CHAPTER 4

Reliability Optimization under High and Low-level Redundancies for Imprecise Parametric Values

- Introduction
- Assumptions
- Low-Level and High-level Redundancy
- Formulation of Reliability-Redundancy Optimization Problems
- Solution Procedure
- Numerical Examples
- Sensitivity Analysis
- Concluding Remarks
4.1 Introduction

Generally, the designing of a modern technological system design depends on the selection of components and their configurations to satisfy the functional as well as performance specifications. For a system with known cost, reliability, 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 known as the redundancy allocation problem. The basic objective of the redundancy allocation problem is to determine the number of redundant components that either maximizes the system reliability or minimize the system cost under several resource constraints. Redundancy allocation problem is nothing but a non-linear integer programming problem. Most of these problems cannot be solved by direct/indirect or mixed search methods due to discrete search space. According to Chern (1992), 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, Hwang and Kuo (1977a) and Kuo and Prasad (2000). In this chapter, we have formulated two types of redundancy, viz. component level redundancy known as low-level redundancy and the system level redundancy known as high-level redundancy for a five-link bridge system where the objective function as well as constraints functions are considered as interval valued. To the best of our knowledge, studies of the system reliability with component reliability as interval valued have already been initiated by Gupta, Bhunia and Roy (2009). Also, a number of researchers has presented different situations and solutions methodologies on redundancy allocation problem in different environments [Park (1987), Mahapatra and Roy (2006) and Liu (2010)].
In this chapter, we have proposed GA-based approaches for solving IVNLP type redundancy allocation problem. To find the optimal solution of this type of problem by GA, order relations of interval numbers assume an important role for GA operators. Here, we have used our proposed interval order relations discussed in Chapter 2. Using these we have developed a real coded elitist GA with tournament selection, intermediate crossover and one-neighborhood mutation for solving the proposed problems. Finally, to illustrate the proposed models, for high-level as well as low-level redundancy, two numerical examples have been presented.

4.2 Assumptions

(i) Reliability of each component is imprecise and interval valued.

(ii) Chances of failures of components are mutually 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.

(vi) The cost coefficients as well as the amount of resources are imprecise and interval valued.

4.3 Low-level and High-level Redundancy

Let us consider a \( n \) component system. Now, we can either provide redundant components, which give a system design diagram as shown in Figure 4.1, or provide a total redundant system as shown in Figure 4.2. The component level redundancy is known as low-level redundancy and the system level redundancy is known as high-level redundancy. Here \( h = \min(x_1, x_2, \ldots, x_n) \).
4.4 Formulation of Reliability-Redundancy Optimization Problems

Let us consider a network with \( n \) subsystems. The goal of the redundancy allocation problem is to determine the number of redundant components in each of \( n \) parallel subsystems so as to maximize the overall system reliability subject to the given resource constraints and also to minimize the overall system cost subject to the given constraint on system reliability.

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

**Problem 1** Maximize \( R_S(x) \)

subject to \( g_i(x) \leq b_i \), \( i = 1, 2, \ldots, m \)

\[ 1 \leq l_j \leq x_j \leq u_j, \quad x_j \text{ integer, } j = 1, \ldots, n \]

The goal of the Problem 1 is to determine the number of redundant components so as to maximize the overall system reliability. This problem belongs to the category of constrained integer non-linear programming problems (INLPP).

The general form of the cost minimization problem is as follows:

**Problem 2** Minimize \( C_S(x) \)

subject to \( R_S(x) \geq R_T \)

This formulation is designed to achieve a minimum total system cost, subject to \( R_T \), a target limit on the system reliability.

For low-level redundant system (see Figure 4.1), the corresponding reliability-redundancy optimization problems are as follows:

**Problem 3** Maximize \( R_S(x) = f(R_1(x_1), R_2(x_2), \ldots, R_q(x_q), \ldots, R_n(x_n)) \)

subject to \( g_i(x) \leq b_i \), \( i = 1, 2, \ldots, m \)

\[ 1 \leq l_j \leq x_j \leq u_j, \quad x_j \text{ integer, } j = 1, \ldots, n \]
where \( R_j(x_j) = [R_{jL}(x_j), R_{jR}(x_j)] = 1 - (1 - [r_{jL}, r_{jR}])^{x_j}, \ j = 1, 2, \ldots, n \nabla \)

and \( r_j = [r_{jL}, r_{jR}] \in (0, 1) \nabla \)

**Problem 4** Minimize \( C_S(x) \nabla \)

subject to \( R_S(x) = f(R_1(x_1), R_2(x_2), \ldots, R_q(x_q), \ldots, R_n(x_n)) \geq R_T \nabla \)

where \( R_j(x_j) = [R_{jL}(x_j), R_{jR}(x_j)] = 1 - (1 - [r_{jL}, r_{jR}])^{x_j}, \ j = 1, 2, \ldots, n \nabla \)

For high-level redundant system (see Figure 4.2), the corresponding reliability-redundancy optimization problems are as follows:

**Figure 4.1:** Low-level redundancy

**Figure 4.2:** High-level redundancy
Problem 5

Maximize \( R_S(h) = [R_{SL}(h), R_{SR}(h)] = 1 - (1 - f([r_{1L}, r_{1R}], \ldots, [r_{qL}, r_{qR}], \ldots, [r_{nL}, r_{nR}]))^h \)

subject to \( g_i(h) \leq b_i, \ i = 1, 2, \cdots, m \)

\( l \leq h \leq u, \ h \ \text{integer} \)

\( r_i = [r_{IL}, r_{IR}] \in (0,1), \ i = 1, 2, \cdots, n \)

Problem 6

Minimize \( C_S(h) \)

subject to \( R_S(h) \geq R_f \)

where \( R_S(h) = [R_{SL}(h), R_{SR}(h)] = 1 - (1 - f([r_{1L}, r_{1R}], \ldots, [r_{qL}, r_{qR}], \ldots, [r_{nL}, r_{nR}]))^h \)

4.5 Solution Procedure

In this section we shall discuss the solution procedure for all the problems mentioned in earlier section i.e., Problems (3)-(6). These problems are non-linear integer programming optimization problems, each with interval valued objective function. Using Big-M penalty technique and real coded genetic algorithm with advanced operators, these problems are converted into unconstrained optimization problems.

The converted problems of Problems (3)-(6) are as follows:

For the constrained optimization Problem 3

Maximize \( R_S(x) = [R_{SL}(x), R_{SR}(x)] \)

subject to \( g_i(x) \leq b_i, \ i = 1, 2, \cdots, m \)

\( 1 \leq l_j \leq x_j \leq u_j, \ x_j \ \text{integer}, \ j = 1, \ldots, n \)

The form of Big-M penalty is as follows:

Maximize \( [\hat{R}_{SL}(x), \hat{R}_{SR}(x)] = [R_{SL}(x), R_{SR}(x)] + \theta(x) \) \hspace{1cm} (4.1)
where $\theta(x) = \begin{cases} [0,0] & \text{if } x \in S \\ \left[ -R_{SL}(x), R_{SR}(x) \right] + [-M,-M] & \text{if } x \notin S \end{cases}$

and $S = \{ x : g_i(x) \leq b_i, \ i = 1,2,\cdots,m \text{ and } 1 \leq l_j \leq x_j \leq u_j, \ x_j \text{ integer, } j = 1,\cdots,n \}$

For the constrained optimization Problem 4

Minimize $C_S(x) = [C_{SL}(x), C_{SR}(x)]$

subject to $R_S(x) \geq R_T$

The form of Big-M penalty is as follows:

Maximize $[\hat{C}_{SL}(x), \hat{C}_{SR}(x)] = [-C_{SL}(x), C_{SR}(x)] + \theta(x)$

where $\theta(x) = \begin{cases} [0,0] & \text{if } x \in S \\ \left[ C_{SL}(x), C_{SR}(x) \right] + [-M,-M] & \text{if } x \notin S \end{cases}$

and $S = \{ x : R_S(x) \geq R_T, \text{ and } 1 \leq l_j \leq x_j \leq u_j, \ x_j \text{ integer, } j = 1,\cdots,n \}$

For the constrained optimization Problem 5

Maximize $R_S(h) = [R_{SL}(h), R_{SR}(h)]$

subject to $g_i(h) \leq b_i, \ i = 1,2,\cdots,m$

$l \leq h \leq u, \ h \text{ integer}$

The form of Big-M penalty is as follows:

Maximize $[\hat{R}_{SL}(h), \hat{R}_{SR}(h)] = [R_{SL}(h), R_{SR}(h)] + \theta(h)$

where $\theta(h) = \begin{cases} [0,0] & \text{if } h \in S \\ \left[ -R_{SL}(h), R_{SR}(h) \right] + [-M,-M] & \text{if } h \notin S \end{cases}$

and $S = \{ h : g_i(h) \leq b_i, \ i = 1,2,\cdots,m \text{ and } 1 \leq l \leq h \leq u, \ h \text{ integer} \}$

For the constrained optimization Problem 6

Minimize $C_S(h) = [C_{SL}(h), C_{SR}(h)]$

subject to $R_S(h) \geq R_T$

The form of Big-M penalty is as follows:
Maximize \[ \hat{C}_{SL}(h), \hat{C}_{SR}(h) = -[C_{SL}(h), C_{SR}(h)] + \theta(h) \] (4.4)

where \[ \theta(h) = \begin{cases} [0, 0] & \text{if } h \in S \\ [C_{SL}, C_{SR}] + [-M, -M] & \text{if } h \not\in S \end{cases} \]

and \( S = \{ h : R_S(h) \geq R_T, \text{ and } 1 \leq l \leq h \leq u, h \text{ integer} \} \)

Here \([\hat{R}_{SL}(x), \hat{R}_{SR}(x)], [\hat{C}_{SL}(x), \hat{C}_{SR}(x)], [\hat{R}_{SL}(h), \hat{R}_{SR}(h)]\) and \([\hat{C}_{SL}(h), \hat{C}_{SR}(h)]\) are the interval valued auxiliary objective functions. Problems (4.1) and (4.2) are integer non-linear unconstrained optimization problems with interval objective of \( n \) integer variables \( x_1, x_2, ..., x_n \), whereas problems (4.3) and (4.4) are integer non-linear unconstrained optimization problems with interval objective of integer variable \( h \).

These problems (4.1)-(4.4) are non-linear unconstrained integer programming problems with interval coefficients.

### 4.6 Numerical Examples

In this section, we have considered the redundancy allocation problem for low-level redundancy (see Figure 4.3) and for high-level redundancy of five-link bridge system (see Figure 4.4) for numerical experiments. Bridge system is of use in system network with subsystem-5 representing a hub and rest of the systems representing servers/client with processors arranged in parallel. This five-link bridge network system [Kuo, Prasad, Tillman and Hwang (2001)] works successfully as long as one of the paths, (subsystem-1-subsystem-2) or (subsystem-3-subsystem-4), is active independently of subsystem-5. However, if the pair of subsystems (1, 4) or (2, 3) fails, then subsystem-5 plays an important role in the system operation. In each subsystem-\( i \), \( i = 1, 2, 3, 4, 5 \), which is imprecise in nature, there is a parallel
configuration consisting of \( x_i \) identical components having reliability \( r_i \). If \( R_i \) be the system reliability of subsystem- \( i, \ i = 1, 2, 3, 4, 5 \) then \( R_i = 1 - (1 - r_i)^{n_i}, \ i = 1, 2, 3, 4, 5 \).

The system reliability of the low-level five-link bridge network system is given by the expression as follows:

\[
R_S(x) = R_1 R_2 + Q_2 R_3 R_4 + Q_1 Q_2 R_3 R_4 + R_1 Q_2 Q_3 R_4 R_5 + Q_1 R_2 R_3 Q_4 R_5,
\]

where \( R_i = 1 - q_i, \ i = 1, 2, 3, 4, 5 \)

The system reliability of the high-level five-link bridge network system is given by the expression as follows:

\[
R_S(h) = 1 - (1 - (r_1 q_2 r_3 r_4 + q_1 r_2 r_3 r_4 + r_1 q_2 q_3 r_4 + q_1 r_2 r_3 q_4 r_5))^h,
\]

where \( r_i = 1 - q_i, \ i = 1, 2, 3, 4, 5 \) and \( h \) be the number of redundant subsystems, arranged in parallel. For low-level redundancy, the corresponding system reliability maximization and cost minimization problems are of the following forms:

**Example 1**

Maximize \([R_{SL}(x), R_{SR}(x)] = [R_{1L}, R_{1R}][R_{2L}, R_{2R}] + [Q_{2L}, Q_{2R}][R_{3L}, R_{3R}][R_{4L}, R_{4R}]

\[+ [Q_{1L}, Q_{1R}][R_{2L}, R_{2R}][R_{3L}, R_{3R}][R_{4L}, R_{4R}]\]

\[+ [R_{1L}, R_{1R}][Q_{2L}, Q_{2R}][Q_{3L}, Q_{3R}][R_{4L}, R_{4R}]\]

\[+ [Q_{1L}, Q_{1R}][R_{2L}, R_{2R}][R_{3L}, R_{3R}][Q_{4L}, Q_{4R}][R_{5L}, R_{5R}]\]

subject to

\[
g_1(x) = \sum_{j=1}^{5} c_j [x_j + \exp(x_j/4)] - b_1 \leq 0
\]

\[
g_2(x) = \sum_{j=1}^{5} w_j x_j \exp(x_j/4) - b_2 \leq 0
\]
Studies on Reliability Optimization Problems by Genetic Algorithm

Figure 4.3: Low-level redundancy of five-link bridge system

Example 2

Minimize $C_S(x) = \sum_{j=1}^{5} c_j \left[ x_j + \exp\left(\frac{x_j}{4}\right) \right]$

subject to $R_S(x) \geq R_T$

where

$[R_{SL}(x), R_{SR}(x)] = [R_{1L}, R_{1R}] [R_{2L}, R_{2R}] + [Q_{2L}, Q_{2R}] [R_{3L}, R_{3R}] [R_{4L}, R_{4R}]$

$+ [Q_{1L}, Q_{1R}] [R_{2L}, R_{2R}] [R_{3L}, R_{3R}] [R_{4L}, R_{4R}]$  

$+ [R_{1L}, R_{1R}] [Q_{2L}, Q_{2R}] [Q_{3L}, Q_{3R}] [R_{4L}, R_{4R}] [R_{5L}, R_{5R}]$

$+ [Q_{1L}, Q_{1R}] [R_{2L}, R_{2R}] [R_{3L}, R_{3R}] [Q_{4L}, Q_{4R}] [R_{5L}, R_{5R}]$

For high-level redundancy, the corresponding system reliability maximization and cost minimization problems are of the form that follows:

Example 3

Maximize $R_S(h) = 1 - (1 - (r_1r_2 + q_2r_3r_4 + q_1r_2q_3r_4 + r_1q_2q_3q_4r_5 + q_1r_2q_3q_4q_5))^h$

subject to
\[ g_1(h) = \left[ h + \exp(h/4) \right] \sum_{j=1}^{5} c_j - b_1 \leq 0 \]

\[ g_2(h) = \left[ h \exp(h/4) \right] \sum_{j=1}^{5} w_j - b_2 \leq 0 \]

where \( r_i = [r_{iL}, r_{iR}], \) \( i = 1, 2, 3, 4, 5 \) and \( q_i = [q_{iL}, q_{iR}] = 1 - [r_{iL}, r_{iR}] \)

**Example 4**

Minimize \( C_S(h) = \left[ h + \exp(h/4) \right] \sum_{j=1}^{5} c_j \)

subject to \( R_S(h) \geq R_T \)

where \( R_S(h) = 1 - (1 - (r_1 r_2 + q_1 r_3 r_4 + q_1 r_2 r_3 r_4 + r_1 q_2 q_3 r_1 r_5 + q_1 r_2 r_3 q_4 r_5))^h \)

and \( r_i = [r_{iL}, r_{iR}], \) \( i = 1, 2, 3, 4, 5 \) and \( q_i = [q_{iL}, q_{iR}] = 1 - [r_{iL}, r_{iR}] \)

All the values of the parameters related to above Examples are given in Table 4.1:

**Table 4.1: Values of the parameters related to Examples 1-4**

<table>
<thead>
<tr>
<th>( j )</th>
<th>( r_j )</th>
<th>( c_j )</th>
<th>( b_1 )</th>
<th>( w_j )</th>
<th>( b_2 )</th>
<th>( R_T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[0.64,0.66]</td>
<td>[3,5]</td>
<td>[1.5,1.6]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>[0.73,0.76]</td>
<td>[4.5,5]</td>
<td>[2,2.5]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>[0.75,0.77]</td>
<td>[5.5,7.5]</td>
<td>[105,115]</td>
<td>[2,2.25]</td>
<td>[30,35]</td>
<td>[0.99,0.999]</td>
</tr>
<tr>
<td>4</td>
<td>[0.83,0.86]</td>
<td>[5.7]</td>
<td>[1.5,1.75]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>[0.88,0.90]</td>
<td>[2,2.5]</td>
<td>[1.75,2]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The proposed method has been coded in C programming language. The computational work has been done on a PC with Intel Core-2 duo processor in LINUX environment. For each example 20 independent runs have been performed to calculate the best found system reliability and best found system cost which are nothing but the optimal values of system reliability and system cost. Also we have to
calculated the mean and variance of system reliability as well as system cost. In this computation, the values of genetic parameters like \( p_{size}, max_{gen}, p_{mute} \) and \( p_{cross} \) have been taken as 100, 100, 0.15 and 0.85 respectively. The computational results have been shown in Table 4.2.

![Figure 4.4: High-level redundancy of five-link bridge system](image)

It has been observed from the computational results that the mean system reliability/mean system cost coincides with the best found system reliability/system cost. This strict coincidence is due to the fact that each trial run provides us the optimum solution. Also, the lower ends of the standard deviations, measured in interval form, assume zero value. This speaks of high precision in our optimization process. It may also be noted that the average CPU time required for implementing the genetic algorithm, is very less.
### Table 4.2: Computational results for Examples 1-4

<table>
<thead>
<tr>
<th>Example</th>
<th>1</th>
<th>3</th>
<th>2</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x's/h$</td>
<td>(2,2,1,2,2)</td>
<td>(1)</td>
<td>(1,1,2,1,1)</td>
<td>(3)</td>
</tr>
<tr>
<td>4.7 Sensitivity Analysis</td>
<td>4.7 Sensitivity Analysis</td>
<td>4.7 Sensitivity Analysis</td>
<td>4.7 Sensitivity Analysis</td>
<td></td>
</tr>
<tr>
<td>Best found system reliability</td>
<td>[0.939545,0.999027]</td>
<td>[0.819842,0.928286]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Mean value of system reliability</td>
<td>[0.939545,0.999027]</td>
<td>[0.819842,0.928286]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Best found system cost</td>
<td>-</td>
<td>-</td>
<td>[53.186,71.904]</td>
<td>[102.34,138.159]</td>
</tr>
<tr>
<td>Mean value of system cost</td>
<td>-</td>
<td>-</td>
<td>[53.186,71.904]</td>
<td>[102.34,138.159]</td>
</tr>
<tr>
<td>Standard deviation of system reliability</td>
<td>[0,0.059482]</td>
<td>[0,0.108444]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Standard deviation of system cost</td>
<td>-</td>
<td>-</td>
<td>[0,18.718]</td>
<td>[0,35.819]</td>
</tr>
<tr>
<td>CPU time in seconds</td>
<td>0.04000</td>
<td>0.07000</td>
<td>0.03000</td>
<td>0.18000</td>
</tr>
</tbody>
</table>

To investigate the overall performance of the proposed GA-based penalty technique for solving low-level redundancy as well as high level redundancy, sensitivity analyses have been carried out graphically on the interval valued system reliability with respect to different GA parameters separately taking other parameters at their original values. These have been shown in Figure 4.5-Figure 4.8. From Figure 4.5 it may be observed that both the bounds of the interval valued system reliability are same for all the values of population size greater than or equal to 30. This implies that our proposed GA is stable when population size exceeds 30. Similarly, from
Figure 4.6 it is clear that our proposed GA is stable when maximum number of generation is greater than or equal to 10. In Figure 4.7 and Figure 4.8, the values of interval valued system reliability have been computed with respect to the probability of crossover within the range 0.45 to 0.95 and the probability of mutation within the range 0.05 to 0.30 respectively. From these figures, it is clear that the proposed GA is stable with respect to probability of crossover as well as the probability of mutation.

**Figure 4.5:** $P_{size}$ vs. interval valued system reliability for Example 1

**Figure 4.6:** $Max_{gen}$ vs. interval valued system reliability for Example 1
4.8 Concluding Remarks

In this chapter, we have investigated two different redundancies known as low-level redundancy and high-level redundancy and proposed four problems where each problem belongs to the category of interval valued non-linear integer programming problems. Then we have solved these problems as constrained single objective interval valued reliability optimization problem. The reduced problem has been converted into unconstrained interval valued integer programming problem using
Big-M penalty technique and solved by genetic algorithm. To solve the problem, we have developed a real coded GA for integer variables with interval valued fitness function, tournament selection, intermediate crossover and one-neighborhood mutation and elitism of size one. It is well known that the penalty coefficient *plays* a crucial role in solving constrained optimization problem by penalty function technique. Therefore, the selection of this parameter is a formidable task. To avoid this difficulty, we have used Big-M penalty technique which does not require any penalty coefficient. This entire approach opens up scope for reliability optimization when reliability values and other design parameters are interval/imprecise valued. Thus, it can be claimed that the generalization attempted in this chapter can handle the real-life problem of imprecise reliability optimization and cost minimization.