ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 10, No. 4, 2015, *pp*. 255-264

# Genetic Algorithm Approach to Solve Integer Nonlinear Programming Problem in Reliability Optimization

/˚öæ Ł˛˚ß˚Ê˚˛#5˚ª łł
Department of Mathematics, Burdwan Raj College, Burdwan-713104, India
(*Received April 04, 2015, accepted July 29, 2015)*

**Abstract.** In the recent years, different types of optimization techniques have been advanced rapidly and applied widely to solve several optimization problems. These advancements are depends on both the upgrading progress of modern computer technologies and that of optimization process for large scale system. The most important optimization problem in reliability optimization is redundancy allocation problem. However, these problems have always been solved in the precise setup. In this paper, the reliability of each component is considered to be imprecise. The goal of this paper is to solve the redundancy allocation problem that maximize the overall system reliability subject to the given resource constraints. This impreciseness has been represented by interval number. This type of problem has been formulated as an interval valued constrained optimization problem. Then the constrained optimization problem has been transformed into unconstrained one by penalty function technique then solved by Genetic Algorithm. Finally, to illustrate the methodology, a numerical example has been presented and solved for illustration purpose.

**Keywords:** Reliability Optimization, Redundancy allocation problem, Genetic algorithm, System reliability, Penalty function technique.

## 1. Introduction

Reliability optimization has become an important issue in recent years because modern high-tech industrial processes with increasing levels of sophistication comprise most engineering systems today. Since last few decades, many publications have addressed this problem using different system structures, performance measures, optimization techniques and options for reliability improvement. The subject "Reliability Optimization" appeared in the literature in due late 1940s and was first applied to communication and transportation systems. One of the goals of the reliability engineer is to find the best way to increase the system reliability. As systems are becoming more complex, the consequences of their unreliable behavior have become severe in terms of cost, effort and so on. For a system with known cost, reliability, weight, volume and other parameters, the corresponding design problem is known as combinatorial optimization problem. The well known reliability design problem is referred to as redundancy allocation problem. The objective of this problem is to determine the number of redundant components that maximize the system reliability or minimize the system cost under the several resource/budget constraints. As redundancy allocation problem is a nonlinear integer programming problem, it cannot be solved by direct/indirect or mixed search methods due to discrete search space. According to Chern [1], redundancy allocation problem is NP-hard. In the existing literature, it is observed that several methods, viz. heuristic methods [2-4], branch and bound method [5], reduced gradient method [6], integer programming method [7] etc. were used to solve such redundancy allocation problem. However, these methods have some advantages as well as disadvantages also. Therefore, with the development of genetic algorithm and other evolutionary algorithms, most of the researchers are motivated to use these methods in solving redundancy allocation problem as these methods provide more flexibility, require fewer assumptions on the objective as well as constraints.

However, in almost all the approaches mentioned earlier, the parameters of redundancy allocation problem have usually been taken to be precise values. However, in reality, the reliability of each component is imprecise due to human errors, improper storage facilities and other unexpected factors relating to environment. To tackle the problem with such imprecise numbers, generally stochastic, fuzzy and fuzzy-stochastic approaches are applied and the corresponding problems are converted to deterministic problems for solving them. In this paper, we prefer to consider interval number because the problem under the three approaches can be converted into a problem in interval representation. Due to this new representation, the objective function of the reduced redundancy allocation problem is interval valued, which is to be

maximized under given constraint. In the recent years, a number of works has been reported in the literature considering the parameters of the problem as interval valued. In this paper, we have considered redundancy allocation problem of series system with four subsystems. Each subsystem is connected in parallel with identical components. The reliability of each component is considered to be interval number. Then the corresponding problem has been formulated as an interval valued constrained optimization problem. Then the constrained optimization problem has been transformed into unconstrained one by Big-M penalty function technique then solved by Genetic Algorithm and Interval order relations developed by Sahoo et al. [8]. Finally, to illustrate the methodology, a numerical example has been solved and the computed results have been presented.

## 2. Assumptions and Notations

A redundancy allocation problem is formulated under the following assumptions and notations.

**2.1. Assumptions:**

(i) Reliability of each component is imprecise and interval valued.
(ii) Failures of components are mutually statistically independent.
(iii)The components as well as the system have two different states, viz. operating state and failure state.

**2.2. Notations:**

| | |
|---|---|
| $n$ | the number of subsystems |
| $x$ | $(x_1, x_2, ..., x_n)$ |
| $x_j$ | the number of components in *j*-th subsystem, arranged in parallel |
| $r_j$ | The reliability j-th *Component* |
| $R_s$ | The system reliability |
| $r_j = [r_{jL}, r_{jR}]$ | Interval valued reliability of *j*-th component |
| $[R_{SL}, R_{SR}]$ | Interval valued system reliability |
| $l_j, u_j$ | lower and upper bounds of $x_j$ |
| $g_i$ | The $i-th$ constraint function |
| $b_i$ | The upper limit on the $i-th$ resource |
| $U(.)$ | Uniform distribution |
| $\Omega$ | Feasible region |
| FS | Feasible Solution |
| NFS | Nonfeasible Solution |

## 3. Mathematical formulation of the Problem

Let us consider a redundancy allocation problem of series system. Each subsystem is connected in parallel with identical components. Our objective is to maximize the overall system reliability subject to the given resource constraints. This can be done by finding the number of redundant components in each subsystem.

The general form of the redundancy allocation problem is as follows:

$$\text{Maximize } R_S = \prod_{j=1}^{n}\left[1-\left(1-r_j\right)^{x_j}\right] \tag{1}$$

$$\text{subject to } g_i(x) \le b_i,\ i = 1, 2, \cdots, m$$

where $1 \le l_j \le x_j \le u_j$, $x_j$ is integer, $j = 1, ..., n$, $b_i$ is the *i*-th available resource, $i = 1, 2, \cdots, m.$

When reliability of each components are interval numbers i.e., $r_j = [r_{jL}, r_{jR}]$ then the problem (3.1) reduces to

$$\text{Maximize } [R_{SL}, R_{SR}] = \prod_{j=1}^{n}\left[1-\left(1-[r_{jL}, r_{jR}]\right)^{x_j}\right] \tag{2}$$

$$\text{subject to } g_i(x) \le b_i,\ i = 1, 2, \cdots, m$$

where $1 \le l_j \le x_j \le u_j$, $x_j$ is integer, $j = 1, ..., n$, $b_i$ is the *i*-th available resource, $i = 1, 2, \cdots, m.$

In Problem (2), reliability of each component is interval valued. For solving such type of problems, interval mathematics and interval ranking play an essential role.

## 4. Finite Interval Mathematics

An interval number $A$ is a closed interval denoted by $A = [a_L, a_R]$ and is defined by $A = [a_L, a_R] = \{x : a_L \le x \le a_R, x \in \square\}$, where $a_L$ and $a_R$ are the left and right limits respectively and $\square$ is the set of all real numbers. An interval $A$ can also be expressed in terms of centre and radius as $A = \langle a_c, a_w \rangle$ $= \{x : a_c - a_w \le x \le a_c + a_w, x \in \square\}$, where $a_c$ and $a_w$ be the centre and radius of the interval $A$ respectively i.e., $a_c = (a_L + a_R)/2$ and $a_w = (a_R - a_L)/2$. Actually, every real number can be treated as an interval, such as for all $x \in \square$, $x$ can be written as an interval $[x, x]$ having zero width.

### 4.1 Arithmetic of interval numbers

Here, we shall give the concise definitions of arithmetical operations like addition, subtraction, multiplication, and division of interval numbers.

**Interval addition:** Let $A = [a_L, a_R] = \langle a_c, a_w \rangle$ and $B = [b_L, b_R] = \langle b_c, b_w \rangle$ be two intervals. Then the addition of two intervals $A$ and $B$ is given by

- $A + B = [a_L, a_R] + [b_L, b_R] = [a_L + b_L, a_R + b_R]$
- or $A + B = \langle a_c, a_w \rangle + \langle b_c, b_w \rangle = \langle a_c + b_c, a_w + b_w \rangle$

**Interval subtraction:** The subtraction of two intervals $A = [a_L, a_R] = \langle a_c, a_w \rangle$ and $B = [b_L, b_R] = \langle b_c, b_w \rangle$ is given by

$A - B = [a_L, a_R] - [b_L, b_R] = [a_L - b_R, a_R - b_L]$

or $A - B = \langle a_c, a_w \rangle - \langle b_c, b_w \rangle = \langle a_c, a_w \rangle + \langle -b_c, b_w \rangle = \langle a_c - b_c, a_w + b_w \rangle$

**Scalar multiplication of an interval number:** The multiplication of an interval $A = [a_L, a_R] = \langle a_c, a_w \rangle$ by a real number $\lambda$ is defined by $\lambda A = \begin{cases} [\lambda a_L, \lambda a_R] & \text{for } \lambda \ge 0, \\ [\lambda a_R, \lambda a_L] & \text{for } \lambda < 0, \end{cases}$

or $\lambda A = \lambda \langle a_c, a_w \rangle = \langle \lambda a_c, |\lambda| a_w \rangle$

**Product of two interval numbers:**

The product of two different intervals $A = [a_L, a_R]$ and $B = [b_L, b_R]$ is defined by $A \times B = [\min(a_L b_L, a_L b_R, a_R b_L, a_R b_R), \max(a_L b_L, a_L b_R, a_R b_L, a_R b_R)]$.

**Division of two interval numbers:** The division of the interval $B = [b_L, b_R]$ by the interval $A = [a_L, a_R]$ is defined as $\frac{B}{A} = B \times \frac{1}{A} = [b_L, b_R] \times [\frac{1}{a_R}, \frac{1}{a_L}]$, provided $0 \notin [a_L, a_R]$.

The above definitions are given in the books written by Moore [9] and Hansen and Walster [10].

### 4.2 Interval Order Relations

For finding the optimum solution in solving the optimization problems with interval valued objectives, we need to define the order relations of interval numbers.

Let $A=[a_L,a_R]$ and $B=[b_L,b_R]$ be two unequal intervals. Then these two intervals may be one of the following types:

Type-1: Two intervals are disjoint [see Fig. 1].

Type-2: Two intervals are partially overlapping [see Fig. 2].

Type-3: One of the intervals contains the other one [see Fig. 3].

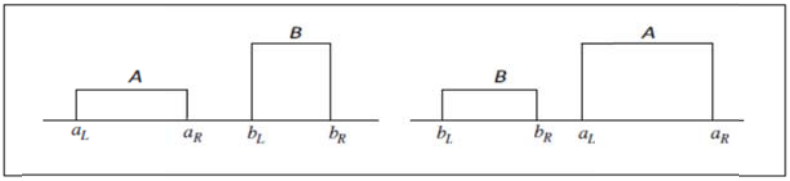


Fig. 1. Type-1 intervals


Fig. 2. Type-2 intervals


Fig. 3. Type-3 intervals

It is to be noted that both the intervals $A=[a_L,a_R]$ and $B=[b_L,b_R]$ will be equal in case of fully overlapping intervals, i.e., $A=B$ iff $a_L=b_L$ and $a_R=b_R$.

Over the last few decades, several definitions have been proposed for finding the interval order relations. Recently, Mahato and Bhunia [11] defined order relations for optimization problems with interval objectives. They defined order relations with respect to optimistic and pessimistic decision-makers. But in their definitions of pessimistic decision-making of Type-3 intervals, it is observed that sometimes optimistic decisions are to be taken. To overcome this situation, Sahoo et al. [8] proposed two new definitions of order relations irrespective of optimistic as well as pessimistic decision-makers for addressing maximization and minimization problems separately.

**Definition 1:**

The order relation $>_{\max}$ between the intervals $A=[a_L,a_R]=\langle a_c,a_w\rangle$ and $B=[b_L,b_R]=\langle b_c,b_w\rangle$, then for maximization problems

(i) $A>_{\max} B \Leftrightarrow a_c>b_c$ for Type-1 and Type-2 intervals and

(ii) $A>_{\max} B \Leftrightarrow$ either $a_c\geq b_c \wedge a_w<b_w$ or $a_c\geq b_c \wedge a_R>b_R$ for Type-3 intervals.

According to this definition, the interval $A$ is accepted for maximization case. Clearly, this order relation $>_{\max}$ is reflexive and transitive but not symmetric.

**Definition 2:**

The order relation $<_{\min}$ between the intervals $A=[a_L,a_R]=\langle a_c,a_w\rangle$ and $B=[b_L,b_R]=\langle b_c,b_w\rangle$, then for minimization problems

(i) $A<_{\min} B \Leftrightarrow a_c<b_c$ for Type-1 and Type-2 intervals and

(ii) $A<_{\min} B \Leftrightarrow$ either $a_c\leq b_c \wedge a_w<b_w$ or $a_c\leq b_c \wedge a_L<b_L$ for Type-3 intervals.

According to this definition, the interval $A$ is accepted for minimization case. Clearly, the order relation $<_{\min}$ is reflexive and transitive but not symmetric.

Given this back drop we reduce (2) as follows:



DOI https://doi.org/2024-JICS-22533 | Generated on 2025-04-12 03:46:48

$$\text{Maximize } [R_{SL}, R_{SR}] = \prod_{j=1}^{n}\left[1-\left(1-r_{jL}\right)^{x_j}, 1-\left(1-r_{jR}\right)^{x_j}\right] \tag{3}$$

$$\text{subject to } g_i(x) \le b_i,\ i = 1,2,\cdots,m$$

where $1 \le l_j \le x_j \le u_j$, $x_j$ is integer, $j = 1,...,n$, $b_i$ is the $i$-th available resource, $i = 1,2,\cdots,m.$

Clearly the problem (3) is an interval valued constrained optimization problem. To solve this optimization problem, we propose to consider heuristic methods. One such method of importance and wide scope is Genetic Algorithm.

## 5. Genetic Algorithm based Constraints handling technique

The problem (3) is an interval valued constrained optimization problem. In the application of Genetic algorithm for solving the problems with interval objectives, there arises an important questions for handling the constraints. During the past, several techniques have been addressed to deal with the constraints in genetic algorithms for solving problems with a non-interval/ fixed objectives. Here, we have to solve the constrained optimization problem by using penalty function technique. In this technique, the given constrained optimization problem is converted to an unconstrained optimization problem in which the reduced objective function involves objective function and a penalty for violating the constraints. In this work we have used Big-M penalty [12].

The form of Big-M penalty is as follows:

For the constrained optimization problem (3)

$$\text{Maximize } [R_{SL}, R_{SR}] = \prod_{j=1}^{n}\left[1-\left(1-r_{jL}\right)^{x_j}, 1-\left(1-r_{jR}\right)^{x_j}\right] \tag{4}$$

$$\text{subject to } g_i(x) \le b_i,\ i = 1,2,\cdots,m$$

where $1 \le l_j \le x_j \le u_j$, $x_j$ is integer, $j = 1,...,n$, $b_i$ is the $i$-th available resource, $i = 1,2,\cdots,m.$

The form of Big-M penalty is as follows:

$$\text{Maximize}[R_{SL}^{次}(x), R_{SR}(x)] = \begin{cases} [R_{SL}(x), R_{SR}(x)] & \text{if } x \in \Omega \\ [-M, -M] & \text{if } x \notin \Omega \end{cases} \tag{5}$$

**where** $\Omega = \{x : g_i(x) \le b_i, i = 1,2,\cdots,m \text{ and } 1 \le l_j \le x_j \le u_j,\ x_j \text{ integer},\ j = 1,...,n\}$

## 6. Genetic Algorithm (GA)

To solve the nonlinear optimization problem (5) with interval objective we have developed an advanced optimization technique, viz. Genetic Algorithm (GA) with interval valued fitness. The most fundamental idea of Genetic Algorithm [13] is to replicate the natural evolution process artificially in which populations undergo continuous changes through genetic operators, like crossover, mutation and selection. In particular, it is very useful for solving complicated optimization problems which cannot be solved easily by direct or gradient based mathematical techniques. It is very effective to handle large-scale, real-life, discrete and continuous optimization problems without making unrealistic assumptions and approximations.

The algorithm for implementing GA is as follows:

### Algorithm

Step-1: Set population size ( $P_s$ ), crossover probability ( $P_c$ ), mutation probability ( $P_m$ ), maximum generation ( $M_g$ ) and bounds of the variables $l_i, u_i$ $(i = 1,...,n)$.

Step-2: $t = 0$ [ $t$ represents the number of current generation].

Step-3: Initialize the chromosome of the population $P(t)$ [ $P(t)$ represents the population at $t$ - $th$ generation].

Step-4: Evaluate the fitness function of each chromosome of $P(t)$ considering the objective function as the fitness function.

Step-5: Find the best chromosome from the population $P(t)$.

Step-6: $t$ is increased by unity.

Step-7: If the termination criterion is satisfied go to step-14, otherwise, go to next step.
Step-8: Select the population $P(t)$ from the population $P(t-1)$ of earlier generation by tournament selection process.
Step-9: Alter the population $P(t)$ by crossover, mutation and elitism operators.
Step-10: Evaluate the fitness function value of each chromosome of $P(t)$.
Step-11: Find the best chromosome from $P(t)$.
Step-12: Compare the best chromosome of $P(t)$ and $P(t-1)$ and store better one.
Step-13: Go to step-6.
Step-14: Print the best chromosome (which is the solution of the optimization problem).
Step-15: End.

The following basic components have been considered to implement the genetic algorithm.

(i) GA parameters ( $P_s$ , $P_c$ , $P_m$ and $M_g$ )
(ii) Chromosome representation
(iii) Initialization of population
(iv) Evaluation of fitness function
(v) Selection process
(vi) Genetic operators (crossover and mutation)

Here we shall discuss the main three important components of GA viz. selection process, crossover and mutation.

In GA, the selection operator plays an important role as it is the first operator applied to the population. The aim of this operator is to select the above average solutions and eliminate below average solutions from the entire population for the next generation under the well known principle "survival of the fittest". Here, we have used tournament selection process of size two with replacement as the selection operator considering the following assumptions:

(i) When both the chromosomes are feasible then one with better fitness value is selected.
(ii) When one chromosome is feasible and another is infeasible then the feasible one is selected.
(iii) When both the chromosomes are infeasible with unequal constraints violation, then the chromosome with less constraints violation is selected.
(iv) When both the chromosomes are infeasible with equal constraints violation, then any one chromosome is selected.

After the selection, a crossover operator is applied to the resulting chromosomes which have survived. It operates on two or more parent chromosomes at a time and creates the offspring by recombining the feature of the parent solutions. In this work, we have used intermediate crossover.

The different steps of intermediate crossover operation are as follows:

Step-1: Find the integral value of $\lfloor P_c * P_s \rfloor$ and store it in $N_c$.

Step-2: Select two parent chromosomes $s_k^{(t)}$ and $s_i^{(t)}$ randomly from the population.

Step-3: (a) Generate $g = U(0, \left| s_{kj}^{(t)} - s_{ij}^{(t)} \right|)$ , $j = 1, 2, ..., n$

(b ) Compute the components $\overline{s}_{kj}^{(t)} = \begin{cases} \overline{s}_{kj}^{(t)} = s_{kj}^{(t)} - g \text{ if } s_{kj}^{(t)} > s_{ij}^{(t)} \\ \overline{s}_{kj}^{(t)} = s_{kj}^{(t)} + g \text{ otherwise} \end{cases}$

and $\overline{s}_{ij}^{(t)} = \begin{cases} s_{ij}^{(t)} + g \text{ if } s_{kj}^{(t)} > s_{ij}^{(t)} \\ s_{ij}^{(t)} - g \text{ otherwise} \end{cases}$

Step-4: Compute $s_k^{(t+1)} = \text{argument of best of } \{ f(s_k^{(t)}), f(s_i^{(t)}), f(\overline{s}_k^{(t)}), f(\overline{s}_i^{(t)}) \}$

and $s_i^{(t+1)} = \text{argument of best of } \{ f(s_k^{(t)}), f(s_i^{(t)}), f(\overline{s}_k^{(t)}), f(\overline{s}_i^{(t)}) \}$

Step-5: Repeat Step-2 and Step-4 for $\frac{N_c}{2}$ times.

The aim of the mutation operation is to introduce the random variations into the population used to prevent the search process from converging to the local optima. Sometimes, it helps to regain the information

lost in earlier generations and responsible for fine tuning of the system. This operator is applied to a single chromosome only. In this work we have used one-neighbourhood mutation [14].

The different steps of mutations operations are as follows:

Step-1: Find the integral value of $\left[P_m * P_s\right]$ and store it in $N_m$.

Step-2: Select one parent chromosome $s_i^{(t)}$ randomly from the population

Step-3: Select a gene $s_{ik}^{(t)}\ (k=1,2,...,n)$ on chromosome $s_i^{(t)}$ for mutation and domain of $s_{ik}^{(t)}$ is $[l_{ik}, u_{ik}]$.

Step-4: Generate $r = U(0,1)$

Step-5: Create new gene $s_{ik}^{\prime(t)}\ (k=1,2,...,n)$ corresponding to the selected gene $s_{ik}^{(t)}$ by mutation process as follows:

$$s_{ik}^{\prime(t)} = \begin{cases} s_{ik}^{(t)} + 1 & \text{if } s_{ik}^{(t)} = l_{ik} \\ s_{ik}^{(t)} - 1 & \text{if } s_{ik}^{(t)} = u_{ik} \\ s_{ik}^{(t)} + 1 & \text{if } r < 0.5 \\ s_{ik}^{(t)} - 1 & \text{if } r \geq 0.5 \end{cases}$$

Step-6: Compute $s_i^{(t+1)} = \text{argument of better of } \{ f(s_i^{(t)}), f(s_i^{\prime(t)}) \}$

Step-7: Repeat Step-2 to Step-6 for $N_m$ times.

## 7. Numerical example

In this section, we have considered the redundancy allocation problem for 4 stage series system. The mathematical formulation of this problem is as follows:

$$\text{Maximize}[R_{SL}, R_{SR}] = \prod_{j=1}^{4}\left[1-(1-r_{jL})^{x_j}, 1-(1-r_{jR})^{x_j}\right] \tag{6}$$

$$\text{subject to, } 1.5x_1 + 3.3x_2 + 3.2x_3 + 4.4x_4 \leq 55$$

$$4x_1 + 5x_2 + 7x_3 + 9x_4 \leq 114.5$$

$$1 \leq x_j \leq 5,\ j = 1,2,3,4 \text{ and are integers.}$$

All the values of the component reliabilities related to above problem are given in Table-1:

Table-1: Parameters of this example

| $j$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $r_{jL}$ | 0.72 | 0.75 | 0.70 | 0.81 |
| $r_{jR}$ | 0.74 | 0.78 | 0.73 | 0.83 |

Then to solve this problem, we have used a real coded genetic algorithm (GA) for integer variables with tournament selection, intermediate crossover and one neighborhood mutation. To demonstrate the different steps like initialization, selection, crossover and mutation, the following GA parameters values have been considered.

Population size=10, Maximum number of generations=50, Probability of crossover=0.85, Probability of mutation=0.15.

Here the termination criterion is that the number of generations reaches maximum number of generations. Simulation results are displayed in Table 2-8.

Table-2: After Initialization

| Chromosome No. | Gene value | | | | Fitness value | Solution status |
|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $[R_{SL}, R_{SR}]$ | |
| 1 | 1 | 3 | 1 | 1 | [0.401861 , 0.443592] | FS |
| 2 | 5 | 5 | 5 | 5 | [-999999 , -999999] | NFS |
| 3 | 2 | 2 | 2 | 5 | [0.786045 , 0.822473] | FS |
| 4 | 2 | 5 | 5 | 2 | [0.885306 , 0.903688] | FS |
| 5 | 2 | 4 | 2 | 2 | [0.805223 , 0.837480] | FS |
| 6 | 4 | 5 | 1 | 2 | [0.669928 , 0.705300] | FS |
| 7 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 8 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |
| 9 | 1 | 3 | 5 | 2 | [0.681504 , 0.709942] | FS |
| 10 | 4 | 1 | 2 | 3 | [0.673652 , 0.716297] | FS |

## GENERATION-I

Table-3: After tournament selection

| Chromosome No. | Gene value | | | | Fitness value | Solution status |
|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $[R_{SL}, R_{SR}]$ | |
| 1 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 2 | 4 | 1 | 2 | 3 | [0.673652 , 0.716297] | FS |
| 3 | 2 | 2 | 2 | 5 | [0.786045 , 0.822473] | FS |
| 4 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 5 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 6 | 2 | 5 | 5 | 2 | [0.885306 , 0.903688] | FS |
| 7 | 4 | 1 | 2 | 3 | [0.673652 , 0.716297] | FS |
| 8 | 2 | 4 | 2 | 2 | [0.805223 , 0.837480] | FS |
| 9 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |
| 10 | 2 | 5 | 5 | 2 | [0.885306 , 0.903688] | FS |

Table-4: After crossover

| Chromosome No. | Gene value | | | | Fitness value | Solution status |
|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $[R_{SL}, R_{SR}]$ | |
| 1 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 2 | 3 | 3 | 2 | 2 | [0.844489 , 0.875066] | FS |
| 3 | 3 | 2 | 2 | 3 | [0.828674 , 0.862464] | FS |
| 4 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 5 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 6 | 2 | 5 | 5 | 2 | [0.885306 , 0.903688] | FS |
| 7 | 2 | 2 | 4 | 4 | [0.855885 , 0.881819] | FS |
| 8 | 3 | 4 | 3 | 3 | [0.941422 , 0.956110] | FS |
| 9 | 3 | 2 | 2 | 3 | [0.828674 , 0.862464] | FS |
| 10 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |

Table-5: After mutation

| Chromosome No. | Gene value | | | | Fitness value | Solution status |
|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $[R_{SL}, R_{SR}]$ | |
| 1 | 2 | 5 | 3 | 5 | [-999999 , -999999] | NFS |
| 2 | 3 | 3 | 2 | 2 | [0.844489 , 0.875066] | FS |
| 3 | 3 | 2 | 2 | 3 | [0.828674 , 0.862464] | FS |
| 4 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 5 | 2 | 5 | 5 | 2 | [0.885306 , 0.903688] | FS |
| 6 | 2 | 2 | 4 | 4 | [0.855885 , 0.881819] | FS |
| 7 | 3 | 4 | 3 | 3 | [0.941422 , 0.956110] | FS |
| 8 | 3 | 2 | 2 | 3 | [0.828674 , 0.862464] | FS |
| 9 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |
| 10 | 3 | 4 | 3 | 3 | [0.941422 , 0.956110] | FS |



DOI https://doi.org/2024-JICS-22533 | Generated on 2025-04-12 03:46:48

**GENERATION-II**

Table-6: After tournament selection

| Chromosome No. | Gene value | | | | Fitness value | Solution status |
|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $[R_{SL}, R_{SR}]$ | |
| 1 | 3 | 4 | 3 | 3 | [0.941422 ,0.956110] | FS |
| 2 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |
| 3 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 4 | 3 | 2 | 2 | 3 | [0.828674 , 0.862464] | FS |
| 5 | 2 | 5 | 5 | 2 | [0.885306 , 0.903688] | FS |
| 6 | 2 | 2 | 4 | 4 | [0.855885 , 0.881819] | FS |
| 7 | 3 | 4 | 3 | 3 | [0.941422 , 0.956110] | FS |
| 8 | 3 | 4 | 3 | 3 | [0.828674 , 0.862464] | FS |
| 9 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |
| 10 | 2 | 2 | 4 | 4 | [0.855885 , 0.881819] | FS |

Table-7: After crossover

| Chromosome No. | Gene value | | | | Fitness value | Solution status |
|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $[R_{SL}, R_{SR}]$ | |
| 1 | 3 | 5 | 5 | 3 | [0.968033 ,0.975692] | FS |
| 2 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |
| 3 | 1 | 5 | 3 | 5 | [0.699703 , 0.724958] | FS |
| 4 | 3 | 2 | 2 | 3 | [0.828674 , 0.862464] | FS |
| 5 | 3 | 4 | 3 | 2 | [0.913703 , 0.933063] | FS |
| 6 | 2 | 2 | 4 | 4 | [0.855885 , 0.881819] | FS |
| 7 | 3 | 4 | 3 | 3 | [0.941422 , 0.956110] | FS |
| 8 | 3 | 3 | 3 | 4 | [0.935551 , 0.952036] | FS |
| 9 | 2 | 2 | 5 | 3 | [0.855989 , 0.881646] | FS |
| 10 | 2 | 3 | 4 | 3 | [0.893680 , 0.913061] | FS |

Table-8: After mutation

| Chromosome No. | Gene value | | | | Fitness value | Solution status |
|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $[R_{SL}, R_{SR}]$ | |
| 1 | 3 | 5 | 5 | 3 | [0.968033 ,0.975692] | FS |
| 2 | 3 | 2 | 5 | 3 | [0.908418 , 0.928947] | FS |
| 3 | 1 | 5 | 4 | 5 | [-999999 , -999999] | NFS |
| 4 | 3 | 2 | 2 | 3 | [0.828674 , 0.862464] | FS |
| 5 | 3 | 4 | 3 | 2 | [0.913703 , 0.933063] | FS |
| 6 | 2 | 2 | 4 | 4 | [0.855885 , 0.881819] | FS |
| 7 | 3 | 4 | 3 | 3 | [0.941422 , 0.956110] | FS |
| 8 | 3 | 3 | 3 | 4 | [0.935551 , 0.952036] | FS |
| 9 | 2 | 2 | 5 | 3 | [0.855989 , 0.881646] | FS |
| 10 | 2 | 3 | 4 | 3 | [0.893680 , 0.913061] | FS |

**Optimal solution:**

Finally, the optimal solution is $x_1 = 5$ , $x_2 = 4$ , $x_3 = 5$ , $x_4 = 3$ with objective function value [ 0.985159,0.990154].

## 8. Conclusion

In this paper, we have proposed a GA based simulations approach for solving redundancy allocation problem. The reliability of a component in a system might be imprecise valued rather than precise for several reasons. This impreciseness may be represented by many ways. In this paper, we have represented this by interval number. For handling the resource constraints, the corresponding problem has been converted into unconstrained optimization problem with the help of Big-M penalty technique. Therefore, the transformed

problem is of unconstrained interval valued optimization problem with integer variables. To solve the transformed problem, we have developed a real coded GA for integer variables. For further research, one may apply other heuristic methods, like, DE, SA, PSO, etc. for solving the problem described in this paper. Again, the proposed approach can be applied in solving the real-life engineering and other optimization problems.

# 9. References

[1] M. S. Chern, On the computational complexity of reliability allocation in series system, Operation Research Letter, 11( 5), (1992), pp. 309-315.

[2] Y. Nakagawa and K. Nakashima, *A Heuristic Method for Determining Optimal Reliability Allocation*, IEEE Transactions on Reliability, 26(3), (1977) , pp. 156-161.

[3] J. H. Kim, and B. J. Yum, *A Heuristic Method for Solving Redundancy Optimization Problems in Complex Systems*, IEEE Transactions on Reliability, 42(**4**), (1993), pp. 572-578.

[4] Kuo, W., C. L. Hwang, and F. A. Tillman, *A Note on Heuristic Methods in Optimal System Reliability*, IEEE Transactions on Reliability, 27, (1978), pp. 320-324.

[5] X.L. Sun, and D. Li, Optimization condition and branch and bound algorithm for constrained redundancy optimization in series system, Optimization and Engineering, 3, (2002), pp. 53-65.

[6] C.L Hwang,., F.A. Tillman, and W. Kuo, *Reliability Optimization by Generalized Lagrangian-function based and reduced-gradient methods*, IEEE Transactions on Reliability, 28, (1979), pp. 316-319.

[7] K. B. Misra, and U. Sharma, An Efficient Algorithm to Solve Integer Programming Problems Arising in System Reliability Design, IEEE Transactions on Reliability, 40, (1991), pp. 81-91.

[8] L. Sahoo, A.K. Bhunia, and, P.K. Kapur, Genetic algorithm based multi-objective reliability optimization in interval environment, Computers and Industrial Engineering, 62, (2012) , pp. 152-160.

[9] R.E. Moore, Methods and Application of Interval Analysis, SIAM Philadelphia, (1979).

[10] E. Hansen, and G.W. Walster, *Global optimization using interval analysis*, Marcel Decker Inc., (2004).

[11] S. K. Mahato, and A. K. Bhunia, *Interval-Arithmetic-Oriented Interval Computing Technique for Global Optimization*, Applied Mathematics Research eXpress, 2006, ( 2006) , pp. 1-19.

[12] R. K., Gupta, A. K. Bhunia, and D. Roy, A GA Based penalty function technique for solving constrained Redundancy allocation problem of series system with interval valued reliabilities of components, Journal of Computational and Applied Mathematics, 232, (2009), pp. 275-284.

[13] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Reading. MA: Addison Wesley, 1989

[14] A. K. Bhunia, L. Sahoo, and D. Roy, Reliability stochastic optimization for a series system with interval component reliability via genetic algorithm, Applied Mathematics and Computations, 216, ( 2010) , pp. 929-939.