MIP: A Versatile Tool for Reliability Design of a System

S.K. Chaturvedi[1] and K.B. Misra[2]

[1]Reliability Engineering Centre, IIT Kharagpur, Kharagpur (WB) India
[2] RAMS Consultants, Jaipur

Abstract: In many reliability design problems, the decision variables can only take integer values. There are many examples such as redundancy allocation, spare parts allocation, repairman allocation that necessitate integer programming formulations and solutions thereof. In other words, the integer programming plays an important role in system reliability optimization. In this chapter, a simple yet powerful algorithm is described, which provides an exact solution to any general class of integer programming formulations and thereby offers reliability designers an efficient tool for system design. The algorithm is presented with an illustration to make the readers understand various steps. Besides, the applications of the algorithm to various reliability design problems are also provided.

33.1 Introduction

Advances in technology have always led system engineers, manufacturers and designers to design and manufacture systems with ever increasing sophistication, complexity, and capacity. Unreliable performance of some of the constituent sub-systems in these systems may lead to disastrous consequences for the system and its environment and loss of lives including economic, legal and sociological implications. Therefore it necessarily requires designers to design systems with the highest possible reliability within the constraints of cost, time, space, volume, technological limits *etc*. As a result, reliability is one of the system attributes that cannot be compromised in system planning, design, development and operation. It is of paramount concern to practicing engineers, manufacturers, economists and administrators. However, it is an established fact that the occurrence of failure can not be completely eliminated even for well-designed, well-engineered, thoroughly tested and properly maintained equipment. As a consequence, a present day user is not prepared to compromise on reliability, yet would like to have its best value for resources consumed in designing a system. Reliability and maintainability design is one of the areas in reliability engineering which makes possible more effective use of resources and helps decrease the wastage of scarce finances, material, and manpower.

An optimal design is one in which all the possible means available to a designer have been explored to enhance the reliability of the system under certain objective(s), operational requirements and allocated resources. Some of the means through which a designer might attempt to enhance system reliability are:

- Reducing the system complexity.
- Increasing the reliability of constituent components through some product improvement program.
- Use of structural redundancy.
- Putting in practice a planned maintenance and repair/replacement policy.

Although each of the aforementioned alternatives has its relative advantages and disadvantages, one may have to strike a balance between them to achieve a system's objectives.

The employment of structural redundancy at subsystem/component level, without disturbing the system topology, can provide a very effective means of improving system reliability to any desired level [1]. In fact, structural redundancy in combination with an appropriate maintenance strategy may lead to provide almost unity reliability. The structural redundancy involves the use of two or more identical components, to ensure that if one fails, the system operation does not get affected and continues to carry on the specified task even in presence of a faulty component. Depending on the type of system, various forms of redundancy schemes, *viz.,* active, standby, partial, voting, *etc*., are available, and this may provide the quickest, easiest, cheapest and sometimes the only solution. However, the only factors, which may influence such a decision, could be the time constraints, existence of an already designed component, a costly and prohibitive redesign, and of course the technological limits.

There are several kinds of reliability design problems a designer may face. For example, it may include reliability allocation, repairman allocation, failure/repair rate allocation, spare parts allocation problem, *etc*., or a combination of these problems. Depending on the situation, appropriate techniques can be adopted.

The present chapter describes an exact and efficient search technique, known as *Misra Integer Programming* (MIP) in the literature to address many system design problems. Although the algorithm was originally conceived to deal with the redundancy allocation problem, it can solve not only several other problems in system reliability design but many other general integer programming problems with equal ease [12].

Before providing the details of the search algorithm and its applications, the next section presents a brief overview of the redundancy allocation problem and the necessity and importance of developing a useful yet very simple algorithm to solve many design problems.

33.2 Redundancy Allocation Problem

33.2.1 An Overview

The problem of redundancy allocation is concerned with the determination of the number of redundant units to be allocated to each subsystem to achieve an optimized objective function (usually the reliability or some other related attribute of the system, *e.g.*, average life, MTTF), subject to one or more constraints reflecting the availability of various resources. Mathematically, the problem can be stated as:

$$Max.\ R_s = f[R_1(x_1), R_2(x_2)...R_n(x_n)], \tag{33.1}$$

$$Sub.\ \text{to}\ g_i(x) = g(x_1, x_2...x_n) \le b_i, i = 1,2...m, \tag{33.2}$$

where reliability R_s (of n sub-systems with x_j redundant units at the j^{th} subsystem, each with a component reliability of r_j) will be a function of its subcomponents' reliabilities, $R_j(x_j)$. The functional form of $f(\cdot)$ depends on the system configuration and the type of redundancy being used. The form of m number of constraints, $g_i(x)$ (linear/nonlinear, separable/non-separable) can usually be determined from physical system considerations. However, if the constraints are separable functions, we can write (33.2) as:

$$g_i(x) = \sum_{j=n}^{n} g_{ij}(x_j) \le b_i, i = 1,2...m. \tag{33.3}$$

The decision variables $x_{j,}$ in the above formulations can take only non-negative integer values; therefore, the problem belongs to the class of non-linear integer programming problems, and expression R_s, in general, may not be separable in x_j. Also, the nonlinear constraints may not necessarily be separable. However, an active-parallel redundant system in a series model consisting of n subsystems with linear constraints can be written in a closed form [10, 13].

In general, a redundancy allocation problem involving integer programming formulation can be stated as:

$$Optimize\; f(x)\,, \tag{33.4}$$

$$Sub.\ to\ \ g_i(x) \le b_i; i = 1,2...m\,. \tag{33.5}$$

The function *f(x)* in (33.4) can be minimized or maximized and could be set to $f(x^*) = \pm\infty$, in general, ("+" for minimization and "–" for maximization) to start the search process. The variable $x = (x_1, x_2, ...x_n)$ is a vector of decision variables in E^n (the n-dimensional Euclidian plane), which is allowed to take positive integer values belonging to the feasible region $\mathcal{R}$ only and bounded by (33.5). Further, some x_i can also assume a value equal to zero. However, most often all x_i, being non-negative integers, are defined between the limits: $x_j^l \le x_j \le x_j^u$. In a redundancy optimization problem, x_i would have positive integer values between $1 \le x_j \le x_j^u$.

The value of x_k^u for the kth subsystem can easily be determined through the consideration of the ith constraint and accepting a minimum of the upper limits computed over all i=1,2...m, while maintaining $x_j^l = 1, \forall j = 1,2...n, j \ne k$ for other subsystems, *i.e.,*

$$x_k^u = \min_i \{x_j^{\max}\}, \forall k = 1,2...n, i = 1,2...m, k \ne j\,. \tag{33.6}$$

Therefore, the search to optimize, $f(x^*)$ could begin at one of the corners of feasible region, *i.e.*, $\left(x_1^u, x_2^l ... x_n^l\right)$ of $\mathcal{R}$ and finish at another point $\left(x_1^l, x_2^l ... x_n^u\right)$. Both of these points are certainly in the feasible region.

Therefore, the MIP basically relies on a systematic search near the boundary of constraints and involves functional evaluations on feasible points satisfying a specified criterion that a feasible point x would lie within the constraints and the current test point is close to the boundary from within the feasible region. However, the stopping criteria can be chosen depending upon the problem and objective of analysis. One of the ways of choosing the stopping criterion could be a test for maximum permissible slacks defined as:

$$mps_i = \min_j \{c_{ij}\} - \varepsilon\,.$$

To initiate the search, we frequently require the computation of $x_1^{\max}$, and in case of linear constraints $x_1^{\max}$ could be computed as:

$$x_1^{\max} = \min \left\{ x_1 : x_1 = \frac{b_i - \sum_j g_i(x_j; j = \mathbf{2}, \mathbf{3}...n)}{\begin{array}{l} \cos t\ coefficients\ of\ i^{th} \\ type\ constra\text{int}\, s \\ corrresponding\ to\ x_1 \end{array}} \right\}, \tag{33.7}$$

where $g_i(\cdot)$ and b_i are the constraint functions of variables and resources available for an ith type of constraint, respectively.

In case of non-linear constraints, $x_1^{\max}$ can be obtained by incrementing x_1, successively, by one unit at a time until at least one of the constraints gets violated, while keeping x_j at a minimum level at all other stages. It would be computationally advantageous if we compute and store the nonlinear incremental costs for the whole range of x_1 in memory rather than evaluate it every time $x_1^{\max}$ is desired.

33.2.2 Redundancy Allocation Techniques: A Comparative Study

Among the well known methods to provide an exact solution are: (i) Dynamic Programming approach, and (ii) Search Technique, *e.g.,* Cutting plane, Branch and Bound, Implicit search, and partial enumerations using functional evaluations and certain rules for detecting an optimal solution from as few feasible solutions as possible.

Besides the exact techniques, several approximate and earlier methods such as Lagrange Multiplier [1, 8], Geometrical programming and the maximum principle approach [8], Differential dynamic programming sequential simplex search, penalty function approaches, *etc*., have also been employed by treating the integer decision variables as real variables and the final solution is obtained by rounding off the optimal variables to the nearest integers. In view of the fact that a decision problem involving integer variables is NP-complete, a number of heuristic procedures for solving design problems have also been proposed. Some evolutionary techniques inspired by some natural phenomenon such as biological (GA) or functioning of brain (ANN) have also been applied to deal with reliability design problems. Such techniques are known as meta-heuristic algorithms in the literature. The interested reader may refer to [6] for a comprehensive survey and a good account of various reliability design problems, their types and classifications, solution approaches along with the applications of some meta-heuristic techniques (such as GA and ANN).

Summarily, most of the exact integer programming techniques mentioned above, except those which are strictly based on some heuristic criteria, are computationally tedious, time-consuming and sometimes unwieldy, and have limitations of one kind or the other. The other simple techniques are mostly heuristic and thus approximate.

However, the approach described later in this chapter can solve a variety of general integer programming problems also, is simple to comprehend, is amenable to computerization, and easy to formulate. The advantages of the approach over the existing techniques are briefly given as under:

1. It does not require the conversion of variables into binary variables as in [5, 7].
2. It is applicable to a very wide variety of problems, with arbitrary nature of objective and constraint functions and without any assumption on the separability of objective functions. However, the functions involved must be non-decreasing functions of decisions variables.
3. It can solve both integer programming as well as zero-one programming problems with ease and effectiveness.
4. As stated earlier, the present approach to solve redundancy allocation problem is a systematic search near the boundary of feasible reason, so it drastically reduces the number of search points.

33.3 Algorithmic Steps to Solve Redundancy Allocation Problem

The entire algorithm can be summarized in following eight steps [9]:

1. Compute the upper and lower bounds of the decision variables to determine the entire feasible region. Lower bounds are generally known from the system description whereas upper bounds are determined from constraints (see (33.6)). Set $t = 2$ and $\nu = 0$, $x \equiv \left(x_1^u, x_2^l \ldots x_n^l\right)$ and $x^* = x$. If this point is within the slack band $\langle b_i - mps_i, b_i \rangle, \forall i = 1,2 \ldots m$, go to step 8.
2. Set $x_2 = x_2 + 1$. If $x_2 \le x_2^u$, go to next step. Otherwise go to step 4.
3. Keeping all other variables, x_j, j=2,3…n at the current level, determine the value of $x_1^{\max}$ which does not violate any of the constraints (refer to (1.8) and subsequent paragraph). If $x_1^{\max} = 0$, go to next step. Otherwise go to step 7.
4. $\nu = \nu + 1$. if $\nu > (n-2)$, STOP and print the optimal result. Otherwise proceed to step 5.
5. Set $k = t + \nu$ and $x_k = x_k + 1$. If $x_k > x_k^u$, return to step 4. Otherwise proceed to step 6.

6. Set $x_j = x_j^l$ for $j = 2,3...k-1$. Also, set v=0. Return to step 3.
7. Calculate slacks for constraints, $s_i, \forall i = 1,2...m$. If the current point lies within the allowable slacks for all *i*, go to next step. Otherwise return to step 2.
8. Evaluate the objective function *f(x)* at the current point *x*. If it is better than $f(x^*)$, then replace $x^* = x$ and $f(x^*) = f(x)$. Return and continue from step 2.

The algorithmic steps are simple and self explanatory. However, for the reader's benefit some of the steps of algorithm are explained in the following illustration.

Illustration:
Consider a SP system with four subsystems, two linear constraints and with subsystem reliabilities as, $r = [0.85\ 0.80\ 0.70\ 0.75]$, respectively.
Mathematically, we can formulate it as:

$$\textit{Maximize} \quad R_s = \prod_{j=1}^{4} (1-(1-r_j)^{x_j}),$$

Sub. to

$$6.2x_1 + 3.8x_2 + 6.5x_3 + 5.3x_4 \le 51.8$$
$$9.5x_1 + 5.5x_2 + 3.8x_3 + 4x_4 \le 67.8\,.$$

Let us determine the upper and lower bounds (step 1) of the variables involved using (33.6) and the search area bounded by its constraints.

By keeping $x_k = 1, \forall j = 1,2,3,4, k \ne j$, the upper bound of a variable, say x_1, can be determined from constraints as:

$$6.2x_1 + 3.8 + 6.5 + 5.3 \le 51.8 \Rightarrow x_1 \le 5.8387$$
$$9.5x_1 + 5.5 + 3.8 + 4 \le 67.8 \Rightarrow x_1 \le 5.7368,$$

i.e., $x_1^u = \min(5.8387, 5.7368) \cong 5$.

Similarly, $x_2^u = 8, x_3^u = 5, x_4^u = 6$, respectively. Therefore, the starting point in the search would be $x = [5111]$, and will finish once we reach $x = [1116]$.

After following the steps of the algorithm with minimum cost difference of 3.7 units, the optimum system reliability, $R^* = 0.8559967$ is obtained for $x^* = [2232]$. Although the total number of search points in the region is 1200, the functional evaluations performed by the algorithm were only done at 43 points, whereas the number of functional value comparisons to obtain maximum reliability was only 5.

The above steps are the essence of the algorithm, and other variables shown in the algorithmic steps are just to make the translation of the algorithm in a suitable programming language easy.

33.4 Applications of MIP to Various System Design Problems

Here we provide an exhaustive list of applications areas and problem formulations, where the MIP has been successfully applied. The areas are as follows:

33.4.1 Reliability Maximization Through Active Redundancy

33.4.1.1 SP System with Linear and/or Nonlinear Constraints

The series parallel (SP) model is one of the simplest and most widely used models in reliability studies. Mathematically, the problem for such systems could be formulated as:

$$\textit{Maximize} \ R_s = \prod_{j=1}^{n} (1-(1-r_j)^{x_j}), \tag{33.8}$$

$$\textit{Sub. to:} \ g_i(x_j) = \sum_{j=1}^{n} c_{ij} x_j \le b_i, i = 1,2...m, \tag{33.9}$$

where the constraints could either be linear, nonlinear or a combination of both (linear and nonlinear).

Example 1: The illustration taken belongs to this category, where the constraints are linear. The optimal solution point provided by the algorithm is $x^* = [2232]$, with optimal system reliability $R_s^* = 0.85599$, with resource consumptions as 50.1 and 49.9, respectively.

Example 2: Consider a SP System with five subsystems, three nonlinear constraints with subsystem reliabilities $r = [0.80\ 0.85\ 0.90\ 0.65\ 0.75]$, respectively. The problem is to

$$\textit{Maximize}\ R_s = \prod_{j=1}^{5}(1-(1-r_j)^{x_j}),$$

Subject to:

$$x_1^2 + 2x_2^2 + 3x_3^2 + 4x_4^2 + 5x_5^2 \le 60,$$

$$7(x_1 + e^{-x_1/4}) + 7(x_2 + e^{-x_2/4}) + 5(x_3 + e^{-x_3/4}) \\ +9(x_4 + e^{-x_4/4}) + 4(x_5 + e^{-x_5/4}) \le 225,$$

and

$$7x_1e^{x_1/4} + 8x_1e^{x_1/4} + 8x_1e^{x_1/4} + 6x_1e^{x_1/4} \\ +9x_1e^{x_1/4} \le 340.$$

The optimal solution point provided by the algorithm is $x^* = [22223]$ with an optimal value of system reliability of $R_s^* = 0.80247$, and resources consumed of 58, 122.63 and 152.78, units, respectively.

33.4.1.2 NSP System with Linear and/or Nonlinear Constraints

The general formulation of the problem for such systems is:

$$\textit{Maximize}\ R_s = f(x,r) \tag{33.10}$$

$$\textit{Sub. to}\ g_i(x) \le b_i,\ i = 1,2...m, \tag{33.11}$$

where $f(x,r)$ is the reliability function of the NSP system.

Example 3: Consider a NSP system with five nodes and seven links as shown in Figure 33.1.

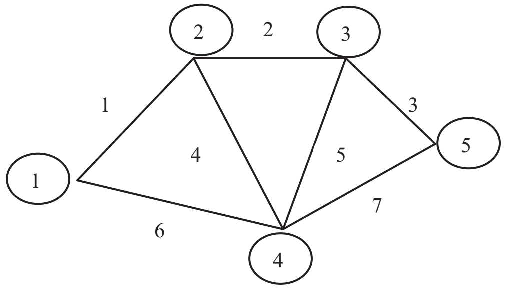

Figure 33.1. A five-node, seven-link NSP system

The objective is to maximize the reliability of the network with the following linear cost constraint:

$$4x_1 + 5x_3 + 2x_4 + 3x_5 + 3x_6 + 5x_7 \le 45,$$

given the component reliabilities as $r = [0.7, 0.9, 0.8, 0.65, 0.7, 0.85, 0.85]$. The optimal reliability of the network computed by the algorithm is $R_s^* = 0.99951$ at $x^* = [1121143]$ with no slacks. Also, out of a total of 11,61,600 search points, it carries out functional evaluations only at 815 points.

Example 4: Consider the bridge network shown in Figure 33.2 with the following three nonlinear constraints:

$$x_1^2 + 2x_2^2 + 3x_3^2 + 4x_4^2 + 5x_5^2 \le 110,$$

$$7(x_1 + e^{-\frac{x_1}{4}}) + 7(x_2 + e^{-\frac{x_2}{4}}) + 5(x_3 + e^{-\frac{x_3}{4}}) \\ +9(x_4 + e^{-\frac{x_4}{4}}) + 5(x_5 + e^{-\frac{x_5}{4}}) \le 175,$$

and

$$7x_1e\frac{x_1}{4} + 8x_2e\frac{x_2}{4} + 8x_3e\frac{x_3}{4} + 6x_4e\frac{x_4}{4} \\ +9x_5e\frac{x_5}{4} \le 200.$$

The optimal allocation for maximizing the reliability of the bridge network is computed to be $x^* = [32343]$, with $R_s^* = 0.99951$, with resource consumption $g_1 = 110, g_2 = 156.55, g_3 = 198.44$,

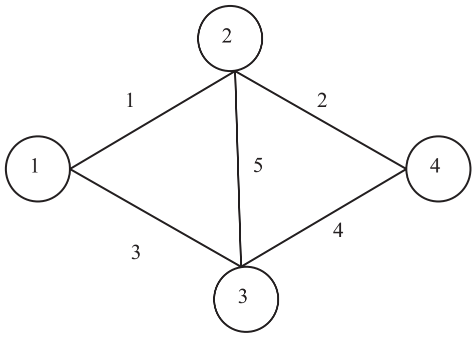

Figure 33.2. A bridge network

respectively. The total number of points visited by the algorithm was 3125, whereas the functional evaluations were at 173 points only.

33.4.2 System with Multiple Choices and Mixed Redundancies

In many engineering applications, it may be possible that the system might need the support of a mixture of available redundancy types (active-parallel, *k*-out-of-*m*, standby, partial, *etc*.) at various subsystems levels. The following example illustrates a typical problem type and its formulation.

Example 5: Consider a three stage series system. The system reliability can be increased by choosing a more reliable component out of four available candidates at stage one. The second stage needs active parallel redundancy whereas the third stage requires a 2-out-of-3:G configuration. The objective is to maximize system reliability with three nonlinear constraints. The formulation of the above problem is as follows.

The objective of the problem can be formulated as:

$$\textit{Maximize}\ R_s = \prod_{j=1}^{n} R_j(x_j), \tag{33.12}$$

where $R_j(x_j)$ is the *j*th subsystem reliability whose form would vary from subsystem to subsystem and therefore no closed form expression for R_s can be obtained. The constraints can either be linear or nonlinear of the type given by (33.11). For this case, we have four choices available for the components of reliability at the first stage, *i.e.*,

$$R_1 = \begin{bmatrix} 0.88 \\ 0.92 \\ 0.98 \\ 0.99 \end{bmatrix} \text{(also called ultiple choice).}$$

At the second stage, there is an active redundant system with single component reliability equal to 0.81. Clearly, the subsystem reliability expression for this stage would be $R_2(x_2) = 1-(1-0.81)^{x_2}$, and third stage has a 2-out-of-x:G subsystem, with unit component reliability = 0.77, *i.e.,* the reliability expression for this stage is given by

$$R_3 = \sum_{2}^{x_3} \binom{x_3}{k} (0.77)(1-0.77)^{x_3-k}.$$

The constraints are:

$$4e^{\left\{\frac{0.02}{1-R_1(x_1)}\right\}} + 5x_2 + 2x_3 \le 45,$$

$$e^{\frac{x_1}{8}} + 3\left(x_2 + e^{\frac{x_2}{4}}\right) + 5(x_3 + e^{\left(\frac{x_3-1}{4}\right)}) \le 65,$$

and

$$8x_2 e^{\frac{x_2}{4}} + 6\left(x_3 - 1\right) e^{\frac{x_3-1}{4}} \le 230.$$

Clearly, the upper and lower bounds of stage one are 1 and 4, respectively, whereas for the others, the bounds would be decided by the constraints and can be computed by using (33.6) (see illustration for how to use the equation).

By following the algorithmic steps, the optimal solution is obtained at $x^* = [3,3,6]$ with $R^* = 0.9702399$. The resources consumed were $g = [37.87, 64.26, 155.52]$. The total search points were 144 with functional evaluations performed at 23 points only.

33.4.3 Parametric Optimization

In many cases, a decision maker would like to know the effects on the solution, if a certain change in constraints values are made. Besides, some constraints values may not be known with certainty (usually they are guessed at). In general, the problems of these types can be transformed into parametric nonlinear programming problems. Several formulations to such problems can be found in [2]. A general parametric programming formulation to reliability design for *n*-stage SP systems is:

$$\text{Maximize } R_s = \prod\left(1-\left(1-r_j\right)^{x_j}\right), \tag{33.13}$$

$$\text{subject to } \sum_{j=1}^{n} g\left(x_j\right) \le b \pm \theta u_i, i = 1,2...m, \tag{33.14}$$

where $0 = \theta_0 < \theta_1 ... < \theta_l = 1, \theta = \theta_0, \theta_1 ... \theta_l$, $x_j \ge 1$ and are integers, and θ, u_i are non-negative constants. The assumptions made in such formulations are:

1. Each stage is essential for overall operational success of the mission.
2. All components are mutually *s*-independent and in the same stage the probability of failure of components is the same.
3. All the components at each stage work simultaneously, and for the stage to fail, all components in that stage must fail.

We provide an illustration for the above formulation.

Example 6: Consider a series system having four stages and two constraints such that we wish to

$$\text{Maximize } \begin{array}{l} R_s = \left(1-\left(1-0.6\right)^{x_1}\right)\left(1-\left(1-0.9\right)^{x_2}\right) \\ \left(1-\left(1-0.55\right)^{x_3}\right)\left(1-\left(1-0.75\right)^{x_4}\right) \end{array},$$

subject to

$$6.2x_1 + 3.8x_2 + 6.5x_3 + 5.3x_4 \le 51.8 + 10\theta$$

$$9.5x_1 + 5.5x_2 + 3.8x_3 + 4.0x_4 \le 67.8 - 15\theta, \quad 0 \le \theta \le 1 .$$

The optimal result was obtained at $\theta = \mathbf{0.37}$, by varying θ between zero to one inclusive, which were the same as obtained by [8]. The optimal allocation was $x^* = [2,2,3,3]$, with $R^* = 0.74401$, and consumed resources were $g = (55.4, 53.4)$.

The summary of optimal results of the numerical examples considered in above sections is shown in Table 33.1.

33.4.4 Optimal Design of Maintained Systems

33.4.4.1 Availability Maximization with Redundancy, Spares and Repair Facility

The availability is a more appropriate measure than reliability or maintainability for maintained systems. So the objective for such systems becomes to maximize availability subjected to multiple constraints (linear and/or nonlinear) taking into account the cost of redundancy, spares and repair facility. Therefore, the formulations to such problems are mostly concerned, directly or

Table 33.1. Summary of the optimal results

Ex. #	Results	Remark
1.	$x^* = [2232]$ $R_s^* = 0.80247$	SP system with two linear constraints
2.	$x^* = [22223]$ $R_s^* = 0.80247$	SP system with three nonlinear constraint
3.	$x^* = [1121143]$ $R_s^* = 0.99951$	NSP with a linear constraint.
4.	$x^* = [32343]$ $R_s^* = 0.99951$	NSP with three nonlinear constraints
5.	$x^* = [3,3,6]$ $R^* = 0.9702399$	Mixed redundant series system with three nonlinear constraints
6.	$x^* = [2,2,3,3]$ $R^* = 0.74401$	Parametric optimization, Series system, two linear constraints

indirectly, with either redundancy or spare parts or repairman allocations. Besides, since these three variables can only assume integer values, availability (a nonlinear function of these variables) optimization would necessitate a nonlinear integer programming formulation but increasing the actual number of variables to three times the number of subsystems as compared to the redundancy allocation problems discussed earlier.

The foregoing section has provided the versatility of the algorithm to deal with redundancy optimization problems related to non-maintained systems. However, it can be applied with equal ease to problems such as spares allocation, repairmen allocation, *etc.*, for maintained system and to multi-criteria optimization.

Let us consider a SP-system of *n* stages, and each stage not only has redundancy but also has a separate maintenance facility like spares and repair in terms of repair. The *j*th stage of such a system is shown in Figure. 33.3, where k_j, σ_j and ρ_j, are the minimum number of components (functional requirement), spares, and used repairmen provided for the *j*th subsystem, respectively. The problem statement of the system is as follows:

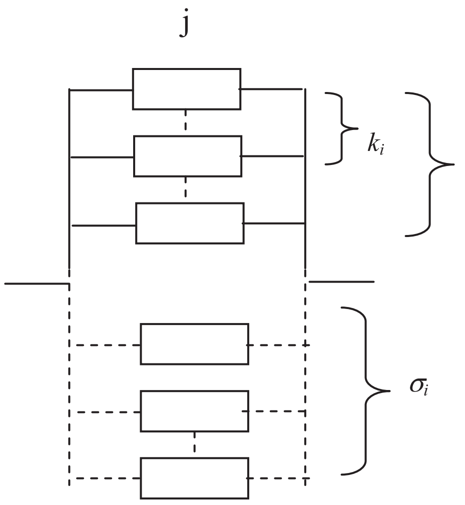

Figure 33.3. A General Subsystem Structure of a j^{th} Stage of a Maintained System

Maximize the availability of a series-parallel maintained system with redundancy, spares, and repair as decision variables, subject to linear constraints. Mathematically, the problem can be expressed as:

$$\textit{Maximize}\ A_s^s = \prod_{j=1}^{n} A_j^s\ ,$$

where the steady-state subsystem availability is expressed as $A_i^s \equiv f(x_j, \sigma_j, \rho_j)$, assuming that all subsystems can be repaired independently. The formulation is subject to the constraints,

$$g_i \equiv \sum_{j=1}^{n} g_{ij}\left\{(x_j - k_j), \sigma_j, \rho_j\right\} \le b_i, i = 1, 2 \ldots m\ .$$

The details of various symbols, assumptions, mathematical formulation and solution of the above problem using the present algorithm have been provided in [15, 17].

33.4.5 Computer Communication Network Design with Linear/Nonlinear Constraints and Optimal Global Reliability/Availability

A CCN is defined as a collection of nodes, *N,* at which the computing resources reside, which communicate with each other via a set of data communicating channels (set of links), *L*. The main objective of such a CCN is to provide efficient communication among various computer centers in order to increase their utility and to make their services available to more users. One of the fundamental desiderata in designing such a system is that of global availability, *i.e.*, the probability that the network is at least simply connected (connectedness), which depends on the topological layout and the availability of individual computer systems and communication facilities.

Assuming link duplexity, a two state model (working or failed), and the presence/absence of a link in a network can be represented by a binary variable, taking a value either zero or one. The problem for such networks can be stated as [16]: determine an optimal CCN topology that gives maximum overall availability within the given permissible cost. In other words, the objective is to find a set of links from a given set of links, which together constitute an optimal CCN topology within the budgetary constraints of C_s. Mathematically, the problem can be expressed as:

$$Maximize\ A_s \equiv f(A_1, A_2 \ldots A_n) = f(x_1, x_2 \ldots x_n),$$

subject to:

$$\sum_{j=1}^{n} x_j(y_{j1} y_{j2} \ldots y_{jn}) = \{11\ldots1\}, \quad with\ \delta(x) \ge N-1 \tag{33.15}$$

$$\sum_{j=1}^{n} c_j x_j \le c_s\ . \tag{33.16}$$

The form of A_s entirely depends on the network topology and is a minimized expression of global reliability/availability of the network, which can be obtained if the spanning trees of the network are known. In the above formulation, (33.15) signifies the continuity constraint, which ensures that the allocation of the decision variables provides a globally available network. The summation and product sign in the constraint represents binary sum and product, respectively. Note that here $x_j(y_{j1} y_{j2} \ldots y_{jn}) = (x_j y_{j1})(x_j y_{j2}) \ldots (x_j y_{jn})$, and $(y_{j1} y_{j2} \ldots y_{jn})$ is a string of binary variables corresponding to the *j*th link, *e.g.,* if L_j connects the *p*th and *q*th nodes, then $y_{jp}, y_{jq} = 1$, and $y_{jk} = 0 \forall k \ne p \ne q$, *e.g.,* if L_4 connects the second and fourth nodes in a five node network then $(y_{41} y_{42} y_{43} y_{44} y_{45}) = \{01010\}$. Variable x_j can be either one or zero and represents the presence/absence of the link in the network. The cost constraint (33.16) is self explanatory. The details of the above problem and application of the algorithm can be found in [17, 18].

33.4.6 Multicriteria Redundancy Optimization

In many situations, the reliability design problems of complex engineering systems may necessitate consideration of several non-commensurable criteria, which may be equally important. In order to offer alternatives to a system designer, it may be better to achieve some kind of balance among the several conflicting properties rather than to optimize just one property. This situation can be mathematically formulated as multicriteria optimization problem in which the designer's goal is to minimize or maximize not a single objective function but several functions simultaneously.

A multi-objective optimization problem, in general, can be stated as follows:

Find a vector $x^* = [x_1^*, x_2^* \ldots x_n^*]$, which satisfies m inequality constraints

$$g_i(x) \ge 0, i = 1, 2 \ldots m$$

and p equality constraints

$$h_u(x) = 0, u = 1, 2 \ldots p < n,$$

such that the vector function

$$f(x) = [f_1(x), f_2(x) \ldots f_k(x)]$$

gets optimized, where $x^* = [x_1^*, x_2^* \ldots x_n^*]$ is a vector of decision variables defined in the n-dimensional Euclidean space of variables E^n $f(x) = [f_1(x), f_2(x) \ldots f_k(x)]$ is a vector function defined in k-dimensional Euclidean space of objectives E^k, and $g_i(x)$, $h_u(x)$, and $f_l(x)$ are linear and/or nonlinear functions of variables $x_1^*, x_2^* \ldots x_n^*$. The constraints (equality and inequality) define the feasible region X and any point x in X defines a feasible solution.

In fact, the task involved in multi-criteria decision making is to find a vector of the decision variables which satisfies constraints and optimizes a vector function whose elements represent several objective functions. These functions form a mathematical description of the performance criteria, which are usually in conflict with each other. Therefore, the term "optimize" here would mean finding a solution which provides acceptable values for all the objective functions simultaneously.

The extension to MIP to such problems provides an efficient approach for solving multicriteria reliability design problems. This is accomplished in combination with the min-max approach for generating Pareto optimal solutions for multicriteria optimization. For detailed discussions, examples and their solutions thereof, the interested reader can refer to [11].

33.5 Conclusions

The search approach presented in this chapter is quite versatile in dealing with problems involving integer programming formulations arising in reliability design. The approach can be easily programmed using any suitable language a user is familiar with. It does not require proving any conditions of convexity, concavity or differentiability of involved functions (objective and constraints) in the optimization process. It is simple, requiring only objective function evaluations for testing the feasibility of very few solution vectors in the search space bounded by the constraints and a comparison with the previous value of the evaluated function.

The major benefit that one can draw from such a search pattern is that for a given set of constraints, the search pattern is independent of the objective function involved. This allows a designer to change the objective function without changing the search pattern and one only need evaluate the objective function to arrive at a different optimal solution. This may be found useful in studying various configurations of constituent subsystems/ components for optimal reliability or any other measures of system performance.

The technique is not only an effective and efficient tool for problems involving single objective functions but is also suitable for problems involving multiple objective functions.

References

[1] Becker PW. The highest and lowest reliability achievable with redundancy. IEEE Transactions on Reliability 1977; R-26:209–213.

[2] Chern MS, Jan RH. Parametric programming applied to reliability optimization problems. IEEE Transactions on Reliability 1985; R-34(2):165–170.

[3] Everett III H. Generalized Lagrangian multiplier method for solving problems of optimum allocation of resources. Operations Research 1963; 11:339–417.

[4] Federowicz AJ, Mazumdar M. Use of geometrical programming to maximize reliability achieved by redundancy. Operations Research 1968; 16:948–954.

[5] Geoffrion AM. Integer programming by implicit enumeration and Bala's method. Society of Industrial and Applied Mathematics Review 1967; 9:178–190.

[6] Kuo W, Prasad VR, Tillman FA, Hwang C. Optimal reliability design: fundamentals and applications. Cambridge University Press, 2001.

[7] Lawler E, Bell MD. A method for solving discrete optimization problems. Operations Research 1966; 14:1098–1112.

[8] Misra KB. Reliability optimization of series-parallel system Part-I: Lagrangian multiplier approach Part:II maximum principle approach. IEEE Transaction on Reliability 1972; R-21(4):230–238.

[9] Misra KB. Search procedure to solve integer programming problems arising in reliability design of a system. International Journal of Systems Science 1991; 22(11):2153–2169.

[10] Misra KB. Reliability analysis and prediction: A methodology oriented treatment. Elsevier, Amsterdam, 1992.

[11] Misra KB. Multicriteria redundancy optimization using an efficient search procedure. International Journal of System Science.1991; 22(11):2171–2183.

[12] Misra K, Misra V. Search method for solving general integer programming problems. International Journal of System Science 1993; 24(12): 2321–2334.

[13] Misra KB (Editor). New trends in system reliability evaluation. Elsevier, Amsterdam, 1993.

[14] Ohno K. Differential dynamic programming for solving nonlinear programming problems. Journal of Operations Research Society of Japan 1978; 21:371–398.

[15] Sharma U, Misra KB. Optimal availability design of a maintained system. Reliability Engineering and System Safety 1988; 20:146–159.

[16] Sharma U, Misra KB, Bhattacharji A.K. Optimization of computer communication networks: Exact and heuristic approaches, Microelectronics and Reliability 1990; 30: 43–50.

[17] Sharma U. On some aspects of reliability design of complex systems. Ph. D. Thesis, Guide: Misra KB, Reliability Engineering Centre, IIT Kharagpur, 1990.

[18] Sharma U, Misra KB, Bhattacharji AK. Applications of an efficient search technique for optimal design of a computer communication network. Microelectronics and Reliability 1991; 31:337–341.