International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2014): 5.611

Maximizing System Reliability of a Four Stage RAP with Two Chance Constraints Having Stochastic Reliability Components

Sanat Kumar Mahato

Department of Mathematics, Mejia Govt. College, West Bengal-722143, India

Abstract: ***This paper deals with the solution of the chance constrained reliability-redundancy allocation problems in imprecise environment. The reliabilities of the components are imprecise numbers and the type of the constraints are chance constraints i.e., stochastic in nature. This paper proposes a stochastic simulation based genetic algorithm approach for solving the reliability optimization problems of the type mentioned. The impreciseness is represented in the stochastic approach. In case of stochastic approach, the reliabilities of the components are taken to be random variables having normal distribution. Then Monte Carlo simulation technique is applied to transform the chance constraints into the deterministic ones. Then Big-M penalty technique is applied to transform the problem to unconstrained one. The altered problem is then solved by the real coded genetic algorithm based on stochastic simulation. Some numerical illustrations are presented to show the performance of the proposed procedure.***

Keywords: System Reliability Optimization, Real Coded Genetic Algorithm, Chance Constraint, Stochastic Simulation Technique, Big-M Penalty Technique.

1. Introduction

In the most of the existing stochastic optimization methods of reliability engineering it has been assumed that all the component reliabilities are precise and fixed real numbers lying in [0,1]. This means that there exists the complete probabilistic information about the component as well as system behavior. This information is dependent on the following two conditions, viz., all the probabilities or probability distributions are known or perfectly determinable, and the system components are independent i.e., all the random variables, describing the component reliability behavior are independent.

Almost in all in the techniques available for reliability optimization, the assumption on uncertainty is based on precise probabilities and the reliabilities of system components are to be known and fixed positive numbers which lying in the interval [0, 1] and well discussed in [4], [8], [15], [16], [17], [22], [23], [24], [31], [32], [33]. The precise system reliability can be computed theoretically if both the above-mentioned conditions are satisfied. However, in the most of the cases of real-life situations, where either the system is newly invented or it persists only as a project, there does not exist sufficient statistical information. This means that only some partial information about the system components is known. So the reliability of the components of a system might be an imprecise numbers rather than precise. To embark upon the problem with such imprecise numbers, generally stochastic, fuzzy and fuzzy-stochastic approaches are applied and the corresponding problems are converted into deterministic problems for solving them. In fuzzy approach, the parameters, constraints and goals are considered as either fuzzy sets with known membership functions or fuzzy numbers whereas in stochastic approach, the system parameters are to be random variables with known probability distributions. On the other hand, in fuzzy-stochastic approach, some of the parameters are considered as fuzzy sets/fuzzy numbers and others as random variables. Apart from these approaches, another one approach, viz. interval approach may be applied for the same purpose. In this approach, an interval number is used to represent the imprecise number. In this area, only a very few works has been done considering the system parameters as interval valued. In this connection, the works of Gupta et. [6], Bhunia et al. [1], Sahoo et al. [27], Bhunia and Sahoo [2], Sahoo et al. [26], Sahoo et al. [28], Mahato et al. [19], Sahoo et al. [29], [30] are worth mentioning.

In this paper, we have proposed a stochastic simulation based genetic algorithm approach [11], [12], for solving the chance constrained [4], [10], [13], [21], reliability optimization problem considering the reliability of each component of a system as either precise or imprecise number lying in $[0,1]$. To symbolize this impreciseness, we have applied stochastic approach. In the stochastic approach, the reliability of each component is considered as a random variable with normal distribution. Firstly, the chance constraints have been transformed into deterministic constraints by Monte Carlo simulation technique [25]. Then the transformed problem has been solved by real coded genetic algorithm based constrained handling technique. Finally, to illustrate the methodology as well as to test the performance of the proposed technique a numerical example has been solved for different set of input data and sensitivity of some parameters are also presented in tabular form.

2. Assumptions

In formulating the chance constraints based stochastic reliability-redundancy optimization problem, the following assumptions have been considered.

a) Reliability of each component is stochastic in nature.
b) Failure of a component of any subsystem will not lead the entire system to failure.
c) Redundancies are active without repair.
d) The components and also the system will have only two states either operating or failure.
e) The resource constraints are chance constraints with resource vector as stochastic in nature.
f) The coefficients in the left hand side of the resource constraints are stochastic in nature.

3. Notations

The notations which have been used in the whole paper are given in the table below.

Notations	**Descriptions**
n	number of subsystems
m	number of constraints
x_j	number of redundant components in the *j*-th subsystem
$x=(x_1,x_2,...,x_n)$	redundant vector
r_j	reliability of each component in the *j*-th subsystem which is precise
$r_j \sim N(\mu_j,\delta_j)$	reliability of each component in the *j*-th subsystem which is stochastic in nature and follows normal distribution with parameters μ_{r_j} and σ_{r_j}
$R_S(x), \tilde{R}_S(x)$	precise system reliability, stochastic system reliability
$g_i(x), \tilde{g}_i(x)$	precise and stochastic valued usability of *i*-th constraint respectively
$b_i, \tilde{b}_i$	precise and stochastic valued total availability of *i*-th resource
$l_j(\geq 1)$, u_j	Lower and upper bounds of x_j
γ_i	level of significance of *i*-th chance constraint
Prob(.)	probability of (.)
$U(a,b)$	uniform distribution over [*a, b*]
$N(\mu,\sigma)$	normal distribution with parameters μ (mean) and σ (standard deviation)
p_s	population size
m_g	maximum number of generations
p_m	probability of mutation
p_c	probability of crossover

4. Some Useful Probability Distributions

4.1 Uniform Distribution

A continuous random variable X over the interval $[a,b]$ is said to follow uniform distribution and denoted by $X \sim U(a,b)$, if its probability density function $f(x)$ is given by

$$f(x)=\begin{cases}\frac{1}{b-a} & \text{if } x\in[a,b]\\ 0 & \text{otherwise.}\end{cases}$$

4.2 Normal Distribution

A continuous random variable X is said to follow normal distribution with parameters μ (mean) and σ (standard deviation), denoted as $X \sim N(\mu,\sigma)$, if its probability density function $f(x)$ is given by

$$f(x)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2\sigma^2}(x-\mu)^2\right];$$
$$-\infty < x,\mu < \infty,\ \sigma > 0.$$

5. Random Numbers Generation

In stochastic simulation, random number plays an important role. So, the generation of random numbers is an essential part of simulation. For this purpose, Jana and Biswal [11], [12] proposed several algorithms based on different probability distributions.

The sub function for the generation of pseudo random numbers between 0 and RAND_MAX has been given in the C library as follows:
#include<stdlib.h>
int rand (void).
where the value of RAND_MAX is defined in <stdlib.h>. Hence, a uniformly distributed random number can be generated from the given interval [a, b] according to the following algorithm.

Algorithm for uniform distribution

Step 1: $m_1 = rand()$
Step 2: $m = m_1 / \text{RAND\_MAX}$
Step 3: Return $a+m(b-a)$.
We denote this random number generator as *U(a,b).*

Algorithm for normal distribution

Based on normal distribution, a random number between $[\mu-\sigma,\mu+\sigma]$ can be generated according to the following algorithm
Step 1: Generate m_2 and m_3 from $U(0,1)$.
Step 2: Compute $y=\left[-2\log_e(m_2)\right]^{\frac{1}{2}}\sin(2\pi m_3)$.
Step 3: Return $(\mu+\sigma y)$.
We denote this $N(\mu,\sigma)$.

6. Mathematical Formulation of the Problem

Let us consider a *n*-stage parallel-series system as shown in Fig.-1. This system is comprised of *n* subsystems connected in series, where *j*-th subsystem consists of x_j number of identical components connected in parallel. Assuming the reliability of each component as precise (fixed), we get the system reliability $R_S(x)$ as

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

We desire to maximize the system reliability subject to several resource constraints. Sometimes, constraints are satisfied depending on the chance. This type of constraints is known as chance constraints. In this case, each constraint is considered as an event of a random experiment.

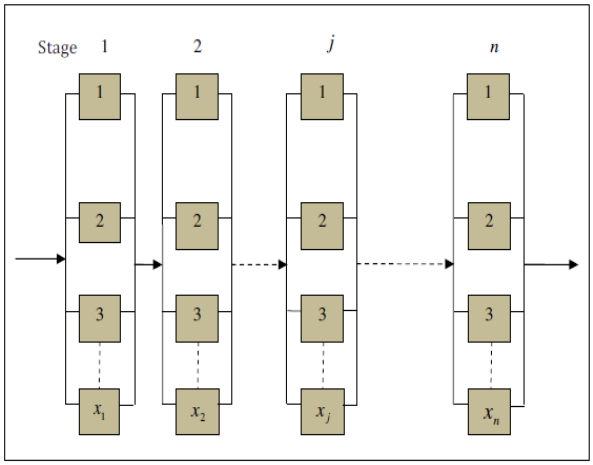

Figure 1: An n-stage parallel series system

Then, the chance constrained reliability-redundancy allocation problem for this parallel-series system with m constraints can be formulated as follows:

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

subject to

$$\text{Prob}\left[g_i(x) \le b_i\right] \ge 1-\gamma_i,\ \ i=1,2,...,m$$

and $l_j \le x_j \le u_j,\ \ j=1,2,...,n.$

In problem (1), it is to be noted that all the parameters are assumed to be precise.

Now, if the reliabilities of components in problem (1) are stochastic in nature and follow normal distribution, then the chance constrained stochastic reliability optimization problem becomes

$$\text{Maximize } \tilde{R}_S(x) = \prod_{j=1}^{n}\left[1-(1-\tilde{r}_j)^{x_j}\right] \qquad (2)$$

subject to

$$\text{Prob}\left[g_i(x) \le \tilde{b}_i\right] \ge 1-\gamma_i,\ \ i=1,2,...,m$$

and $l_j \le x_j \le u_j,\ \ j=1,2,...,n.$

where $\tilde{r}_j \sim N\left(\mu_{r_j}, \sigma_{r_j}\right),\ \ j=1,2,...,n.$ and

$\tilde{b}_i \sim N\left(\mu_{b_i}, \sigma_{b_i}\right),\ \ i=1,2,...,m.$

We aim to solve the problem (2) for different set of data. The problem is nonlinear all integer programming problem with chance constraints. By transforming the problem into deterministic problem, the reduced problems can be solved by existing methods. However, in an alternative way, we can solve the same by stochastic simulation based genetic algorithm technique.

7. Stochastic Simulations

In the stochastic simulation for chance constrained optimization problem, at first the stochastic constraints are converted into their respective deterministic equivalent forms to the given level of confidence. Let us consider the chance constraints as follows:

$$\text{Prob}\left[g_i(x,r) \le b_i\right] \ge 1-\gamma_i,\ 0<\gamma_i<1,\ i=1,2,...,m$$

where $r=(r_1,r_2,....,r_n)$ is a n-dimensional continuous vector and each r_i has a known distribution. Monte Carlo simulation technique is used for estimating these chance constraints for a given x. Let us generate N independent vectors $r^{(s)}=\left(r_1^{(s)},r_2^{(s)},....,r_n^{(s)}\right),\ \ s=1,2,...,N$

from their probability distributions. Let $N_i'(i=1,2,...,m)$ be the number of occurrences when all the constraints $g_i\left(x,r^{(s)}\right) \le b_i (i=1,2,...,m)$ are satisfied. Then by the definition of probability we have

$$\frac{N_i'}{N} \ge 1-\gamma_i,\ \ i=1,2,...,m. \qquad (3)$$

A solution is said to be feasible, if the condition (3) is satisfied for all i $(i=1,2,3,...,m)$.

The algorithm for calculating the value of N_i'/N from the given chance constraints is as follows:

Step-1: Initialize $N_i'=0(i=1,2,...,m)$.

Step-2: Generate random numbers according to the known distribution of the random variables R_i.

Step-3: Find all the values of $g_i(x,r)$, ***for all*** $i,\ i=1,2,...,m.$

Step-4: If $g_i\left(x,r^{(s)}\right) \le b_i$, ***then*** $N_i'=N_i'+1, i=1,2,...,m.$

Step-5: Repeat Steps 2 -4 for N times.

Step-6: Find the ratio $N_i'/N, i=1,2,...,m.$

Step-7: Stop.

8. Big-M Penalty Technique

It is to be observed that the optimization problem (2) is constrained optimization problem. During the past, several techniques [20] have been proposed to handle the constraints in genetic algorithms for solving the optimization problem. Recently, Gupta et al. [6] and Bhunia et al. [1] solved the optimization problem using Big-M penalty method. In this method, the given constrained optimization problem is converted into 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 transformed problem of (2) is as follows:

$$\text{Maximize } \hat{\tilde{R}}_S(x) \qquad (4)$$

$$\text{where } \hat{\tilde{R}}_S(x) = \begin{cases} \tilde{R}_S & \text{if } x \in S \\ -M & \text{if } x \notin S \end{cases}$$

and $S = \left\{ x : \text{Prob}\left[g_i(x) \le \tilde{b}_i \right] \ge 1 - \gamma_i,\ \ i = 1,2,...,m \right\}$ be the feasible space.

Problem (4) is integer non-linear unconstrained optimization problem. For solving this problem, we have developed stochastic simulation based genetic algorithm (GA) with advanced operators for integer variables.

9. The Genetic Algorithm

The different steps of genetic algorithm [5], [9] are given in the following algorithm:

The Algorithm

Step-1: Initialize GA parameters (p_s, m_g, p_m and p_c) and the bounds of each decision variables.
Step-2: Set iteration $= 0$.
Step-3: Initialize the population i.e. p_s number of chromosomes.
Step-4: Set iteration $= iteration + 1$.
Step-5: Check the constraints using stochastic simulation.
Step-6: Evaluate fitness function for each chromosome.
Step-7: Use tournament selection to select chromosomes having better fitness values.
Step-8: Apply crossover, mutation and elitist operators to update the chromosomes.
Step-9: If iteration $< m_g$ ***, go to Step-4; otherwise go to Step-10.***
Step-10: Print the best chromosome along with the fitness value.
Step-11: Stop.

There are several GA parameters, viz. population size (p_s), maximum number of generation (m_g), crossover rate i.e., the probability of crossover (p_c) and mutation rate i.e., the probability of mutation (p_m). 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 dimension 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.

At the beginning, GA needs the initialization of the population of solutions. If $x_1, x_2, ..., x_n$ be the decision variables of the optimization problem to be solved, then each chromosome can be represented as $X_p = (x_1, x_2, ..., x_n)_p$, $p = 1,2,..., p_s$. Here the integer values of $x_j \left(j = 1,2,...,n \right)$ are initialized uniformly between l_j and $u_j \left(j = 1,2,...,n \right)$. There are several procedures for selecting a random number of integer types. In this work, we have used the following algorithm for selecting an integer number randomly. A random 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 $|a - b|$.

As the constraints of the problem are chance constraints with some known degree of significance, stochastic simulation technique has been applied for checking the constraints.

Fitness function plays an important role in GA. This role is same for natural evolution process in the biological and physical environments. In our work, the value of objective function of the 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 well known 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 the 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.

10. Numerical Examples

To illustrate the methodology and also to test the performance of the proposed algorithm, we have solved the example for several stages and with different set of input data. In this example, component reliabilities, coefficients of the chance constraints and available resources are normally distributed.

Example: A *n*-stage series system with *m* stochastic chance constraints is considered as a pure stochastic integer

programming problem with stochastic reliability components. The problem in this case becomes

$$\text{Maximize } R_S(x) = \prod_{j=1}^{n}\left[1-(1-\tilde{r}_j)^{x_j}\right]$$

subject to

$$\text{Prob}\left[\sum_{j=1}^{4}\tilde{a}_{ij}x_j \le \tilde{b}_i\right] \ge 1-\gamma_i, \quad i=1,2,...,m$$

where, $\tilde{r}_j \sim N\left(\mu_{r_j}, \sigma_{r_j}\right)$, $1 \le x_j \le 10$, $j=1,2,...,n$

and $\tilde{b}_i \sim N\left(\mu_{b_i}, \sigma_{b_i}\right)$, $i=1,2,...,m$.

Now, we desire to solve this problem with 4-stages with two number of stochastic constraints i.e., *n=4* & *m=2* in different situations. The input data are all taken in stochastic form which are distributed normally and are given in the Tables 1-5.

Table 1: Input data

j	1	2	3	4
$\tilde{r}_j$	N(0.75, 0.01)	N(0.80, 0.02)	N(0.75, 0.01)	N(0.85, 0.02)
a_{1j}	N(1.5, 0.01)	N(3.3, 0.05)	N(3.2, 0.02)	N(4.4, 0.01)
a_{2j}	N(4.0, 0.03)	N(5.0, 0.04)	N(7.0, 0.03)	N(9.0, 0.02)
Available resource: $b_1 \sim$N(55, 2), $b_2 \sim$N(125, 3), $\gamma_1=0.10$, $\gamma_2=0.15$				

Table 2: Input data

j	1	2	3	4
$\tilde{r}_j$	N(0.85, 0.01)	N(0.85, 0.02)	N(0.80, 0.01)	N(0.90, 0.02)
a_{1j}	N(1.5, 0.01)	N(3.3, 0.05)	N(3.2, 0.02)	N(4.4, 0.01)
a_{2j}	N(4.0, 0.03)	N(5.0, 0.04)	N(7.0, 0.03)	N(9.0, 0.02)
Available resource: $b_1 \sim$N(55, 2), $b_2 \sim$N(125, 3), $\gamma_1=0.10$, $\gamma_2=0.15$				

Table 3: Input data

j	1	2	3	4
$\tilde{r}_j$	N(0.85, 0.01)	N(0.90, 0.02)	N(0.85, 0.01)	N(0.95, 0.02)
a_{1j}	N(1.5, 0.01)	N(3.3, 0.05)	N(3.2, 0.02)	N(4.4, 0.01)
a_{2j}	N(4.0, 0.03)	N(5.0, 0.04)	N(7.0, 0.03)	N(9.0, 0.02)
Available resource: $b_1 \sim$N(55, 2), $b_2 \sim$N(125, 3), $\gamma_1=0.10$, $\gamma_2=0.15$				

Table 4: Input data

j	1	2	3	4
$\tilde{r}_j$	N(0.86, 0.01)	N(0.89, 0.02)	N(0.85, 0.01)	N(0.94, 0.02)
a_{1j}	N(1.5, 0.01)	N(3.3, 0.05)	N(3.2, 0.02)	N(4.4, 0.01)
a_{2j}	N(4.0, 0.03)	N(5.0, 0.04)	N(7.0, 0.03)	N(9.0, 0.02)
Available resource: $b_1 \sim$N(55, 2), $b_2 \sim$N(125, 3), $\gamma_1=0.10$, $\gamma_2=0.15$				

Table 5: Input data

j	1	2	3	4
$\tilde{r}_j$	N(0.89, 0.01)	N(0.85, 0.02)	N(0.90, 0.01)	N(0.93, 0.02)
a_{1j}	N(1.5, 0.01)	N(3.3, 0.05)	N(3.2, 0.02)	N(4.4, 0.01)
a_{2j}	N(4.0, 0.03)	N(5.0, 0.04)	N(7.0, 0.03)	N(9.0, 0.02)
Available resource: $b_1 \sim$N(55, 2), $b_2 \sim$N(125, 3), $\gamma_1=0.10$, $\gamma_2=0.15$				

Table 6: Best found results for input data taken from Table 1-5

Input from Table No.	$r=(r_1,r_2,r_3,r_4)$	$x=(x_1,x_2,x_3,x_4)$	R
1	(0.74,0.79,0.76,0.85)	(6,4,5,3)	0.998010
2	(0.85,0.87,0.79,0.87)	(6,4,5,3)	0.999649
3	**(0.83,0.88,0.85,0.95)** **(0.86,0.92,0.85,0.93)**	**(6,4,5,3)** **(4,4,3,1)**	**0.999968**
4	(0.85,0.87,0.82,0.93)	(6,4,5,3)	0.999947
5	(0.89,0.83,0.90,0.91)	(5,5,4,3)	0.999940

Table 7: Sensitivity of b_1 using others data from Table 3

b_1	x	r	R
N(55,1)	**(7,4,5,3)**	**(0.87,0.94,0.85,0.96)**	**0.9999639311**
N(55,2)	(6,4,5,3)	(0.84,0.91,0.85,0.95)	0.9999631286
N(55,3)	(6,4,5,3)	(0.84,0.91,0.88,0.99)	0.9999409114
N(55,4)	(5,4,5,3)	(0.83,0.90,0.85,0.92)	0.9999097329
N(55,5)	(5,4,5,3)	(0.84,0.92,0.85,0.96)	0.9998751529

Table 8: Sensitivity of b_2 using others data from Table 3

b_2	x	r	R
N(125,1)	(6,4,5,3)	(0.86,0.91,0.87,0.93)	0.9999600432
N(125,2)	(6,4,5,3)	(0.86,0.87,0.84,0.94)	0.9999487544
N(125,3)	**(6,4,5,3)**	**(0.84,0.91,0.85,0.95)**	**0.9999631286**
N(125,4)	(6,4,5,3)	(0.84,0.91,0.86,0.97)	0.9999558163
N(125,5)	(6,4,5,3)	(0.86,0.89,0.85,0.92)	0.9999454725

Table 9: Sensitivity of γ_1 using others data from Table 3

γ_1	x	r	R
0.10	(6,4,5,3)	(0.85,0.92,0.84,0.94)	0.9999542631
0.15	(6,4,5,3)	(0.84,0.92,0.85,0.93)	0.9999596902
0.20	(6,4,5,3)	(0.82,0.88,0.84,0.94)	0.9999579307
0.25	(7,4,5,3)	(0.84,0.90,0.85,0.95)	0.9999607027
0.30	(10,4,2,1)	(0.86,0.92,0.84,0.96)	0.9999730732
0.35	(6,4,5,3)	(0.85,0.89,0.84,0.94)	0.9999509162
0.40	(6,4,5,3)	(0.85,0.92,0.87,0.98)	0.9999540116
0.45	(6,4,5,3)	(0.85,0.89,0.86,0.93)	0.9999604335
0.50	(6,4,5,3)	(0.86,0.85,0.85,0.95)	0.9999737764
0.55	**(6,4,5,3)**	**(0.86,0.92,0.86,0.94)**	**0.9999929499**

Table 10: Sensitivity of γ_2 using others data from Table 3

γ_2	x	r	R
0.15	(6,4,5,3)	(0.85,0.92,0.84,0.94)	0.9999542631
0.20	(6,4,5,3)	(0.85,0.93,0.84,0.94)	0.9999523536
0.25	(6,4,5,3)	(0.86,0.91,0.87,0.95)	0.9999538756
0.30	(6,4,5,3)	(0.84,0.87,0.84,0.97)	0.9999549780
0.35	(6,4,5,3)	(0.85,0.92,0.84,0.93)	0.9999545930
0.40	(6,4,5,3)	(0.86,0.89,0.86,0.91)	0.9999524183
0.45	**(6,4,5,3)**	**(0.86,0.92,0.87,0.93)**	**0.9999581263**
0.50	(6,4,5,3)	(0.86,0.93,0.85,0.95)	0.9999550624
0.55	(6,4,5,3)	(0.84,0.90,0.85,0.97)	0.9999512939
0.60	(6,4,5,3)	(0.84,0.89,0.85,0.98)	0.9999535680

We have used real coded genetic algorithm to solve the problem under consideration. In this algorithm, we have used tournament selection, intermediate crossover and one neighborhood mutation as genetic operators. For this purpose, we have prepared the code for this algorithm in C++ Programming language. The corresponding computational work has been done on a PC with Intel i3 processor in LINUX environment. For each problem, fifty independent runs have been performed to determine the best found system reliability which is nothing but the optimal/near to optimal value of system reliability. In this computation, the values of genetic parameters, like p_s, m_g, p_c and p_m have been taken as 150, 100, 0.85 and 0.15 respectively.

Table 11: Input data in crisp case

j	1	2	3	4
r_j	0.75	0.80	0.75	0.85
a_{1j}	1.5	3.3	3.2	4.4
a_{2j}	4.0	5.0	7.0	9.0
Available resource: b_1 =55 b_2 =125, and $\gamma_1 = 0.10$, $\gamma_2 = 0.15$				

11. Result Discussion

The numerical example has been solved for different input data set. The input data are given in the Tables 1-5. It is to be noted that the reliability components follow the normal distribution with different means and standard deviations in each input data table. In these data tables, standard deviation of the *i*-th component reliability is kept fixed; however, the mean has been changed to make different input data set. For every input data set, we have considered 50 independent run of the program coded in C++ in Ubuntu 10.1 LINUX operating system to find the best fitness function of the problem. The best found results obtained in these five cases are presented in Table-6 and it is to be noted that using Table-3, we get comparatively better outcome. Further, we observe from Table-6 that in case of input Table-3, there are two different optimal solutions with same objective function value **0.999968.** Earlier works available in the literature, in case of crisp input data set given in Table-11, show that the optimum objective value of this problem is **0.995946** with x=(5,4,5,4). Thus, the outcomes in this work are far better than that of the crisp case achieved earlier. The sensitivities of the resource availabilities b_1 and b_2 are presented in Tables 7 & 8 respectively. While, the Tables 9 & 10 represent the sensitivities of γ_1 and γ_2 respectively. The best found results have been indicated with bold face in each case.

12. Conclusions

In this paper, we have presented a simulation based genetic algorithm for solving Chance Constrained redundancy allocation problem considering stochastic component reliabilities. To transform the chance constraints to its deterministic equivalent form, Monte Carlo simulation technique have been applied. Then the transformed problem has been converted into unconstrained optimization problem with the help of Big-M penalty technique. To solve the transformed problem, we have applied real coded genetic algorithm with tournament selection, intermediate crossover and one-neighbourhood mutation. For solving the optimization problem, we have used the GA based Big-M penalty approach. In this approach, the value of fitness function is not computed for infeasible solutions. For infeasible solutions, the value of M may be taken depending on the fitness function value. A small value (in case of maximization problem) or a large value (in case of minimization problem) may be considered for M to solve the constrained optimization problem.

13. Scope of Future Research

In future, the researchers may apply the proposed methodology and simulation based genetic algorithm for solving optimization problems which are mostly arising in the areas of reliability engineering disciplines and management sciences and also in the different areas of optimization.

14. Acknowledgement

The author would like to acknowledge the help and cooperation extended by my research guide Dr. A. K. Bhunia, Professor, Department of Mathematics, Burdwan University.

References

[1] A.K. Bhunia, L. Sahoo, D. Roy, "Reliability stochastic optimization for a series system with interval component reliability via genetic algorithm", Applied Mathematics and Computations, 216, pp. 929-939, 2010.
[2] A.K. Bhunia, L. Sahoo, "Genetic algorithm based reliability optimization in interval environment", in Innovative Computing Methods and Their Applications to Engineering Problems, N. Nedjah (Eds.), SCI 357, pp. 13-36, Springer-Verlog, Berlin Heidelberg, 2011.
[3] A. Charnes, W.W. Cooper, G.H. Symonds, "Cost horizons and certainty equivalents: An approach to stochastic programming of heating oil", Management Sciences, 4, pp. 235-263, 1958.
[4] M.S. Chern, "On the computational complexity of reliability redundancy allocation in a series system", Operations Research Letters, 11(5), pp. 309-315, 1992.
[5] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison- Wesley, Longman Publishing Co., Boston, 1989.
[6] R.K. Gupta, A.K. Bhunia, 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(2), pp. 275-284, 2009.
[7] E. Hansen, G.W. Walster, Global optimization using interval analysis, Marcel Dekker Inc., New York, 2004.
[8] M.Y. Hikita, K. Nakagawa, Nakashima, H. Narihisa, "Reliability optimization of systems by a surrogate-constraints algorithm", IEEE Transactions on Reliability, 41(3), pp. 473-480, 1992.
[9] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI, 1975.
[10] K. Iwamura, B. Liu, "A genetic algorithm for chance constraint programming", Journal of Information and Optimization Sciences, 17(2), pp. 409-422, 1996.
[11] R.K. Jana, M.P. Biswal, "Stochastic simulation based genetic algorithm for chance constraint programming problems with continuous random variables", International Journal of Computer Mathematics, 81(9), pp. 1069-1076, 2004.
[12] R.K. Jana, M.P. Biswal, "Stochastic simulation based genetic algorithm for chance constraint programming problems with continuous random variables", International Journal of Computer Mathematics, 81(12), pp. 1455-1463, 2004.

[13] P. Kall, S.W. Wallace, Stochastic programming, John Wiley & Sons, Chichester, 1994.
[14] S. Karmakar, S. Mahato, A.K. Bhunia, "Interval oriented multi-section techniques for global optimization", Journal of computational and Applied Mathematics, 224, pp. 476-491, 2009.
[15] W. Kuo, H. Lin, Z. Xu, W. Zhang, "Reliability optimization with the Lagrange-multiplier and Branch-and-Bound technique", IEEE Transactions on Reliability, 36(5), pp. 624-630, 1987.
[16] W. Kuo, V.R. Prasad, "An annotated overview of system reliability optimization", IEEE Transactions on Reliability, 49(2), pp. 487-493, 2000.
[17] W. Kuo, V.R. Prasad, F.A. Tillman, C.L. Hwang, Optimal Reliability Design: Fundamentals and Applications, Cambridge University Press, 2001.
[18] S.K. Mahato, A.K. Bhunia, "Interval-arithmetic-oriented interval computing technique for global optimization", Applied Mathematics Research Express, 2006, 01-18, 2006.
[19] S.K. Mahato, L. Sahoo, A.K. Bhunia, "Reliability-redundancy optimization problem with interval valued reliabilities of components via genetic algorithm", Journal of Information and Computing Science, 7(4), pp. 284-295, 2012.
[20] K. Miettinen, M.M. Mäkelä, J. Toivanen, "Numerical comparison of some penalty-based constraint handling techniques in genetic algorithms", Journal of Global Optimization, 27, pp. 427-446, 2003.
[21] L.B. Miller, H. Wagner, "Chance-constrained programming with joint constraints", Operations Research, 13, pp. 930-945, 1965.
[22] B. Misra, U. Sharma, "An efficient algorithm to solve integer-programming problems arising in system-reliability design", IEEE Transactions on Reliability, 40(1), pp. 81-91, 1991.
[23] Y. Nakagawa, S. Miyazaki, "Surrogate constraints algorithm for reliability optimization problems with two constraints", IEEE Transactions on Reliability, 30(2), pp. 181-184, 1981.
[24] V. Ravi, B.S.N. Murty, P.J. Reddy, "Nonequilibrium simulated-annealing algorithm applied to reliability optimization of complex systems", IEEE Transactions on Reliability, 46(2), pp. 233-239, 1997.
[25] R.Y. Rubinstein, Simulation and Monte Carlo method, John Wiley and Sons, New York, 1981.
[26] L. Sahoo, A.K. Bhunia, P.K. Kapur, "Genetic algorithm based multi-objective reliability optimization in interval environment", Computer and Industrial Engineering, 62(1), pp. 152-160, 2012.
[27] L. Sahoo, A.K. Bhunia, D. Roy, "A genetic algorithm based reliability redundancy optimization for interval valued reliabilities of components", Journal of Applied Quantitative Methods, 5(2), pp. 270-287, 2010.
[28] L. Sahoo, A.K. Bhunia, D. Roy, "An application of genetic algorithm in solving reliability optimization problem under interval component Weibull parameters", Mexican Journal of Operations Research, 1(1), pp. 2-19, 2012.
[29] L. Sahoo, A.K. Bhunia, D. Roy, "Reliability optimization in Stochastic Domain via Genetic Algorithm", International Journal of Quality & Reliability Management, 31(6), pp.698 - 717, 2014.
[30] L. Sahoo, S.K. Mahato, A.K. Bhunia, "Optimization of System Reliability for Series System with Fuzzy Component Reliabilities by Genetic Algorithm", Journal of Uncertain System, 8(2), pp. 136-148, 2014.
[31] M. Sheikhalishahi, V. Ebrahimipour, H. Zaman, M. Jeihoonian, "A hybrid GA-PSO approach for reliability optimization in redundancy allocation problem", International Journal of Advanced Manufacturing Technology, 68, pp. 317-338, 2013.
[32] X. Sun, D. Li, "Optimal condition and Branch and Bound algorithm for constrained redundancy optimization in series system", Optimization and Engineering, 3, pp. 53-65, 2002.
[33] C.S. Sung, Y.K. Cho, "Branch and Bound redundancy optimization for a series system with multiple-choice constraints", IEEE Transactions on Reliability, 48 (2), pp. 108-117, 1999.

Author Profile

Dr. Sanat Kumar Mahato is an Assistant Professor of Mathematics, Mejia Govt. College, West Bengal, India. He has been awarded the Ph. D. degree in Mathematics by The University of Burdwan in August, 2014. Dr. Mahato has published nine research papers in different peer reviewed international journals. His area of research work includes Application of Genetic Algorithm in inventory, in reliability optimization, interval analysis and its application, particle swarm optimization etc