0               | 140 0.0           |
| eil76                   | 186 0.54                 | 186 0.54                         | 186 0.54                         | 186 0.54                         | 186 0.54                         | 186 0.54              | 90 0.0            |
| rat99                   | 455 0.0                  | 455 0.0                          | 455 0.0                          | 455 0.0                          | 455 0.0                          | 455 0.0               | 201 0.0           |
| kroA100                 | 9159 0.0                 | 9159 0.0                         | 9159 0.0                         | 9159 0.0                         | 9159 0.0                         | 9159 0.0              | 6236.2 0.5        |
| kroB100                 | 9240 0.0                 | 9240 0.0                         | 9240 0.0                         | 9240 0.0                         | 9240 0.0                         | 9240 0.0              | 6693.2 0.6        |
| kroC100                 | 9171 0.0                 | 9171 0.0                         | 9171 0.0                         | 9171 0.0                         | 9171 0.0                         | 9171 0.0              | 6349.2 0.6        |
| kroE100                 | 8991 0.0                 | 8991 0.0                         | 8991 0.0                         | 8991 0.0                         | 8991 0.0                         | 8991 0.0              | 6709.3 0.8        |
| kroA150                 | 10062.4 0.06            | 10057.6 0.02                     | 10056 0.02                       | 10057 0.02                       | 10056 0.02                       | 10056 0.02            | 8106.2 0.5        |
| ali535                  | 3 25206.0 .32           | 25206.0 0.31                     | 25182.8 0.22                     | 25224.2 0.39                     | 25133.4 0.03                     | 25219.4 0.26         | 19921.75 2.54     |
| p654                    | 23258.4 0.23            | 23224.8 0.09                     | 23289.4 0.36                     | 23291.0 0.37                     | 23215.6 0.05                     | 23278.4 0.29         | 20250.625 2.24    |
|                         |                         |                                  |                                  |                                  |                                  |                       | 19381.75 2.54     |
| Instance | NC | Best tour cost found for different NB and coverage range $r$ |
|----------|----|--------------------------------------------------------|
|          |    | NB  8  9  10  11  12 |
|          |    | $r$  500 600 700 500 600 700 500 600 700 500 600 700 500 600 700 |
| 7 Result |    | 2607 2198 1844 2782 2115 1913 3089 2867 2518 3166 2342 2102 3368 2698 2382 |
|          |    | NB  8  9  10  11  12 |
|          |    | $r$  500 600 700 500 600 700 500 600 700 500 600 700 500 600 700 |
| berlin52 | 9 Result | 2560 2040 2008 2692 2328 1985 3089 2867 2518 2977 2445 2129 3152 2636 2426 |
|          |    | NB  7  8  9 |
|          |    | $r$  500 600 700 500 600 700 500 600 700 500 600 700 500 600 700 |
| 11 Result |    | 2547 2094 1808 2597 2040 1993 2692 2316 1985 3089 2867 2518 2977 2474 2089 |
|          |    | NB  10  11  12  |
|          |    | $r$  50 90 100 50 90 100 50 90 100 50 90 100 50 90 100 |
| st70     | 9 Result | 169 143 142 187 157 167 205 185 182 202 193 190 214 205 202 |
|          |    | NB  7  8  9 |
|          |    | $r$  50 90 100 50 90 100 50 90 100 50 90 100 50 90 100 |
| 11 Result |    | 162 131 123 153 143 150 178 161 157 202 185 182 202 193 190 |
|          |    | NB  11  12  |
|          |    | $r$  30 40 50 30 40 50 30 40 50 30 40 50 30 40 50 |
| 7 Result |    | 156 130 123 132 129 132 147 143 155 142 142 142 156 157 157 |
|          |    | NB  9  10  11  |
|          |    | $r$  30 40 50 30 40 50 30 40 50 30 40 50 30 40 50 |
| eil76    | 9 Result | 117 93 90 122 102 98 122 120 118 132 132 128 145 134 134 |
|          |    | NB  7  8  9 |
|          |    | $r$  30 40 50 30 40 50 30 40 50 30 40 50 30 40 50 |
| 11 Result |    | 117 88 68 112 83 78 107 97 85 118 102 93 126 115 111 |
|          |    | NB  12  13  |
|          |    | $r$  1000 1100 1200 1000 1100 1200 1000 1100 1200 1000 1100 1200 1000 1100 1200 |
| 7 Result |    | 8054 8202 8056 8840 8766 8639 9408 8917 8917 9339 9189 8903 10594 10131 9532 |
|          |    | NB  12  13  |
|          |    | $r$  1000 1100 1200 1000 1100 1200 1000 1100 1200 1000 1100 1200 1000 1100 1200 |
| kroA100  | 9 Result | 8859 7903 7748 8289 8174 7857 8401 8165 8077 8130 8018 7966 8241 8349 8201 |
|          |    | NB  10  11  12  |
|          |    | $r$  1000 1100 1200 1000 1100 1200 1000 1100 1200 1000 1100 1200 1000 1100 1200 |
| 11 Result |    | 7827 7552 7552 7785 7646 7170 7814 7801 7631 8220 7753 7770 8401 8165 8077 |
Table 5 continued

| Instance | NC | Best tour cost found for different NB and coverage range r |
|----------|----|----------------------------------------------------------|
|          | NB | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
|          | r  | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 |
| 7        | Result | 8285 | 8090 | 8090 | 8647 | 8671 | 8344 | 8690 | 8690 | 8250 | 9079 | 8771 | 8771 | 9585 | 9175 | 9071 |
|          | NB | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 7        | r  | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 |
| kroC100  | 9   | Result | 7415 | 7510 | 7415 | 7794 | 7747 | 7173 | 7954 | 7927 | 7801 | 7947 | 7553 | 753 | 8360 | 8050 | 7846 |
|          | NB | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 11       | r  | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 |
| kroD100  | 9   | Result | 8422 | 8341 | 8169 | 9130 | 8431 | 9717 | 9083 | 8828 | 9426 | 8559 | 8959 | 9754 | 9346 | 9251 |
|          | NB | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 7        | r  | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 |
| kroE100  | 9   | Result | 8263 | 7784 | 7093 | 7919 | 7911 | 7441 | 8018 | 8033 | 7783 | 8587 | 7917 | 7817 | 8202 | 8202 | 8202 |
|          | NB | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 11       | r  | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 |
| kroA150  | 9   | Result | 9288 | 9013 | 8766 | 9850 | 9514 | 9163 | 9775 | 9312 | 9054 | 9943 | 9802 | 9643 | 10378 | 9282 | 9282 |
|          | NB | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 11       | r  | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 | 1000 | 1100 | 1200 |
| Instance | NC | Best tour cost found for different NB and coverage range $r$ |
|----------|----|----------------------------------------------------------|
|          | NB | 22          | 23          | 24          | 25          | 26          |
|          | $r$ | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 |
|          | Result | 9776 9631 9631 | 9661 9661 9661 | 9781 9710 9710 | 10381 10347 10318 | 10361 10195 9992 |
|          | NB | 18          | 19          | 20          | 21          | 22          |
|          | $r$ | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 |
| kroB150  | Result | 9019 8891 8203 | 9830 9218 9146 | 9259 9143 9143 | 9499 9387 9387 | 9528 9304 9132 |
|          | NB | 15          | 16          | 17          | 18          | 19          |
|          | $r$ | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 |
|          | Result | 8577 8335 8217 | 8414 8400 8345 | 8768 8547 8199 | 8975 8660 8628 | 9204 8760 8685 |
|          | NB | 29          | 30          | 31          | 32          | 33          |
|          | $r$ | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 |
|          | Result | 10335 10335 10223 | 11630 11314 11198 | 11341 11203 11133 | 11103 11103 11103 | 11868 11548 11480 |
|          | NB | 23          | 24          | 25          | 26          | 27          |
|          | $r$ | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 |
| kroA200  | Result | 10804 10407 10064 | 10104 9390 9390 | 10216 9655 10188 | 10387 9814 9773 | 10074 10074 10074 |
|          | NB | 19          | 20          | 21          | 22          | 23          |
|          | $r$ | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 | 1000 1100 1200 |
|          | Result | 9253 8783 8507 | 9311 8988 8988 | 9413 9233 9039 | 9581 9581 9268 | 9396 9396 9288 |
|          | NB | 77          | 78          | 79          | 80          | 81          |
|          | $r$ | 150 160 170 | 150 160 170 | 150 160 170 | 150 160 170 | 150 160 170 |
|          | Result | 907 743 743 | 817 806 806 | 875 805 780 | 781 781 735 | 843 814 814 |
|          | NB | 60          | 61          | 62          | 63          | 64          |
|          | $r$ | 150 160 170 | 150 160 170 | 150 160 170 | 150 160 170 | 150 160 170 |
| ali535   | Result | 733 733 704 | 746 705 660 | 796 736 727 | 736 736 726 | 734 734 729 |
|          | NB | 50          | 51          | 52          | 53          | 54          |
|          | $r$ | 150 160 170 | 150 160 170 | 150 160 170 | 150 160 170 | 150 160 170 |
|          | Result | 648 615 615 | 663 652 652 | 681 681 681 | 739 720 710 | 635 617 575 |
|          | NB | 192         | 193         | 194         | 195         | 196         |
|          | $r$ | 1500 1600 1700 | 1500 1600 1700 | 1400 1500 1700 | 1400 1500 1600 | 1500 1600 1800 |
|          | Result | 22583 22514 22123 | 22250 22250 22250 | 22600 22600 22573 | 22711 21977 21977 | 22977 22568 22293 |
|          | NB | 115         | 116         | 117         | 118         | 119         |
|          | $r$ | 1400 1600 1800 | 1500 1700 1800 | 1500 1600 1700 | 1300 1400 1500 | 1300 1400 1500 |
| u574     | Result | 18077 17750 17631 | 17237 17237 17081 | 16655 16655 16655 | 17933 17304 17304 | 18101 17182 17182 |
|          | NB | 82          | 83          | 84          | 85          | 86          |
|          | $r$ | 1400 1500 1600 | 1500 1600 1700 | 1400 1500 1600 | 1400 1500 1600 | 1400 1500 1600 |
|          | Result | 14899 14899 14773 | 14385 13340 13430 | 14938 14712 14712 | 14840 14629 14629 | 15329 14792 14774 |
Table 6 The computational results due to different coverage range ($r$) for restricted covering problem.

| Instance | NC | Best tour cost found for different NB |
|----------|----|-------------------------------------|
| Berlin52 | 9  | 2581 1829 70% 80% 100% 1172 – – 1742 1649 1112 1703 1644 1443 953 2127 2037 1813 1487 1122 |
| NB       | 8  | 2576 2088 1969 1922 1528 2895 2985 2244 2157 2895 2552 2513 2244 2228 2157 1455 |
| Size     | 50%| 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% |
| 11 Result| –  | – – – 1929 1029 – 1703 1644 1443 953 2127 2037 1813 1487 1122 |
| NB       | 7  | 231 203 201 201 172 231 214 200 200 185 – 271 249 233 205 |
| Size     | 50%| 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% |
| 9 Result | –  | – – – 1929 1029 – 1703 1644 1443 953 2127 2037 1813 1487 1122 |
| NB       | 10 | 276 2088 1969 1922 1528 2895 2985 2244 2157 2895 2552 2513 2244 2228 2157 |
| Size     | 50%| 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% |
| 11 Result| –  | – – – 1929 1029 – 1703 1644 1443 953 2127 2037 1813 1487 1122 |
| NB       | 8  | 2576 2088 1969 1922 1528 2895 2985 2244 2157 2895 2552 2513 2244 2228 2157 |
| Size     | 50%| 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% |
| st70     | 9  | 221 176 161 147 140 222 213 162 158 231 203 201 176 |
| NB       | 11 | 233 198 200 200 181 267 246 236 229 267 246 236 229 222 |
| Size     | 50%| 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% |
| 11 Result| –  | – – – 1929 1029 – 1703 1644 1443 953 2127 2037 1813 1487 1122 |
| NB       | 10 | 276 2088 1969 1922 1528 2895 2985 2244 2157 2895 2552 2513 2244 2228 2157 |
| Size     | 50%| 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% 50% 60% 70% 80% 100% |
| 12 Result| –  | – – – 1929 1029 – 1703 1644 1443 953 2127 2037 1813 1487 1122 |
| Instance | NC | Best tour cost found for different NB |
|----------|----|--------------------------------------|
|          |    | NB 1 (r = 60)                        |
|          |    | Size 50% 60% 70% 80% 100%            |
|          |    | Result 158 150 132 115               |
|          |    | NB 2 (r = 45)                        |
|          |    | Size 50% 60% 70% 80% 100%            |
|          |    | Result 179 154 148 147 148           |
|          |    | NB 3 (r = 25)                        |
|          |    | Size 50% 60% 70% 80% 100%            |
|          |    | Result 76 90 62 125 125             |
|          |    | NB 4 (r = 35)                        |
|          |    | Size 50% 60% 70% 80% 100%            |
|          |    | Result 930 926 922 878 874           |
|          |    | NB 5 (r = 34)                        |
|          |    | Size 50% 60% 70% 80% 100%            |
|          |    | Result 8804 8792 8734 8660 8347       |
|          |    | NB 6 (r = 25)                        |
|          |    | Size 50% 60% 70% 80% 100%            |
|          |    | Result 6508 6156 6156 6244 5651       |
|          |    | NB 7 (r = 20)                        |
|          |    | Size 50% 60% 70% 80% 100%            |
|          |    | Result 7799 7431 7182 7182 6293       |
| Instance | NC | Best tour cost found for different NB |
|----------|----|---------------------------------------|
| NB 7     | 15 \( (r = 2233) \) | 16 \( (r = 2423) \) | 17 \( (r = 2532) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 8664 | 8343 | 8051 | 7538 | 8343 | 8534 | 8585 | 8365 | 8341 | 9158 | 8227 | 8177 | 7822 | 8180 |
| NB       | 18 \( (r = 2216) \) |            | 19 \( (r = 1244) \) |            | 20 \( (r = 1519) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 9003 | 8822 | 8773 | 8659 | 8659 | –   | 11177 | 9898 | 9227 | 9227 | 10846 | 10014 | 9684 | 9544 | 9147 |
| NB       | 12 \( (r = 1191) \) |            | 13 \( (r = 1468) \) |            | 14 \( (r = 1842) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 7004 | 7009 | 6839 | 6729 | 6729 | –   | 9968 | 8203 | 7423 | 6913 | 9158 | 8227 | 8177 | 7822 | 8180 |
| NB       | 15 \( (r = 2240) \) |            | 16 \( (r = 1961) \) |            | 17 \( (r = 2457) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 8308 | 8106 | 8073 | 7966 | 7966 | 9035 | 8621 | 8248 | 8284 | 8284 | 8572 | 8168 | 7807 | 7807 | 7366 |
| NB       | 10 \( (r = 2212) \) |            | 11 \( (r = 1747) \) |            | 12 \( (r = 2281) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 6642 | 6194 | 6251 | 6247 | 6135 | 8108 | 7015 | 6972 | 6872 | 6349 | 6795 | 6849 | 6361 | 6361 | 6361 |
| NB       | 13 \( (r = 2556) \) |            | 14 \( (r = 1842) \) |            | 15 \( (r = 2192) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 7322 | 7215 | 7175 | 7167 | 7167 | 8832 | 7927 | 8066 | 7614 | 7597 | 8214 | 7847 | 7847 | 7797 | 7366 |
| NB       | 15 \( (r = 2835) \) |            | 16 \( (r = 2003) \) |            | 17 \( (r = 2374) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 8137 | 7704 | 7428 | 7382 | 7272 | 8999 | 7854 | 8060 | 7952 | 7543 | 8674 | 8513 | 8384 | 8279 | 7864 |
| NB       | 18 \( (r = 2359) \) |            | 19 \( (r = 1635) \) |            | 20 \( (r = 1757) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 8484 | 8443 | 8443 | 8242 | 8137 | 9912 | 9332 | 8708 | 8708 | 8664 | 10351 | 9891 | 9486 | 9342 | 8795 |
| NB       | 12 \( (r = 2535) \) |            | 13 \( (r = 1478) \) |            | 14 \( (r = 1902) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 7545 | 7205 | 7155 | 6599 | 6349 | –   | 8195 | 7921 | 7384 | 6825 | 8219 | 7819 | 7146 | 7325 | 7317 |
| NB       | 15 \( (r = 1802) \) |            | 16 \( (r = 1316) \) |            | 17 \( (r = 3027) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 8372 | 7952 | 7617 | 7501 | 7206 | –   | 9343 | 8607 | 8342 | 7612 | 8058 | 7981 | 7708 | 7708 | 7708 |
| NB       | 10 \( (r = 1907) \) |            | 11 \( (r = 2417) \) |            | 12 \( (r = 2662) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 9095 | 8267 | 7934 | 7219 | 5851 | –   | 8621 | 8316 | 6422 | –   | 9027 | 8969 | 8626 | 6515 | 7366 |
| NB       | 13 \( (r = 1246) \) |            | 14 \( (r = 2998) \) |            | 15 \( (r = 3440) \) |
|          | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | –   | –   | 10777 | 6705 | –   | –   | 9867 | 9867 | 7884 | 7399 | 7392 | 7392 | 7392 | 7392 | 7366 |
| Instance | NC | Best tour cost found for different NB |
|----------|----|-------------------------------------|
| 7        | 15 ($r = 2508$) | 16 ($r = 1414$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 9095 | 8267 | 7934 | 7219 | 6866 | – | – | 8621 | 8316 | 8032 | – | 9027 | 8969 | 8626 | 8030 |
| NB 18 ($r = 1200$) | 19 ($r = 1900$) | 20 ($r = 1600$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | – | – | 10377 | 8647 | – | – | 9867 | 9867 | 9084 | 10675 | 9671 | 9355 | 9355 | 9221 |
| NB 22 ($r = 3400$) | 23 ($r = 3500$) | 24 ($r = 1900$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | – | 8415 | 7376 | 6474 | 6474 | 7805 | 7386 | 7070 | 7011 | 6846 | 9585 | 8211 | 7876 | 7876 | 7778 |
| kroD100  | 9   | 10 ($r = 1960$) | 11 ($r = 1950$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 7478 | 6761 | 6145 | 5686 | 5525 | 7245 | 6754 | 6561 | 6076 | 6076 | 7681 | 7004 | 6857 | 6465 | 6026 |
| NB 13 ($r = 1592$) | 14 ($r = 1592$) | 15 ($r = 1653$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | – | 9104 | 7356 | 7593 | 7176 | 9712 | 9004 | 8594 | 7538 | – | 9587 | 8506 | 8506 | 7929 | 7924 |
| NB 11 ($r = 1960$) | 12 ($r = 1665$) | 13 ($r = 2140$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | – | – | – | 10377 | 8647 | – | – | 9867 | 9867 | 9084 | 10675 | 9671 | 9355 | 9355 | 9221 |
| kroA150  | 9   | 10 ($r = 1960$) | 11 ($r = 1950$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 10181 | 9762 | 9762 | 9546 | 9220 | 9931 | 10282 | 9567 | 9542 | 542 | 10673 | 10788 | 10470 | 10611 | 10226 |
| NB 25 ($r = 3660$) | 26 ($r = 2890$) | 27 ($r = 1186$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 10427 | 10427 | 10395 | 10395 | 10264 | 10900 | 10953 | 10386 | 10314 | 10314 | 13393 | 12257 | 12190 | 11424 | 10951 |
| NB 17 ($r = 3200$) | 18 ($r = 3200$) | 19 ($r = 2770$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 8487 | 8487 | 8487 | 8033 | 8106 | 10303 | 10214 | 9850 | 9335 | 9339 | 10501 | 10655 | 10092 | 10172 | 10069 |
| kroA150  | 9   | 25 ($r = 3660$) | 26 ($r = 2710$) | 27 ($r = 1870$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 10427 | 10800 | 10395 | 10366 | 10264 | 10737 | 10783 | 10249 | 10249 | 10249 | 10737 | 10737 | 10249 | 10249 | 10249 |
| NB 15 ($r = 2090$) | 18 ($r = 1620$) | 19 ($r = 1240$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 8366 | 7562 | 7562 | 7805 | 7408 | 10181 | 9149 | 8822 | 8524 | 7967 | – | 11735 | 9918 | 9535 | 8671 |
| NB 20 ($r = 2050$) | 21 ($r = 2870$) | 22 ($r = 1820$) |
| Size     | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% | 50% | 60% | 70% | 80% | 100% |
| Result   | 9609 | 8955 | 9282 | 9245 | 8980 | 9677 | 9395 | 9454 | 9547 | 9547 | 9670 | 9571 | 8845 | 8845 | 8845 |
Table 7 Results under the restriction of group coverage cost NC(r/3)

| Instance | NC       | Result                |
|----------|----------|-----------------------|
|           |          | 8(r = 982) 9(r = 1186) 10(r = 1124) 11(r = 1125) 12(r = 1192) 13(r = 1315) |
| 7        | Best cost | 1790 1919 2161 2112 2371 2529 |
| Berlin52  |          | 6(r = 960) 7(r = 1082) 8(r = 982) 9(r = 1125) 10(r = 1124) 11(r = 1125) |
| 9        | Best cost | 1546 1487 1455 1929 2181 2070 |
|           |          | 5(r = 780) 6(r = 955) 7(r = 1082) 8(r = 982) 9(r = 955) 10(r = 1124) |
| 11       | Best cost | 1923 1559 1487 1463 1839 2168 |
| Berlin52  |          | 11(r = 82) 12(r = 72) 13(r = 52) 14(r = 56) 15(r = 54) 16(r = 62) |
| 7        | Best cost | 124 155 155 149 172 198 |
| Berlin52  |          | 9(r = 72) 10(r = 57) 11(r = 82) 12(r = 72) 13(r = 49) 14(r = 56) |
| 9        | Best cost | 141 118 123 133 124 154 |
| St70     |          | 7(r = 84) 8(r = 51) 9(r = 52) 10(r = 82) 11(r = 72) 12(r = 52) |
| 11       | Best cost | 111 105 95 107 121 122 |
| Berlin52  |          | 10(r = 60) 11(r = 33) 12(r = 34) 13(r = 43) 14(r = 33) 15(r = 27) |
| 7        | Best cost | 208 202 227 235 240 252 |
| St70     |          | 8(r = 34) 9(r = 45) 10(r = 43) 11(r = 50) 12(r = 58) 13(r = 32) |
| 9        | Best cost | 162 203 191 188 226 235 |
| St70     |          | 7(r = 43) 8(r = 39) 9(r = 43) 10(r = 42) 11(r = 46) 12(r = 50) |
| 11       | Best cost | 136 178 197 191 186 224 |
| Berlin52  |          | 15(r = 1875) 16(r = 947) 17(r = 2852) 18(r = 2160) 19(r = 2352) 20(r = 1520) |
| 7        | Best cost | 8191 8191 8261 8664 8880 9448 |
| Berlin52  |          | 12(r = 1976) 13(r = 2569) 14(r = 2560) 15(r = 1449) 16(r = 2426) 17(r = 2267) |
| 9        | Best cost | 7170 7333 7759 8565 8032 8046 |
| Berlin52  |          | 10(r = 2995) 11(r = 2445) 12(r = 2052) 13(r = 2641) 14(r = 2560) 15(r = 1835) |
| 11       | Best cost | 6276 6966 7360 7306 7375 7821 |
| Berlin52  |          | 15(r = 2233) 16(r = 2423) 17(r = 2532) 18(r = 2216) 19(r = 1244) 20(r = 1519) |
| 7        | Best cost | 8074 8421 8159 8943 9511 9384 |
| Berlin52  |          | 12(r = 1191) 13(r = 1468) 14(r = 1842) 15(r = 2440) 16(r = 1961) 17(r = 2457) |
| 9        | Best cost | 7919 8104 7903 8031 8413 8141 |
| Berlin52  |          | 10(r = 2212) 11(r = 1747) 12(r = 2281) 13(r = 2556) 14(r = 1842) 15(r = 2192) |
| 11       | Best cost | 6298 6811 6745 7347 7494 7915 |
| Berlin52  |          | 15(r = 2835) 16(r = 2003) 17(r = 2374) 18(r = 2359) 19(r = 1635) 20(r = 1757) |
| 7        | Best cost | 7678 8209 8212 8365 8886 8958 |
| Berlin52  |          | 12(r = 2535) 13(r = 1478) 14(r = 1902) 15(r = 1802) 16(r = 1316) 17(r = 3027) |
| 9        | Best cost | 6764 7813 7270 7684 8012 7997 |
| Berlin52  |          | 10(r = 1907) 11(r = 2417) 12(r = 2662) 13(r = 1246) 14(r = 2998) 15(r = 3440) |
| 11       | Best cost | 6771 6489 6786 7797 7325 7371 |
| Berlin52  |          | 15(r = 2508) 16(r = 1414) 17(r = 1699) 18(r = 1197) 19(r = 1908) 20(r = 1590) |
| 7        | Best cost | 7447 8952 8988 9181 9181 9203 |
| Berlin52  |          | 12(r = 1665) 13(r = 2140) 14(r = 1592) 15(r = 1601) 16(r = 1282) 17(r = 1601) |
| 9        | Best cost | 7751 7140 8028 7930 8579 8053 |
| Berlin52  |          | 10(r = 1599) 11(r = 1950) 12(r = 2154) 13(r = 1592) 14(r = 1592) 15(r = 1653) |
| 11       | Best cost | 6032 6489 6683 6927 7803 7210 |
| Berlin52  |          | 22(r = 3408) 23(r = 3511) 24(r = 1915) 25(r = 3664) 26(r = 2912) 27(r = 2892) |
| 7        | Best cost | 7645 7252 8051 9090 8880 |
| Berlin52  |          | 17(r = 3202) 23(r = 2302) 24(r = 2773) 25(r = 3664) 26(r = 2719) 27(r = 1876) |
| 9        | Best cost | 6577 7774 6713 7548 8648 10926 |
| Berlin52  |          | 15(r = 2091) 18(r = 1627) 19(r = 1240) 20(r = 2056) 21(r = 2871) 22(r = 1826) |
| 11       | Best cost | 8220 6438 8614 |
Table 8 Results of the CSPs with fuzzy cost matrices using two proposed approaches (same result)

| Instance | NC | Best tour cost found for different NB |
|----------|----|--------------------------------------|
|          |    | 9                                    |
|          |    | 10                                   |
|          |    | 11                                   |
|          |    | 12                                   |
|          |    | 13                                   |
| 7        | [1446.8,1449, 1450.4] | [1359.7,1542, 1641.3] | [2276.3,2279, 2610.5] |
| NB=8     | 7  | 8                                    |
|          |    | 9                                    |
|          |    | 10                                   |
|          |    | 11                                   |
| berlin52 | 9  | [958.5,960, 961.3] | [1178.8,1483, 1538.2] | [2154.4,2157, 2158.7] |
| NB =5    | 6  | 7                                    |
|          |    | 8                                    |
|          |    | 9                                    |
|          |    | 10                                   |
| 11       | [742.1,743, 743.7] | [951.8,953, 1105.5] | [1480.9,1483, 1484.7] | [1540.8,1543, 1544.9] |
| NB =11   | 12 | 13                                   |
|          |    | 14                                   |
|          |    | 15                                   |
| 7        | [181.2,186, 188.8] | [199.7,219, 239.3] | [237.3,241, 244.3] | [251.8,256, 260.4] |
| NB =8    | 9  | 10                                   |
|          |    | 11                                   |
|          |    | 12                                   |
|          |    | 13                                   |
| st70     | 9  | [1386.8,140, 141.0] | [152.3,155, 165.8] | [173.4,176, 178.1] | [178.5,181, 184.2] |
| NB =7    | 8  | 9                                    |
|          |    | 10                                   |
|          |    | 11                                   |
| 11       | [1054.1,108, 109.5] | [140.2,143, 165.8] | [178.2,181, 184.2] | [188.7,193, 196.4] |
| NB =11   | 12 | 13                                   |
|          |    | 14                                   |
|          |    | 15                                   |
| 7        | [113.2,116, 119.1] | [123.9,126, 146.5] | [139.9,143, 153.6] | [148.3,152, 161.5] |
| NB =9    | 10 | 11                                   |
|          |    | 12                                   |
|          |    | 13                                   |
| cil76    | 9  | [876.9,90, 92.1] | [99.7,113, 127.2] | [112.0,125, 153.8] |
| NB =7    | 8  | 9                                    |
|          |    | 10                                   |
|          |    | 11                                   |
| 11       | [60.6,62, 62.5] | [59.2,92, 109.7] | [105.9,108, 112.7] | [122.0,125, 127.2] |
| NB =15   | 16 | 17                                   |
|          |    | 18                                   |
|          |    | 19                                   |
|          |    | 20                                   |
| 7        | [7054.3,7058, 7062.8] | [8091.2,8013, 8127.9] | [8923.4,8928, 8932.6] | [9350.3,9356, 9360.8] |
| NB =12   | 13 | 14                                   |
|          |    | 15                                   |
|          |    | 16                                   |
|          |    | 17                                   |
| kroA100  | 9  | [6327.6,6333, 6333.7] | [6733.3,6736, 7618.9] | [7716.7,7721, 7724.5] | [8253.1,8257, 8260.8] |
| NB =10   | 11 | 12                                   |
|          |    | 13                                   |
|          |    | 14                                   |
|          |    | 15                                   |
| 11       | [5662.9,5665, 5667.5] | [6346.9,6350, 6475.1] | [6885.9,6889, 6812.1] | [7611.5,7615, 7618.9] |
| NB =15   | 16 | 17                                   |
|          |    | 18                                   |
|          |    | 19                                   |
|          |    | 20                                   |
| 7        | [7210.2,7214, 7217.0] | [8148.1,8152, 8725.6] | [8725.6,8730, 8861.5] | [9311.4,9315, 9318.5] |
| NB =12   | 13 | 14                                   |
|          |    | 15                                   |
|          |    | 16                                   |
|          |    | 17                                   |
| kroB100  | 9  | [6523.9,6527, 6530.2] | [7136.1,7139, 7418.4] | [7469.8,7674, 7678.4] | [8266.5,8271, 8275.3] |
| NB =10   | 11 | 12                                   |
|          |    | 13                                   |
|          |    | 14                                   |
|          |    | 15                                   |
Table 8 continued

| Instance | NC | Best tour cost found for different NB |
|----------|----|---------------------------------------|
|          |    | NB=8 | 9 | 10 | 11 | 12 | 13 |
| 11       | [5808.8,5812, 5814.7] | [6397.7,6400, 6402.4] | [6537.5,6541, 6543.4] | [6891.2,6895, 6899.2] | [7411.1,7415, 7418.4] | [7688.0,7691, 7695.4] |
|          |     | 16 | 17 | 18 | 19 | 20 |
| 7        | [7335.8,7340, 7344.5] | [7811.9,7818, 7820.4] | [7961.8,7965, 7974.9] | [8160.1,8165, 8170.0] | [8804.9,8810, 8815.3] | [8850.7,8856, 8860.7] |
|          |     | 13 | 14 | 15 | 16 | 17 |
| kroC100  | 9   | [6558.7,6562, 6564.2] | [6659.3,6663, 6666.4] | [7051.8,7055, 7085.6] | [6915.4,6919, 6922.5] | [7411.1,7415, 7418.4] |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 5 | 6 | 7 | 8 | 9 |
| kroD100  | 9   | [6339.7,6343, 6346.6] | [6947.2,6950, 6953.8] | [7406.6,7410, 7413.1] | [7815.3,7819, 7822.5] | [7934.9,7939, 7941.9] |
|          |     | 11 | 12 | 13 | 14 | 15 |
| kroE100  | 9   | [6673.1,6676, 6679.9] | [7233.7,7238, 7240.9] | [7551.2,7555, 7557.9] | [7605.7,7610, 7614.0] | [8121.8,8126, 8130.0] |
|          |     | 11 | 12 | 13 | 14 | 15 |
| rd100    | 9   | [2397.0,2401, 2403.1] | [2590.5,2593, 2597.8] | [2579.6,2583, 2586.9] | [2745.6,2751, 2755.3] | [2843.4,2848, 2852.4] |
|          |     | 12 | 13 | 14 | 15 | 16 |
| kroA150  | 9   | [7965.6,7971, 7975.0] | [8439.4,8444, 8447.9] | [8725.4,8730, 8735.4] | [8743.9,8749, 8753.0] | [8936.3,8941, 8947.6] |
|          |     | 14 | 15 | 16 | 17 | 18 |

NB = 10

| Instance | NC | Best tour cost found for different NB |
|----------|----|---------------------------------------|
|          |    | NB=15 | 9 | 10 | 11 | 12 | 13 |
|          |     | 16 | 17 | 18 | 19 | 20 |
|          |     | 7  | [7397.9,7401, 7404.9] | [7678.2,7682, 7686.7] | [8110.5,8115, 8120.0] | [8935.8,8940, 8945.0] | [9406.0,9410, 9415.1] |
|          |     | 12 | 13 | 14 | 15 | 16 |
| kroD100  | 10  | [6512.3,5615, 5617.7] | [6301.7,6305, 6307.8] | [6811.8,6815, 6819.2] | [7270.7,7274, 7278.0] | [7399.3,7403, 7406.1] |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 11 | 12 | 13 | 14 | 15 |
| kroE100  | 10  | [6673.1,6676, 6679.9] | [7233.7,7238, 7240.9] | [7551.2,7555, 7557.9] | [7605.7,7610, 7614.0] | [8094.9,8100, 8105.0] |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 11 | 12 | 13 | 14 | 15 |
| rd100    | 10  | [2222.3,2225, 2228.5] | [2384.2,2388, 2392.4] | [2442.6,2446, 2450.2] | [2581.2,2586, 2590.3] | [2379.0,2385, 2393.8] |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 7  | [2397.0,2401, 2403.1] | [2590.5,2593, 2597.8] | [2579.6,2583, 2586.9] | [2745.6,2751, 2755.3] | [2843.4,2848, 2852.4] |
|          |     | 12 | 13 | 14 | 15 | 16 |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 7  | [9178.8,9184, 9190.1] | [9719.5,9725, 9730.6] | [10212.8,10219, 10224.7] | [10497.3,10504, 10510.3] | [10974.4,10981, 10988.8] |
|          |     | 17 | 18 | 19 | 20 | 21 |
|          |     | 11 | 12 | 13 | 14 | 15 |
|          |     | 7  | [9178.8,9184, 9190.1] | [9719.5,9725, 9730.6] | [10212.8,10219, 10224.7] | [10497.3,10504, 10510.3] | [10974.4,10981, 10988.8] |
|          |     | 17 | 18 | 19 | 20 | 21 |
|          |     | 7  | [7397.9,7401, 7404.9] | [7678.2,7682, 7686.7] | [8110.5,8115, 8120.0] | [8935.8,8940, 8945.0] | [9406.0,9410, 9415.1] |
|          |     | 12 | 13 | 14 | 15 | 16 |
| kroA150  | 9   | [7965.6,7971, 7975.0] | [8439.4,8444, 8447.9] | [8725.4,8730, 8735.4] | [8743.9,8749, 8753.0] | [8936.3,8941, 8947.6] |
|          |     | 14 | 15 | 16 | 17 | 18 |
Table 8 continued

| Instance | NC | Best tour cost found for different NB |
|----------|----|--------------------------------------|
|          |    | NB=8                                 |
|          |    | 9 | 10 | 11 | 12 | 13 |
| 11       |    | [7313,9,7318, | [7729,0,7732, | [7520,4,7525, | [8084,0,8088, | [8614,7,8620, | [8691,4,8696, |
|          |    | 7320.9] | 7735.6] | 7529.3] | 8092.2] | 8625.0] | 8701.1] |
| 7        |    | [10678,6,10686, | [11095,3,11102, | [11240,2,11247, | [11400,6,11408, | [11608,1,11616, | [11755,1,11765, |
|          |    | 10692.6] | 11109.9] | 11254.5] | 11414.8] | 11624.3] | 11774.4] |
| kroA200  |    | [8509,9,8515, | [8669,7,8676, | [9381,8,9388, | [9547,9,9552, | [9545,5,9552, | |
|          |    | 8519.2] | 8937.2] | 9392.6] | 9547] | 9557] | |

Table 9 Results of the CSPs with rough cost matrices using two proposed approaches (same result)

| Instance | NC | Best tour cost found for different NB |
|----------|----|--------------------------------------|
|          |    | NB=8                                 |
|          |    | 9 | 10 | 11 | 12 | 13 |
| 7        |    | [1446,1462.5, | [1547,1559.8, | [1639,1661.6, | [1604,4,1694.4, | [2316,1,2400.1, | [2090,2,2164.6, |
|          |    | 1422,1482.4] | 1462.4,1582.4] | | 1694.4] | 2164.6] | 1697] |
| Berlin52  |    | [909,923.6, | [1181,1196.8, | [1456,8,1524.5, | [1531,9,1579.7, | [2090,2,2164.6, | |
|          |    | 889,3,934.4] | 1178,9,1224] | | | | |
| 11       |    | [743,751.8, | [915,927.5, | [1456,8,1524.5, | [1531,9,1579.7, | [2090,2,2164.6, | |
|          |    | 729,9,764.8] | 901,5,938.8] | | | | |
| st70     |    | [138,157.1, | [162,178.4, | [173,5,253.4, | [197,5,253.4, | [200,2,2164.6, | |
|          |    | 125,9,174.5] | 173,5,253.4] | | | | |
| eil76    |    | [87,97,7, | [98,119.4, | [120,135.5, | [125,154.4, | [136,163.3, | |
|          |    | 69,3,130.1] | 120,135.5] | | | | |
|          |    | [87,97,7, | [98,119.4, | [120,135.5, | [125,154.4, | [136,163.3, | |
|          |    | 69,3,130.1] | 120,135.5] | | | | |
| Instance | NC | Best tour cost found for different NB |
|----------|----|--------------------------------------|
|          |    | NB= 8  | 9    | 10     | 11    | 12    | 13    |
| rat99    | 9  | [304,327.8, 325,352.7, 349,380.5, 372,403.2, 370,404.9, 385,425.3, 279,3,531.1, 294,5,382.5, 320,8,402.1, 347,5,431.2, 333,9,441.3, 346,9,465.2] | | | | | |
|          |    | NB= 9  | 10    | 11     | 12    | 13    | 14    |
|          | 11 | [260,276.9, 269,293.7, 304,327.8, 330,352.5, 346,379, 372,403.0, 234,7,290.8, 250,1,315, 279,3,351.1, 306,1,378.2, 316,4,409.8, 347,3,431.2] | | | | | |
|          |    | NB= 15 | 16    | 17     | 18    | 19    | 20    |
|          | 7  | [2401,2434.2, 2599,2636.5, 2534,2565.3, 2712,2761.6, 2848,2896.6, 2927,3032.8, 2364,3,2473.9, 2563,6,2686.1, 2495,7,2604.2, 2656,8,2809.5, 2796,2,2940.5, 2913,4,3071.1] | | | | | |
| rd100    | 9  | [2161,2189.9, 2190,2215.4, 2428,2459.9, 2512,2546.7, 2633,2665.2, 2640,2686.4, 2128,1,2226.2, 2157,4,2237.8, 2376,7,2496.7, 2471,2,2582.4, 2607,8,2710.7, 2607,8,2728.9] | | | | | |
|          |    | NB= 10 | 11    | 12     | 13    | 14    | 15    |
|          | 11 | [1952,1975.7, 2027,2054.8, 2127,2154.8, 2262,2288.6, 2428,2459.9, 2527,2564.1, 1921,3,2005.8, 1999,1,2084.2, 1999,1,2084.2, 2226,8,2323.9, 2376,7,2496.7, 2490,6,2606.1] | | | | | |
|          |    | NB= 15 | 16    | 17     | 18    | 19    | 20    |
|          | 7  | [7291,7327.8, 8038,8077.5, 8023,8060.6, 8199,8236.3, 8996,9035.3, 9356,9387.4, 7241,6,7374, 8006,8,8120.9, 7985,4,8109.2, 8171,8,8284.8, 8958,2,9069.3, 9298,5,9439.3] | | | | | |
| kroA100  | 9  | [6331,6357.4, 6736,6756.8, 7684,7722, 7721,7754.2, 8257,8287.5, 7916,7951, 6235,6388.9, 6697,8,6794.2, 7644,7,7759.2, 7862,9,8327.5, 7885,6,7995.6] | | | | | |
|          |    | NB= 10 | 11    | 12     | 13    | 14    | 15    |
|          | 11 | [5665,5683.9, 6350,6375.7, 6472,6506.8, 6869,6905.3, 7684,7722, 7614,7645.9, 5636,6,5708.7, 6321,1,6401, 6441,5,6534.3, 6837,1,6942.6, 7644,7,7755.2, 7570,3,7697.2] | | | | | |
|           | | | | | | | |
| Instance | NC | Best tour cost found for different NB |
|----------|----|--------------------------------------|
|          |    | NB= 15 | 16    | 17     | 18    | 19    | 20    |
|          | 7  | [7214,7235.2, 8152,8185.1, 8056,8085.4, 8692,8734.1, 8686,8730.8, 9135,9176.4, 7182,9,7291.4, 8109,2,8222.7, 8014,7,8125.1, 8649,2,8773.7, 8636,5,8782.5, 9090,7,9229.4] | | | | | |
| kroB100  | 9  | [6527,6558.8, 7139,7165.4, 7415,7447.7, 7674,7708, 8271,8300.8, 8135,8173.6, 6504,3,6589.8, 7098,3,7200.6, 7386,2,7483.5, 7643,7,7570.8, 8222,5,8341.7, 8089,8225.4] | | | | | |
|          |    | NB= 15 | 16    | 17     | 18    | 19    | 20    |
|          | 11 | [5779,5800.3, 6400,6420.1, 6514,6540.5, 7247,7276.6, 7415,7447.7, 7691,7720, 5744.5,5817.9, 6366,7,6440.9, 6481,4,6578.8, 7201,9,7310.3, 7386,2,7483.5, 7638,7,7767] | | | | | |
| kroD100  | 9  | [6343,6364.2, 6950,6977.8, 7271,7303.4, 6965,7001.7, 7926,7968.4, 7939,7978.9, 6315,7,6398.4, 6950,6977.8, 7238,7,7339.1, 6927,7,7307.2, 7894,7,7995.8, 7890,8,8018.4] | | | | | |
|          |    | NB= 15 | 16    | 17     | 18    | 19    | 20    |
|          | 11 | [5615,5633.3, 5795,5820.6, 6305,6324.2, 6768,6793.8, 7271,7303.1, 7359,7388.2, 5587.7,5655.5, 5768,7,5844.6, 6272,4,6353.7, 6738,9,6823.1, 7238,7,7339.1, 7324,7424.8] | | | | | |
|          |    | NB= 22 | 23    | 24     | 25    | 26    | 27    |
environment. To solve the problem in imprecise environment, some approach is followed where tour is searched without transferring the imprecise optimisation problem into a crisp optimisation problem. Moreover, fuzzy and rough simulation approaches are proposed to solve the problem for type of fuzzy data set and any measure of rough data set. For the restricted problems, path is searched in such a manner that the total distance/cost of different cities of a group from the visited city of the group should not exceed a predefined limit. The algorithm is tested with standard benchmark crisp problems available in the literature. To test the algorithm in the imprecise environment, imprecise instances are derived randomly from the standard crisp instances using a specified rule. Different numerical studies and the comparison studies with different well-established algorithms for CSPs establish the efficiency of the proposed approach for solving the CSPs in the crisp as well as in the imprecise environments. The approaches presented in this study are applicable in any routing problems, like, TSP, GTSP, CTP, BCSP, Vehicle Routing Problem, etc., in different environments. The simulation approaches presented here can be applied to solve different real-life optimisation problems in science and technology. In future work, stochastic simulation approach can be used to deal with the problem with the random data sets. Also, the study can be done with the mixed data sets, i.e. the data sets involving different types of imprecise data. Multi-objective CSPs in imprecise environments can also be studied in the future.

### Funding
No funding used for this research work.

### Data availability
Enquiries about data availability should be directed to the authors.

### Declarations

#### Conflict of interest
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. The authors declare the following financial interests/personal relationships which may be considered as potential competing interests.

### Table 9 continued

| Instance | NC | Best tour cost found for different NB |
|----------|----|---------------------------------------|
|          |    | NB=8  | 9 | 10 | 11 | 12 | 13 |
| 7 kroA150 |    | [9184, 9228.3, 9132.6, 9294.5] | 9823, 9872.4 | 10344, 10407.1 | 10583, 10643.5 | 10299, 10351, 10917, 10981.1 | 10836.3, 11047.5 |
| 9        |    | 18 | 18 | 20 | 21 | 22 | |
| 7764.8, 7879.9 |    | [7806, 7841, 8511.6, 8645] | 8560, 8607.7 | 8695, 8735.1 | 8749, 8798.6 | [9163, 9211.1, 9471, 9531.8] | |
| NB=14    |    | [7523, 7559.4, 7481.8, 7590.9] | [7732, 7757.2, 8018.4, 8134.7] | 8058, 8100.7 | 8088, 8121.8 | [8561, 8602.2, 8808, 8854.1, 8766.1, 8897.9] | |

### References

Current JR, Schilling David A (1989) The covering salesman problem. The Ohio University, Columbus, Ohio

Campbell AM, Vandenbussche D, Hermann W (2008) Routing for relief efforts. Transport Sci 42(2):127–145

Dorigo M, Gambardella LM (1997) Ant colonies for the traveling salesman problem. BioSystems 43:73–81

Dorigo M, Di Caro G (1999) The ant colony optimization meta-heuristics. In: Corne D, Dorigo M, Glover F (eds) New ideas in optimization. McGraw-Hill, London, pp 11–32

Gerhard R (1999) TSPLIB—a traveling salesman problem library. ORSA J Comput 3(4):376–384

Gendreau M, Laporte G, Semet F (1992) The covering tour problem. Comput Oper Res 4010:86–94

Golden BL, Nari-Azimi Z, Raghavan S, Salari M, Toth P (2012) The generalized covering salesman problem. INFORMS J Comput 24(4):534–553

Hachicha M, Hodgson M, Laporte G, Semet F (2000) Heuristics for the multi-vehicle covering tour problem. Comput Oper Res 27:29–42

Kersgosien Y, Lenté C, Billaut J (2009) Home health care problem: An extended multiple Traveling Salesman Problem. In: Multi-disciplinary international conference on scheduling: theory and applications (MISTA 2009), Dublin, Ireland 10-12 August 2009

Khanra A, Maiti MK, Maiti M (2016) Profit maximization of TSP with uncertain parameters through a hybrid algorithm. In: Proceedings of the 4th international conference on frontiers in intelligent computing: theory and applications (FICTA), Advances in Intelligent Systems and Computing Science. Springer https://doi.org/10.1007/978-81-322-2695-6_26

Khan I, Maiti MK (2018) A novel hybrid algorithm for generalized traveling salesman problems in different environments. Vietnam J Comput Sci 5(1):27–43. https://doi.org/10.1007/s40595-017-0099-z

Khan I, Pal S, Maiti MK (2018) A modified particle swarm optimization algorithm for solving traveling salesman problem with imprecise cost matrix. In: 4th International conference on recent advances in information technology (RAIT). https://doi.org/10.1109/RAIT.2018.8389060

Khan I, Pal S, Maiti MK (2019) A hybrid PSO-GA algorithm for traveling salesman problems in different environments. Int J Uncertain Fuzziness Knowl Based Syst 27(5):693–717

Kotenko I, Senenko I (2019) Hierarchical fuzzy situational networks for online decision-making. Appl Telecommun Syst Comput Oper Res 39:2594–2602

Kumar R (2012) Blending roulette wheel selection & rank selection in genetic algorithms. Int J Mach Learn Comput 2(4):365–370
Lin S, Kernighan B (1973) An effective heuristics algorithm for traveling salesman problem. Oper Res 21:498–516
Labbe M, Laporte G, Martin I, Salzar G (2004) The ring star problem: polyhedral analysis and exact algorithm. Networks 43(3):177–189
Labbe M, Laporte G, Martin I, Salzar G (2005) Locating median cycles in networks. Eur J Oper Res 160:457–470
Liu B (2002) Theory and practice of uncertain programming. Physica-Verlag, Heidelberg
Liu B, Iwamura K (1998) A note on chance constrained programming with fuzzy coefficients. Fuzzy Sets Syst 100:229–233
Maurya AK, Kumar N (2017) Localization problem in disaster management smartphone application. Int J Adv Res Comput Sci 8(9):177–180
Pandiri V, Singh A, Rossi A (2020) Two hybrid metaheuristic approaches for the covering salesman problem. Neural Comput Appl 32(19):15643-15663
Pramanik P, Maiti MK, Maiti M (2017) Three level partial trade credit with promotional cost sharing. Appl Soft Comput 58:553–575
Salari M, Naji-Azimi Z (2012) An integer programming-based local search for the covering salesman problem. Comput Oper Res 39:2594–2602
Salari M, Reihaneh M, Sabbagh MS (2015) Combining ant colony optimization algorithm and dynamic programming technique for solving the covering salesman problem. Comput Ind Eng 83:244–251
Sierksma G (2014) Hamiltonicity and the 3-Opt procedure for the traveling salesman problem. Appl Math 22(2):351–358
Tripathy S, Tulshyan A, Kar S, Pal T (2017) A metameric genetic algorithm with new operator for covering salesman problem with full coverage. Int J Control Theory Appl 10(7):245–252
Ungureanu V (2006) Traveling salesman problem with transportation. Comput Sci J Moldova 14(2):41
Venkatesh P, Srivastava G, Singh A (2019) A multi-start iterated local search algorithm with variable degree of perturbation for the covering salesman problem. Harmony search and nature inspired optimization algorithms. Springer, Singapore, pp 279–292
Zadeh L (1965) Fuzzy sets. Inf Control 8:338–356
Zang Z, Jiang L, Ratli M, Ding B (2020) A parallel variable neighborhood search for solving covering salesman problem. Optim Lett 16:175–190

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.