Rough Genetic Algorithm for Constrained Solid TSP with Interval Valued Costs and Times

Samir Maity a, Arindam Roy b and Manoranjan Maiti c

a Operations Management Group, Indian Institute of Management Calcutta, Kolkata, India; b Department of Computer Sciences & Applications, P.K. College, Contai, India; c Department of Applied Mathematics, Vidyasagar University, Medinipur, India

ABSTRACT
This paper presents new rough set based genetic algorithms (RSGAs) to solve constrained solid travelling salesman problems (CSTSPs) with restricted conveyances (CSTSPwR) having uncertain costs and times as interval values. To grow the impreciseness in soft computing (SC), the proposed RSGAs, a rough set based age-dependent selection technique and an age-oriented min-point crossover are used along with three types of probability, $p_m$-dependent random mutations. A number of benchmark problems from standard data set, TSPLIB are tested against the proposed algorithms and existing simple GA (SGA). CSTSPwRs are formulated as constrained linear programming problems and solved by both proposed RSGAs and SGA. These are illustrated numerically by some empirical data and the results from the above methods are compared. Statistical significance of the proposed algorithms are demonstrated through statistical analysis using standard deviation. Moreover, the non-parametric test, Friedman test, is performed with the proposed algorithms. In addition, a post hoc paired comparison is applied and the out performance of the RSGAs.

1. Introduction
TSP (travelling salesman problem) is an NP-hard combinatorial optimisation problem [1]. During last two decades, several types of TSPs have been solved by the researchers. In TSP, a Travelling Salesman (TS) is sometimes asked to visit a given number of cities/nodes for which there are time windows for the earliest and the latest service start times. Focacci et al. [2] formulated a TSP with time window (TSPTW) and solved using a hybrid exact algorithm. Later, Chang et al. [3] extended the TSPTW with stochastic travel time and tight time-windows (TWs) called stochastic dynamic TSP with hard time windows (SDTSPHTD). To avoid congestion, the double TSP with multiple stacks (DTSPMS) was considered by Petersen and Madsen [4]. It finds the shortest routes performing pickups and deliveries in two separated networks/regions.

All the above-mentioned models assumed the edge costs to be symmetric. Majumder and Bhunia [5] formulated a TSP with asymmetric costs and imprecise travel times and
solved using GA. Moon et al. [6] applied precedence constraints before visiting the nodes/cities in a TSP and solved using an improved GA. Xing et al. [7] presented a hybrid approach that combines an improved GA and optimisation strategies for solving the asymmetric TSP (ATSP). Bai et al. [8] proposed a max–min ant colony optimisation method for the solution of ATSPs bridging the gap between hybridisation and theoretical analysis. Jula et al. [9] considered a routing problem with stochastic travel times and time windows estimating means and variances of arrival times at nodes and removing routes that are dominated by others. Chang et al. [3] solved a stochastic dynamic TSP with hard time windows following more or less same procedures of Jula et al. [9]. Chang and Mao [10] developed a modified ant algorithm to solve TSPTWs for minimum cost tour. Dong et al. [11] proposed a new hybrid algorithm, cooperative genetic ant system to solve TSP. Yuan et al. [12] proposed a new crossover operator called two-part chromosome crossover for solving the multiple travelling salesman problem (MTSP). Recently, Miranda Bront et al. [13] formulated and solved a time-dependent travelling salesman problem (TTSP). Perturbed Decomposition Algorithm applied for multi-objective TSP by Cornu et al. [14].

In spite of the above different types of developments in the area of TSP, there are some lacunae in the realistic formulation of the problem and development of solution techniques. These are as follows.

In TSP, it is generally assumed that a salesman travels from one city to another using only one conveyance. But in real life, a set of conveyances may be available at each city. In that case, a salesman has to design his/her tour for minimum cost/time/distance maintaining the TSP conditions and using the suitable conveyances at different cities. This problem is called Solid Travelling Salesman Problem (STSP). In realistic situations, all conveyances may not be available from all places/stations. According to the availability of the conveyances, the model is set up with some restricted conveyances at each station, i.e. in each node, equal number of conveyances may not be available to travel from one station to another. All the above-mentioned earlier TSP formulations considered only one conveyance for travel between different cities. Here we have considered several different conveyances at different stations.

Travelling cost and time from one city to another city depend on the several factors such as types of conveyances, condition of roads, geographical areas, weather condition such as rainy/foggy weather at the time of the travel, conditions of vehicles and so on. So there always prevail some uncertainties/vagueness. Under these circumstances, time and cost parameters may be considered as imprecise in nature in the form of uncertain interval values instead of a fixed (deterministic) values. Except Changdar et al. [15], earlier investigations did not consider TSPs with imprecise/uncertain costs and/or times. Here we have formulated STSPs with interval valued costs and times in the objective and constraint.

Moreover, a salesman may be asked to complete the entire trip within a specified time which acts as a constraint on the model. Such kind of constraint is called time constraint, which may also be taken as uncertain interval values. In realistic approach of TSP, as the conveyances are restricted, we formulate CSTSP with restricted conveyances say CSTSPwR. Till now, none has considered such CSTSPwR in crisp or imprecise environments.

Basically to grow the impreciseness in SC methods that prompted us to design such kind of inclusions of rough set in the proposed GA. The uncertainty is come by nature in realistic situation of the proposed model also. Some historical study is presented in Table 1.
Table 1. Historical study of soft computing techniques.

| Methodology          | Algorithms          | Authors                    | Year  |
|----------------------|---------------------|----------------------------|-------|
| Soft Computing       | Zadeh et al. [16,17]| 1994, 1998                 |       |
| Genetic Algorithm    | Goldberg et al. [18]| 1989                       |       |
| Simulated Annealing  | Chiang et al. [19]  | 1997                       |       |
| Tabu Search          | Knox et al. [20]    | 1989                       |       |
| Ant Colony           | Dorigo et al. [21]  | 2002                       |       |
| Simple GA            | Holland et al. [22] | 1975                       |       |
| PSO                  | Eberhart et al. [23]| 1995                       |       |
| TLO                  | Črepinšek et al. [24–26]| 2000                   |       |
| ABC                  | Mernik et al. [27]  | 2014, 2015                 |       |
| SA vs EA             | Piotrowski et al. [28]| 2016                    |       |
| Pareto GA            | Ni et al. [29]      | 2013                       |       |
| Localised            | Ursani et al. [30]  | 2011                       |       |
| Adaptive             | Neungmathchana et al. [31]| 2013             |       |
| Enhanced             | Huang et al. [32]   | 2012                       |       |
| Efficient            | Enigin et al. [33]  | 2011                       |       |
| Fuzzy GA             | Last, Roy, Fedz et al. [34–36]| 2005, 2009|       |
| GA                   | Elyam et al. [37]   | 2016                       |       |
| Generic GA           | Liang et al. [38]   | 2009                       |       |
| Rough set, GA        | Cheng et al. [39]   | 2010                       |       |
| Rough penalty GA     | Lin et al. [40]     | 2013                       |       |
| GA                   | Mirhosseini et al. [41]| 2014                   |       |

In our proposed RSGAs, we consider age of each chromosome in the form of a rough set/extended rough set as a selection operation which provides the creation of the probability of crossover (p_c). Thus p_c is a function of the chromosome ‘age’. For crossover operation, we determine the convex combination of the interval values, after modifying the parents, compare the costs between each node of the pair of chromosomes and the best one is concatenated in the new route/offspring. For mutation operation, a generation dependent new probability of mutation (p_m) is created and with this p_m, total number of nodes to be mutated are calculated. Then identified nodes/positions/locations are randomly selected and mutated within a chromosome/path.

In this paper, CSTSPwRs are formulated with asymmetric parametric crisp interval values for time and costs. A total time is imposed for the entire tour in the form of a constraint. CSTSPwR formulated in different environments are solved by both proposed RSGAs and SGA for some empirical data set. Alternative near-optimum paths along with optimum time and cost are presented for each CSTSPwR.

To establish the efficiency of the RSGAs, some standard test problems from TSPLIB [42] are solved by both RSGAs and SGA and their performance are compared. The transformed CSTSPwRs are also solved using both proposed RSGAs and SGA. These models are illustrated with some numerical data and the results from both methods are compared. In crisp environment, some sensitivity analyses are performed with different allowable times. Through standard deviation (SD) and error analyses, the non-parametric Friedman test is applied also post-hoc paired analyses is done.

The present problem is more complicated due to the uncertainty/imprecision of the costs and times through uncertain interval values. As optimisation of interval valued objective functions is not well defined, it is very difficult to formulate and optimise these. Tiwari et al. [43] proposed fuzzy closer and fuzzy-rough set. We use fuzzy possibility and necessity based approaches [44,45] to represent and to solve the CSTSPwR with uncertain interval values. Thus present investigation contains the following new features:
• CSTSPwR are more realistic TSPs. None considered this type of uncertain problems with interval valued parameters.
• For the first time, new algorithms of RSGAs with ‘Rough set based selection’, ‘min-point crossover’ and generation-dependent mutations are developed.
• New RSGAs are tested with different data sets from TSPLIB [42] and the efficiency of the proposed algorithms is established.
• Here objective function is formulated using geometric mean (this is a new concept) and constraint values are compared using different interval ranking procedures.
• Non-parametric Friedman test is performed with the proposed algorithms to establish the differences amongst the algorithms.
• A post hoc paired comparison is made to determine the degree of differences and thus the proposed algorithm outperforms the others.

This paper is organised as follows. In Section 2, we describe mathematical preliminaries with definitions. Section 3 gives different types of CSTSPwRs. In Section 4, RSGAs are presented. Finally, we illustrate the above problems using some empirical data in Section 5. In Section 6, some statistical tests using SD are performed. Last, in Section 7, we conclude the paper identifying the scope of future development.

In next section, some mathematical requisite are described.

2. Mathematical preliminaries

2.1. Interval number

A (real) interval is a nonempty set of real numbers

$$\mathbf{A} = [a_L, a_R] = \{ x \in \Re | a_L \leq A \leq a_R \},$$

where $a_L$ and $a_R$ are the lower and upper bounds (end points) of the interval number $\mathbf{A}$, respectively. $A$ is a generic (arbitrary) element in the interval $\mathbf{A}$. $\Re$ denotes the all real numbers. $\mathbf{A}$ is a more complex expression, the lower bound and the upper bound are also written as

$$a_L \equiv \inf(\mathbf{A}), \quad a_R \equiv \sup(\mathbf{A}).$$

The set of all interval is denoted by $\mathcal{IR}$. To represent an unknown number as an approximation plus/minus an error bound, the midpoint $\tilde{A}$ and with of an interval $\mathbf{A}$ are respectively introduced as

$$\tilde{A} \equiv \text{mid}(x) = \frac{a_L + a_R}{2}, \quad \text{and} \quad \text{wid}(\mathbf{A}) = a_R - a_L.$$

Hence $\mathbf{A}$ can be wid(\mathbf{A}) represented as

$$\hat{\mathbf{A}} = [\tilde{A}, \text{wid}(\mathbf{A})] = (a_c, a_w).$$

**Definition 2.1 (Majumder et al. [5]):** Let $\ast \in \{+, -, \cdot, \div \}$ be a binary operation on the set of all real numbers. If $A$ and $B$ are two closed intervals, then $A \ast B = \{a \ast b : a \in A, b \in B \}$ defines a binary operation on the set of closed intervals. In the case of division, it is assumed that $0 \notin B$. 
The operations on intervals used in this paper may be explicitly calculated for two interval numbers, \( A = [a_L, a_R] \), \( B = [b_L, b_R] \) and \( \hat{A} = [a_c, a_w] \), \( \hat{B} = [b_c, b_w] \) from above definition as

\[
\begin{align*}
A + B &= [a_L, a_R] + [b_L, b_R] = [a_L + b_L, a_R + b_R], \\
\hat{A} + \hat{B} &= [a_c, a_w] + [b_c, b_w] = [a_c + b_c, a_w + b_w], \\
kA &= k[a_L, a_R] = [ka_L, ka_R], & \text{for } k \geq 0, \\
k\hat{A} &= k[a_c, a_w] = [ka_c, |k|a_w], & \text{where } k \text{ is a real number}
\end{align*}
\]

(1)

Let the uncertain costs from two alternatives be represented by two closed intervals \( \hat{A} = [a_c, a_w] \), \( \hat{B} = [b_c, b_w] \) respectively. It is also assumed that the cost of each alternative lies in the corresponding interval. These two intervals \( \hat{A} \) and \( \hat{B} \) may be of the following three types: Type I: Both the intervals are disjoint.
Type II: Intervals are partially overlapping.
Type III: One interval is contained in the other.

In optimistic decision making, the decision maker (DM) expects the lowest cost ignoring the uncertainty. According to Majumder et al. [5], the order relations of the interval numbers for minimisation problems in case of optimistic decision making are as follows:

**Definition 2.2 (Majumder et al. [5]):** Let us define the order relation \( \leq_{omin} \) between \( A = [a_L, a_R] \) and \( B = [b_L, b_R] \) as

\[
\begin{align*}
A \leq_{omin} B &\iff a_L \leq b_L, \\
A \leq_{omin} B &\iff A \leq_{omin} B \land A \neq B.
\end{align*}
\]

(2)

**Pessimistic decision making**

For pessimistic decision making, the DM expects the minimum cost for minimisation problems according to the principle ‘Less uncertainty is better than more uncertainty’. According to Karmakar et al. [44] and Majumder et al. [5], the order relations of interval numbers for minimisation problems in case of pessimistic decision making are as follows:

**Definition 2.3 (Majumder et al. [5]):** Let us define the order relation \( \leq_{pmin} \) between \( \hat{A} = [a_c, a_w] \) and \( \hat{B} = [b_c, b_w] \) as

\[
\begin{align*}
\hat{A} \leq_{pmin} \hat{B} &\iff a_c \leq b_c \text{ for Type I and Type II intervals} \\
\hat{A} \leq_{pmin} \hat{B} &\iff (a_c \leq b_c) \land (a_w < b_w) \text{ for Type III intervals.}
\end{align*}
\]

(3)

However, for Type III intervals with \( (a_c \leq b_c) \land (a_w < b_w) \), the pessimistic decision cannot be taken. Here, the optimistic decision is to be considered.

**Remark 2.4:** Now as the interval valued objectives are not well defined, so we use common features of arithmetic mean (AM) and geometric mean (GM) as follows. Let \( A = [a_L, a_R] \) be a common interval for a particular objective function. Since we know that

\[
AM \geq GM \Rightarrow \frac{m_1 \ast a_L + m_2 \ast a_R}{m_1 + m_2} \geq \left( a_L^{m_1} \ast a_R^{m_2} \right)^{1/(m_1 + m_2)}
\]

(4)

for a minimisation problem, determine the minimum of the objective function \( (a_L^{m_1} \ast a_R^{m_2})^{1/(m_1 + m_2)} \), here \( m_1 \) and \( m_2 \) are the given weights.
2.2. Optimisation problem with interval values

An optimisation problem can be stated as follows:

Minimise \( f(X) = \sum_{j=1}^{n} \hat{c}_j x_j \)

subject to \( \sum_{j=1}^{n} \hat{a}_{ij} x_j \leq \hat{b}_i \) \hspace{1cm} for \( i = 1, 2, \ldots, m \),

\( x_j \geq 0 \) \hspace{1cm} for \( j = 1, 2, \ldots, n \),

where \( \hat{c}_j, \hat{a}_{ij} \) and \( \hat{b}_i \) are interval values with known lower and upper bounds. Then the above equation is transformed as

Minimise \( f(X) = \sum_{j=1}^{n} [\hat{c}_j, \bar{c}_j] x_j \)

subject to \( [A_i, \bar{A}_i]x_j \subseteq [b_i, \bar{b}_i] \) \hspace{1cm} for \( i = 1, 2, \ldots, m \),

\( x_j \geq 0 \) \hspace{1cm} for \( j = 1, 2, \ldots, n \),

where \( A_i = \sum_{j=1}^{N} a_{ij}, \bar{A}_i = \sum_{j=1}^{N} \bar{a}_{ij} \).

The interval valued \( c, \bar{c} \) are the upper and lower bound values of \( \hat{c} \), similar values are taken for \( \hat{a} \) and \( \hat{b} \).

Now as the above objective functions are not well defined, so it is converted according to Equation (4) and interval valued constraints are represented with different approaches [46] as

Minimise \( f(X) = \sum_{j=1}^{n} [\hat{c}_j, \bar{c}_j] x_j \)

subject to \( [A_i, \bar{A}_i]x_j \subseteq [b_i, \bar{b}_i] \) \hspace{1cm} for \( i = 1, 2, \ldots, m \),

\( x_j \geq 0 \) \hspace{1cm} for \( j = 1, 2, \ldots, n \).

Following different approaches for transformation on an interval constraint.

According to Moore’s approaches [46], the above constraint in Equation (7) can be written as

subject to \( A_i x_j \geq b_i, \bar{A}_i x_j \leq \bar{b}_i \),

following Ishibuchi and Tanaka’s approaches [46]

subject to \( A_i x_j \leq b_i, \bar{A}_i x_j \leq \bar{b}_i \),

due to Chanas and Kuchta’s approaches [46] for \( 0 \leq s_0 < s_1 \leq 1 \),

subject to \( (A_i + s_0 \ast (\bar{A}_i - A_i)) x_j \leq (b_i + s_0 \ast (\bar{b}_i - b_i)), (A_i + s_0 \ast (\bar{A}_i - A_i) + A_i + s_1 \ast (\bar{A}_i - A_i)) x_j \leq (b_i + s_0 \ast (\bar{b}_i - b_i) + b_i + s_1 \ast (\bar{b}_i - b_i)) \)
by Hu and Wang’s approaches [46]

$$\frac{(A_i + \overline{A_i})}{2} x_j \leq \frac{(\overline{b_i} + b_j)}{2},$$

subject to \(
(A_i - \overline{A_i}) x_j \geq (\overline{b_i} - b_j)
\) (11)

and using Mahato and Bhunia’s approaches [46]

subject to \((\text{optimistic case})\) \(A_i x_j \leq b_j\),

\((\text{pessimistic case})\) \(\frac{(A_i + \overline{A_i})}{2} x_j \leq \frac{(\overline{b_i} + b_j)}{2}\), for Type I & II intervals, (12)

\((A_i - \overline{A_i}) x_j \geq (\overline{b_i} - b_j)\), for Type III intervals.

In every case of the constraint evaluation from the above equations (8)–(12) \(x_j \geq 0\), for \(j = 1, 2, \ldots, n\).

Rough variable (cf. Mondal et al. [47])

Let a rough variable \(\xi\) is a measurable function from the rough space \((\Lambda, \Delta, \kappa, \Pi)\) to the set of real numbers. i.e. for every Borel set of \(\Re\), \(\{\lambda \in \Lambda | \xi(\lambda) \in B\} \in \kappa\).

The lower \((\underline{\xi})\) and upper \((\overline{\xi})\) approximations of the rough variable \(\xi\) are given by \(\xi = \{\xi(\lambda) | \lambda \in \Lambda\}\).

Trust measure (cf. Mondal et al. [47])

Let \((\Lambda, \Delta, \kappa, \Pi)\) be a rough space. The trust measure of event \(A\) is denoted by \(Tr\{A\}\) and defined by \(Tr\{A\} = \frac{1}{2}(Tr\{A\} + Tr\{\overline{A}\})\). Upper and lower trust measures of event \(A\) are respectively defined by \(Tr\{A\} = \frac{\Pi\{A\}}{\Pi\{\Delta\}}\) and \(Tr\{A\} = \frac{\Pi\{A \cap \Delta\}}{\Pi\{\Delta\}}\). When enough information about the measure \(\Pi\) is not available, it may be treated as the Lebesgue measure.

Then we can get the trust measure of the rough event \(\hat{\xi} \geq r\) as \(Tr\{\hat{\xi} \geq r\}\) and its function curves (Figure 1) are presented below where \(r\) is a crisp number, \(\hat{\xi}\) is a rough variable given by \(\hat{\xi} = ([a, b], [c, d]), 0 \leq c \leq a \leq b \leq d\).

\[
Tr\{\hat{\xi} \geq r\} = \begin{cases} 
0 & \text{for } d \leq r, \\
\frac{(d - r)}{2(d - c)} & \text{for } b \leq r \leq d, \\
\frac{1}{2} \left( \frac{(d - r)}{(d - c)} + \frac{(b - r)}{(b - a)} \right) & \text{for } a \leq r \leq b, \\
\frac{1}{2} \left( \frac{(d - r)}{(d - c)} + 1 \right) & \text{for } c \leq r \leq a, \\
1 & \text{for } r \leq c.
\end{cases}
\] (13)

\[
Tr\{\hat{\xi} \leq r\} = \begin{cases} 
0 & \text{for } r \leq c, \\
\frac{(r - c)}{2(d - c)} & \text{for } c \leq r \leq a, \\
\frac{1}{2} \left( \frac{(r - c)}{(d - c)} + \frac{(r - a)}{(b - a)} \right) & \text{for } a \leq r \leq b, \\
\frac{1}{2} \left( \frac{(r - c)}{(d - c)} + 1 \right) & \text{for } b \leq r \leq d, \\
1 & \text{for } d \leq r.
\end{cases}
\] (14)
In this paper, we introduce a new mathematical extension on the rough intervals. In the earlier research papers, rough intervals are taken as three subintervals (Figure 1 and Figure 2) but in our present investigation, we consider a modification/refinement of the rough intervals. Here we consider in Equations (15) and (16) five subintervals on the rough intervals (Figure 3 and Figure 4).
Figure 4. $\text{Tr}\{\hat{\xi} \leq r\}$ function curve.

If the interval is divided into more regions, then the trust values of $\hat{\xi}$ are a rough variable given by $\hat{\xi} = ([a, b], [c, d]), 0 \leq c \leq c_1 \leq a \leq b \leq c_2 \leq d$ is

$$
\begin{align*}
\text{Tr}\{\hat{\xi} \leq r\} &= \begin{cases} 
0, & \text{for } d \leq r, \\
\frac{(d - r)}{3(d - c)}, & \text{for } c_2 \leq r \leq d, \\
\frac{(d - r)}{3(d - c)} + \frac{(c_2 - r)}{3(c_2 - c_1)}, & \text{for } b \leq r \leq c_2, \\
\frac{1}{3} \left( \frac{(d - r)}{d - c} + \frac{(c_2 - r)}{c_2 - c_1} + \frac{(b - r)}{b - a} \right), & \text{for } a \leq r \leq b, \\
\frac{1}{3} \left( \frac{(d - r)}{d - c} + \frac{(c_2 - c_1)}{c_2 - c_1} + 1 \right), & \text{for } c_1 \leq r \leq a, \\
\frac{1}{3} \left( \frac{(d - r)}{d - c} + 2 \right), & \text{for } c \leq r \leq c_1, \\
1, & \text{for } r \leq c.
\end{cases}
\end{align*}
$$

(15)

$$
\begin{align*}
\text{Tr}\{\hat{\xi} \geq r\} &= \begin{cases} 
0, & \text{for } r \leq c, \\
\frac{(r - c)}{3(d - c)}, & \text{for } c \leq r \leq c_1, \\
\frac{1}{3} \left( \frac{(r - c)}{d - c} + \frac{(r - c_1)}{c_2 - c_1} \right), & \text{for } c_1 \leq r \leq a, \\
\frac{1}{3} \left( \frac{(r - c)}{d - c} + \frac{(r - c_1)}{c_2 - c_1} + \frac{(r - a)}{b - a} \right), & \text{for } a \leq r \leq b, \\
\frac{1}{3} \left( \frac{(r - c)}{d - c} + \frac{(r - c_1)}{c_2 - c_1} + 1 \right), & \text{for } b \leq r \leq c_2, \\
\frac{1}{3} \left( \frac{(r - c)}{d - c} + 2 \right), & \text{for } c_2 \leq r \leq d, \\
1, & \text{for } d \leq r.
\end{cases}
\end{align*}
$$

(16)

In dividing the interval $[c, d]$ into five subintervals instead of three intervals, we get more specific small intervals and define the age of the chromosomes by five linguistic words
in the said intervals. Later identifying the parents by five words, we get several types of their combinations and five types of offsprings, where only three types of children were produced for three subdivisions of \([c, d]\).

In the following section, mathematical formulation of the proposed problem is presented.

3. Problem formulation

3.1. Classical TSP with time constraints (CTSP)

In a classical two-dimensional TSP, a salesman has to travel \(N\) cities at minimum cost. In this tour, salesman starts from a city, visit all the cities exactly once and comes to the starting city using minimum cost. Here time in travelling from one city to another is considered. The salesman should choose such a path in which the maximum allowable time limit is maintained.

Now when the cost and times are interval numbers, then the classical TSP mathematically written as

\[
\text{Minimise} \quad Z = \sum_{i \neq j} \left[ c_{L_{ij}}, c_{R_{ij}} \right] x_{ij}
\]

subject to

\[
\sum_{i=1}^{N} x_{ij} = 1 \quad \text{for } j = 1, 2, \ldots, N,
\]

\[
\sum_{j=1}^{N} x_{ij} = 1 \quad \text{for } i = 1, 2, \ldots, N,
\]

\[
\sum_{i=1}^{N} \sum_{j=1}^{N} \left[ t_{L_{ij}}, t_{R_{ij}} \right] x_{ij} \subseteq \left[ t_{\text{max}L}, t_{\text{max}R} \right],
\]

where \(x_{ij} \neq x_{k}, \quad i, j = 1, 2, \ldots, N.\)

Then the above CTSP reduces to determine a complete tour \((x_1, x_2, \ldots, x_N, x_1)\)

\[
\text{Minimise} \quad Z = \sum_{i=1}^{N-1} \left[ c_{L_{ij+1}}, c_{R_{ij+1}} \right] x_{ij+1} + \left[ c_{L_{N1}}, c_{R_{N1}} \right] x_{N,1}
\]

subject to

\[
\sum_{i=1}^{N-1} \left[ t_{L_{ij+1}}, t_{R_{ij+1}} \right] x_{ij+1} + \left[ t_{L_{N1}}, t_{R_{N1}} \right] x_{N,1} \subseteq \left[ t_{\text{max}L}, t_{\text{max}R} \right],
\]

where \(x_i \neq x_j, \quad i, j = 1, 2, \ldots, N.\)

3.2. STSP with time constraints (CSTSP)

In an STSP, a salesman has to travel \(N\) cities by choosing any one of the \(P\) different types of conveyances available using minimum cost maintaining maximum allowable time. Time taken to travel from one city to another using different conveyances are different. Let \(c(i, j, k)\) be the cost for travelling from \(i\)th city to \(j\)th city using \(k\)th type conveyance and \(t(i, j, k)\) be the time taken for this travel. Then the salesman has to determine a complete tour \((x_1, x_2, \ldots, x_N, x_1)\) and corresponding conveyance types \((v_1, v_2, \ldots, v_P)\) to be used for the tour, where \(x_i \in \{1, 2, \ldots, N\}\) for \(i = 1, 2, \ldots, N\), \(v_i \in \{1, 2, \ldots, P\}\) for \(i = 1, 2, \ldots, N\) and all \(x_i\)s are distinct. Then the problem can be mathematically formulated.
Determine a complete tour \((x_1, x_2, \ldots, x_N, x_1)\) using any one of the available conveyance in each step from the vehicle types \((v_1, v_2, \ldots, v_p)\) so as

\[
\text{to minimise } Z = \sum_{i=1}^{N-1} \left[ c_{L, i+1} + c_{R, i+1} \right] (x_{i, j+1}, v_i) + \left[ c_{L, i+1} + c_{R, i+1} \right] x_{N, 1}, v_i,
\]

subject to \[
\sum_{i=1}^{N-1} \left[ t_{L, i+1} + t_{R, i+1} \right] (x_{i, j+1}, v_i) + \left[ t_{L, i+1} + t_{R, i+1} \right] x_{N, 1}, v_i) \subseteq [t_{\max L}, t_{\max R}],
\]

where \(x_i \neq x_j, \quad i, j = 1, 2, \ldots, N, \quad v_i, v_j \in \{1, 2, \ldots, orP\} \).

### 3.3. STSP using restricted conveyances with time constraints (CSTSPwR)

In real life, it is seen that in all stations, all types of conveyances may not be available due to the geographical position of the station, weather conditions, etc. So it is more realistic that restricted conveyances are available in different stations. Considering the availability of the conveyances, we design the STSP with restricted condition with time constraints as below.

Let \(c(i, j, k)\) be the cost for travelling from \(i\)th city to \(j\)th city using \(k\)th type conveyance and \(t(i, j, k)\) be the time taken in travelling from \(i\)th city to \(j\)th using \(k\)th type conveyance. Then the salesman has to determine a complete tour \((x_1, x_2, \ldots, x_N, x_1)\) and corresponding conveyance types \((v_1, v_2, \ldots, v_5)\) to be used for the tour, where \(x_i \in \{1, 2, \ldots, N\}\) for \(i = 1, 2, \ldots, N\), \(v_i \in \{1, 2, \ldots, 5\}\) for \(i = 1, 2, \ldots, N\) and all \(x_i\)s are distinct. Also \(v_i \in \{1, 2, \ldots, 5\}\) provides maximum available \(S(\leq P)\) types of conveyances. Then the problem can be mathematically formulated.

Determine a complete tour \((x_1, x_2, \ldots, x_N, x_1)\) using any one available corresponding conveyance in each step from the vehicle types \((v_1, v_2, \ldots, v_5)\) so as

\[
\begin{align*}
\text{Minimise} & \quad Z = \sum_{i=1}^{N-1} \left[ c_{L, i+1} + c_{R, i+1} \right] (x_{i, j+1}, v_i) + \left[ c_{L, i+1} + c_{R, i+1} \right] x_{N, 1}, v_i, \\
\text{subject to} & \quad \sum_{i=1}^{N-1} \left[ t_{L, i+1} + t_{R, i+1} \right] (x_{i, j+1}, v_i) + \left[ t_{L, i+1} + t_{R, i+1} \right] x_{N, 1}, v_i) \subseteq [t_{\max L}, t_{\max R}],
\end{align*}
\]

where \(x_i \neq x_j, \quad i, j = 1, 2, \ldots, N, \quad v_i, v_j \in \{1, 2, \ldots, 5\}\).

Thus the above model is written as

\[
\begin{align*}
\text{Minimise} & \quad Z = [C_L, C_R] x_i, \\
\text{subject to} & \quad [T_L, T_R] x_i \subseteq [t_{\max L}, t_{\max R}],
\end{align*}
\]

where \(x_i \neq x_j, \quad i, j = 1, 2, \ldots, N, \quad v_i \in \{v_1, v_2, \ldots, v_5\}\)

and

\[
\begin{align*}
[C_L, C_R] x_i & = \sum_{i=1}^{N-1} \left[ c_{L, i+1} + c_{R, i+1} \right] x_{i+1, j}, v_i + \left[ c_{L, i+1} + c_{R, i+1} \right] x_{N, 1}, v_i, \\
[T_L, T_R] x_i & = \sum_{i=1}^{N-1} \left[ t_{L, i+1} + t_{R, i+1} \right] x_{i+1, j}, v_i + \left[ t_{L, i+1} + t_{R, i+1} \right] x_{N, 1}, v_i.
\end{align*}
\]

The interval valued objective and constraint are transformed as given in Section 2 following Karmakar et al. [44]. Thus the crisp version of the above model is Minimise \(Z = (C_L m_1 \times C_R m_2)^1/(m_1+m_2)\), for \(m_1, m_2 \in (0, 1)\) with one of the following constraints.
According to Moore’s approaches [46]

\[
\begin{align*}
\text{subject to} & \quad t_{\text{maxL}} \leq T_L, \\
& \quad T_R \leq t_{\text{maxR}}.
\end{align*}
\]  

(22)

Ishibuchi and Tanaka’s approaches [46]

\[
\begin{align*}
\text{subject to} & \quad T_L \leq t_{\text{maxL}}, \\
& \quad \frac{(T_L + T_R)}{2} \leq \frac{(t_{\text{maxR}} + t_{\text{maxL}})}{2}.
\end{align*}
\]  

(23)

Chanas and Kuchta’s approaches [46] for \(0 \leq s_0 < s_1 \leq 1,\)

\[
\begin{align*}
\text{subject to} & \quad \frac{(T_L + s_0 \ast (T_R - T_L))}{2} \leq \frac{(t_{\text{maxL}} + s_0 \ast (t_{\text{maxR}} - t_{\text{maxL}}))}{2}, \\
& \quad \frac{(T_L + s_0 \ast (T_R - T_L) + T_L + s_1 \ast (T_R - T_L))}{2} \leq \frac{(t_{\text{maxL}} + s_0 \ast (t_{\text{maxR}} - t_{\text{maxL}}) + t_{\text{maxL}} + s_1 \ast (t_{\text{maxR}} - t_{\text{maxL}}))}{2}.
\end{align*}
\]  

(24)

Hu and Wang’s approaches [46]

\[
\begin{align*}
\text{subject to} & \quad \frac{(T_L + T_R)}{2} \leq \frac{(t_{\text{maxR}} + t_{\text{maxL}})}{2}, \\
& \quad (t_{\text{maxR}} - t_{\text{maxL}}) \leq (T_R - T_L).
\end{align*}
\]  

(25)

Mahato and Bhunia’s approaches [46]

\[
\begin{align*}
\text{subject to} & \quad (\text{optimistic case}) T_L \leq t_{\text{maxL}}, \\
& \quad (\text{pessimistic case}) \frac{(T_L + T_R)}{2} \leq \frac{(t_{\text{maxR}} + t_{\text{maxL}})}{2}, \quad \text{for Type I & Type II intervals}, \\
& \quad (t_{\text{maxR}} - t_{\text{maxL}}) \leq (T_T), \quad \text{for Type III intervals}.
\end{align*}
\]  

(26)

Now we go to the proposed algorithm.

4. Proposed RSGAs

4.1. Simple genetic algorithm (SGA)

GA provides a general-purpose search methodology, which uses principles inspired by natural genetics to evolve solutions to problems [18,21]. The SGA starts off with a population of randomly generated chromosomes, each representing a candidate solution to the problem and advances towards better chromosomes by applying genetic operators. An SGA for a particular problem must have the procedure such as follows.

4.2. Rough set based genetic algorithms (RSGAs)

Here RSGA with the rough set-based age-dependent selection, min-point crossover and \(p_m\)-dependent random mutation is developed and used, among a set of potential solutions to get a new set of solutions. As usual, it is continued until terminating conditions are encountered.
Algorithm of SGA:
1. Start
2. Set iteration counter \( t = 0 \) and maximum generation \( M = M_0 \)
3. Randomly generate initial population \( p(t) \)
4. Evaluate initial population \( p(t) \)
5. while \( t \leq M \) do
6. \( t = t + 1 \)
7. Select the mating pool
8. for each pair of parents do
9. Perform crossover with probability of crossover \( p_c \)
10. end for
11. Mutate randomly selected chromosome with probability of mutation \( p_m \)
12. Store offspring’s into offspring set
13. Select a percent of better offsprings from the offspring set and insert into \( p(t) \)
14. Evaluate \( p(t) \)
15. Remove all offspring from the offspring set
16. end while
17. End Algorithm

4.2.1. Representation
Here a complete tour on \( N \) cities represents a solution. So an \( N \)-dimensional integer vector \( X_i = (x_{i1}, x_{i2}, \ldots, x_{iN}) \) is used to represent a solution (path), where \( x_{i1}, x_{i2}, \ldots, x_{iN} \) represent \( N \) consecutive cities in a tour. Population size number \( M \), and \( i \)th solution \( X_i = (x_{i1}, x_{i2}, \ldots, x_{iN}) \), where \( x_{i1}, x_{i2}, \ldots, x_{iN} \) are randomly generated by random number generator between 1 to \( N \) with maintaining the TSP conditions such as not repetition of cities (nodes) and also satisfying the constraint. Fitness is evaluated by through summing the cost between the consecutive cities (nodes) of each solution (chromosome). The solution \( f(X_i) \) represents for \( i \)th solution fitness in the solution space. Since the maximum population size \( M \), so \( M \) numbers of solutions (chromosomes) are generated by randomly.

4.2.1.1. Rough set-based age-dependent selection. In Last et al. [34], for the first time, an attempt was made to improve the performance of genetic algorithms by providing a new fuzzy-based extension of the Life Time feature. They used a Fuzzy Logic Controller (FLC) to adopt the crossover probability as a function of the chromosomes age. Also Fdez et al. [36], Roy et al. [35] used it in some refinement of the mechanism on inventory control system. They used the age of the chromosome in fuzzy environment. But, for the first time, we model the age in rough environment as rough set is more uncertain than fuzzy set in the uncertain paradigms. So rough set-based age is more effective. The general principle is that for both young and old individuals, the crossover probability is naturally low, while there is a certain age interval, where this probability is high. The concepts of young, old and middle-aged are modelled as linguistic variables. Again we extend these linguistic variables in rough environment.

Here, the concept of representation of age using linguistic variables in both fuzzy and rough environments are same, but the mathematical representations are different.
Moreover, the classification of linguistic variables by earlier authors [34–36] has been extended in the present analysis.

In our proposed RSGAs, the age of a chromosome is determined based on their fitness values and then a ‘rough set-based age-dependent selection’ (REA) technique is applied. Here the age of each chromosome lies in a region of the common age represented by a rough set. These regions are termed as young, middle, and old for RSGA-1. So for the age of each chromosome, a linguistic value – young, middle, or old – is created. Now according to the age distributions of the members (in pair) of the mating pool, similar linguistic variables such as low, medium, and high are generated for the said chromosomes to fix $p_c$’s. Using the trust measures of rough set, the probability of crossover, $p_c$ for each chromosome is assigned by the corresponding linguistic variables.

Now $M$ solutions in a generation with fitnesses represented by $f(X_i)$ of the $i$th chromosomes. At the time of initialization, each chromosome age is defined as null. Now in every generation the age is counted as using the following mechanism:

$$\text{If } \text{avgfit} > f(X_i), \text{age } (x_i) = \text{avg(age)} + \frac{k \ast (\text{avgfit} - f(X_i))}{(\text{avgfit} - \text{minfit})}$$

$$\text{If } \text{avgfit} \leq f(X_i), \text{age}(x_i) = \frac{\text{avg(age)}}{2} + \frac{k \ast (f(X_i) - \text{avgfit})}{(\text{maxfit} - \text{avgfit})}$$

where avgfit is the average fitness values, maxfit and minfit are maximum and minimum fitness values of the last generation and $k = (\text{maxfit} + \text{minfit}) / 2$. Also avg(age) means the average age of the set of chromosomes. Here the maximum and minimum ages depend on the requirement of the problems.

Now since age calculated as crisp values, we construct the common rough values from it,

$$\text{Rough Age} = ([r_1 \ast \text{avg age}, r_2 \ast \text{avg age}], [r_3 \ast \text{avg age}, r_4 \ast \text{avg age}]),$$

where $r_1 = \frac{\text{Max Age} - \text{Avg Age}}{\text{Avg Age}}$, $r_2 = \frac{\text{Max Age} + \text{Min Age}}{2}$, $r_3 = \frac{\text{Max Age} - \text{Min Age}}{2}$, $r_4 = \frac{\text{Avg Age} - \text{Min Age}}{\text{Avg Age}}$.

According to the age of the chromosome, it belongs to any one of the common rough age defined as Young, Middle, and Old. For common rough age ($[a, b], [c, d]$), it is described as below

$$\text{Age} = \begin{cases} \text{Young}, & \text{for } c \leq \text{age} < a, \\ \text{Middle}, & \text{for } a \leq \text{age} \leq b, \\ \text{Old}, & \text{for } b < \text{age} \leq d. \end{cases} \quad (27)$$

Outcomes of different types of ages are given in Equation 27. The above equation (27) shows that if common rough age region is ($[a, b], [c, d]$), then the space $c$ to $a$ refers to as young age, $a$ to $b$ as middle age and $b$ to $d$ as old age. Also the pictorial representation is given in Figure 5. Then in Table 2, according to the linguistic values, rough trust-based probability of crossovers ($p_c$) is assigned which is shown in Figure 6.
4.2.1.2. Rough extended age-based selection. To have more accurate classification, we make five classifications instead of above three and then, the region of common age is divided into very young, young, middle, old and very old for RSGA-II. As before, combining the eligible parents, the very low, low, medium, high and very high linguistic variables are assigned for $p_c$’s of chromosomes.

Now we consider the age in a different extended linguistic code, i.e. Young, Middle and Old, are replaced by Very Young, Young, Middle, Old and Very Old scales. So it is more realistic in the sense of classification and acceptable to design for the real-world problems. According to the requirement of the five linguistic values, we expanded the trust measure levels in five sections which are shown in Equations (15) and (16). Determined $p_c$ values of the extended linguistics are also given below in Figures 7 and 8.

According to the extended age of the chromosome in Table 3, it belongs to any one of the common rough age intervals such as Very Young, Young, Middle, Old and Very Old. The common rough age $([a, b], [c, d])$ is extended to $0 \leq c \leq c_1 \leq a \leq b \leq c_2 \leq d$ and is
Figure 7. Rough extended age distribution of Interval.

Figure 8. Rough extended age distribution of $p_c$.

Table 3. Rough extended trust based linguistic.

| Chromosomes | Very young | Young | Middle | Old | Very old |
|-------------|------------|-------|--------|-----|----------|
| Very young  | Very low   | Low   | Medium | Low | Very low |
| Young       | Low        | Low   | High   | Low | Very low |
| Middle      | Medium     | High  | Very high | High | Medium |
| Old         | Low        | Low   | High   | Low | Very low |
| Very old    | Very low   | Very low | Medium | Very low | Very low |

described as below:

\[
\text{Age} = \begin{cases} 
\text{VeryYoung}, & \text{for } c \leq \text{age} < c_1, \\
\text{Young}, & \text{for } c_1 \leq \text{age} < a, \\
\text{Middle}, & \text{for } a \leq \text{age} \leq b, \\
\text{Old}, & \text{for } b < \text{age} \leq c_2, \\
\text{VeryOld}, & \text{for } c_2 < \text{age} \leq d.
\end{cases}
\] (28)

4.2.2. Min-point crossover

4.2.2.1. Crossover mechanism. To select two individuals (parents) from the matting pool, generate the random number between $[0,1]$. If $r < p_c$, then select that population for first parent (say $P_{r_1}$). Similarly choose the another parents (say $P_{r_2}$). Let these are $P_{r_1} : a_1, a_2, \ldots, a_N$ and $P_{r_2} : s_1, s_2, \ldots, s_N$. Here $(a_1, a_2, \ldots, a_N)$ and $(s_1, s_2, \ldots, s_N)$ are nodes within $(1, 2,$
3, \ldots, N \), and these are numbers of cities. Then choose a city randomly from 1 to \( N \), say \( a_i = s_p \ (i = 1, 2, \ldots, N) \). Parents are modified by placing \( a_i \) or \( s_p \) in the first place of \( P_r_1 \) and \( P_r_2 \). Now modified parents are given by \( P_r_1 : a_i, a_1, a_2, \ldots, a_i \prec \prec 1, a_i \prec \prec 1, \ldots, a_N, P_r_2 : s_p, s_1, s_2, \ldots, s_p \prec \prec 1, s_p \prec \prec 1, \ldots, s_N \). To get the first child (Ch1), placing \( a_i \) in the first place of Ch1, compare the next route \( a_i \) to \( a_1 \) and \( a_i \) to \( s_1 \), minimum cost route be selected in Ch1. As considered the uncertain interval values, so we determine the values as convex combination of the upper and lower interval values, such as for any route cost between two nodes \( a_i \) to \( a_2 \) becomes as 
\[
c(a_i, a_1) = \lambda \ast CL_{i1} + (1 - \lambda) \ast CR_{i1},
\]
\( \lambda \in \text{rand}[0, 1] \), CL and CR, lower and upper values of the corresponding intervals between two node and then compare with minimum values which are defined as min-point values. The procedure is as follows:

\[
\text{if} \quad (c(a_i, a_1) \prec c(a_i, s_1)) \quad \text{concatenate} \ a_1 \text{in Ch}1. \\
\text{else} \quad \text{concatenate} \ s_1 \text{in Ch}1. \\
\text{Ch}_1 : a_i, s_1 (\text{say}).
\]

Thus by the repetition of this process, find the first child \( \text{Ch}_1 : a_i, s_1, a_1, \ldots, a_N \) (say). Now for the second child, we modifying the parents by placing \( a_i \) or \( s_p \) at the end of \( P_r_1 \) and \( P_r_2 \). The modified parents are given by \( P_r_1 : a_1, a_2, \ldots, a_i \prec \prec 1, a_i \prec \prec 1, \ldots, a_N, a_i \) and \( P_r_2 : s_1, s_2, \ldots, s_p \prec \prec 1, s_p \prec \prec 1, \ldots, s_N, s_p \). For the second child (Ch2), placing \( a_i \) in the first place of Ch2, compare the next route \( a_i \) to \( a_N \) and \( a_i \) to \( s_N \), minimum cost route be selected in Ch2. According to first child it also form second child \( \text{Ch}_2 : a_i, s_N, a_i, \ldots, a_1 \) (say). Hence by the above mechanism, using two modified parents \( (P_{r1}) \) and \( (P_{r2}) \), find the two children (offspring) \( \text{Ch}_1 \) and \( \text{Ch}_2 \).

4.2.3. Three different forms of \( p_m \) dependent random mutations

4.2.3.1. Selection for mutation. For each solution of \( P(t) \), generate a random number \( r \) from the range \([0, 1]\). If \( r < p_m \), then the solution is taken for mutation where \( p_m \) be the probability of mutation.

4.2.3.2. Mutation process. At first determined the total number of mutated node \( (T) \). To mutate a solution \( X = (x_1, x_2, \ldots, x_N) \), number of mutated node \( T = p_m \ast N, N = \text{total number of nodes} \).

(i) Random location method (Type I): Generate two different integer randomly between \([1, N]\). Interchange the nodes \( x_i, x_j \) according to generate two random integers up-to \( T \) times to get new solutions which replace the parent solution.

(ii) Fixed location method (Type II): If \( T \) becomes even then selected \( T \) consecutive numbers of node in a solution \( X = (x_1, x_2, \ldots, x_N) \) and select any of the two nodes \( x_i \) and \( x_j \) and interchange their places. So here change is done up to \( T/2 \) times not generating any random number. On the other hand, if \( T \) becomes odd, then similarly interchanges the places of the solutions up to \( (T/2) + 1 \) times.
(iii) **Generation oriented mutation (variable method):** Here we model a new form of mutation mechanism where probability of mutation \( p_m \) is determined as follows:

\[
p_m = \frac{k}{\text{Current generation number}}, \quad k \in [0, 1].
\]

So, here proposed mutation mechanism follows the real-world demand and \( p_m \) decreases smoothly as generation may increase. After calculating the \( p_m \), then mutation operation works both of the two methods, Type I and Type II.

### 4.3. Algorithm for RSGA

**Input:** max_gen, pop_size, Max_age, Min_age, Problem Data (cost matrix, risk matrix).

**Output:** The optimum and near optimum solutions.

1. **Start**
2. \( g \leftarrow 0 \) // \( g \): iteration/generation number,
3. **Initialise** \( P(g) \) //According to representation in Section 4.2.1, randomly generate initial population \( P(g) \) for generation \( g \),
4. **Evaluate** \( f(P(g)) \); //Evaluate fitness of each chromosome according to Section 4.2.2a initial population \( P(g) \).
5. **while** \( (g \leq \text{max}_\text{gen}) \)
6. Evaluate the average fitness,
7. **if** average fitness > current fitness,
8. \( \text{age}(x_i) = \text{avg}(\text{age}) + \frac{k(\text{avgfit} - f(X_i))}{(\text{avgfit} - \text{minfit})} \),
9. **else**
10. \( \text{age}(x_i) = \frac{\text{avg}(\text{age})}{2} + \frac{k(f(X_i) - \text{avgfit})}{(\text{maxfit} - \text{avgfit})} \),
11. **if** \( \text{age}(x_i) > \text{maximum age} \)
12. \( \text{age}(x_i) = \text{maximum age} \),
13. **else if** \( \text{age}(x_i) < \text{minimum age} \)
14. \( \text{age}(x_i) = \text{minimum age} \),
15. Determine average age,
16. Determine common rough age,
17. Switch (Choice)
18. **Case I:** // RSGA-I
19. (a) Developed linguistic variables young, middle, old,
20. (b) **for** each pair of parents **do**
21. (c) Trust based \( p_c \) created,
22. (d) **end for**
23. **Case II:** // RSGA-II
24. (a) Developed variables very young, young, middle, old, very old,
25. (b) **for** each pair of parents **do**
26. (c) Extended trust based \( p_c \) created,
27. (d) **end for**// end switch,
28. **for** \( i = 1 \) to Pop Size // **min-point crossover**
29. Choose pair of chromosomes according to \( p_c \),
30. Randomly generate node between 1 and \( N \) (say \( a_r \)),
31. Replace \( a_r \) at first place of each parents chromosomes,
Determine min-point value of each corresponding node,
for \( j = 1 \) to \( N \),
Compare min-point value,
Check the existence of corresponding node in child,
Concatenated node to the child (offspring),
end for
Replace \( a_r \) at end place of each parents chromosomes,
Compare min-point value from end of the each next nodes,
for \( j = 1 \) to \( N \),
Compare min-point value,
Check the existence of corresponding node in child,
Concatenated node to the child (offspring),
end for
Replace the child’s in offspring’s set,
end for
Switch (Choice) // Mutation
Case-I (simple):
(a) for \( i = 0 \) to pop_size 
(b) Select chromosome depending \( p_m \),
(c) Randomly select two different nodes between \([1, N]\)
(d) Replace the places of the selected two nodes,
end for
Case-II(variable):
(a) \( p_m = \frac{k}{g}, k \in [0, 1]\),
(b) Determine \( T = p_m * N \) // total number of mutated node,
(c) for \( i = 0 \) to pop_size 
(d) Select chromosome depending \( p_m \),
(e) for \( j = 1 \) to \( T \) // Type-I 
(f) Randomly select two different nodes between \([1, N]\)
(g) Replace the places of the selected two nodes,
end for
end for
Case-III(variable):
(a) \( p_m = \frac{k}{g}, k \in [0, 1]\),
(b) Determine \( T = p_m * N \),
(c) for \( i = 0 \) to pop_size 
(d) Select chromosome depending \( p_m \),
(e) for \( j = 1 \) to \( \frac{T}{2} \) or \( (\frac{T}{2} + 1) \) // According \( T \) even or odd (Type-II),
(f) Replace the places of the any two nodes,
end for
end for
Store the new off springs into offspring set,
Reproduce a new \( P(g) \),
Evaluate \( f(P(g)) \); // evaluate the fitness of reproduce chrom.,
Store the local optimum and near optimum solutions,
g ← g+1,
end while
49. Store the global optimum and near optimum results,
50. End Algorithm.

Flow chart of this algorithm is depicted in Figure 9.

4.4. Termination criteria
RSGA-I (Rough set based) and RSGA-II (Rough extended set based) algorithms are terminated by any one of the following conditions is satisfied (which over is earlier):

(a) the best solution does not improve within 20 consecutive generations
(b) number of generations reaches user-defined iterations (generations).

The same termination criteria are used for SGA, SGA-I, II, III, IV, V and FGA which are different combinations of the GA operators presented in Table 5.

Some numerical experiments are done in the following section.

5. Numerical examples

5.1. Testing with problems from TSPLIB [42]
To validate the feasibility and effectiveness of the proposed algorithms, we apply the proposed RSGAs on some standard TSP problems taken from TSPLIB [42]. The proposed algorithm was implemented in C++ with following parameters as 100 chromosomes and 2000 iterations (maximum). The best optimal results are presented.

5.1.1. Comparison of results of test problems by RSGA-II and SGA
Table 4 gives the results of the test problems using RSGA-II and SGA, the results are compared in terms of optimal cost, iterations and computational time (CPU time in minutes) also individual run of each algorithm are presented. It is seen that the number of iterations and computational times are less in RSGA-II than SGA. In each instance, average result (Avg), best found results (Cost) and the SD are presented. Here BKS stands for as “best known solution” in the literature.

5.1.2. Comparison of results of RSGAs with respect to different types of operators
Moreover, for a particular test problem bayg29, both SGA and proposed RSGAs are used with different operators and parameters ($p_c$’s, $p_m$’s, $p_s$’s). The obtained results are presented in Tables 5 and 6. Here in each case 25 individual run with best found result 1610 is considered.

In Table 5, we survey the importance of different types of selection, crossover and mutation parameters in proposed algorithms. It indicates that for the optimal solution of the standard TSP data bayg29, optimal result is found in the rough extended age-based selection mechanism with min point crossover and fixed mutation. These results are obtained quickly by 64 iterations only. Here also, RSGAs perform better than SGA. In this testing,
Figure 9. Flowchart of RSGA.
Table 4. Results for standard TSP problem (TSPLIB).

| Instances | BKS | Cost Avg SD | Iter. | Time | Run |
|-----------|-----|-------------|-------|------|-----|
| fri26     | 937 | 939.49      | 1.64  | 2020 | 67  | 0.23 | 25 |
|           | 2023 | 1.87        |       |      |     |      |    |
|           | 1610 | 0.78        |       |      |     |      |    |
|           | 699  | 1.46        |       |      |     |      |    |
| bays29    | 2020 | 939.49      | 25    | 1.64 | 426 | 98  | 1.78 | 40 |
|           | 2021.3 | 6.63       |       |      |     |      |    |
| bayg29    | 1610 | 700.53      | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
|           | 699  | 426.86      |       |      |     |      |    |
| dantzig42 | 699  | 700.53      | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
| eil51     | 426  | 700.53      | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
|           | 426.86 | 6.63       |       |      |     |      |    |
| berlin52  | 7542 | 7556.15     | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
|           | 675  | 687.21      |       |      |     |      |    |
| st70      | 675  | 538         | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
|           | 538  | 539.16      |       |      |     |      |    |
| eil76     | 538  | 108159      | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
|           | 1211 | 1211.7      |       |      |     |      |    |
|           | 21282 | 2.79      |       |      |     |      |    |
| pr76      | 108159 | 675        | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
|           | 108201.17 | 2.42    |       |      |     |      |    |
|           | 1211 | 1211.7      |       |      |     |      |    |
| rat99     | 1211 | 21282       | 25    | 6.53 | 123 | 141 | 1.41 | 25 |
|           | 21282 | 2.79      |       |      |     |      |    |
| kroa100   | 21282 | 21289.34   | 25    | 6.53 | 123 | 141 | 1.41 | 25 |

Table 5. Comparison of RSGAs and SGAs for bayg29 with different parameters.

| Algorithm | Selection | Crossover | Generation | p_c | p_m | p_l | Avg | SD |
|-----------|-----------|-----------|------------|-----|-----|-----|-----|----|
| SGA-I     | Roulette Wheel | Cyclic     | 678        | 0.3 | 0.4 | —   | 1603.31 | 2.37 |
| SGA-II    | Probabilistic   | Cyclic     | 309        | 0.31| 0.4 | 0.3 | 1605.8 | 1.83 |
| SGA-III   | Probabilistic   | Comparison | 256        | 0.4 | 0.4 | —   | 1604.72 | 3.17 |
| SGA-IV    | Probabilistic   | Comparison | 176        | 0.4 | 0.4 | 0.3 | 1607.81 | 1.54 |
| RSGA-I    | Rough age based | Min point  | 66         | 0.4 | 0.4 | —   | 1609.21 | 0.94 |
| RSGA-II   | Rough extended age based | Min point | 64         | 0.4 | 0.4 | —   | 1609.76 | 0.78 |
| SGA-V     | Roulette Wheel  | Min point  | 211        | 0.4 | 0.4 | —   | 1608.32 | 2.05 |
| SGA-I     | Roulette Wheel  | Cyclic     | 411        | 0.5 | 0.4 | —   | 1605.54 | 2.39 |

“probabilistic” selection takes less generations than that required for “Roulette Wheel” selection. Again keeping every thing same, with RW selection, higher value of p_c requires more number of generations and hence it is undesirable.

In Table 6, optimum results for the standard TSP, “bayg29” are obtained in different environments using different selection and mutation techniques. It is observed that though all approaches furnish the same optimum result, the RSGAs with Min-point crossover and
### Table 6. Comparison of RSGAs for bayg29 with different types of mutations.

| Algorithm   | Selection | Crossover | Mutation | Generation | \( p_m \) | Avg | SD  |
|-------------|-----------|-----------|----------|------------|-----------|-----|-----|
| Simple      | 753       | 0.4       | 1617.34  | 0.78       |
| Random      | 598       | 0.3       | 1617.92  | 1.09       |
| Rough Age   | 634       | 0.2       | 1616.54  | 0.85       |
| Based Min   | 256       | 0.4       | 1613.46  | 0.75       |
| Fixed       | 145       | 0.3       | 1613.43  | 0.91       |
| Rough Age   | 98        | 0.2       | 1612.63  | 1.82       |
| Based Min   | 66        | 0.4       | 1613.21  | 0.77       |
| Fixed       | 71        | 0.3       | 1612.52  | 0.97       |
| Rough Age   | 87        | 0.2       | 1617.78  | 1.03       |
| RSGA-I      |           |           |          |            |           |     |     |
| Variable    | 47        |           |          |            |           |     |     |
| Fixed       | 664       | 0.4       | 1617.53  | 1.28       |
| Simple      | 564       | 0.3       | 1617.05  | 0.59       |
| Fixed       | 605       | 0.2       | 1617.03  | 0.96       |
| Random      | 234       | 0.4       | 1615.57  | 0.67       |
| Rough Age   |           |           |          |            |           |     |     |
| Based Min   | 121       | 0.3       | 1613.85  | 0.94       |
| Fixed       | 64        | 0.4       | 1611.34  | 1.11       |
| RSGA-II     |           |           |          |            |           |     |     |
| Variable    | 47        |           |          |            |           |     |     |
| Fixed       | 664       | 0.4       | 1617.53  | 1.28       |
| Simple      | 564       | 0.3       | 1617.05  | 0.59       |
| Fixed       | 605       | 0.2       | 1617.03  | 0.96       |
| Random      | 234       | 0.4       | 1615.57  | 0.67       |

**Note:** RSGAs with “simple” mutation require maximum numbers of generations for optimum results, whereas “random” and “fixed” mutations take the value in between these numbers.

### Table 7. Input data: interval CTSP.

| \( i/j \) | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 |
|-----------|----|----|----|----|----|----|----|----|----|----|
| Crisp cost matrix (10 \( \times \) 10) |    |    |    |    |    |    |    |    |    |    |
| 1         | \( \infty \) | [35, 39] | [18, 23] | [20, 27] | [17, 19] | [36, 45] | [37, 42] | [42, 49] | [33, 34] | [44, 48] |
| 2         | [24, 30] | \( \infty \) | [20, 26] | [28, 32] | [35, 39] | [40, 44] | [30, 36] | [43, 47] | [28, 34] | [14, 16] |
| 3         | [38, 42] | [27, 34] | \( \infty \) | [25, 28] | [22, 26] | [35, 36] | [9, 13] | [32, 35] | [40, 42] | [30, 33] |
| 4         | [28, 32] | [10, 14] | [7, 12] | \( \infty \) | [20, 22] | [25, 28] | [30, 33] | [35, 39] | [22, 25] | [37, 42] |
| 5         | [27, 32] | [22, 26] | [35, 38] | [30, 33] | \( \infty \) | [20, 24] | [25, 30] | [30, 33] | [9, 13] | [28, 33] |
| 6         | [15, 17] | [30, 33] | [25, 30] | [8, 12] | [28, 30] | \( \infty \) | [33, 36] | [40, 44] | [32, 34] | [30, 36] |
| 7         | [38, 44] | [25, 32] | [30, 33] | [22, 24] | [37, 39] | [40, 44] | \( \infty \) | [32, 35] | [20, 22] | [25, 27] |
| 8         | [40, 45] | [5, 9] | [32, 35] | [40, 44] | [35, 38] | [25, 26] | [40, 44] | \( \infty \) | [37, 39] | [38, 42] |
| 9         | [40, 42] | [40, 46] | [23, 26] | [25, 29] | [20, 25] | [2, 5] | [37, 45] | [32, 35] | \( \infty \) | [28, 34] |
| 10        | [28, 33] | [30, 34] | [28, 32] | [20, 25] | [11, 15] | [32, 36] | [37, 39] | [40, 44] | [30, 34] | \( \infty \) |

**Interval time matrix**

| \( i/j \) | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 |
|-----------|----|----|----|----|----|----|----|----|----|----|
| 1         | \( \infty \) | [5, 53] | [8, 9] | [7, 79] | [82, 9] | [59, 64] | [58, 6] | [59, 62] | [6, 64] | [57, 6] |
| 2         | [78, 84] | [81, 88] | [75, 8] | [5, 56] | [6, 64] | [7, 76] | [58, 62] | [75, 79] | [9, 92] | [7, 76] |
| 3         | [59, 63] | [79, 82] | \( \infty \) | [85, 87] | [78, 84] | [65, 7] | [81, 86] | [68, 72] | [6, 64] | [7, 76] |
| 4         | [72, 76] | [9, 92] | [94, 95] | \( \infty \) | [8, 84] | [75, 8] | [7, 76] | [65, 66] | [78, 8] | [63, 7] |
| 5         | [83, 88] | [79, 86] | [69, 74] | [72, 74] | \( \infty \) | [82, 88] | [79, 82] | [71, 73] | [9, 92] | [72, 74] |
| 6         | [88, 9] | [7, 74] | [75, 77] | [91, 92] | [72, 75] | \( \infty \) | [67, 69] | [6, 7] | [7, 73] | [77, 8] |
| 7         | [68, 7] | [59, 6] | [8, 84] | [7, 73] | [6, 65] | [61, 65] | \( \infty \) | [68, 7] | [8, 83] | [77, 8] |
| 8         | [6, 64] | [94, 95] | [69, 73] | [6, 63] | [59, 62] | [79, 81] | \( \infty \) | [59, 63] | [73, 76] | [72, 74] |
| 9         | [6, 63] | [81, 83] | [77, 8] | [75, 78] | [8, 82] | [9, 99] | [63, 65] | [68, 7] | \( \infty \) | [72, 74] |
| 10        | [85, 9] | [8, 76] | [73, 75] | [53, 55] | [9, 96] | [69, 73] | [64, 66] | [59, 63] | [7, 74] | \( \infty \) |
Table 8. Optimum results of CTSP in crisp environment.

| Algorithm | Selection | Mutation | Path                  | Gen | Value     | Avg   | SD   | $T_{\text{max}}$ |
|-----------|-----------|----------|-----------------------|-----|-----------|-------|------|-----------------|
| Rough     | Random    | 6-4-3-7-10-5-9-8-2-1 | 87  | [130, 166] | 146.21 | 1.21 | T_{\text{max}} |
| (RSGA-I)  | Variable  | 6-4-3-7-10-5-9-8-2-1 | 71  | [130, 166] | 147.53 | 0.92 | Without         |
| RSGA      | Rough     | 5-7-2-10-3-4-6-9-1-8 | 59  | [130, 166] | 108.75 | 1.45 | T_{\text{max}} |
| (RSGA-II) | Variable  | 5-7-2-10-3-4-6-9-1-8 | 28  | [92, 127]  | 104.32 | 0.82 | T_{\text{max}} |
| Rough     | Random    | 8-2-10-5-9-6-1-4-3-7 | 92  | [155, 182] | 164.35 | 0.57 | T_{\text{max}} |
| (RSGA-I)  | Variable  | 9-5-6-4-3-7-10-8-2-1 | 32  | [143, 165] | 153.37 | 0.66 | [7.0, 8.25]    |
| RSGA      | Rough     | 6-8-2-10-4-3-7-9-1-5 | 76  | [149, 169] | 157.56 | 0.89 | T_{\text{max}} |
| (RSGA-II) | Variable  | 6-8-2-10-4-3-7-9-1-5 | 53  | [149, 169] | 157.8  | 0.56 | T_{\text{max}} |
| SGA-I     | RW        | 10-8-2-5-9-6-1-4-3-7 | 188 | [192, 259] | 223.43 | 0.77 | [7, 7.75]     |
| RSGA-II   | REA       | 4-1-2-5-9-6-10-8-3-7 | 48  | [165, 189] | 174.45 | 1.22 | [6.5, 7.5]    |
| SGA-I     | RW        | 1-2-5-10-4-3-7-9-6-4 | 231 | [232, 354] | 290.7  | 0.99 | [6.5, 7.5]    |
| RSGA-II   | REA       | 7-2-6-9-1-4-8-5-10-3 | 45  | [272, 312] | 287.17 | 0.97 | [5.5, 6.75]   |

5.2. Experiments for CTSP with and without time constraint

Here we consider a deterministic TSP of 10×10 size given by Equation (20), whose cost and time matrices are given in Table 7.

For the above input data, the problem given by Equation 22 is solved by RSGAs and SGAs and the optimum results are presented in Table 8. Here we took maximum generation = 100, independent run of each algorithms and deterministic constraint obtained using only Moore approaches. In all cases, 20 individual run done.

From Table 8, it is observed that for the cases of CTSP having “without $T_{\text{max}}$” and same “$T_{\text{max}}$”, the better results are obtained with respect to the cost as well as number of generations by RSGA-II and variable mutation. Also when $T_{\text{max}}$ decreases, corresponding cost increases according to the realistic conditions of the present investigation.

5.3. Experiment for CSTSPwR with time constraint

Now for a CSTSPwR, we consider maximum available three types of conveyances. The cost and time matrices for the CSTSPwR of 10×10 size are presented below in Table 9.

Here we took maximum generation as 200 with 20 independent runs, and for proposed RSGA-II, REA selection with variable mutation is used. Only feasible constraints that are satisfied only in the corresponding interval are considered. The optimum results are presented in Table 10.

Comparing the corresponding results from the above table, we see that Table 10 supports the usual expectation, i.e. as the total travel times decrease, the corresponding costs increase for all approaches. Here, the evaluated costs are of dominated types expect one or two, i.e. lower and upper values of the cost intervals with higher $T_{\text{max}}$ are larger than the corresponding values with lower $T_{\text{max}}$. Considering all the approaches, Moore’s approach with RSGA-II furnishes the lowest travel cost. This cost is much less than the corresponding costs with SGA-I for both values of $T_{\text{max}}$. 
Table 9. Input data: CSTSPwR.

| i/j | 1     | 2     | 3     | 4     | 5     | 6     | 7     | 8     | 9     | 10    |
|-----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 1   | ∞     | 32.35 | [17.19] | [17.21] | [29.30] | [5.7] | [15.18] | [25.29] | [39.42] | [20.23] |
| 2   | [34.38] | ∞     | [40.44] | [16.19] | [32.37] | [39.41] | [39.41] | [39.41] | [39.41] | [39.41] |
| 3   | [36.39] | [16.20] | ∞     | [10.14] | [40.45] | [33.36] | [17.20] | [30.33] | [28.32] | [29.31] |
| 4   | [18.21] | [19.23] | [7.10] | ∞     | [17.20] | [15.18] | [30.32] | [32.32] | [20.24] | [47.49] |
| 5   | [9.12] | [12.15] | [27.30] | [23.24] | [25.28] | [30.33] | [16.18] | [32.35] | [37.40] |       |
| 6   | [16.19] | [41.44] | [34.37] | [17.21] | [29.34] | [42.46] | [27.30] | [20.24] | [22.23] | [26.29] |
| 7   | [14.18] | [21.24] | [35.37] | [12.14] | ∞     | [20.23] | [14.18] | [30.32] | [8.11]   | [25.27] |
| 8   | [6.9]  | [32.34] | [33.39] | [40.44] | [40.44] | [25.27] | [12.16] | [7.9]  | [25.28] |       |
| 9   | [13.16] | [26.30] | [4.6]  | [6.9]  | [26.29] | ∞     | [31.34] | [39.42] | [30.33] | [28.31] |
| 10  | [5.8]  | [20.23] | [25.27] | [7.11]  | [26.30] | ∞     | [40.44] | [30.31] | [22.23] | [40.42] |
|     | [5.8]  | [27.30] | [27.30] | [10.13] | [38.41] | [23.26] | [20.23] | [35.36] | [30.34] |       |
|     | [36.39] | [23.26] | [27.32] | [21.24] | [35.38] | [38.41] | [7.11] | [31.34] | [19.22] |       |
|     | [37.40] | [53.55] | [37.39] | [40.44] | [56.60] | [20.22] | ∞     | [40.44] | [33.35] | [13.16] |
|     | [38.32] | [25.27] | [24.27] | [23.25] | [37.40] | [43.46] | [11.14] | [34.37] | [25.28] |       |
|     | [39.42] | [24.28] | [30.33] | [38.42] | [34.37] | [23.26] | [39.42] | [20.23] | [35.38] |       |
|     | [41.43] | [5.7]  | [52.54] | [19.22] | [34.37] | [15.18] | [19.22] | ∞     | [52.54] | [35.38] |
|     | [20.24] | [16.18] | [43.46] | [40.43] | [46.48] | [46.48] | [46.48] | [46.48] | [46.48] | [46.48] |
|     | [38.41] | [39.42] | [4.9]  | [23.26] | [20.23] | [22.25] | [5.8]  | [30.33] | [27.30] |       |
|     | [10.13] | [38.40] | [34.37] | [33.36] | [31.33] | [31.34] | [36.39] | [32.34] | ∞     | [18.20] |
|     | [31.33] | [34.37] | [36.39] | [28.30] | [20.22] | [23.26] | [38.41] | [11.15] | [24.27] |       |
|     | [15.18] | [28.31] | [26.29] | [18.21] | [9.12]  | [30.34] | [35.39] | [40.43] | [29.32] |       |
|     | [25.30] | [28.20] | [18.20] | [29.32] | [32.34] | [10.13] | [26.29] | [41.43] | [51.54] | ∞     |
|     |       |       |       |       |       |       |       |       |       |       |

Cost matrix (10 × 10) with three conveyances

|     |     |     |     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

Timematrix(10×10) with maximum available three conveyances

|     |     |     |     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|

Inputdata: CSTSPwR.
Table 10. Results of CSTSPwR.

| Algorithm | Optimum path (vehicle) | Cost | Avg | SD | Approaches |
|-----------|------------------------|------|-----|----|------------|
| 1(1)-10(1)-5(2)-4(1)-2(1)-9(1)-3(2)-7(1)-8(3)-6(2) | [98,112] | 104.37 | 1.26 | Moore |
| 9(1)-7(1)-8(1)-6(2)-1(1)-3(1)-4(1)-2(1)-10(1)-5(3) | [120,136] | 127.68 | 1.17 | Ishibuchi & Tanka |
| 8(3)-2(2)-10(1)-9(1)-6(2)-4(2)-3(3)-7(2)-5(2)-1(3) | [116,132] | 121.32 | 0.95 | Chanas & Kuchta’s |
| 7(1)-8(1)-6(2)-1(1)-10(3)-5(2)-4(3)-2(1)-10(1)-5(3) | [128,156] | 141.23 | 1.67 | Hu & Wang’s |
| 6(1)-4(3)-2(1)-9(1)-6(2)-1(1)-3(2)-7(1)-8(3)-6(2) | [116,132] | 127.68 | 1.17 | Ishibuchi & Tanka |
| 5.4. CSTSPwR with virtual data set (for large TSP) |

For the large-scale data set, we randomly generate the costs within a range. Here, costs $c_{Lij}$ and $c_{Rij}$ ($i \neq j$) are taken for second and third conveyances respectively as follows:

$c_{Lij} = 10(1 + \text{random integer on}[0,8]),$ $c_{Rij} = c_{Lij} + 0.5(1 + \text{random integer on}[0,8])$

$c_{Lij} = 9(1 + \text{random integer on}[0,8]),$ $c_{Rij} = c_{Lij} + 0.5(1 + \text{random integer on}[0,8])$

$c_{Lij} = 11(1 + \text{random integer on}[0,8]),$ $c_{Rij} = c_{Lij} + 0.5(1 + \text{random integer on}[0,8])$

Similarly randomly generated time matrix for three conveyances is as follows:

$t_{Lij} = 0.25(1 + \text{random number on}[0,1]),$ $t_{Rij} = t_{Lij} + 0.15(1 + \text{random number on}[0,1])$

$t_{Lij} = 0.2(1 + \text{random number on}[0,1]),$ $t_{Rij} = t_{Lij} + 0.12(1 + \text{random number on}[0,1])$

$t_{Lij} = 0.3(1 + \text{random number on}[0,1]),$ $t_{Rij} = t_{Lij} + 0.18(1 + \text{random number on}[0,1])$

Data set are randomly generated using rand() function of C programming language. For the CSTSPwR, cost and time matrices for different size problems ($N = 20, N = 40, N = 60, N = 80, N = 100, N = 150$ and $N = 200$) are generated randomly. For the results, we considered the CSTSPwR formulations in Equations (22)–(26) and solved using RSGA-II. For comparison, SGA-I with Moore’s approach (Equation 22) only is taken. The optimum results of this randomly generated CSTSPwRs are presented in Table 11.

From Table 11, it is evident that for all sizes of CSTSPwR, Moore’s approach gives the best results, i.e. minimum cost is less than the other four approaches. Moreover, with the sizes of the cost matrices, the optimum cost increases which is obvious and expected. For comparison, here SGA-I with Moore’s approach is used and it is seen that RSGA-II fetches much better results than SGA-I for all sizes of cost matrix. SGA-I gives the worst result than all other approaches for a particular size of cost matrix.

In next, some statistical tests are done.

6. Statistical test for RSGAs

6.1. Dispersion tests for RSGA-II

6.1.1. Against different test problems only

Performance of the proposed method is statistically tested running it 25 times and calculating the average value, standard deviation and percentage relative error according to optimal solution against some standard test problems. The results obtained by the
Table 11. Results of CSTSPwR with larger number of cities.

| Instances | Algorithm  | Approaches                        | Costs         | $T_{\text{max}}$ |
|-----------|------------|-----------------------------------|---------------|-----------------|
| 20 x 20   | RSGA-II    | Chanas and Kuchta’s               | [267, 308]    | [14.4, 21.8]    |
|           |            | Moore                             | [282, 320]    |                 |
|           |            | Hu and Wang’s                     | [280, 315.6]  |                 |
|           |            | Mahato and Bhunia’s               |               |                 |
|           | SGA-I      | Moore                             | [328, 360]    | [21.25, 34.7]   |
|           |            | Moore                             | [466, 513]    |                 |
|           |            | Ishibuchi                         | [490, 567]    |                 |
| 40 x 40   | RSGA-II    | Chanas and Kuchta’s               | [546, 587]    | [19.7, 28.4]    |
|           |            | Moore                             | [544, 593.5]  |                 |
|           |            | Hu and Wang’s                     | [576, 632.5]  |                 |
|           |            | Mahato and Bhunia’s               |               |                 |
|           | SGA-I      | Moore                             | [598, 648]    | [23.7, 27.3]    |
|           |            | Moore                             | [690, 810]    |                 |
|           |            | Ishibuchi                         | [719, 842]    |                 |
| 60 x 60   | RSGA-II    | Chanas and Kuchta’s               | [765, 896]    | [23.5, 33.6]    |
|           |            | Moore                             | [784, 867]    |                 |
|           |            | Hu and Wang’s                     | [793, 923]    |                 |
|           |            | Mahato and Bhunia’s               |               |                 |
|           | SGA-I      | Moore                             | [839, 978]    | [24.25, 34.5]   |
|           |            | Moore                             | [1042, 1224]  |                 |
|           |            | Ishibuchi                         | [1196, 1376]  |                 |
| 80 x 80   | RSGA-II    | Chanas and Kuchta’s               | [1264, 1433]  | [26.5, 64.7]    |
|           |            | Moore                             | [1335, 1479]  |                 |
|           |            | Hu and Wang’s                     | [1273, 1452]  |                 |
|           |            | Mahato and Bhunia’s               |               |                 |
|           | SGA-I      | Moore                             | [1598, 1864]  | [41.25, 62.7]   |
|           |            | Moore                             | [1425, 1663]  |                 |
|           |            | Ishibuchi                         | [1562, 1678]  |                 |
| 100 x 100 | RSGA-II    | Chanas and Kuchta’s               | [1647, 1783]  | [46.5, 77.8]    |
|           |            | Moore                             | [1724, 1890]  |                 |
|           |            | Hu and Wang’s                     | [1738, 1975]  |                 |
|           |            | Mahato and Bhunia’s               |               |                 |
|           | SGA-I      | Moore                             | [1983, 2251]  | [64.2, 76.4]    |
|           |            | Moore                             | [2042, 2490]  |                 |
|           |            | Ishibuchi                         | [2368, 2789]  |                 |
| 150 x 150 | RSGA-II    | Chanas and Kuchta’s               | [2945, 3272]  | [70.6, 121.7]   |
|           |            | Moore                             | [3243, 3364]  |                 |
|           |            | Hu and Wang’s                     | [2173, 2505]  |                 |
|           |            | Mahato and Bhunia’s               |               |                 |
|           | SGA-I      | Moore                             | [3581, 3743]  | [91.25, 135.2]  |
|           |            | Moore                             | [2724, 3602]  |                 |
|           |            | Ishibuchi                         | [3192, 3945]  |                 |
| 200 x 200 | RSGA-II    | Chanas and Kuchta’s               | [3347, 4178]  | [104.3, 176.2]  |
|           |            | Moore                             | [3441, 4732]  |                 |
|           |            | Hu and Wang’s                     | [3703, 4926]  |                 |
|           |            | Mahato and Bhunia’s               |               |                 |
|           | SGA-I      | Moore                             | [4398, 5231]  | [141.5, 188.9]  |

The percentage relative error is defined as

$$\text{Error(\%)} = \frac{\text{(average solution} - \text{best known solution})}{\text{best known solution}} \times 100.$$  

Here BKS stands for best known solution. Examining Table 12, it is concluded that the proposed method, RSGA-II has generated the closer results to the optimal solutions with minimal standard deviations for the problems fri26, bays29, bayg29, dantzig42, eil51, berlin52, st70, pr76, rat99, Lin105 and Eil101. It can be seen that except one problem kroa200, for all other 14 problems, best results by RSGA-II are the same as the corresponding best results in the literature.
Table 12. Dispersion results of RSGA-II for different test problems.

| Instances | BKS | Best | Worst | Average | SDb | Error (%) |
|-----------|-----|------|-------|---------|-----|-----------|
| fri26     | 937 | 937  | 939   | 937.32  | 1.31| 0.19      |
| bays29    | 2020| 2020 | 2034  | 2020.25 | 2.37| 1.21      |
| bayg29    | 1610| 1610 | 1616  | 1610.42 | 0.46| 1.24      |
| dantzig42 | 699 | 699  | 704   | 700.71  | 1.52| 1.49      |
| eil51     | 426 | 426  | 429   | 427.15  | 0.98| 0.17      |
| berlin52  | 7542| 7542 | 7567  | 7544.45 | 0.76| 1.37      |
| st70      | 675 | 675  | 686   | 679.4   | 1.43| 0.23      |
| eil76     | 538 | 538  | 557   | 543.3   | 23.57| 0.53      |
| pr76      | 108159| 108159 | 108343 | 108211.73 | 2.12| 2.70      |
| rat99     | 1211| 1211 | 1220  | 1217.5  | 0.74| 0.29      |
| Kroa100   | 21282| 21282 | 21604  | 21432.30 | 56.17| 1.07      |
| Lin105    | 14379| 14379 | 14431  | 14387.25 | 1.35| 0.94      |
| Eil101    | 629 | 629  | 646   | 629.7   | 1.23| 0.07      |
| Ch105     | 6528| 6528 | 6636  | 6543.7  | 31.62| 3.46      |
| Kroa200   | 29368| 29468 | 29874  | 297036.15 | 103.28| 2.87      |

6.1.2. Against different test problems and different algorithms
In Table 13, average value SDs and the corresponding errors have been calculated for 11 problems using 7 methods. In all cases, the average results given by RSGA-I and RSGA-II are less than the corresponding average results by SGA-I, II, III, IV and V. Moreover, as the SDs in RSGA-I and II are quite small except three cases, it indicates that these methods are stable, results in different runs do not differ much from the mean. We also obtained the least percentage relative error in different cases. These errors are also very small indicating that derived average solutions are nearer to the best known solution in the literature. Thus the proposed methods have produced closer results to optimum.

6.1.3. The Friedman test
To compare the performance of the algorithms SGA-I, II, III, IV, V, RSGA-I and RSGA-II, we perform the Friedman test (cf. Derrac et al. [48]). Since it is a non-parametrical statistical procedure whose main aim is to detect significant difference between the behaviour of two or more algorithms.

The assumptions of Friedman test are as follows:

- The results over instances are mutually independent (i.e. the results within one instances do not influence the results within other instances)
- Within each instance, the observations (average objective values) can be ranked.

Hypothesis:
$H_0$: Each ranking of the algorithms within each problem is equally likely (i.e. there is no difference between them)
$H_1$: At least one of the algorithms tends to yield larger objective functions than at least one of the other algorithms.

Here number of algorithms ($k$) = 7 , number of instances ($b$) = 11. The Friedman ranking table is given below which is prepared according to the average results of Table 13.

Now $A_2 = \sum_{j=1}^{b} \sum_{i=1}^{k} (R(X_{ij}))^2$, $R_j = \sum_{i=1}^{b} R(X_{ij})$ for $j = 1, 2, \ldots, k$, and $B_2 = \frac{1}{b} \sum_{j=1}^{k} R_j^2$.

The test statistic is given by $T_2 = \frac{(b (b-1)(B_2 - bk(k+1)^2/4))/(A_2 - B_2)}$. Hence from Table 14, we calculate
Table 13. Results of RSGA and other methods for different test problems.

| Algorithm | Problem   | fri26 | bays29 | bayg29 | dantzig42 | eil51 | berlin52 | st70 | eil76 | pr76 | rat99 | kroa100 |
|-----------|-----------|-------|--------|--------|-----------|-------|----------|------|-------|------|-------|---------|
| BKS       |           | 937   | 2020   | 1610   | 699       | 426   | 7542     | 675  | 538   | 108159 | 1211  | 21282   |
| Avg       |           | 989.23| 2076.9 | 1639.5 | 731.4     | 452.3 | 7667.4   | 722.84 | 584.5 | 108354.5 | 1246.7 | 21757.67 |
| SGA-I     | SD        | 5.93  | 1.73   | 20.48  | 5.78      | 7.65  | 7.32     | 5.90 | 5.5   | 31.58 | 24.7  | 34.9    |
| Error(%)  |           | 1.75  | 0.78   | 1.82   | 1.80      | 0.68  | 0.84     | 2.89 | 0.97  | 4.3   | 2.57  | 3.85    |
| Avg       |           | 984.3 | 2075.2 | 1637.8 | 732.6     | 452.7 | 7666.8   | 721.7 | 580.63 | 108360.5 | 1244.65 | 21731.3 |
| SGA-II    | SD        | 2.37  | 1.80   | 1.48   | 2.70      | 26.58 | 10.32    | 4.64 | 6.56  | 52.8  | 24.7  | 87.98   |
| Error(%)  |           | 1.52  | 0.92   | 0.74   | 1.79      | 0.81  | 0.59     | 3.45 | 2.7   | 3.65  | 0.86  | 2.97    |
| Avg       |           | 979.73| 2074.4 | 1637.71| 728.47    | 450.8 | 7655.41  | 718.58 | 578.3 | 108725.42 | 1243.6 | 21702.3 |
| SGA-III   | SD        | 2.23  | 1.71   | 2.48   | 1.23      | 11.56 | 1.32     | 2.19 | 0.135 | 2.15  | 1.7   | 32.5    |
| Error(%)  |           | 0.74  | 2.72   | 1.95   | 0.56      | 1.79  | 1.02     | 0.86 | 1.75  | 3.22  | 1.63  | 2.7     |
| Avg       |           | 966.37| 2056.21| 1633.8 | 721.4     | 445.92| 7617.46  | 712.72 | 562.2 | 108674.61 | 1233.7 | 21678.7 |
| SGA-IV    | SD        | 4.13  | 1.98   | 2.54   | 3.01      | 2.51  | 1.33     | 2.32 | 1.56  | 2.58  | 2.37  | 84.89   |
| Error(%)  |           | 2.7   | 3.01   | 1.72   | 2.8       | 1.81  | 0.93     | 1.06 | 2.45  | 0.87  | 1.36  | 2.11    |
| Avg       |           | 958.52| 2050.3 | 1627.43| 717.42    | 442.7 | 76129.9  | 701.25 | 558.2 | 108521.75 | 1231.53 | 21502.26 |
| SGA-V     | SD        | 2.63  | 2.81   | 1.48   | 2.17      | 1.65  | 0.82     | 1.9  | 0.76  | 2.08  | 4.7   | 78.91   |
| Error(%)  |           | 1.12  | 1.78   | 0.95   | 2.36      | 1.02  | 1.9      | 0.93 | 1.78  | 4.45  | 2.31  | 3.27    |
| Avg       |           | 953.2 | 2036.17| 1621.43| 710.12    | 432.8 | 7589.6   | 686.2 | 544.3 | 108344.8 | 1223.49 | 21457.2 |
| RSGA-I    | SD        | 2.76  | 2.75   | 0.54   | 1.78      | 1.15  | 1.02     | 2.31 | 0.61  | 2.58  | 1.03  | 67.8    |
| Error(%)  |           | 0.78  | 1.39   | 0.31   | 1.51      | 0.67  | 1.59     | 0.76 | 2.25  | 0.72  | 0.35  | 1.23    |
| Avg       |           | 937.32| 2020.25| 1610.42| 700.7     | 427.15| 7544.45  | 679.4 | 543.3 | 108211.5 | 1217.5 | 21432.3 |
| RSGA-II   | SD        | 1.31  | 2.37   | 0.46   | 1.52      | 0.98  | 0.76     | 1.43 | 2.37  | 2.12  | 0.71  | 56.17   |
| Error(%)  |           | 0.19  | 1.21   | 0.24   | 1.49      | 0.17  | 1.37     | 0.23 | 0.53  | 2.7   | 0.29  | 1.07    |
Table 14. Ranking of the Friedman test.

| Algorithms (k) | SGA-I | SGA-II | SGA-III | SGA-IV | SGA-V | RSGA-I | RSGA-II |
|---------------|-------|--------|---------|--------|-------|--------|---------|
| fri26         | R(X₁) | R(X₂)  | R(X₃)   | R(X₄)  | R(X₅) | R(X₆)  | R(X₇)   |
| bay29         | 7     | 6      | 5       | 4      | 3     | 2      | 1       |
| bayg29        | 7     | 6      | 5       | 4      | 3     | 2      | 1       |
| dantzig42     | 6     | 7      | 5       | 4      | 3     | 2      | 1       |
| eili51        | 6     | 7      | 5       | 4      | 3     | 2      | 1       |
| berlin52      | 7     | 6      | 5       | 4      | 3     | 2      | 1       |
| st70          | 7     | 6      | 5       | 4      | 3     | 2      | 1       |
| eili76        | 7     | 6      | 5       | 4      | 3     | 2      | 1       |
| pr76          | 3     | 4      | 7       | 6      | 5     | 2      | 1       |
| rat99         | 7     | 6      | 5       | 4      | 3     | 2      | 1       |
| kroa100       | 7     | 6      | 5       | 4      | 3     | 2      | 1       |
| Average Rank  | 6.45  | 6      | 5.18    | 4.18   | 3.18  | 2      | 1       |
| Summation     | 71    | 66     | 57      | 46     | 35    | 22     | 11      |

Table 15. Paired comparison of the Friedman test.

| | Rᵢ − Rⱼ | SGA-II | SGA-III | SGA-IV | SGA-V | RSGA-I | RSGA-II |
|---|---------|--------|---------|--------|-------|--------|---------|
| SGA-I | 5 | 14 | 25 | 36 | 49 | 60
| SGA-II | — | 9 | 20 | 31 | 44 | 55
| SGA-III | — | — | 11 | 22 | 35 | 46
| SGA-IV | — | — | — | 11 | 24 | 35
| SGA-V | — | — | — | — | 13 | 24
| RSGA-I | — | — | — | — | — | 11

\[ A_2 = 473 + 402 + 299 + 196 + 115 + 44 + 11 = 1540, \]
\[ B_2 = \frac{1}{11} [71^2 + 66^2 + 57^2 + 46^2 + 35^2 + 22^2 + 11^2] = 1508.36 \]

With the values of \( A_2 \) and \( B_2 \), calculate the test statistic,
\[
T_2 = \frac{(11-1)(1508.36 - 11 \times 7(1+1)^2/4)}{1540 - 1508.36} = 87.34
\]

Using a table for the F distribution with a significance level \( \alpha = 0.01 \), we found that
\[
F_{(1-\frac{\alpha}{2},(k-1),(b-1)(k-1))} = F_{0.99,6,60} = 3.12.
\]

Since \( T_2 > F_{0.99,6,60} \), we reject the null hypothesis. Hence there exist some algorithms whose performances are significantly different from others.

6.1.4. (Post hoc) paired comparisons

Here the algorithms \( i \) and \( j \) are considered different after the rejection of the null hypothesis from the Friedman test. Following the post hoc paired comparison technique (cf. Derrac et al. [48]), calculate the absolute differences of the summation of the ranks of algorithms \( i \) and \( j \) and declare \( i \) and \( j \) different if:
\[
|Rᵢ − Rⱼ| > t_{1-\frac{\alpha}{2}} \sqrt{\frac{2b(A_2 - B_2)}{(b-1)(k-1)}}^{1/2},
\]
where \( t_{1-\frac{\alpha}{2}} \) is the \( 1 - (\alpha/2) \) quantile of the t distribution with \( (b-1)(k-1) \) degrees of freedom. Here \( t_{1-\frac{\alpha}{2}} \) for \( \alpha = 0.01 \) and 60 degrees of freedom is 2.660 and the critical value for the difference is 2.660. The following table summarises the paired comparisons, underline values indicated that the algorithms are different.
From Table 15, we conclude that RSGA-I and RSGA-II have outperformed than all other algorithms and RSGA-II is the best out performer amongst the other algorithms.

In the last section, overall conclusion and future plan is described.

7. Conclusion

Here, for the first time, new GA (RSGAs) has been proposed with rough set-based selection, min-point crossover and generation-dependent mutation processes. Here rough set-based age-dependent selection with three and five (extended) classifications, min-point crossover and three different $p_m$-dependent mutations are developed. For STSPs, $p_m$-oriented random mutation accelerates to get wide variety of node combinations. If $p_m$ is high, then the mutation rate is also much high. So it is in fine tuning to the optimisation problem, particularly this type of node oriented problems such as TSP, vehicle routing problem, network optimisation, etc. Again in accounting the complexity in mutation mechanism, type I is much high against the type II because in type I, random exchange occurs with searching the node in each step of the mutation whereas type II does no searching in location exchange. But type I is more affective to find the global optimum. For type III, its complexity is better against other two and efficiency is much high. With these new features, RSGAs are used for test problems from TSPLIB and its efficiency is proved. The supremacy of RSGA is established through the Friedman test and post hoc paired comparison. Later, two TSP problem-constrained TSP and constrained solid TSP are solved and the optimum results along with near optimum results are presented. The developed RSGAs are quite general, these can be used for the decision making problems in other areas such as inventory control system, supply chain and portfolio management. Moreover, RSGAs will be very useful for the large problems with large-scale data. The proposed RSGAs can be extended/modified to be applied for the optimisation of multi-objective problems.

Acknowledgements

Peer review under responsibility of Fuzzy Information and Engineering Branch of the Operations Research Society of China.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This research article is supported by University Grants Commission of India by grant number PSW-150/14-15(ERO).

ORCID

Samir Maity http://orcid.org/0000-0002-7209-0784

References

[1] Lawler EL, Lenstra JK, RinnooyKan AHG, et al. The traveling salesman problem: G. E. Re guided tour of combinatorial optimization. New York: John Wiley and Sons; 1985.
[2] Focacci F, Lodi A, Milano M. A hybrid exact algorithm for the TSPTW. Inform J Comput. 2002;14(4):403–417.

[3] Chang T, Wan Y, Tooi W. A stochastic dynamic travelling salesman problem with hard time windows. Eur J Oper Res. 2009;198(3):748–759.

[4] Petersen HL, Madsen OBG. The double travelling salesman problem with multiple stack – formulation and heuristic solution approaches. Eur J Oper Res. 2009;198:339–347.

[5] Majumder J, Bhunia AK. Genetic algorithm for asymmetric traveling salesman problem with imprecise travel times. J Comput Appl Math. 2011;235(9):3063–3078.

[6] Moon C, Ki J, Choi G, et al. An efficient genetic algorithm for the traveling salesman problem with precedence constraints. Eur J Oper Res. 2002;140:606–617.

[7] Xing LN, Chen YW, Yang KW, et al. A hybrid approach combining an improved genetic algorithm and optimization strategies for the asymmetric travelling salesman problem. Artif Intell. 2008;21:1370–1380.

[8] Bai J, Yang GK, Chen YW, et al. A model induced max-min ant colony optimization for asymmetric travelling salesman problem. Appl Soft Comput. 2013;13:1365–1375.

[9] Jula H, Dessouky M, Ioannou PA. Truck Route Planning in non-stationary stochastic networks with time windows at customer location. IEEE Trans Intell Transp Syst. 2006;7(1):51–52.

[10] Yuan S, Skinner B, Huang S, et al. A new crossover approaches for solving the multiple travelling salesman problem using genetic algorithms. Eur J Oper Res. 2013;228:72–82.

[11] Miranda Bront J, Mendez Diaz I, Zabala P. Facts and valid inequalities for time dependent travelling salesman problem. Eur J Oper Res. 2014;236:891–902.

[12] Chiang WC, Russell RA. Simulated annealing metaheuristics for the vehicle routing problem with time windows. Ann Oper Res. 1997;63:3–27.

[13] Khonl J. The application of Tabu search to the symmetric traveling salesman problem, Ph.D. Dissertation, University of Colorado, 1989.

[14] Bianchi L, Dorigo M, Gambardella LM. Ant colony optimization approach to the probabilistic travelling salesman problem. PPSN VII, LNCS. 2002;24(3):883–892.

[15] Holland JH. An efficient genetic algorithm for the traveling salesman problem. Eur J Oper Res. 1975;145:606–617.

[16] Eberhart RC, Kennedy J. Particle swarm optimization. In: Proc IEEE Conf Neural Networks. 1995;4:1942–1948.

[17] Črepinšek M, Liu SH, Mernik L. A note on teaching learning based optimization algorithm. Inf Sci (Ny). 2012;212:79–93.

[18] Črepinšek M, Liu S.H, Mernik L, et al. Is a comparison of results meaningful from the in exact replications of computational experiments?. Soft Comput. 2016;20(1):223–235.
[26] Črepinšek M, H. Liu S, Mernik M. Replication and comparison of computational experiments in applied evolutionary computing: Common pitfalls and guidelines to avoid them. Appl Soft Comput. 2014;19:161–170.

[27] Mernik M, Liu SH, Karaboga MD, et al. On clarifying misconceptions when comparing variants of the artificial Bee colony algorithm by offering a new implementation. Inf Sci (NY). 2015;291:115–127.

[28] Piotrowski AP, Napiorkowski MJ, Napiorkowski JJ, et al. Swarm intelligence and evolutionary algorithms: performance versus speed. Inf Sci (NY). 2016. doi:10.1016/j.ins.2016.12.028.

[29] Ni H, Wang Y. Stack index tracking by Pareto efficient GA. Appl Soft Comput. 2013;13(12):4519–4535.

[30] Ursani Z, Essam D, Cornforth D, et al. Localized genetic algorithm for vehicle routing problem with time windows. Appl Soft Comput. 2011;11(8):5375–5390.

[31] Neungmatcha W, Sethanan K, Gen M, et al. Adaptive genetic algorithm for solving sugarcane loading stations with multi-facility services problem. Comput Electron Agric. 2013;98(10):85–99.

[32] Huang KY. An enhance classification method comparing a GA rough set theory and modified PBM-index function. Appl Soft Comput. 2012;12(10):46–63.

[33] Enigin O, Ceron G, Yilmaz MK. An efficient GA for hybrid flow shop scheduling with multiprocessor task problem. Appl Soft Comput. 2011;11(3):3056–3065.

[34] Last M, Eyal S. A fuzzy-based lifetime extension of genetic algorithms. Fuzzy Sets Syst. 2005;149:131–147.

[35] Roy A, Kar S, Maiti M. A production inventory model with stock dependent demand incorporating learning and inflationary effect in a random planning horizon: A fuzzy genetic algorithm with varying population size approach. Comput Ind Eng. 2009;57:1324–1335.

[36] Fdez JA, Alcal R, Gacto MJ, et al. Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. Fuzzy Sets Syst. 2009;160:905–921.

[37] Elyan E, Gaber MM. A genetic algorithm approach to optimising random forests applied to class engineered data. Inf Sci (NY). 2016. doi:10.1016/j.ins.2016.08.007.

[38] Liang WY, Huang CC. The generic genetic algorithm incorporates with rough set theory: an application of the web services composition. Expert Syst Appl. 2009;36:5549–5556.

[39] Cheng CH, Chen TL, Wei LY. A hybrid model based on rough sets theory and genetic algorithms for stock price forecasting. Inf Sci (NY). 2010;180:1610–1629.

[40] Lin CH. A rough penalty genetic algorithm for constrained optimization. Inf Sci (NY). 2013;241:119–137.

[41] Mirhosseini M, Mashinchi M, Nezambadi-pour H. Improving n-similarity problem by genetic algorithm and its application in text document resemblance. Fuzzy Inf Eng. 2014;6:263–278.

[42] TSPLIB. Available at http://comopt.ifi.uni-dueidelberg.de/software/TSPLIB95/.

[43] Tiwari SP, Sharan S, Jadav VK. Fuzzy closure and fuzzy-rough set. Fuzzy Inf Eng. 2014;6:93–100.

[44] Karmakar S, Mahato SK, Bhunia AK. Interval oriented multi-section techniques for global optimization. J Comput Appl Math. 2009;224(2):476–491.

[45] Pal K, Mondal SK. Multi-item supplier selection model with fuzzy risk analysis studied by possibility and necessity constraints. Fuzzy Infor Eng. 2015;7:451–474.

[46] Karmakar S, Bhunia AK. A comparative study of different order relations of intervals. Reliable Comput. 2012;16:360–369.

[47] Mondal M, Maity AK, Maiti MK, et al. A production repairing inventory model with fuzzy rough coefficients under inflation and time value of money. Appl Math Model. 2012;37(5):3200–3215.

[48] Derrac J, Garcia S, Molina D, et al. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput. 2011;1:3–18.