Journal of Uncertain Systems
Vol.8, No.2, pp.136-148, 2014
Online at: www.jus.org.uk

Optimization of System Reliability for Series System with Fuzzy Component Reliabilities by Genetic Algorithm

Laxminarayan Sahoo[1,*], Asoke Kumar Bhunia[2], Sanat Kumar Mahato[3]
[1]*Department of Mathematics, Raniganj Girls' College, Raniganj-713358, India*
[2]*Department of Mathematics, The University of Burdwan, Burdwan-713104, India*
[3]*Department of Mathematics, Durgapur Govt. College, Durgapur-713214, India*

Received 26 November 2012; Revised 18 June 2013

Abstract

This paper deals with redundancy allocation problem that maximizes the overall system reliability subject to the given resource constraints where reliability, cost and weight of each component as well as the amount of resources are imprecise and fuzzy valued. For fuzzification of the problem, reliability, cost, weight and amount of resources are assumed to be triangular fuzzy numbers. As a result, this type of problem has been formulated as a fuzzy valued nonlinear integer programming problem. Thereafter, we have used graded mean integration representation of fuzzy number based on the integral value of graded mean α-level of fuzzy number for defuzzifying the objective function as well as the resource constraints. Then the transformed problem has been formulated as an unconstrained integer programming problem with the help of Big-M penalty function technique and to solve it, we have developed genetic algorithm for integer variables with tournament selection, intermediate crossover and one neighborhood mutation. Finally, three numerical examples have been solved and the computational results for crisp problem have been compared with the same available in the existing literature.

Keywords: reliability optimization, fuzzy number, defuzzification, penalty function technique, genetic algorithm

1 Introduction

Development of modern high-tech system design depends on the selection of components and configurations to meet the functional requirements as well as performance specifications. For a system with known reliability, cost, weight, volume and other system parameters, the corresponding design problem becomes a combinatorial optimization problem. The best known reliability design problem of this type is referred to the redundancy allocation problem.

The basic objective of redundancy allocation problem is to find the number of redundant components that either maximize the system reliability or minimize the system cost under several resource constraints. Redundancy allocation problem is basically a nonlinear integer programming problem. Most of these problems can not be solved by direct/indirect or mixed search methods due to discrete search space. According to Chern [4], redundancy allocation problem with multiple constraints is quite often hard to find feasible solutions. This redundancy allocation problem is NP-hard and it has been well discussed in Tillman et al. [29], Kuo and Prasad [11] and Kuo et al. [12]. Earlier, several deterministic methods like the heuristic methods [19, 28], the reduced gradient method [9], the branch and bound method [27, 10, 26], integer programming [16, 13, 8, 18] and other well-developed mathematical programming techniques were used to solve such redundancy allocation problem. However, these methods have both advantages and disadvantages. Dynamic programming is not useful for reliability optimization of a general system as it can be used only for few particular structures of the objective function and constraints that are decomposable. In branch and bound method, the effectiveness depends on sharpness of the bound and required memory that increases exponentially with the problem size. As a result, with the development of genetic algorithm [6] and other evolutionary algorithms, researchers took more attention on redundancy allocation problem as these methods provide more flexibility and require less assumptions on the objective as well as the constraints. These methods are also

[*] Corresponding author. Email: lxsahoo@gmail.com (L. Sahoo).

effective irrespective of whether the search space is discrete or not. In this connection, one may refer to the works of Ravi et al. [20], Shelokar et al. [25], Sheikhalishahi et al. [24], Coelho [5] and Najafi et al. [17].

In the existing literature, in almost all the studies, the design parameters of redundancy allocation problem have usually been taken as precise values. However, in real life situations, design parameters are not precise due to human errors, improper storage facilities and other unexpected factors relating to environment. Therefore, these parameters may be imprecise. 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 those. In an alternative way, imprecise numbers are also represented by deterministic interval numbers. In this area, very few researchers have solved redundancy allocation problems considering the imprecise parameters as deterministic interval numbers. In this regard, the recent works of Bhunia et al. [2], Sahoo et al. [22], Sahoo et al. [21], Gupta et al. [7] are worth mentioning.

In this paper, we have considered the redundancy allocation problem with fuzzy valued reliabilities of components. Then to handle the fuzziness, we have used GMIV technique to defuzzify the fuzzy number. Due to this defuzzification of fuzzy numbers, the objective function as well as the constraints are converted into non-fuzzy valued. Then, the transformed problem is formulated as a non-linear constrained integer programming problem. Thereafter, to solve the constrained optimization problem, we have converted it into an unconstrained one by using the penalty function technique. For solving such optimization problem, we have developed a real coded elitist GA with tournament selection, intermediate crossover and one-neighborhood mutation. Finally, to illustrate the proposed model, three numerical examples have been solved for different cases and the computational results for crisp case have been compared with the existing results.

2 Assumptions and Notations

The following assumptions and notations have been used in the entire paper.

2.1 Assumptions

(i) Reliability of each component is imprecise and fuzzy valued.
(ii) Failures of components are statistically independent.
(iii) The system will not be damaged or failed due to failed components.
(iv) All redundancy is active and there is no provision for repair.
(v) The components as well as the system have two different states, viz. operating state and failure state.

2.2 Notations

n	number of subsystems
x_i	number of components in i-th subsystem
x	$(x_1, x_2, ..., x_n)$, the redundancy vector
r_i	reliability of *i-th* component
R_S	system reliability
S	feasible region
b_j	availability of j-th resource ($j = 1, 2, ..., m$)
l_i, u_i	lower and upper bounds of x_i
$\mu_{\tilde{A}}(x)$	membership function of x of fuzzy set $\tilde{A}$
$L(x)$	left shape function of x of fuzzy set $\tilde{A}$
$R(x)$	right shape function of x of fuzzy set $\tilde{A}$
$P_{dGw}(\tilde{A})$	graded mean integral value of $\tilde{A}$ with degree of optimism w
p_cross	probability of crossover/ crossover rate
p_mute	probability of mutation/ mutation rate
p_size	population size
max_gen	maximum number of generations

3 Representation of Fuzzy Number

The word ‘fuzzy’ was first introduced by Zadeh [31] in the year 1965 in his famous paper “Fuzzy Sets” for representing impreciseness/fuzziness or vagueness mathematically. The approach of fuzzy set is an extension of classical set theory and it is used in fuzzy logic. In classical set theory, the membership of each element in relation to a set is assessed in binary terms according to a crisp conditions; an element either belongs to or does not belong to the set. By contrast, a fuzzy set theory permits the gradual assessment of the membership of each element in relation to a set; this is discussed with the aid of a membership function. Fuzzy set is an extension of classical set theory since, for a certain universe, a membership function may act as an indicator function, mapping all elements to either 1 or 0, as in the classical notation. He used this word to generalize the mathematical concept of the set to fuzzy set or fuzzy subset, where in a fuzzy set, a membership function is defined for each element of the referential set. After Zadeh [31], this subject was enhanced by Zimmermann [32, 33, 34] and Bellman and Zadeh [1]. To tackle the problem with fuzzy parameters, first of all the problem is to be defuzzified. In this defuzzification, there are several methods available in the literature. For this connection, the works of Ming et al. [15], Yager et al. [30], Saneifard [23], Chen and Hsieh [3] are worth mentioning. Chen and Hsieh [3] introduced the graded mean integration representation method based on the integral value of graded mean α -level of generalized fuzzy number for defuzzification of generalized fuzzy number to achieve the computational efficiency.

Fuzzy set: A fuzzy set $\tilde{A}$ in a universe of discourse X is defined as the set of pairs: $\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) : x \in X\},$ where the mapping $\mu_{\tilde{A}} : X \to [0,1]$ is called the membership function or grade of membership of x in $\tilde{A}$.

Convex fuzzy set: A fuzzy set $\tilde{A}$ is called convex if and only if for all $x_1, x_2 \in X,$ $\mu_{\tilde{A}}(\lambda x_1 + (1-\lambda)x_2)$ $\geq \min\{\mu_{\tilde{A}}(x_1), \mu_{\tilde{A}}(x_2)\}$ where $\lambda \in [0,1]$.

Support of a fuzzy set: The support of fuzzy set $\tilde{A}$ denoted by $S(\tilde{A})$ is the crisp set of all $x \in X$ such that $\mu_{\tilde{A}}(x) > 0$.

α **-level set:** The set of elements that belong to the fuzzy set $\tilde{A}$ at least to the degree α is called the α -level set or α -cut, $\tilde{A}_\alpha = \{x \in X : \mu_{\tilde{A}}(x) \geq \alpha\}.$ If $\tilde{A}_\alpha = \{x \in X : \mu_{\tilde{A}}(x) > \alpha\},$ it is called strong α -level set or strong α -cut.

Normal fuzzy set: A fuzzy set $\tilde{A}$ is called a normal fuzzy set if there exists at least one $x \in X$ such that $\mu_{\tilde{A}}(x) = 1$.

Fuzzy number: A fuzzy number is a fuzzy set which is both convex and normal. A fuzzy number is a special case of a fuzzy set. Different definitions and properties of fuzzy numbers are encountered in the literature but they all agree on that a fuzzy number represents the conception of a set of real numbers ‘closer to a’ where ‘a’ is the number being fuzzified.

Graded mean integration representation of fuzzy number: Suppose $\tilde{A}$ is a generalized fuzzy number as shown in Figure 1. It is described as a fuzzy subset of $\mathbb{R}$, whose membership function $\mu_{\tilde{A}}(x)$ is given by

$$\mu_{\tilde{A}}(x) = \begin{cases} L(x) & a \leq x \leq b \\ 1 & b \leq x \leq c \\ R(x) & c \leq x \leq d \\ 0 & \text{otherwise} \end{cases}$$

where $L(x)$ is continuous and strictly monotonic increasing function of x in $a \leq x \leq b$, $R(x)$ is continuous and strictly monotonic decreasing function of x in $c \leq x \leq d$. According to Chen and Hsieh [3], graded mean integral value of $\tilde{A}$ is defined by

$$P_{dGw}(\tilde{A}) = \frac{\int_0^1 x\left\{(1-w)L^{-1}(x) + wR^{-1}(x)\right\} dx}{\int_0^1 x dx} = 2\int_0^1 x\left\{(1-w)L^{-1}(x) + wR^{-1}(x)\right\} dx$$

where the pre-assigned parameter $w \in [0,1]$, a pre-assigned parameter is called degree of optimism. Here, $w = 1$ represents the optimistic decision makers’ viewpoint, $w = 0$ represents a pessimistic viewpoint of the decision maker, and $w = 0.5$ reflects a moderately optimistic decision makers’ point of view.

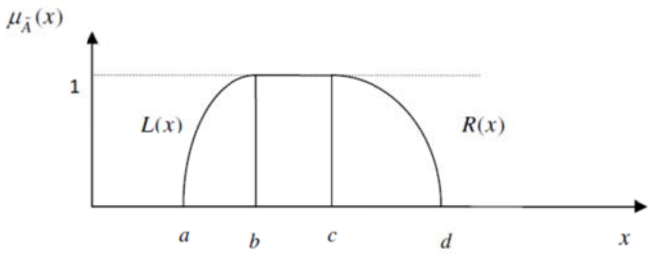

Figure 1: Generalized fuzzy number

Triangular Fuzzy Number (TFN): A TFN $\tilde{A}$ is specified by the triplet (a_1, a_2, a_3) and is defined by its continuous membership function $\mu_{\tilde{A}}(x): X \to [0,1]$ as follows:

$$\mu_{\tilde{A}}(x) = \begin{cases} \dfrac{x-a_1}{a_2-a_1} & \text{if } a_1 \le x \le a_2 \\ 1 & \text{if } x = a_2 \\ \dfrac{a_3-x}{a_3-a_2} & \text{if } a_2 \le x \le a_3 \\ 0 & \text{otherwise.} \end{cases}$$

Here

$$L(x) = \frac{x-a_1}{a_2-a_1} \text{ and } R(x) = \frac{a_3-x}{a_3-a_2}.$$

Therefore,

$$L^{-1}(x) = a_1 + (a_2 - a_1)x \text{ and } R^{-1}(x) = a_3 - (a_3 - a_2)x.$$

GMIV formula for triangular fuzzy numbers with degree of optimism w: If $\tilde{A} = (a_1, a_2, a_3)$ is the Triangular Fuzzy Number, then GMIV of $\tilde{A}$ with the degree of optimism w is given by

$$\begin{aligned} P_{dGw}(\tilde{A}) &= 2\int_0^1 x\left\{(1-w)L^{-1}(x) + wR^{-1}(x)\right\} dx \\ &= 2\int_0^1 x\left\{(1-w)[a_1 + (a_2 - a_1)x] + w[a_3 - (a_3 - a_2)x]\right\} dx \\ &= \frac{1}{3}\left[(1-w)a_1 + 2a_2 + wa_3\right]. \end{aligned}$$

If $a_1 = a_2 = a_3 = a,$ then $P_{dGw}(\tilde{A}) = a$ is the real number $a.$

4 Mathematical Formulation of the Problem

Here, we have considered redundancy allocation problem of parallel-series system. Our objective is to find the optimal number of redundant components $x_i, i = 1, 2, ..., n$ of a given n-stage system (Figure 2), which maximizes system reliability R_S subject to the resource constraints arising on volume, weight and cost.

The mathematical formulation for this problem is as follows:

$$\text{Maximize } R_S = \prod_{i=1}^{n}\left[1-(1-r_i)^{x_i}\right] \tag{1}$$

subject to

$$g_j(x_1, x_2, ..., x_n) \le b_j, \quad j = 1, 2, ..., m$$
$$l_i \le x_i \le u_i, \; i = 1, 2, ..., n.$$

When all the parameters are fuzzy valued number, then the problem (1) reduces to

$$\text{Maximize } \tilde{R}_S = \prod_{i=1}^{n} \left[1-(1-\tilde{r}_i)^{x_i}\right] \tag{2}$$

subject to

$$\tilde{g}_j(x_1, x_2, ..., x_n) \le \tilde{b}_j, \quad j = 1, 2, ..., m$$
$$l_i \le x_i \le u_i, \ i = 1, 2, ..., n.$$

Here $\tilde{R}_S$, $\tilde{r}_i, \tilde{g}_j$ and $\tilde{b}_j$ are all fuzzy valued system reliability, i-th component reliability, j-th constraint and j-th resource availability respectively.

Now, we use graded mean defuzzification method to convert the problem (2) in crisp form. By graded mean defuzzification method, problem (2) reduces to the following form:

$$\text{Maximize } P_{dGw}(\tilde{R}_S) = \prod_{i=1}^{n} \left[1-(1-P_{dGw}(\tilde{r}_i))^{x_i}\right] \tag{3}$$

subject to

$$P_{dGw}(\tilde{g}_j(x_1, x_2, ..., x_n)) \le P_{dGw}(\tilde{b}_j), \quad j = 1, 2, ..., m$$
$$l_i \le x_i \le u_i, \ i = 1, 2, ..., n.$$

Here $P_{dGw}(\tilde{R}_S)$ is the defuzzified/GMIV (with degree of optimism w) valued objective function.

To solve this problem, we have used Genetic Algorithm based constraints handling approach.

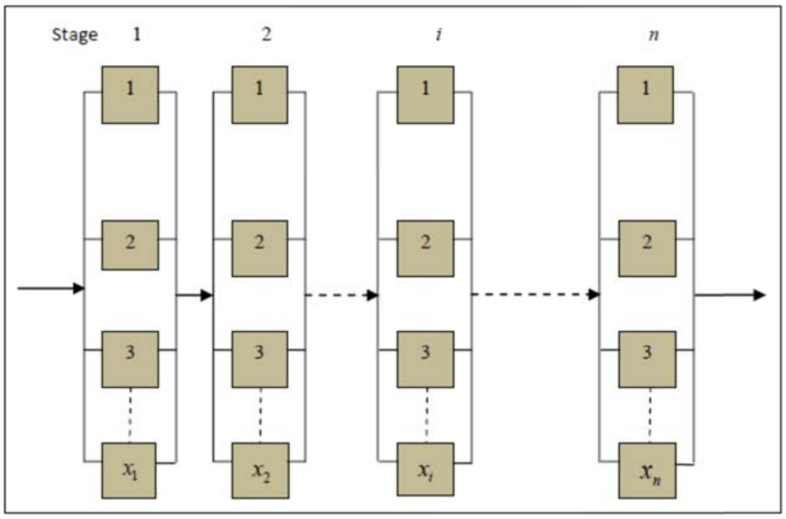

Figure 2: A n-stage parallel-series system

5 Genetic Algorithm based Constraints Handling Approach

Clearly the optimization problem (3) is a constrained optimization problem. During the past, several techniques [14] have been proposed to handle the constraints in genetic algorithms for solving the optimization problem. Recently, Gupta et al. [7] and Bhunia et al. [2] solved the optimization problem using Big-M penalty method. In this method, the given constrained optimization problem is converted to an unconstrained optimization problem by penalizing a large positive number say, M and called this penalty as Big-M penalty. In this work, we have used the Big-M penalty technique.

The converted problem of (3) is as follows:

$$\text{Maximize } \hat{P}_{dGw}(\tilde{R}_S) = P_{dGw}(\tilde{R}_S) + \theta(x) \tag{4}$$

where

$$\theta(x) = \begin{cases} 0 & \text{if } x \in S \\ -P_{dGw}(\tilde{R}_S) - M & \text{if } x \notin S \end{cases}$$

and $S = \left\{ x : P_{dGw}(\tilde{g}_j(x_1, x_2, ..., x_n)) \le P_{dGw}(\tilde{b}_j), j = 1, 2, ..., m \right\}$ is the feasible space.

Problem (4) is an integer non-linear unconstrained optimization problem with defuzzified valued objective of n integer variables $x_1, x_2, ..., x_n$. For solving this problem, we have developed a real coded genetic algorithm (GA) with advanced operators for integer variables.

6 Genetic Algorithm

Genetic algorithm (GA) is a well-known stochastic search iterative method based on the evolutionary theory "survival of the fittest" of Charles Darwin and natural genetics. The most fundamental idea of Genetic Algorithm is to imitate 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. This algorithm starts with an initial population of probable solutions, called individuals, to a given problem where each individual is represented as a chromosome using different form of coding. These chromosomes are evaluated for their fitness. Based on their fitness, chromosomes in the population are to be selected for reproduction and selected individuals are improved by two known genetic operations, viz. crossover and mutation. The crossover operation is applied to create offspring from a pair of selected individuals. The mutation operation is used for a slight modification/change in the individual. The repeated applications of these genetic operators to the relatively fit individuals result in an increase of the average fitness of the population over generations and identification of improved solutions to the problem under investigation. This process is applied iteratively until the termination criterion is satisfied.

The procedural algorithm of the working principle of GA is as follows:

```
Algorithm genetic:
begin
    t ← 0; [ t represents the number of current generation]
    Compute initial population P(t);
    Evaluate the fitness function of P(t);
    Obtain the best found result from P(t);
    while termination criterion not fulfilled do
        t ← t + 1;
        Select P(t) from P(t − 1) by selection process;
        Alter P(t) by crossover and mutation;
        Evaluate the fitness function of P(t);
        Obtain the best found result from P(t) and compare with P(t-1) and store the better one;
        Replace the worst result of P(t) by the best found result of P(t-1) if it is better than that of P(t);
    end while
    Print the best found result;
end
```

To implement the GA, the following basic components are to be considered:

i) GA parameters (population size, maximum number of generations, crossover rate and mutation rate)
ii) Chromosome representation
iii) Initialization of population
iv) Evaluation of fitness function
v) Selection process
vi) Genetic operators (crossover, mutation and elitism).

There are several GA parameters, viz. population size (*p_size*), maximum number of generations (*max_gen*), crossover rate, i.e., the probability of crossover (*p_cross*) and mutation rate, i.e., the probability of mutation (*p_mute*). There is no hard and fast rule for selecting the population size for GA, how large it should be. The population size is problem dependent and will need to increase with the dimensions of the problem. Regarding the maximum number of generations, there is no clear indication for considering this value. It varies from problem to problem and depends upon the number of genes (variables) of a chromosome and prescribed as stopping/termination criteria to make sure that the solution has converged. From natural genetics, it is obvious that the rate of crossover is always greater than that of the rate of mutation. Generally, the crossover rate varies from 0.60 to 0.95 whereas the mutation rate varies from 0.05 to 0.20. Sometimes the mutation rate is considered as $1/n$ where n is the number of genes (variables) of the chromosome.

To represent an appropriate chromosome is an important issue in the application of GA for solving the optimization problem. There are different types of representations, like, binary, real, octal, hexadecimal coding, available in the existing literature. Among these representations, the real coding representation is very popular. In this representation, a chromosome is coded in the form of vector/matrix of integer/ floating point or combination of the both numbers and every component of that chromosome represents a decision variable of the problem. In this representation, each chromosome is encoded as a vector of integer numbers, with the same component as the vector of decision variables of the problem. This type of representation is accurate and more efficient as it is closed to the real design space and moreover, the string length of each chromosome is the number of design variables. In this representation, for a given problem with n decision variables, a n-component vector $x=(x_1,x_2,...,x_n)$ is used as a chromosome to represent a solution to the problem. A chromosome denoted as v_k $(k=1,2,...,p\_size)$ is an ordered list of n genes as $v_k=\{v_{k1},v_{k2},...,v_{ki},...,v_{kn}\}$.

After representation of chromosome, the next step is to initialize the chromosome that will take part in the artificial genetics. To initialize the population, first of all we have to find the independent variables and their bounds for the given problem. Then the initialization process produces population size number of chromosomes in which every component for each chromosome is randomly generated within the bounds of the corresponding decision variable. There are several procedures for selecting a random number of integer types. In this work, we have used the following algorithm for selecting of an integer random number. An integer random number between a and b can be generated as either $x=a+g$, or $x=b-g$ where g is a random integer between 1 and $|a-b|$.

Evaluation/fitness function plays an important role in GA. This role is same for natural evolution process in the biological and physical environments. After initialization of chromosomes of potential solutions, we need to see how relatively good they are. Therefore, we have to calculate the fitness value for each chromosome. In our work, the value of objective function of the reduced unconstrained optimization problems corresponding to the chromosome is considered as the fitness value of that chromosome.

The selection operator which is the first operator in artificial genetics plays an interesting role in GA. This selection process is based on the Darwin’s principle on natural evolution “survival of the fittest”. The primary objective of this process is to select the above average individuals/chromosomes from the population according to the fitness value of each chromosome and eliminate the rest of the individuals/chromosomes. There are several methods for implementing the selection process. In this work, we have used well known tournament selection with size two.

The exploration and exploitation of the solution space can be made possible by exchanging genetic information of the current chromosomes. After the selection process, other genetic operators, like crossover and mutation are applied to the resulting chromosomes those which have survived. Crossover is an operator that creates new individuals/chromosomes (offspring) by combining the features of both parent solutions. It operates on two or more parent solutions at a time and produces offspring for next generation. In this work, we have used intermediate crossover for integer variables.

The aim of mutation operator is to introduce the random variations into the population and is used to prevent the search process from converging to the local optima. This operator helps to regain the information lost in earlier generations and is responsible for fine tuning capabilities of the system and is applied to a single individual only. Usually, its rate is very low; because otherwise it would defeat the order building being generated through the selection and crossover operations. In this work we have used one-neighborhood mutation for integer variables.

In any generation of GA, sometimes there arises a situation when the best chromosome may get lost from the population when a new population is created by crossover and mutation operations. To overcome this situation, the worst individual/chromosome of the current generation is replaced by the best individual/chromosome of previous generation. Instead of single chromosome one or more chromosomes may take part in this operation. This process is named as elitism.

7 Numerical Examples

To illustrate our proposed GA based penalty function technique for solving the reliability optimization problem with fuzzy valued as well as precise/fixed valued reliabilities of components, the three numerical examples of parallel-series system have been considered for different choices of the parametric values.

Example 1:

$$\text{Maximize } R_S = \prod_{i=1}^{4}\left\{1-(1-r_i)^{x_i}\right\}$$

subject to

$$C_S = \sum_{i=1}^{4} C_i x_i \le C$$

$$W_S = \sum_{i=1}^{4} W_i x_i \le W$$

$$x_i \in \mathbb{Z}^+, i = 1, 2, 3, 4.$$

Example 2:

$$\text{Maximize } R_S = \prod_{i=1}^{5}\left\{1-(1-r_i)^{x_i}\right\}$$

subject to

$$V_S = \sum_{i=1}^{5} V_i x_i^2 \le V$$

$$C_S = \sum_{i=1}^{5} C_i \left[x_i + \exp\left(\frac{x_i}{4}\right)\right] \le C$$

$$W_S = \sum_{i=1}^{5} W_i \left[x_i \exp\left(\frac{x_i}{4}\right)\right] \le W$$

$$x_i \in \mathbb{Z}^+, i = 1, 2, ..., 5.$$

Example 3:

$$\text{Maximize } R_S = \prod_{i=1}^{15}\left\{1-(1-r_i)^{x_i}\right\}$$

subject to $C_S = \sum_{i=1}^{15} C_i x_i \le C$

$$W_S = \sum_{i=1}^{15} W_i x_i \le W$$

$$x_i \in \mathbb{Z}^+, i = 1, 2, ..., 15.$$

We have solved the problems considering the following three cases:
Case 1: All parameters are fuzzy valued.
Case 2: Only component reliabilities are fuzzy valued.
Case 3: Crisp parametric values.

The values of various parameters are given in Tables 1-6. The proposed method is coded in C programming language and run in a LINUX environment. The computational work has been done on a PC with Intel core-2 duo processor with 2.5 GHz. For each case, 20 independent runs have been performed to calculate the best found system reliability. The computational results are presented in Tables 7-10. In this computation, we have taken population size, maximum number of generations, crossover rate and mutation rate as 100, 100, 0.85 and 0.15, respectively.

Table 1: Values of different parameters of Example 1 for Cases 1 and 2

	Case 1			Case 2		
i	$\tilde{r}_i$	$\tilde{C}_i$	$\tilde{W}_i$	$\tilde{r}_i$	$\tilde{C}_i$	$\tilde{W}_i$
1	(0.74, 0.80, 0.88)	(0.2, 1.2, 2.4)	(4,5,6)	(0.74, 0.80, 0.88)	(1.2, 1.2, 1.2)	(5, 5, 5)
2	(0.63, 0.70, 0.78)	(2.0, 2.3, 2.8)	(3,4,5)	(0.63, 0.70, 0.78)	(2.3, 2.3, 2.3)	(4, 4, 4)
3	(0.68, 0.75, 0.82)	(3.0, 3.4, 3.9)	(7,8,9)	(0.68, 0.75, 0.82)	(3.4, 3.4, 3.4)	(8, 8, 8)
4	(0.78, 0.85, 0.92)	(4.0, 4.5, 4.8)	(6,7,8)	(0.78, 0.85, 0.92)	(4.5, 4.5, 4.5)	(7, 7, 7)
	$\tilde{C}$ = (50, 56, 60), $\tilde{W}$ = (115, 120, 125)			$\tilde{C}$ = (56, 56, 56), $\tilde{W}$ = (120, 120, 120)		

Table 2: Values of different parameters of Example 2 for Cases 1 and 2

	Case 1				Case 2			
i	$\tilde{r}_i$	$\tilde{V}_i$	$\tilde{C}_i$	$\tilde{W}_i$	$\tilde{r}_i$	$\tilde{V}_i$	$\tilde{C}_i$	$\tilde{W}_i$
1	(0.74, 0.80, 0.88)	(0.4, 1, 1.5)	(6.4, 7, 7.9)	(6.1, 7, 7.6)	(0.74, 0.80, 0.88)	(1, 1, 1)	(7, 7, 7)	(7, 7, 7)
2	(0.82, 0.85, 0.93)	(1.2, 2, 2.7)	(6.3, 7, 7.8)	(7.2, 8, 8.9)	(0.82, 0.85, 0.93)	(2, 2, 2)	(7, 7, 7)	(8, 8, 8)
3	(0.83, 0.90, 0.98)	(2.3, 3, 3.8)	(4.6, 5, 5.7)	(7.4, 8, 8.7)	(0.83, 0.90, 0.98)	(3, 3, 3)	(5, 5, 5)	(8, 8, 8)
4	(0.61, 0.65, 0.79)	(3.4, 4, 4.9)	(8.2, 9, 9.7)	(5.7, 6, 6.9)	(0.61, 0.65, 0.79)	(4, 4, 4)	(9, 9, 9)	(6, 6, 6)
5	(0.72, 0.75, 0.84)	(1.2, 2, 2.9)	(3.2, 4, 4.7)	(8.2, 9, 9.7)	(0.72, 0.75, 0.84)	(2, 2, 2)	(4, 4, 4)	(9, 9, 9)
	$\tilde{V}$ = (100, 110, 115), $\tilde{C}$ = (165, 175, 180), $\tilde{W}$ = (195, 200, 210)				$\tilde{V}$ = (110,110,110), $\tilde{C}$ = (175,175,175), $\tilde{W}$ = (200,200,200)			

Table 3: Values of different parameters of Example 1 for Case 3

i	$\tilde{r}_i$	$\tilde{C}_i$	$\tilde{W}_i$
1	(0.80, 0.80, 0.80)	(1.2, 1.2, 1.2)	(5, 5, 5)
2	(0.70, 0.70, 0.70)	(2.3, 2.3, 2.3)	(4, 4, 4)
3	(0.75, 0.75, 0.75)	(3.4, 3.4, 3.4)	(8, 8, 8)
4	(0.85, 0.85, 0.85)	(4.5, 4.5, 4.5)	(7, 7, 7)

Table 4: Values of different parameters of Example 2 for Case 3

i	$\tilde{r}_i$	$\tilde{V}_i$	$\tilde{C}_i$	$\tilde{W}_i$
1	(0.80, 0.80, 0.80)	(1, 1, 1)	(7, 7, 7)	(7, 7, 7)
2	(0.85, 0.85, 0.85)	(2, 2, 2)	(7, 7, 7)	(8, 8, 8)
3	(0.90, 0.90, 0.90)	(3, 3, 3)	(5, 5, 5)	(8, 8, 8)
4	(0.65, 0.65, 0.65)	(4, 4, 4)	(9, 9, 9)	(6, 6, 6)
5	(0.75, 0.75, 0.75)	(2, 2, 2)	(4, 4, 4)	(9, 9, 9)

Table 5: Values of different parameters of Example 3 for Cases 1 and 2

	Case 1			Case 2		
i	$\tilde{r}_i$	$\tilde{C}_i$	$\tilde{W}_i$	$\tilde{r}_i$	$\tilde{C}_i$	$\tilde{W}_i$
1	(0.80, 0.90, 0.98)	(4, 5, 6)	(7, 8, 9)	(0.80, 0.90, 0.98)	(5, 5, 5)	(8, 8, 8)
2	(0.60, 0.75, 0.90)	(2, 4, 5)	(8, 9, 11)	(0.60, 0.75, 0.90)	(4, 4, 4)	(9, 9, 9)
3	(0.50, 0.65, 0.75)	(8, 9, 11)	(4, 6, 7)	(0.50, 0.65, 0.75)	(9, 9, 9)	(6, 6, 6)
4	(0.70, 0.80, 0.90)	(6, 7, 9)	(5, 7, 9)	(0.70, 0.80, 0.90)	(7, 7, 7)	(7, 7, 7)
5	(0.70, 0.85, 0.95)	(5, 7, 9)	(7, 8, 10)	(0.70, 0.85, 0.95)	(7, 7, 7)	(8, 8, 8)
6	(0.85, 0.93, 0.99)	(3, 5, 8)	(6, 8, 10)	(0.85, 0.93, 0.99)	(5, 5, 5)	(8, 8, 8)
7	(0.70, 0.78, 0.85)	(4, 6, 7)	(8, 9, 10)	(0.70, 0.78, 0.85)	(6, 6, 6)	(9, 9, 9)
8	(0.55, 0.66, 0.75)	(7, 9, 10)	(5, 6, 7)	(0.55, 0.66, 0.75)	(9, 9, 9)	(6, 6, 6)
9	(0.70, 0.78, 0.90)	(2, 4, 7)	(6, 7, 9)	(0.70, 0.78, 0.90)	(4, 4, 4)	(7, 7, 7)
10	(0.80, 0.91, 0.98)	(4, 5, 7)	(6, 8, 9)	(0.80, 0.91, 0.98)	(5, 5, 5)	(8, 8, 8)
11	(0.75, 0.79, 0.90)	(5, 6, 7)	(8, 9, 10)	(0.75, 0.79, 0.90)	(6, 6, 6)	(9, 9, 9)
12	(0.60, 0.77, 0.85)	(6, 7, 9)	(5, 7, 9)	(0.60, 0.77, 0.85)	(7, 7, 7)	(7, 7, 7)
13	(0.60, 0.67, 0.80)	(7, 9, 12)	(4, 6, 8)	(0.60, 0.67, 0.80)	(9, 9, 9)	(6, 6, 6)
14	(0.70, 0.79, 0.90)	(6, 8, 9)	(4, 5, 6)	(0.70, 0.79, 0.90)	(8, 8, 8)	(5, 5, 5)
15	(0.55, 0.67, 0.80)	(5, 6, 8)	(5, 7, 9)	(0.55, 0.67, 0.80)	(5, 5, 5)	(7, 7, 7)
	$\tilde{C}$ = (395, 400, 405), $\tilde{W}$ = (410, 414, 420)			$\tilde{C}$ = (400, 400, 400), $\tilde{W}$ = (414, 414, 414)		

Table 6: Values of different parameters of Example 3 for Case 3

i	$\tilde{r}_i$	$\tilde{C}_i$	$\tilde{W}_i$
1	(0.90, 0.90, 0.90)	(5, 5, 5)	(8, 8, 8)
2	(0.75, 0.75, 0.75)	(4, 4, 4)	(9, 9, 9)
3	(0.65, 0.65, 0.65)	(9, 9, 9)	(6, 6, 6)
4	(0.80, 0.80, 0.80)	(7, 7, 7)	(7, 7, 7)
5	(0.85, 0.85, 0.85)	(7, 7, 7)	(8, 8, 8)
6	(0.93, 0.93, 0.93)	(5, 5, 5)	(8, 8, 8)
7	(0.78, 0.78, 0.78)	(6, 6, 6)	(9, 9, 9)
8	(0.66, 0.66, 0.66)	(9, 9, 9)	(6, 6, 6)
9	(0.78, 0.78, 0.78)	(4, 4, 4)	(7, 7, 7)
10	(0.91, 0.91, 0.91)	(5, 5, 5)	(8, 8, 8)
11	(0.79, 0.79, 0.79)	(6, 6, 6)	(9, 9, 9)
12	(0.77, 0.77, 0.77)	(7, 7, 7)	(7, 7, 7)
13	(0.67, 0.67, 0.67)	(9, 9, 9)	(6, 6, 6)
14	(0.79, 0.79, 0.79)	(8, 8, 8)	(5, 5, 5)
15	(0.67, 0.67, 0.67)	(6, 6, 6)	(7, 7, 7)
$\tilde{C}$ = (400, 400, 400), $\tilde{W}$ = (414, 414, 414)			

Table 7: Computational results of Examples 1 and 2 for Case 1

	Example 1				Example 2				
	x	R_S	C_S	W_S	x	R_S	V_S	C_S	W_S
w=0	(5,7,5,4)	0.996709	53.403	114.007	(3,2,2,3,3)	0.888137	74.997	141.0333	186.7051
w=0.5	(5,6,5,4)	**0.997522**	55.116	117.000	(3,2,2,3,3)	0.914625	78.285	140.6763	198.6564
w=1	(5,5,5,4)	0.997468	56.037	121.660	(3,2,2,3,4)	**0.945221**	108.003	155.688	201.6754

Table 8: Computational results of Examples 1 & 2 for Case 2

	Example 1				Example 2				
	x	R_S	C_S	W_S	x	R_S	V_S	C_S	W_S
w=0	(5,6,5,4)	0.995943	54.8	117	(3,2,2,3,3)	0.888137	83	146.125	166.589
w=0.5	(5,6,5,4)	0.997522	54.8	117	(3,2,2,3,3)	0.914625	83	146.125	166.589
w=1	(5,6,5,4)	**0.998569**	54.8	117	(3,2,2,3,3)	**0.937352**	83	146.125	166.589

Table 9: Computational results of Example 3 for Case 1

	x	R_S	C_S	W_S
w=0	(3, 4, 6, 4, 3, 3, 4, 6, 4, 3, 3, 5, 5, 4, 5)	0.93461492	386.32	412.00
w=0.5	(3, 4, 6, 4, 3, 2, 3, 5, 4, 3, 3, 4, 5, 4, 5)	0.94506465	393.89	413.34
w=1	(2, 4, 5, 3, 3, 2, 4, 5, 3, 2, 3, 4, 5, 4, 5)	**0.95529990**	399.36	413.68

Table 10: Computational results of Example 3 for Case 2

	x	R_S	C_S	W_S
w=0	(3, 4, 6, 4, 3, 2, 3, 5, 4, 3, 3, 4, 5, 4, 5)	0.90620540	391	413
w=0.5	(3, 4, 6, 4, 3, 2, 3, 5, 4, 3, 3, 4, 5, 4, 5)	0.94506465	391	413
w=1	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	**0.97045922**	392	414

Table 11: Comparison of the best found results of Example 1 and 2

		Example 1		Example 2	
Method	Case	x	R_S	x	R_S
Proposed Method	Case 1	(5, 6, 5, 4)	**0.997522**	(3, 2, 2, 3, 3)	**0.945221**
	Case 2	(5, 6, 5, 4)	**0.998569**	(3, 2, 2, 3, 3)	**0.937352**
	Case 3	(5, 6, 5, 4)	**0.997470**	(3, 2, 2, 3, 3)	**0.904467**
ACO[25]	-	(5, 6, 5, 4)	0.997500	(3, 2, 2, 3, 3)	0.904500
SA[20]	-	(5, 6, 5, 4)	0.997500	(3, 2, 2, 3, 3)	0.904500
I-NESA[20]	-	(5, 6, 5, 4)	0.997500	(3, 2, 2, 3, 3)	0.904500
NLIP [13]	-	(5, 6, 5, 4)	0.997500	(3, 2, 2, 3, 3)	0.904500

Table 12: Comparison of the best found results of Example 3

Method	Case	x	R_S	C_S	W_S
Proposed Method	Case 1	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	**0.95529990**	399.36	413.68
	Case 2	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	**0.97045922**	392.00	414.00
	Case 3	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	**0.94561336**	392.00	414.00
ACO[25]	-	(3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5)	0. 94561300	392	414
SA[20]	-	(2, 4, 5, 4, 3, 2, 4, 5, 4, 3, 3, 4, 5, 5, 4)	0. 94325900	380	414
I-NESA[20]	-	(3, 4, 5, 3, 3, 2, 4, 5, 4, 3, 3, 4, 5, 5, 5)	0. 94474900	389	414
NLIP [13]	-	(3, 4, 5, 3, 3, 2, 4, 5, 4, 3, 3, 4, 5, 5, 5)	0. 94474900	389	414

We have compared our results for these examples with the results of some earlier results and are presented in Tables 11 and 12. From Tables 11 and 12, it is observed that the computational results are either better or same in different cases with the existing ones. For Example 1 and 3, the respective best results are obtained when only reliability components are considered as fuzzy numbers with moderately decision makers' point of view while in Example 2, the largest value of system reliability is obtained when all the parameters are fuzzy valued with optimistic decision makers' point of view.

8 Concluding Remarks

In this paper, reliability redundancy allocation problem with imprecise parameters has been formulated to maximize the system reliability subject to the given resource constraints. Here the values of imprecise parameters are considered as triangular fuzzy numbers and the corresponding problem has been formulated as a fuzzy valued nonlinear integer programming problem. Then the problem has been converted to crisp one by defuzzification technique with the help of graded mean integration method. Then the transformed problem has been solved with respect to optimistic, pessimistic and moderately optimistic decision makers' point of view. We have also solved the crisp problem to compare the results with the same of existing methods. From the simulation results, it is observed that the results in cases 1 and 2 are better than that of in crisp case.

For further research, one may use the proposed technique to formulate and solve the other reliability optimization problems (single or multi-objective).

Acknowledgements

The authors are grateful for the valuable comments and suggestions from the respected reviewer. These important comments and suggestions have enriched the strength and significance of our paper. The second author would like to acknowledge the financial support provided by the Council of Scientific and Industrial Research (CSIR), New Delhi, India, for carrying out the research works presented in this paper.

References

[1] Bellman, R.E., and L.A. Zadeh, Decision making in a fuzzy environment, *Management Science*, vol.17, no.4, pp.B141–B164, 1970.

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

[3] Chen, S.H., and C.H. Hsieh, Graded mean integration representation of generalized fuzzy numbers, *Jounal of Chinese Fuzzy Systems Association*, vol.5, no.2, pp.1–7, 1999.

[4] Chern, M.S., On the computational complexity of reliability redundancy allocation in a series system, *Operations Research Letters*, vol.11, no.5, pp.309–315, 1992.

[5] Coelho, L.S., An efficient particle swarm approach for mixed-integer programming in reliability-redundancy optimization applications, *Reliability Engineering and System Safety*, vol.94, pp.830–837, 2009.

[6] Goldberg, D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Longman Publishing Co., Boston, 1989.

[7] Gupta, R.K., Bhunia, A.K., 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*, vol.232, no.2, pp.275–284, 2009.

[8] Hikita, M., Nakagawa, Y., Nakashima, K., and H. Narihisa, Reliability optimization of systems by a surrogate-constraints algorithm, *IEEE Transactions on Reliability*, vol.41, no.3, pp.473–480, 1992.

[9] Hwang, C.L., Tillman, F.A., and W. Kuo, Reliability optimization by generalized Lagrangian-function and reduced-gradient methods, *IEEE Transactions on Reliability*, vol.28, no.4, pp.316–319, 1979.

[10] Kuo, W., Lin, H., Xu, Z., and W. Zhang, Reliability optimization with the Lagrange-multiplier and Branch-and-Bound technique, *IEEE Transactions on Reliability*, vol.36, no.5, pp.624–630, 1987.

[11] Kuo, W., and V.R. Prasad, An annotated overview of system reliability optimization, *IEEE Transactions on Reliability*, vol.49, no.2, pp.487–493, 2000.

[12] Kuo, W., Prasad, V.R., Tillman, F.A., and C.L. Hwang, *Optimal Reliability Design: Fundamentals and Applications*, Cambridge University Press, 2001.

[13] Luus, R., Optimization of system reliability by a new nonlinear integer programming procedure, *IEEE Transactions on Reliability*, vol.24, no.1, pp.14–16, 1975.

[14] Miettinen, K., Mäkelä, M.M., and J. Toivanen, Numerical comparison of some penalty-based constraint handling techniques in genetic algorithms, *Journal of Global Optimization*, vol.27, pp.427–446, 2003.

[15] Ming, M., Kandel, A., and M. Friedman, A new approach for defuzzification, *Fuzzy Sets and Systems*, vol.111, pp.351–356, 2000.

[16] Misra, B., and U. Sharma, An efficient algorithm to solve integer-programming problems arising in system-reliability design, *IEEE Transactions on Reliability*, vol.40, no.1, pp.81–91, 1991.

[17] Najafi, A.A., Karimi, H., Chambari, A., and F. Azimi, Two metaheuristics for solving the reliability redundancy allocation problem to maximize mean time to failure of series-parallel system, *Scientia Iranica*, 2012, doi:10.1016/j.scient.2012.12.022.

[18] Nakagawa, Y., and S. Miyazaki, Surrogate constraints algorithm for reliability optimization problems with two constraints, *IEEE Transactions on Reliability*, vol.30, no.2, pp.181–184, 1981.

[19] Nakagawa, Y., and K. Nakashima, A heuristic method for determining optimal reliability allocation, *IEEE Transactions on Reliability*, vol.26, no.3, pp.156–161, 1977.

[20] Ravi, V., Murty, B.S.N., and P.J. Reddy, Nonequilibrium simulated-annealing algorithm applied to reliability optimization of complex systems, *IEEE Transactions on Reliability*, vol.46, no.2, pp.233–239, 1997.

[21] Sahoo, L., Bhunia, A.K., and P.K. Kapur, Genetic algorithm based multi-objective reliability optimization in interval environment, *Computer and Industrial Engineering*, vol.62, no.1, pp.152–160, 2012.

[22] Sahoo, L., Bhunia, A.K., and D. Roy, A genetic algorithm based reliability redundancy optimization for interval valued reliabilities of components, *Journal of Applied Quantitative Methods*, vol.5, no.2, pp.270–287, 2010.

[23] Saneifard, R., A method for defuzzification by weighted distance, *International Journal of Industrial Mathematics*, vol.3, pp.209–217, 2009.

[24] Sheikhalishahi, M., Ebrahimipour, V., Shiri, H., Zaman, H., and M. Jeihoonian, A hybrid GA-PSO approach for reliability optimization in redundancy allocation problem, *The International Journal of Advanced Manufacturing Technology*, vol.68, pp.317–338, 2013.

[25] Shelokar, P.S., Jayaraman, V.K., and B.D. Kulkarni, Ant algorithm for single and multiobjective reliability optimization problems, *Quality and Reliability Engineering*, vol.18, no.6, pp.497–514, 2002.

[26] Sun, X., and D. Li, Optimal condition and Branch and Bound algorithm for constrained redundancy optimization in series system, *Optimization and Engineering*, vol.3, pp.53–65, 2002.

[27] Sung, C.S., and Y.K. Cho, Branch and Bound redundancy optimization for a series system with multiple-choice constraints, *IEEE Transactions on Reliability*, vol.48, no.2, pp.108–117, 1999.

[28] Tian, Z., Levitin, G., and M.J. Zuo, A joint reliability-redundancy optimization approach for multi-state series-parallel systems, *Reliability Engineering and System Safety*, vol.94, pp.1568–1576, 2009.

[29] Tillman, F.A., Hwang, C.L., and W. Kuo, Optimization techniques for system reliability with redundancy: a review, *IEEE Transactions on Reliability*, vol.26, no.3, pp.148–155, 1977.

[30] Yager, R.R., and D.P. Filev, On the issue of defuzzification and selection based on a fuzzy set, *Fuzzy Sets and Systems*, vol.55, pp.255–272, 1993.

[31] Zadeh, L.A., Fuzzy sets, *Information and Control*, vol.8, pp.338–353, 1965.

[32] Zimmermann, H.J., Description and optimization of fuzzy systems, *International Journal of General Systems*, vol.2, pp.209–215, 1976.

[33] Zimmermann, H.J., Fuzzy programming and linear programming with several objective functions, *Fuzzy Sets and Systems*, vol.1, pp.45–55, 1978.

[34] Zimmermann, H.J., *Fuzzy Set Theory and Its Applications*, Kluwer Academic Press, Dordrecht, 1991.