STUDIES ON OPTIMAL DESIGN OF HIGH RELIABLE SYSTEM: SINGLE AND MULTIPLE OBJECTIVE NONLINEAR INTEGER PROGRAMMING

Yuji Nakagawa

December 1978

STUDIES ON OPTIMAL DESIGN OF HIGH RELIABLE SYSTEM: SINGLE AND MULTIPLE OBJECTIVE NONLINEAR INTEGER PROGRAMMING

by

Yuji Nakagawa

Submitted in partial fulfillment of the requirement for the degree of

DOCTOR OF ENGINEERING

at

KYOTO UNIVERSITY

Kyoto, JAPAN

December 1978

ABSTRACT

This dissertation is devoted to the development of methods for the optimal design of high reliable systems.

Chapter 1 defines and describes reliabilities of units, subsystems and systems, and formulates reliability optimization problems with a single and multiple objective functions in the general form. This chapter mainly prepares for discussions of the succeeding chapters.

In Chapter 2, a method is presented for generating systematically all possible series-parallel (s-p) structures for 1,...,M units. The method is modified in order to calculate the failure probabilities and reliabilities of all possible s-p redundant structures for 1,...,M units with two failure states. Then we apply the present methods to a problem of finding an optimal s-p redundant structure subject to four reliability, two failure probabilities and fail-safe.

Chapter 3 presents a heuristic method for obtaining an optimal reliability allocation of a series system. In each subsystem, redundant components can be added (in parallel, stand-by, or k-out-of-n:G, etc.), or a more reliable unit can be used to improve the system reliability. The solution is obtained by repeatedly using a more reliable candidate at each subsystem that has the greatest value of a 'weighted sensitivity function'. The balance between the objective function and the constraints is controlled by a 'balancing coefficient'. The overall computaional procedure is given and an example is presented. The computations are given for a set of randomly generated test problems

in which the optimal parallel redundancy under linear constraints is determined. The proposed method is then compared with other methods.

In Chapter 4, this dissertation presents an efficient method for finding the exact optimal solutions of reliability allocation problems that are formulated as an integer nonlinear programming problem generalized to handle nonlinear constraints and non-separalbe problems. The method is based on branch-and-bound principle and developed by considering separation and relaxation techniques.

In Chapter 5, we consider the problem of determining an optimal design of a reliability system with multiple properties and integer variables. This problem is formulated as a multiobjective nonlinear pure-integer programming problem. This dissertation presents a method for solving the multiobjective programming problem, which consists of three techniques: 1)narrowing a given feasible region if the region is too wide, 2)obtaining exactly the set of Pareto-optimal solutions, and 3)selecting the best one for decision makers from the set of Pareto-optimal solutions. This method has an interesting merit that an increase in the number of objective functions scarcely affects the size of problem. An example is included.

ACKNOWLEDGMENTS

The auther is grateful for this opportunity to express his appreciation to the following individuals for their aid and encouragement during the completion of this study and the pursuit of his doctorate:

Dr. Y. Hattori, the author's major advisor, for his guidance, encouragement and helpful suggestions during all stages of this study.

Dr. H. Mine, Professor of Kyoto University, and Dr. M. Kodama, Professor of Nagoya Institute of Technology, for their encouragement and confidence they so willingly offered.

Dr. R. A. Evans, Editor of IEEE Transactions on Reliability, for his useful comments and suggestions for this study.

Dr. T. Ibaraki, Associate Professor of Kyoto University, and Dr. K. Ohno, Associate Professor of Kyoto University, for helpful discussions.

Mr. K. Nakashima for having called the author's attention to the study of reliability theory when the author was a student of Himeji Institute of Technology and for his continual encouragement and valuable suggestions.

The member of Professor Hattori's Laboratory, Institute of Atomic Energy, Kyoto University, Dr. H. Kokame and Mr. D. Tanaka for advice and discussions.

Mr. T. Inagaki, doctoral candidate in the Department of Precision Mechanics of Kyoto University, for valuable discussions.

The author's wife, Keiko, who sacrificed her own needs in order to help his pursuit of the doctorate and who offered constant encouragement when it was greatly needed.

Mrs. E. Hayashi for preparing and typing the original draft of the manuscript.

Mis. Y. Nakatani for typing the final draft of the manuscript.

CONTENTS

Chapter 1

INTRODUCTION

1.1 INTRODUCTION

There is a basic conflict in increasing the reliability of a system. The improvement of reliability is causative of increasing the consumed amounts of resources; e.g., cost, weight, volume, area, time. This conflict cannot be circumvented, but it can be minimized through optimum design. The conflict between quality and the outlay of resources is presented everywhere. It is prominent, for example, in the design of complex electronic equipment for space use. There are constraints on some of the resources. In the case of space systems, the payload weight is limited by the capability of the launch vehicle. There are also often minimum acceptable reliability requirements.

This dissertation is devoted to the development of methods for the optimal design of reliability systems. Chapter 2 presents a method for generating systematically and exactly all the possible series-parallel (s-p) structures for 1,...,M units. The method is modified in order to calculate the failure probabilities and reliabilities of all the possible s-p redundant structures for 1,...,M units with two failure states. Then we apply the present methods to a problem of finding an optimal s-p redundant structure subject to four constraints for reliability, two failure probabilites and fail-safe.

Chapter 3 presents a heuristic method for obtaining an optimal reliability allocation of a series system. In each subsystem, redundant components can be added (in parallel, stand-by, k-out-of-n:G, etc.), or a more

reliable unit can be used in order to improve the system reliability. The solution is obtained by repeatedly using a more reliable candidate at each subsystem that has the greatest value of a 'weighted sensitivity function'. The balance between the objective function and the constraints is controlled by a 'balancing coefficient'. The overall computational procedure is given and an example is presented. The computations are given for a set of randomly generated test problems in which the optimal parallel redundancy under linear constraints is determined. The proposed method is then compared with other methods.

In Chapter 4, this dissertation presents an efficient method for finding the exact optimal solutions of reliability allocation problems that are formulated as an integer nonlinear programming problem generalized to handle nonlinear constraints and non-separable problems. The method is based on branch-and-bound principle and developed by concidering separation and relaxation techniques.

In Chapter 5, we consider the problem of determining an optimal design of a reliability system with multiple properties and integer variables. This problem is formulated as a multiobjective nonlinear pure-integer programming problem. This dissertation presents a method for solving the multiobjective programming problem, which consists of three techniques; 1)narrowing a given feasible region if the region is too wide, 2)obtaining exactly the set of Pareto-optimal solutions, and 3)selecting the best one for decision makers from the set of Pareto-optimal solutions. This method has an interesting merit that an increase in the number of objective functions scarcely affects the size of problem. An example is included.

Through the dissertation, the term 'reliability' is taken to mean the probability that a system, a subsystem or a unit will operate successfully during a given time interval $[0,t^*]$. The 'system'means what is constructed by multiple 'subsystems', each of which is composed of single or multiple 'units'. The unit is used to denote the basic element level under consideration. A unit may be a relay or a piece

of equipment. All units in a system are assumed to be statistically independant and identically distributed.

The definitions and descriptions about reliabilities of units, subsystem and system that are relevant to this dissertation are given in the following Sections 1.2, 1.3, 1.4. In Section 1.5, we formulate reliability optimization problems with a single and multiple objective functions in the general form.

1.2 UNIT RELIABILITY

1.2.1 1-failure-state unit

If we need not to consider multiple failure states of a unit when we treat its reliability, the unit will be called a 1-failure-state unit and has two states: success and failure. The reliability p_i of a unit used in a subsystem i means the probability that the unit will operate successfully during $[0,t^*]$. The unreliability (failure probability) $q_i = 1 - p_i$ can be obtained as follows:

1) If a given collection of the units is tested for the time t^*, we can obtain the estimate $\hat{q}_i$ of q_i, which is called empirical reliability.

$$\hat{q}_i = \frac{\text{number of units failed}}{\text{number of units tested}} . \tag{1.1}$$

When a great many of units are tested,

$$q_i \simeq \hat{q}_i . \tag{1.2}$$

2) Assume that the failure distribution of the unit is identical. If the failure distribution $F(t)$ $(-\infty < t < \infty)$ has a density $f(t)$, the failure rate function $r(t)$ is defined for values of t by

$$r(t) = \frac{f(t)}{1 - F(t)}. \tag{1.3}$$

These quantities are related by

$$F(t) = \int_{-\infty}^{t} f(\tau)\, d\tau \tag{1.4}$$

$$= 1 - \exp\left[-\int_{-\infty}^{t} r(\tau)\, d\tau \right]. \tag{1.5}$$

Then the failure probability q_i is obtained by

$$q_i = F(t^*). \tag{1.6}$$

Typical useful failure laws which have been assumed are given next. The failure rate is given when it is a simple expression.

(i) The exponential distribution:

$$f(t) = \begin{cases} 0 & (t \leq 0) \\ \lambda e^{-\lambda t} & (t > 0), \end{cases} \tag{1.7}$$

$$r(t) = \lambda, \tag{1.8}$$

$$F(t) = 1 - e^{-\lambda t}, \tag{1.9}$$

where $\lambda > 0$.

(ii) The Rayleigh distribution:

$$f(t) = \begin{cases} 0 & (t \leq 0) \\ \lambda t e^{-\lambda t^2/2} & (t > 0), \end{cases} \tag{1.10}$$

$$r(t) = \lambda t, \tag{1.11}$$

$$F(t) = 1 - e^{-\lambda t^2/2} \tag{1.12}$$

where $\lambda > 0$.

(iii) The Weibull distribution:

$$f(t) = \begin{cases} 0 & (t \leq 0) \\ \lambda \alpha t^{\alpha-1} e^{-\lambda t^{\alpha}} & (t > 0), \end{cases} \tag{1.13}$$

$$r(t) = \lambda \alpha t^{\alpha-1}, \tag{1.14}$$

$$F(t) = 1 - e^{-\lambda t^{\alpha}}, \tag{1.15}$$

where $\lambda, \alpha > 0$.

(iv) The modified extreme value distribution:

$$f(t) = \begin{cases} 0 & (t < 0) \\ \frac{1}{\lambda} \exp\left[-\frac{e^t - 1}{\lambda} + t \right] & (t > 0), \end{cases} \tag{1.16}$$

$$r(t) = \frac{e^t}{\lambda} \tag{1.17}$$

$$F(t) = 1 - \exp[- \frac{e^t - 1}{\lambda}], \tag{1.18}$$

where $\lambda > 0$.

(v) The normal distribution:

$$f(t) = \frac{1}{\sigma\sqrt{2\pi}} \exp[- \frac{(t - \mu)^2}{2\sigma^2}], \tag{1.19}$$

$$F(t) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{t} \exp[- \frac{(t - \mu)^2}{2\sigma^2}] \, d\tau , \tag{1.20}$$

where $\sigma > 0$, $-\infty < \mu \leq \infty$.

(vi) The gamma distribution:

$$f(t) = \begin{cases} 0 & (t \leq 0) \\ \dfrac{\lambda^{\alpha} t^{\alpha-1} e^{-\lambda t}}{\Gamma(\alpha)} & (t > 0), \end{cases} \tag{1.21}$$

$$F(t) = \frac{\lambda^{\alpha}}{\Gamma(\alpha)} \int_0^t \tau^{\alpha-1} e^{-\lambda\tau} \, d\tau, \tag{1.22}$$

where $\lambda, \alpha > 0$ and

$$\Gamma(\alpha) = \int_0^\infty \tau^{\alpha-1} e^{-\tau}\, d\tau. \tag{1.23}$$

(vii) The log normal distribution:

$$f(t) = \begin{cases} 0 & (t \le 0) \\ \dfrac{1}{t\sigma\sqrt{2\pi}} \exp\left[-\dfrac{(\log t - \mu)^2}{2\sigma^2} \right] & (t > 0), \end{cases} \tag{1.24}$$

$$F(t) = \frac{1}{\sigma\sqrt{2\pi}} \int \frac{1}{\tau} \exp\left[-\frac{(\log\tau - \mu)^2}{2\sigma^2} \right] d\tau, \tag{1.25}$$

where $-\infty < \mu < \infty$, $\sigma > 0$.

1.2.2 2-Failure-State Unit

If we must consider 2 failure states of a unit (as relay, switch, sensor, valve, etc.) to treat its reliability, the unit will be called 2-failure-state unit (Fig. 1.1). There is no restriction on the number or kind of inputs to the unit; the inputs are irrelevant to this discussion except as they are implicit in the definition of success. There is only one output, and there are exactly 2 output states (called '0' and '1').

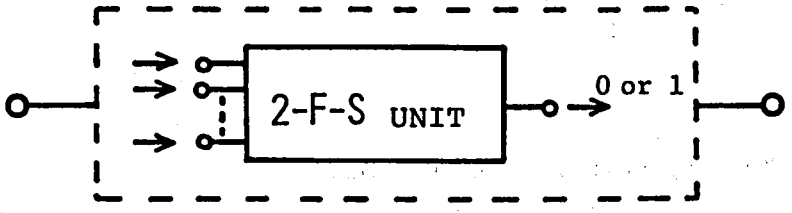


Fig. 1.1 2-failure-state unit.

The unit has exactly 3 states:

(i) success: the output follows the inputs according to the nature of the device.

(ii) or (iii) failure 'i': the output is stuck-at-i, for i=0,1.

The two failure probabilities q_i^0 and q_i^1 are defined as follows:

q_i^0 the probability that a unit used in subsystem i will fail as 0 during $[0,t^*]$,

q_i^1 the probability that a unit used in subsystem i will fail as 1 during $[0,t^*]$.

The empirical unreliabilities $\hat{q}_i^0$ and $\hat{q}_i^1$ are

$$\hat{q}_i^0 = \frac{\text{number of units failed as 0}}{\text{number of units tested}}, \tag{1.26}$$

$$\hat{q}_i^1 = \frac{\text{number of units failed as 1}}{\text{number of units tested}}. \tag{1.27}$$

When we know the failure distribution functions $F^0(t)$, $F^1(t)$ of failures '0', '1' of a unit, respectively, q_i^0 and q_i^1 are given by

$$q_i^0 = F^0(t^*), \tag{1.28}$$

$$q_i^1 = F^1(t^*). \tag{1.29}$$

1.3 SUBSYSTEM RELIABILITY

1.3.1 Improvement of Subsystem Reliability

There are essentially two basic methods of improving the reliability of any subsystem. One method is to increase the reliability of unit by 1)using high reliable unit, 2)redesigning, 3) derating, 4) making stronger against the environmental stress, etc. Another method is the use of redundant units. By using the methods, we can obtain some candidates (alternatives), having different reliabilities and different amounts of resources, for each of subsystems.

When units in a candidate have 1 or 2 failure states, the candidate itself is a 1-failure-state or a 2-failure-state device, respectively.

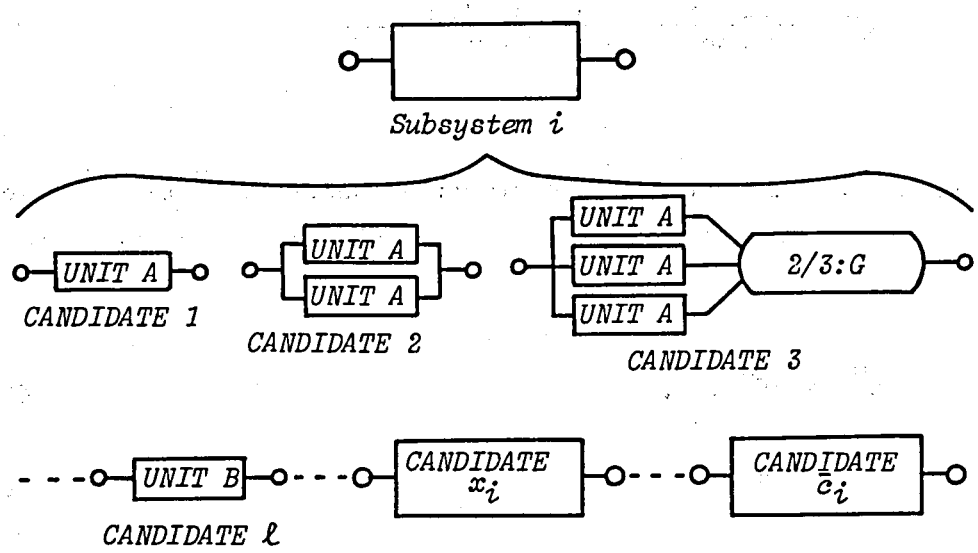


Fig. 1.2 An example of candidates for subsystem i.

We use the following notation.

x_i	candidate number; $\underline{c}_i \le x_i \le \bar{c}_i$,
$Q_i(x_i)$	probability that a 1-failure-state candidate x_i for subsystem i will fail during [0,t*],
$Q_i^0(x_i)$	probability that a 2-failure-state candidate x_i for subsystem i will fail as 0 during [0,t*],
$Q_i^1(x_i)$	probability that a 2-failure-state candidate x_i for subsystem i will fail as 1 during [0,t*].

The reliability of subsystem i is as follows:

(i) When the candidate x_i is a 1-failure-state device,

$$R_i(x_i) = 1 - Q_i(x_i) \tag{1.30}$$

(ii) When the candidate x_i is a 2-failure-state device,

$$R_i(x_i) = 1 - Q_i^0(x_i) - Q_i^1(x_i) \ . \tag{1.31}$$

If the candidate x_i is composed of a single unit, then

$$Q_i(x_i) = q_i \tag{1.32}$$

or

$$Q_i^0(x_i) = q_i^0 \tag{1.33}$$

$$Q_i^1(x_i) = q_i^1 \ , \tag{1.34}$$

where q_i is the failure probability of x_i for 1-failure-state case, and q_i^0, q_i^1 are the failure probabilities of x_i for 2-failure-state case.

1.3.2 Redundancies for 1-Failure-State Units

All units in each of subsystems are assumed to be statistically alike. The several redundancies for 1-failure-state units are as follows:

(i) Parallel redundancy: The candidate x_i is failed if and only if all units are failed.

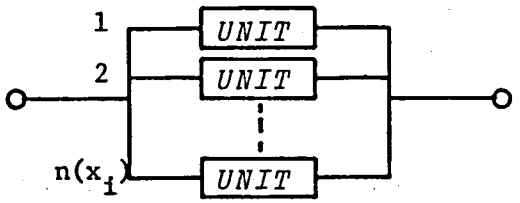


Fig. 1.3 Parallel redundancy

$$Q_i(x_i) = q_i^{n(x_i)} \tag{1.35}$$

where q_i is the failure probability of the unit used in candidate x_i and $n(x_i)$ is the number of units used in candidate x_i.

(ii) k-out-of-$n(x_i)$:G [or k-out-of-$n(x_i)$:F] redundancy: The candidate x_i is good [failed] if and only if at least k of its $n(x_i)$ units are good [failed].

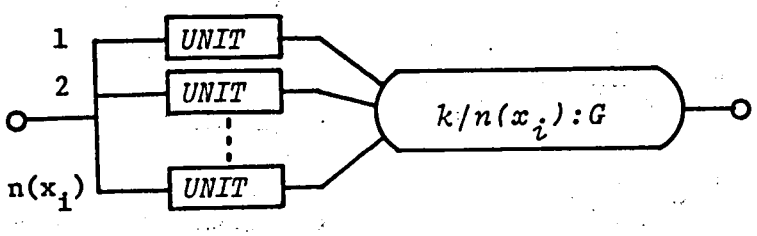


Fig. 1.4 k-out-of-$n(x_i)$:G redundancy

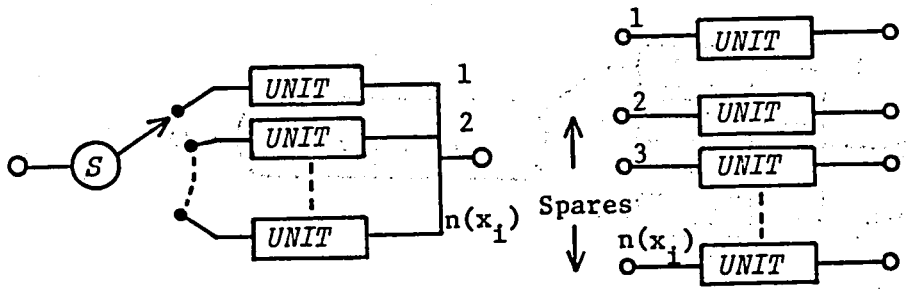
$$Q_i(x_i) = \sum_{j=k}^{n(x_i)} \binom{n(x_i)}{j} (1 - q_i)^j q_i^{n(x_i)-j} \tag{1.36a}$$

$$\left[= \sum_{j=k}^{n(x_i)} \binom{n(x_i)}{j} q_i^{\,j} (1 - q_i)^{n(x_i)-j} \right] \tag{1.36b}$$

where the $k/n(x_i)$:G [$k/n(x_i)$:F] device is assumed to be perfect. Note that the $k/n(x_i)$:G [or $k/n(x_i)$:F] device does not often exist explicitly.

(iii) Stand-by [or spare] redundancy:

If we assume that perfect failure detection and switching [replacement] are made and the stand-by [spare] units do not age, then the candidate x_i is failed if only if and only if all $n(x_i)$ stand-by [spare] units are failed.



(a) Stand-by (b) Spare

Fig. 1.5 Stand-by and spare redundancy

The failure probability of the candidate depends on the failure distribution function of units.

a) The exponential distribution:

$$Q_i(x_i) = \sum_{j=k}^{n(x_i)-1} \frac{(\lambda t^*)^j}{j!} e^{-\lambda t^*} \tag{1.37}$$

$$\simeq \frac{(\lambda t^*)^{n(x_i)}}{n(x_i)!\left(1 - \frac{\lambda t^*}{n(x_i)+1}\right)} e^{-\lambda t^*} . \tag{1.38}$$

b) The normal distribution:

$$Q_i(x_i) = \frac{1}{n(x_i)\sigma\sqrt{2\pi}} \int_{-\infty}^{t^*} \exp\left[- \frac{(\tau - n(x_i)\mu)^2}{2(n(x_i)\sigma)^2} \right] d\tau, \tag{1.39}$$

c) The gamma distribution:

$$Q_i(x_i) = \frac{\lambda^{n(x_i)\alpha} (t^*)^{n(x_i)\alpha-1}}{\Gamma(n(x_i)\alpha)} e^{-\lambda t^*}, \qquad (1.40)$$

1.3.3 Redundancies for 2-Failure-State Units

In this section, too, we assume that all units in each of candidates are statistically alike. The several redundancies for 2-failure-state units are as follows:

(i) Series-parallel redundancy:

This redundancy will be treated in detail in the next chapter.

(ii) series and (iii) parallel redundancies:

These redundancies are special cases of (i).

(iv) k-out-of-$n(x_i)$:1 [or k-out-of-$n(x_i)$:0] redundancy:

The output of candidate x_i is 1 [0] if and only if at least k outputs of its $n(x_i)$ units are 1 [0].

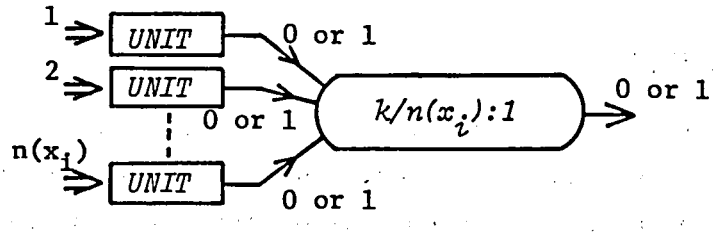


Fig. 1.6 k-out-of-$n(x_i)$:1 redundancy

The failure probabilities that the candidate will fail stuck-at 0 and 1 are, respectively,

$$Q_i^0(x_i) = q_v^0 + (1 - q_v^0 - q_v^1) \sum_{j=n(x_i)-k+1}^{n(x_i)} \binom{n(x_i)}{j} (q_i^0)^j (1 - q_i^0)^{n(x_i)-j} \tag{1.41a}$$

$$\left[= q_v^0 + (1 - q_v^0 - q_v^1) \sum_{j=k}^{n(x_i)} \binom{n(x_i)}{j} (q_i^0)^j (1 - q_i^0)^{n(x_i)-j} \right] \tag{1.41b}$$

$$Q_i^1(x_i) = q_v^1 + (1 - q_v^0 - q_v^1) \sum_{j=k}^{n(x_i)} \binom{n(x_i)}{j} (q_i^1)^j (1 - q_i^1)^{n(x_i)-j} \tag{1.42a}$$

$$\left[= q_v^1 + (1 - q_v^0 - q_v^1) \sum_{j=n(x_i)-k+1}^{n(x_i)} \binom{n(x_i)}{j} (q_i^1)^j (1 - q_i^1)^{n(x_i)-j} \right] \tag{1.42b}$$

where q_v^0 , q_v^1 are the probabilities that $k/n(x_i):1$ [or $k/n(x_i):0$] device (voter) will fail stuck-at 0 and 1, respectively.

1.4 SYSTEM RELIABILITY

The configurations of reliability systems can be roughly divided into two groups: series and non-series. Any system in which the system success is obtained by the success of all its subsystems is a

series configuration. The other systems are non-series configuration. Optimization problems treating series system are generally separable for each variable. In the case of non-series system, however, the optimization problems are non-separable.

When we use candidates x_i (i=1,...,n) as subsystem i of a system, the reliability R(x) of the system (the probability that the system operates successfully during [0,t*]) is as follows:

(i) Series system:

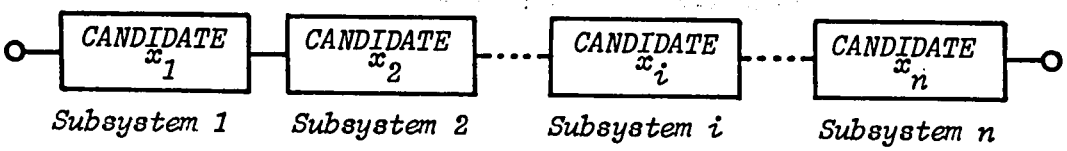


Fig. 1.7 Series system

$$R(X) = \prod_{i=1}^{n} R_i(x_i). \tag{1.43}$$

(ii) Non-series system:

Since the reliability functions of non-series systems can not be written in the general form, let us give two examples of series-parallel and non-series-parallel systems:

a) Series-parallel system:

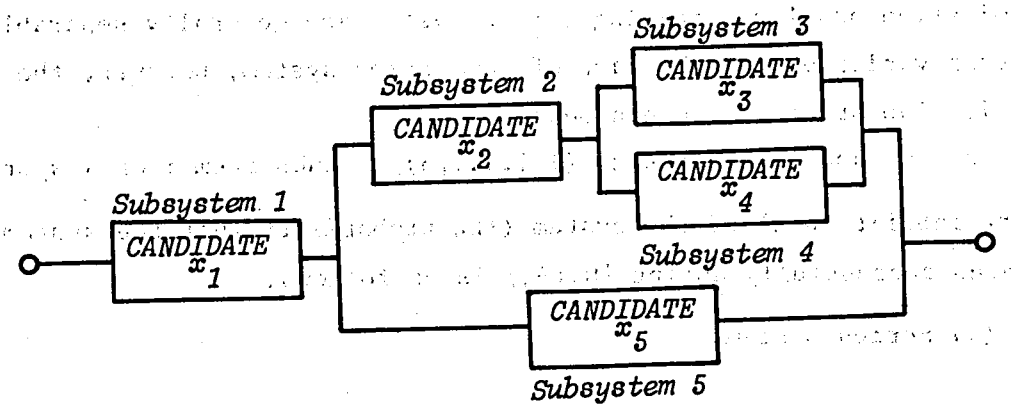


Fig. 1.8 Series-parallel system

$$R(X) = R_1(x_1)R_2(x_1)\{1 - (1 - R_3(x_3))(1 - R_4(x_4))(1 - R_5(x_5))\}$$
$$+ R_1(x_1)(1 - R_2(x_2))R_5(x_5)$$

$$= R_1(x_1)R_5(x_5) + R_1(x_1)R_2(x_2)R_3(x_3) + R_1(x_1)R_2(x_2)R_4(x_4)$$
$$- R_1(x_1)R_2(x_2)R_3(x_3)R_4(x_4) - R_1(x_1)R_2(x_2)R_3(x_3)R_5(x_5)$$
$$- R_1(x_1)R_2(x_2)R_4(x_4)R_5(x_5) + R_1(x_1)R_2(x_2)R_3(x_3)R_4(x_4)R_5(x_5). \quad (1.44)$$

b) Non series-parallel system:

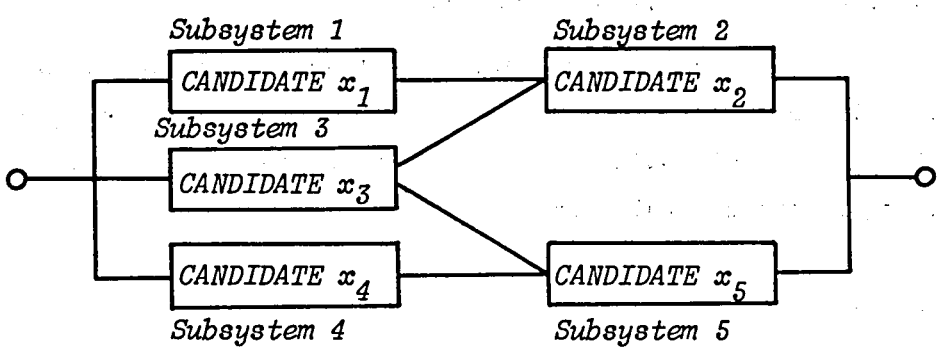


Fig. 1.9 Non series-parallel system

$$R(X) = R_3(x_3)\{1 - (1 - R_2(x_2))(1 - R_5(x_5))\}$$

$$+ (1 - R_3(x_3))\{1 - (1 - R_1(x_1)R_2(x_2)(1 - R_4(x_4)R_5(x_5))\}$$

$$= R_1(x_1)R_2(x_2) + R_2(x_2)R_3(x_3) + R_3(x_3)R_5(x_5)$$

$$- R_1(x_1)R_2(x_2)R_3(x_3) - R_2(x_2)R_3(x_3)R_5(x_5) - R_3(x_3)R_4(x_4)R_5(x_5)$$

$$- R_1(x_1)R_2(x_2)R_4(x_4)R_5(x_5) + R_1(x_1)R_2(x_2)R_3(x_3)R_4(x_4)R_5(x_5) . \quad (1.45)$$

1.5 RELIABILITY OPTIMIZATION PROBLEM

In this section, we formulate a reliability optimization problem with a single objective function (scalar optimization problem) and a multiobjective reliability optimization problem (vector optimization problem) in the general form.

1.5.1 Scalar Optimization Problem

Consider a reliability system having several properties (e.g., reliability, availability, cost, volume , weight and time). When we try to design the system, we will be faced with the conflict that, the better a design alternative is for a property, the worse it becomes for some other properties. We can reduce the conflict by considering a scalar optimization problem formulated by selecting the most important or appropriate property as an objective function and by treating the other important properties as the constraints.

A general formulation of reliability optimization problem with a single objective function is

Problem A: Choose the optimal candidate x_i for each subsystem i from the candidates prepared, so that the objective function is maximized (or minimized) without violating nonlinear constraints:

$$\text{maximize (or minimize)} \quad f(X)$$

$$\text{subject to} \quad g_j(X) \leq b_j \quad (j = 1,\ldots,m),$$

$$X = (x_1,\ldots,x_n),$$

$$\underline{c}_j \leq x_i \leq \bar{c}_i, \; x_i \text{ integer}.$$

We will treat the special cases of this problem in Chapters 3 and 4.

Problem A includes most of the conventional reliability optimization problems as its special cases. Some of the conventional problems are as follows:

(i) Problem A_1: Allocate the number x_i of parallel redundant units (its unreliability is q_i) to each subsystem i so that the reliability of series system is maximized subject to a single or multiple linear constraints [B8,M7]:

$$\text{maximize} \prod_{i=1}^{n} (1 - q_i^{x_i+1})$$

$$\text{subject to} \sum_{i=1}^{n} a_{ji} x_i \leq b_j \quad (j = 1,\ldots,m),$$

$$x_i \geq 0 \quad \text{integer}.$$

(ii) Problem A_2: Minimize a linear cost function without allowing the system reliability to drop below a given acceptable level R_{min} [M25]:

$$\text{minimize} \sum_{i=1}^{n} c_i x_i$$

$$\text{subject to} \prod_{i=1}^{n} (1 - q_i^{x_i+1}) \leq R_{min},$$

$$x_i \geq 0 \quad \text{integer}.$$

(iii) Problem A_3: The nonlinear separable constraint case of Problem A_1[A1,S5]:

$$\text{maximize} \quad \prod_{i=1}^{n} (1 - q_i^{x_i+1})$$

$$\text{subject to} \quad \sum_{i=1}^{n} g_{ji}(x_i) \leq b_j \quad (j = 1,\ldots,m),$$

$$x_i \geq 0 \text{ integer},$$

where $g_{ji}(x_i)$ are monotone non-decreasing.

(iv) Problem A_4: Choose the optimal candidate x_i for each subsystem i so that the reliability of a series system is maximized without violating nonlinear separable constraints [M17,N3]:

$$\text{maximize} \quad \prod_{i=1}^{n} R_i(x_i)$$

$$\text{subject to} \quad \sum_{i=1}^{n} g_{ji}(x_i) \leq b_j \quad (j = 1,\ldots,m),$$

$$x_i \geq 1 \text{ integer},$$

where $g_{ji}(x_i)$ are momotone nondecreasing.

(v) Problem A_5: The nonseries system case of Problem A_3 [A2]:

$$\text{maximize} \quad R(X)$$

$$\text{subject to} \quad \sum_{i=1}^{n} g_{ji}(x_i) \leq b_j \quad (j = 1,\ldots,m),$$

$$X = (x_1,\ldots,x_n),$$

$$x_i \geq 1 \quad \text{integer},$$

where $R(X)$ and $g_{ji}(x_i)$ are monotone nondecreasing.

1.5.2 Vector optimization Problem

Another way for reducing the conflict mentioned in Section 1.5.1 is the multiobjective programming. All properties are assumed to be better when they are smaller (e.g., use unreliability rather than reliability). A general formulation of vector optimization problem is as follows:

Problem B: Choose the optimal candidate for each subsystem from the prepared candidates so that the objective functions are minimized without violating nonlinear constraints:

$$\text{minimize} \quad f(X) = [f_1(X),\ldots,f_m(X)]$$

$$\text{subject to} \quad f_j(X) \leq b_j \quad (j \in M^{sub}),$$

$$g_j(X) \leq d_j \quad (j = 1,\ldots,u),$$

$$X = (x_1,\ldots,x_n),$$

$$\underline{c}_i \leq x_i \leq \bar{c}_i, \; x_i \text{ integer},$$

where M^{sub} is a subset of $\{i,\ldots,m\}$.

Generally speaking, this problem can not be solved without the information about decision maker's preference. The set of Pareto-optimal solutions is obtainable mathematically. However we can not select a best one from the Pareto-optimal solution set without the information from the decision maker.

We will treat a special case of problem B in the last chapter.

Chapter 2

RELIABILITY CALCULATION AND OPTIMIZATION OF SERIES-PARALLEL REDUNDANT STRUCTURES FOR 2-FAILURE-STATE UNITS

2.1 INTRODUCTION

There are several kinds of redundancy for units with two failure states, e.g., hammock and bridge [M21], k-out-of-m [M5], and series-parallel (s-p). The redundancies treated in [B4,G5-G7,T7] are special cases of s-p. Tillman [T7] has formulated problems with several modes of failure. The formulation in his Type 1 failures involves a careless assumption. Thus, e.g., when h=2, s=4, $q_1=q_2=q_3=q_4=0.3$, m=4, both equations (8) and (18) in [T7] give 1.669 as the probability of failure. His treatment for Type 2 failure is the same as the conventional one for a unit with two failure modes. A formulation in [G5-G7] has the same error as [T7].

This chapter treats s-p redundancy; each unit and the system are a device with two failure states. For practical usage of the device, e.g., valve, relay, and sensor in protection systems of nuclear power plant, chemical plant, electric power system, etc., it is always desired to decrease not only the unreliability of a fail-safe device, but also both of the two failure probability of the device so that one

failure (unsafe failure) happens much more rarely than another failure (safe failure), e.g., see Section 2.7. In such cases, s-p redundancy can be used; k-out-of-m redundancy can not be used for mechanical valves. Which redundancy of s-p and k-out-of-m is better depends on the kind of device, reliability, and cost of k-out-of-m devices (voter), etc. The difficulty in treating the s-p redundancy is that the number of possible structures increases exponentially with increasing number of redundant units. The total number of such s-p structures has been calculated by Riordan-Shannon[R1].

In this chapter, a method is presented for generating systematically all possible s-p structures for 1,...,M units. The method is modified to calculate the failure probabilities and reliabilities of all possible s-p redundant structures for 1,...,M 2-failure-state units. Furthermore we apply the present methods to a problem of finding an optimal s-p structure with minimum units subject to four constraints for reliability, two failure probabilities and fail-safe.

2.2 SERIES AND PARALLEL REDUNDANCIES OF 2-FAILURE-STATE UNIT

There is no restriction on the number or kind of inputs to a unit; the inputs are irrelevant to the discussion, except as they are implicit in the definition of success. There is only one output and there are exactly 2 output states (called '0' and '1').

The unit has exactly 3 states:

(i) success: the output follows the inputs according to the nature of the unit,

(ii) or (iii) failure 'i': the output is stuck-at-i, for i = 0, 1.

An example is a gate-valve (Fig. 2.1). Output '1' is open; output

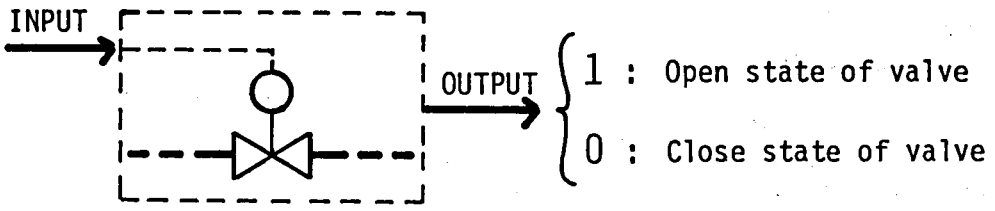


Fig. 2.1 Valve.

'0' is closed. There are 2 input signals, one to close the valve, one to open the valve. Its states are:

Success: output is 1 or 0 depending on which input is activated.
Failure '0': the valve is stuck closed, regardless of the input.
Failure '1': the valve is stuck open, regardless of the input.

A pair of units can be combined in either of 2 ways. A combined pair of units is a single device and has several inputs, but only one output with 2 states. The 2 combinations are called series (Fig. 2.2) and parallel (Fig. 2.3) in defference to usual terminology, but the terms can be very misleading if used in any way but as defined by the truth table.

Truth Table for Unit Pair

	series				parallel			
output of Unit 1	1	1	0	0	1	1	0	0
output of Unit 2	1	0	1	0	1	0	1	0
output of comb.	1	0	0	0	1	1	1	0

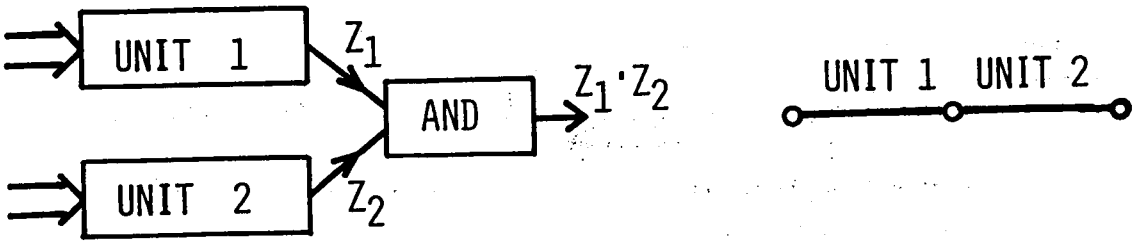


Fig. 2.2 Series redundancy.

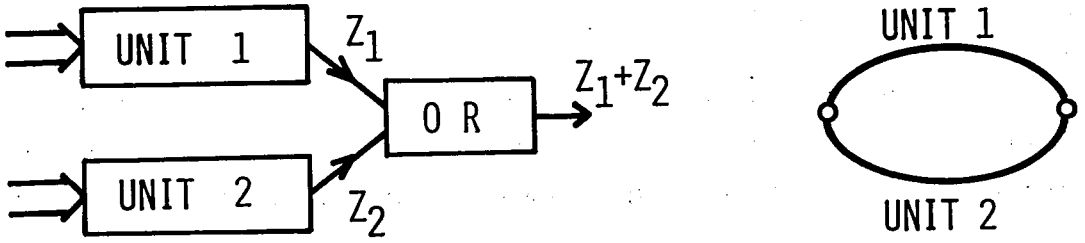


Fig. 2.3 Parallel redundancy.

An example is the gate valve mentioned above; series and parallel refer to the physical connection, not necessarily the logic connection. Series and parallel are duals. If the output definitions are switched, then series turns into parallel and vice-virsa.

2.3. NOMENCLATURE & NOTATION

Nomenclature

structure: A collection of units connected together.

series-parallel (s-p): A structure is called 's-p' if it is either a series or a parallel connection of two s-p structures. A 1-unit structure is 's-p'.

essentially series (e.s.) or seesntially parallel (e.p.): A s-p structure is called 'e.s.' or 'e.p.' if it is the series or parallel connection of two s-p structures, respectively. A 1-unit structure can be called either 'e.s.' or 'e.p.'.

dual: The 'dual' is generated when the words series and parallel are interchanged in describing the structure.

Note: All possible e.s. structures are in one-to-one correspondence with all possible e.p. structures, because each e.s. structure has an e.p. dual. Hence it is sufficient to generate all possible e.s. structures.

structure number; A number corresponding to an e.s. structure and its e.p. dual, see the definition of i in Notation below.

NOTATION

M	maximum number of units.
m	number of units; $1 \leq m \leq M$.
N	number of all possible e.s. (or e.p.) structures for 1,...,M units.
n_m	number of all possible e.s. (or e.p.) structures for m units.
N_m	number of all possible e.s. (or e.p.) structures for 1,...,m units; $N_m = N_{m-1} + n_m = \sum_{t=1}^{m} n_t$, $N_M = N$.
$\sim$	implies dual.
$\cdot$	implies series connection.
i	structure number; $i = 1,\ldots,N$.

S_i e.s. structure.

$\tilde{S}_i$ e.p. dual of S_i.

S_i^* means either S_i or $\tilde{S}_i$; viz., $S_i^* \in \{S_i, \tilde{S}_i\}$.

$u(S_i^*)$ number of units forming a s-p structures S_i^*; $u(S_i) = u(\tilde{S}_i)$.

$S_{i_1}^* \cdot S_{i_2}^* \cdot \ldots \cdot S_{i_p}^*$ structure having p structures $S_{i_1}^*, S_{i_2}^*, \ldots, S_{i_p}^*$ in series.

$\min\text{-sn}(S_i^*)$ minimum of structure numbers of e.p. structures forming a structure S_i^* in series, see (2,3).

$\lambda_{\mu,\kappa}$ minimum of structure numbers of e.s. structures S_i such that $u(S_i) = \mu$ and $\min\text{-sn}(S_i) = \kappa$, see (2,13).

$[y]^+$ maximum integer $\leq y$.

q^0, q^1 failure probabilities of units for the cases output is always '0', '1', respectively.

$Q_i^0 (=1-p_i^0)$, $\tilde{Q}_i^0$ failure probabilities of S_i, $\tilde{S}_i$, respectively, for the case output is always '0'.

Q_i^1, $\tilde{Q}_i^1 (=1-\tilde{P}_i^1)$ failure probabilities of S_i, $\tilde{S}_1$, respectively, for the case output is always '1'.

R_i, $\tilde{R}_i$ reliabilities of S_i, $\tilde{S}_i$, respectively.

2.4. PROCEDURE FOR GENERATION OF STRUCTURES

Figure 2.4 shows all possible s-p structures for 1,2,3,4 units. In this section, we present a procedure for systematically generating all possible s-p structures for 5,...,M units. (Because of the duality between e.s. and e.p. structures, it is sufficient to generate all possible e.s. structures)

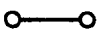
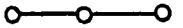
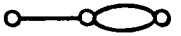
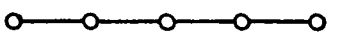
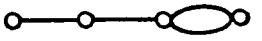
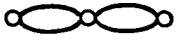
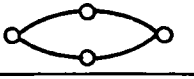
NUMBER OF UNITS m	HALF OF THE STRUCTURES (e.s. structures) S_i	DUALS (e.p. structures) $\tilde{S}_i$	NUMBER OF STRUCTURES
1	S_1 	$\tilde{S}_1$ 	1
2	S_2 	$\tilde{S}_2$ 	2
3	S_3  S_4 	$\tilde{S}_3$  $\tilde{S}_4$ 	4
4	S_5  S_6  S_7  S_8  S_9 	$\tilde{S}_5$  $\tilde{S}_6$  $\tilde{S}_7$  $\tilde{S}_8$  $\tilde{S}_9$ 	10

Fig. 2.4 Series-parallel structures for 1,2,3,4 units.

2.4.1 Preparation for Describing Procedure

In Fig.2.4, each of s-p structures is designated by S_i or S_i ($i = 1, \ldots, 9$) according to a rule mentioned in Section 2.4.1.2. Using this figure, we explain two standard forms for expressing e.s. structures and a rule for designating all possible e.s. structures by S_i ($i = 1, \ldots, N$), and then explain a rudimental method for generating e.s. structures and useful properties for systematic generation of e.s. structure.

2.4.1.1 Standard Forms for Expressing E.S. Structures

Form the definition of 'e.s.' in Section 4.3, it is confirmed that any e.s. structure for $2, \ldots, M$ units can be expressed by series-connections among 2 or more e.p. structures. That is, assume

Condition 1: All possible e.s. structures for $1, \ldots, m-1$ units have been generated and designated by

$$S_1 , S_2 , \ldots , S_{N_{m-1}}$$

according to a rule that will be mentioned in Section 2.4.1.2.

Then any e.s. structure S_i ($u(S_i) = m$) can be expressed as:

Form A: $S_i = \tilde{S}_{i_1} \cdot \tilde{S}_{i_2} \cdot \ldots \cdot \tilde{S}_{i_p}$ ($1 \leq i_1 \leq i_2 \leq \ldots \leq i_p \leq N_{m-1}$, $p \geq 2$),

where

$$\sum_{t=1}^{p} u(\tilde{S}_{i_t}) = m. \qquad (2.1)$$

(For example, in Fig. 2.4, $S_6 = \tilde{S}_1 \cdot \tilde{S}_1 \cdot \tilde{S}_2$ and $S_9 = \tilde{S}_2 \cdot \tilde{S}_2$)

As for Form A, note that

$$\tilde{S}_{i'} \cdot \tilde{S}_{i''} = \tilde{S}_{i''} \cdot \tilde{S}_{i'}, \qquad (2.2)$$

for any i', i'' ($1 \leq i' \leq N_{m-1}$, $1 \leq i'' \leq N_{m-1}$).

We, here, introduce the definition such as

$$\text{min-sn}(S_i^*) = \begin{cases} \min\limits_{t=1,\ldots,p} \{i_t\} = i_1, & \text{when } S_i^* \text{ is e.s. and expressed as Form A.} \\ i, & \text{when } S_i^* \text{ is e.p.} \end{cases}$$

(for example, in Fig. 2.4, $\text{min-sn}(S_6) = \text{min-sn}(\tilde{S}_1 \cdot \tilde{S}_1 \cdot \tilde{S}_2) = \min\{1,1,2\} = 1$ and $\text{min-sn}(\tilde{S}_6) = 6$)

Let us rewrite Form A into another form which is convenient to generate e.s. structures. Under Condition 1, any e.s. structure S_i ($u(S_i) = m$) can be expressed as:

Form B: $S_i = \tilde{S}_k \cdot S_\ell^*$ $(1 \leq k \leq \ell \leq N_{m-1})$,

where

$$k = \text{min-sn}(S_i) \tag{2.4}$$

$$S_\ell^* = \begin{cases} S_\ell, & \text{when } p > 2 \text{ in Form A.} \\ \tilde{S}_\ell, & \text{when } p = 2 \text{ in Form A.} \end{cases} \tag{2.5}$$

$$k \leq \text{min-sn}(S_\ell^*), \tag{2.6}$$

$$u(S_k) + u(S_\ell^*) = m. \tag{2.7}$$

(For example, in Fig. 2.4, $S_6 = \tilde{S}_1 \cdot S_4$ and $S_9 = \tilde{S}_2 \cdot \tilde{S}_2$)

(The proof that Form A can be rewritten into Form B is obvious)

2.4.1.2 A Rule for Designating E.S. Structures by S_i

Suppose that all possible e.s. structures for $1,\ldots,m'$ $(3 \leq m' \leq M)$ units have been generated. Let us now designate N_m, e.s. structures by $S_1, S_2, \ldots, S_{N_m}$. There are $1 \times 2 \times \ldots \times N_m$, different manners for designating. A rule is presented here, which is convenient to generate all possible e.s. structures for $m + 1$ units.

Our rule is inductive:

i. (For $m = 1, 2$) Designate the e.s. structure having 1 unit by S_1 and the e.s. structure having 2 units by S_2, as shown in Fig. 2.4.

ii. (For m = 3,..., m') Assume that all e.s. structures for 1,..., m-1 units have been designated by $S_1,\ldots, S_{N_{m-1}}$. Then a rule for designating n_m e.s. structures by $S_{N_{m-1}+1}, S_{N_{m-1}+2},\ldots, S_{N_m}$ is as follows:

Rule A: (For Form A) For two e.s. structures such as

$$S_{i'} = \tilde{S}_{i'_1}\cdot\ldots\cdot\tilde{S}_{i'_p} \quad (1 \le i'_1 \le \ldots \le i'_p \le N_{m-1},\ q > 2)$$

$$S_{i''} = \tilde{S}_{i''_1}\cdot\ldots\cdot\tilde{S}_{i''_q} \quad (1 \le i''_1 \le \ldots \le i''_q \le N_{m-1},\ q > 2),$$

where $u(S_{i'}) = u(S_{i''}) = m$, if it is true that $i' < i''$, then $S_{i'}$ and $S_{i''}$ must satisfy

$$i' = i'',$$

$$\vdots$$

$$i'_{t-1} = i''_{t-1}$$

$$i'_t < i''_t \quad (1 \le t \le \min\{p,q\}).$$

(For example, in Fig. 2.4, for m = 3, $S_3 = \tilde{S}_1\cdot\tilde{S}_1\cdot\tilde{S}_1$, $S_4 = \tilde{S}_1\cdot\tilde{S}_2$, and then for m = 4, $S_5 = \tilde{S}_1\cdot\tilde{S}_1\cdot\tilde{S}_1\cdot\tilde{S}_1$, $S_6 = \tilde{S}_1\cdot\tilde{S}_1\cdot\tilde{S}_2$, $S_7 = \tilde{S}_1\cdot\tilde{S}_3$, $S_8 = \tilde{S}_1\cdot\tilde{S}_4$, $S_9 = \tilde{S}_2\cdot\tilde{S}_2$)

Using Rule A, which is lexicographical, we can uniquely write $S_{N_{m-1}+1},\ldots, S_{N_m}$ for n_m e.s. structures having m units.

Rule A can be rewritten as:

Rule B: (For Form B) For two e.s. structures such as

$$S_{i'} = \tilde{S}_{k'} \cdot S^*_{\ell'},$$

$$S_{i''} = \tilde{S}_{k''} \cdot S^*_{\ell''},$$

where $u(S_{i'}) = u(S_{i''}) = m$, $k' = \text{min-sn}(S_{i'})$ and $k'' = \text{min-sn}(S_{i''})$, if it is true that $i' < i''$, then $S_{i'}$ and $S_{i''}$ must satisfy one of the following Cases 1, 2, 3, 4.

Case 1: $k' < k''$.
Case 2: $k' = k''$, $S_{i'} = \tilde{S}_{k'} \cdot S_{\ell'}$, $S_{i''} = \tilde{S}_{k''} \cdot S_{\ell''}$ and $\ell' < \ell''$.
Case 3: $k' = k''$, $S_{i'} = \tilde{S}_{k'} \cdot S_{\ell'}$, $S_{i''} = \tilde{S}_{k''} \cdot \tilde{S}_{\ell''}$.
Case 4: $k = k''$, $S_{i'} = \tilde{S}_{k'} \cdot \tilde{S}_{\ell'}$, $S_{i''} = \tilde{S}_{k''} \cdot \tilde{S}_{\ell''}$ and $\ell' < \ell''$.

(The proof that Rule B is equivalent to Rule A is given in Appendix A)

As for Rule B, see Table 2.1. This table shows all possible e.s. structures for $1,\ldots,8$ units in the form $\tilde{S}_k \cdot S_\ell$ or $\tilde{S}_k \cdot \tilde{S}_\ell$. We read Table 2.1 as follows; e.g., see the row $m = 8$, $j = 3$, $k = 4$, the row shows that, for all S_i ($i = 379, 380, \ldots, 390$)

$$m = u(S_i) = 8,$$
$$k = \text{min-sn}(S_i) = 4,$$
$$j = u(\tilde{S}_k) = 3,$$

and

$$S_{379} = \tilde{S}_4 \cdot \tilde{S}_{10},$$
$$S_{380} = \tilde{S}_4 \cdot \tilde{S}_{11},$$
$$\vdots$$
$$S_{390} = \tilde{S}_4 \cdot \tilde{S}_{21}.$$

Table 2.1

Essentially Series Structures for 1,...,8 Units

m	j	k	$\tilde{S}_k \cdot s_\ell$ or $\tilde{S}_k \cdot \tilde{S}_\ell$	s_i	$\lambda_{m,k}$	n_m
1	1	1		s_1	1	1
2	1	1	$\tilde{S}_1 \cdot \tilde{S}_1$	s_2	2	1
3	1	1	$\tilde{s}_1 \cdot \begin{cases} s_2 \\ \tilde{s}_2 \end{cases}$	s_3 s_4	3	2
4	1	1	$\tilde{s}_1 \cdot \begin{cases} s_\ell, & \ell=3,4 \\ \tilde{s}_\ell, & \ell=3,4 \end{cases}$	s_5, s_6 s_7, s_8	5	5
	2	2	$\tilde{S}_2 \cdot \tilde{S}_2$	s_9	9	
5	1	1	$\tilde{s}_1 \cdot \begin{cases} s_\ell, & \ell=5,\ldots,9 \\ \tilde{s}_\ell, & \ell=5,\ldots,9 \end{cases}$	$s_{10},\ldots,s_{14}$ $s_{15},\ldots,s_{19}$	10	12
	2	2	$\tilde{S}_2 \cdot \tilde{s}_\ell$, $\ell=3,4$	s_{20}, s_{21}	20	
6	1	1	$\tilde{s}_1 \cdot \begin{cases} s_\ell, & \ell=10,\ldots,21 \\ \tilde{s}_\ell, & \ell=10,\ldots,21 \end{cases}$	$s_{22},\ldots,s_{33}$ $s_{34},\ldots,s_{45}$	22	33
	2	2	$\tilde{s}_2 \cdot \begin{cases} s_9 \\ \tilde{s}_\ell, & \ell=5,\ldots,9 \end{cases}$	s_{46} $s_{47},\ldots,s_{51}$	46	
	3	3	$\tilde{S}_3 \cdot \tilde{s}_\ell$, $\ell=3,4$	s_{52}, s_{53}	52	
		4	$\tilde{S}_4 \cdot \tilde{S}_4$	s_{54}	54	
7	1	1	$\tilde{s}_1 \cdot \begin{cases} s_\ell, & \ell=22,\ldots,54 \\ \tilde{s}_\ell, & \ell=22,\ldots,54 \end{cases}$	$s_{55},\ldots,s_{87}$ $s_{88},\ldots,s_{120}$	55	90
	2	2	$\tilde{s}_2 \cdot \begin{cases} s_\ell, & \ell=20,21 \\ \tilde{s}_\ell, & \ell=10,\ldots,21 \end{cases}$	s_{121}, s_{122} $s_{123},\ldots,s_{134}$	121	
	3	3	$\tilde{S}_3 \cdot \tilde{s}_\ell$, $\ell=5,\ldots,9$	$s_{135},\ldots,s_{139}$	135	
		4	$\tilde{S}_4 \cdot \tilde{s}_\ell$, $\ell=5,\ldots,9$	$s_{140},\ldots,s_{144}$	140	
8	1	1	$\tilde{s}_1 \cdot \begin{cases} s_\ell, & \ell=55,\ldots,144 \\ \tilde{s}_\ell, & \ell=55,\ldots,144 \end{cases}$	$s_{145},\ldots,s_{234}$ $s_{235},\ldots,s_{324}$	145	261
	2	2	$\tilde{s}_2 \cdot \begin{cases} s_\ell, & \ell=46,\ldots,54 \\ \tilde{s}_\ell, & \ell=22,\ldots,54 \end{cases}$	$s_{325},\ldots,s_{333}$ $s_{334},\ldots,s_{366}$	325	
	3	3	$\tilde{S}_3 \cdot \tilde{s}_\ell$, $\ell=10,\ldots,21$	$s_{367},\ldots,s_{378}$	367	
		4	$\tilde{S}_4 \cdot \tilde{s}_\ell$, $\ell=10,\ldots,21$	$s_{379},\ldots,s_{390}$	379	
	4	5	$\tilde{S}_5 \cdot \tilde{s}_\ell$, $\ell=5,\ldots,9$	$s_{391},\ldots,s_{395}$	391	
		6	$\tilde{S}_6 \cdot \tilde{s}_\ell$, $\ell=6,\ldots,9$	$s_{396},\ldots,s_{399}$	396	
		7	$\tilde{S}_7 \cdot \tilde{s}_\ell$, $\ell=7,8,9$	$s_{400}, s_{401}, s_{402}$	400	
		8	$\tilde{S}_8 \cdot \tilde{s}_\ell$, $\ell=8,9$	s_{403}, s_{404}	403	
		9	$\tilde{S}_9 \cdot \tilde{S}_9$	s_{405}	405	

$k = \min\text{-}sn(S_i)$, $j = u(\tilde{S}_k)$

Any two e.s. structures satisfy Rule B for each m = 3,...,8; e.g., S_{325} and S_{367} in Table 2.1 satisfy Case 1 in Rule B; S_{325} and S_{326} satisfy Case 2; S_{325} and S_{334} satisfy Case 3; S_{334} and S_{335} satisfy Case 4.

2.4.1.3 A Rudimental Method for Generating E.S. Structures

In this section, we present a rudimental method for generating e.s. structures for m units.

Under Condition 1, we obtain

Property 1: For any e.s. structure S_i ($u(S_i) = m$), it is true that

$$u(\tilde{S}_k) \leq [m/2]^+,$$

where $k = \min\text{-}sn(S_i)$.

(The proof of Property 1 is given in Appendix B-1)

By using Property 1 and Form B, under Condition 1 we can generate all possible e.s. structures for m units.

The method is as follows: generate the following two groups of e.s. structures, for each k such that

$$u(\tilde{S}_k) \leq [m/2]^+. \tag{2.8}$$

Group 1: structures $\tilde{S}_k \cdot S_\ell$ for all ℓ such that

$$u(S_\ell) = m - u(\tilde{S}_k), \tag{2.9}$$

$$\text{min-sn}(S_\ell) \geq k. \tag{2.10}$$

Group 2: structures $\tilde{S}_k \cdot \tilde{S}_\ell$ for all ℓ such that

$$u(\tilde{S}_\ell) = m - u(\tilde{S}_k), \tag{2.11}$$

$$\text{min-sn}(\tilde{S}_\ell) \geq k. \tag{2.12}$$

(For example, when m = 5 (i.e., we start with Fig. 2.4), we obtain $u(S_k) \leq 2$ from (2.8); i.e., k = 1, 2 (see Fig. 2.4). When k = 1, we obtain $u(S_\ell) = u(\tilde{S}_\ell) = 4$ from (2.9) and (2.11) because $u(\tilde{S}_k) = 1$, and then Fig. 2.4 show that $\{\ell |\ u(S_\ell) = 4\} = \{\ell |\ u(\tilde{S}_\ell) = 4\} = \{5,6,7,8,9\}$, and all $\ell \in \{5,6,7,8,9\}$ satisfy (2.10) and (2.12); i.e., the structures obtained are $\tilde{S}_1 \cdot S_\ell$ (ℓ = 5,...,9) and $\tilde{S}_1 \cdot \tilde{S}_\ell$ (ℓ = 5,...,9). When k = 2, we obtain $u(S_\ell) = u(\tilde{S}_\ell) = 3$ from (2.9) and (2.11) because $u(\tilde{S}_k) = 2$, and then Fig. 2.4 shows that $\{\ell |\ u(S_\ell) = 3\} = \{\ell |\ u(\tilde{S}_\ell) = 3\} = \{3,4\}$, and all $\ell \in \{3,4\}$ satisfy (2.12) but do not satisfy (2.10), i.e., the structures obtained are $\tilde{S}_2 \cdot \tilde{S}_3$, $\tilde{S}_2 \cdot \tilde{S}_4$. We have obtained all possible e.s. structures for 5 units, see the row m = 5 in Table 2.1.)

This method can exactly generate all possible e.s. structures for m units, since a) it is ovbious that all of the structures generated by this method have m units and differ from each other, b) from a fact that any e.s. structure is expressed in Form B, we easily see that the set of the generated structures includes all possible e.s. structures for m units (note that (2.9) and (2.11) are obtained from (2.7), and (2.10) and (2.12) are obtained from (2.6)).

In order to generate furthermore e.s. structures for m+1 units, we must designate each of all structures generated for m units by S_i ($i = N_{m-1}+1, \ldots, N_m$). This designating can be done by using Rule B in Section 2.4.1.2 (In Procedure A, as seen in Section 2.4.2, e.s. structures are generated in the order of the structure numbers $i = N_{m-1}+1, \ldots, N_m$).

2.4.1.4 Useful Properties for Systematic Generation of E.S. Structures

To make the method presented in Section 2.4.1.3 a systematic one, under Condition 1 we introduce $\lambda_{\mu,\kappa}$ ($\mu \leq N_{m-1}$) defined as

$$\lambda_{\mu,\kappa} = \min\{i \mid u(S_i) = \mu \text{ and } \min\text{-}sn(S_i) = \kappa\}. \tag{2.13}$$

(For example, see Table 2.1, $\lambda_{5,1} = \min\{10,\ldots,19\} = 10$ and $\lambda_{5,2} = \min\{20,21\} = 20$)

Definition (2.13) leads to

$$\min\{i \mid u(S_i) = \mu\} = N_{\mu-1} + 1 = \lambda_{\mu,1}, \tag{2.14}$$

$$\max\{i \mid u(S_i) \leq \mu\} = N_{\mu} = \lambda_{\mu+1,1} - 1, \tag{2.15}$$

$$\min\{i \mid u(S_i) = \mu \text{ and } \min\text{-}sn(S_i) \geq \kappa\} = \lambda_{\mu,\kappa}. \tag{2.16}$$

By using the definition (2.13), we have the following properties, which are convenient to generate all possible e.s. structures for m units.

a) Common properties to Groups 1 and 2 in Section 2.4.1.3. We can rewrite Property 1 by using (2.15):

Property 1': For any e.s. structures S_i ($u(S_i) = m$), it is true that

$$1 \leq k \leq \lambda_{[m/2]^+ +1,1} - 1,$$

where $k = \min\text{-}sn(S_i)$.

Furthermore we have:

Property 2: Both sets of ℓ satisfying (2.9) and ℓ satisfying (2.11) are

$$\{\lambda_{m-j,1}, \lambda_{m-j,1}{}^{+1}, \ldots, \lambda_{m-j+1,1}{}^{-1}\}$$

for each $k = 1,\ldots,\lambda_{[m/2]^+ +1,1}{}^{-1}$, where $j = u(\tilde{S}_k)$.

(The proof of Property 2 is obvious from (2.14) and (2.15))

b) Properties for Group 1.

Property 3: ℓ satisfying both (2.9) and (2.10) exists i.f.f.

$$1 \leq k \leq \lambda_{[m/3]^+ +1,1}{}^{-1} \quad (= N_{[m/3]^+}).$$

(The proof of Property 3 is given in Appendix B-2)

Property 4: For each $k = 1,\ldots, \lambda_{[m/3]^+ +1,1}{}^{-1}$, it is true that

$$\text{min-sn}(S_\ell) \begin{cases} < k & \text{for all } \ell = \lambda_{m-j,1},\ldots, \lambda_{m-j,k}{}^{-1} \\ \geq k & \text{for all } \ell = \lambda_{m-j,k},\ldots, \lambda_{m-j+1,1}{}^{-1}, \end{cases}$$

where $j = u(\tilde{S}_k)$.

(The proof of Property 4 is obvious from (2.16))

c) Properties for Group 2.

Property 5: For each $k = 1,\ldots, \lambda_{[(m+1)/2]^+,1}{}^{-1}$ (i.e., $k = 1,\ldots,$

$\lambda_{[m/2]^+,1}{}^{-1}$ when m is odd, and k = 1,..., $\lambda_{[m/2]^+,1}{}^{-1}$

when m is even), it is true that

$$\text{min-sn}(S_\ell) = \ell \geq k \quad \text{for all } \ell = \lambda_{m-j,1},\ldots, \lambda_{m-j+1,1},$$

where $j = u(\tilde{S}_k)$.

(The proof of Property 5 is given in Appendix B-3)

Property 6: When m is even, for each $k = \lambda_{[m/2]^+,1},\ldots, \lambda_{[m/2]^+,1}{}^{-1}$, it is true that

$$\text{min-sn}(S_\ell) \begin{cases} < k & \text{for all } \ell = \lambda_{[m/2]^+,1},\ldots, \quad k-1 \\ \geq k & \text{for all } \ell = k,\ldots, \lambda_{[m/2]^+,1}{}^{-1}. \end{cases}$$

(The proof of property 6 is obvious)

d) Properties 1', 2,..., 6 can be summarized as follows:

i. (As for Group 1) Properties 2, 3 and 4 show that it is sufficient to generate

$$\tilde{S}_k \cdot S_\ell \quad (\ell = \lambda_{m-j,k},\ldots, \lambda_{m-j+1,1}{}^{-1})$$

for each $k = 1,\ldots, \lambda_{[m/3]^+,1}{}^{-1}$, where $j = u(\tilde{S}_k)$ $(1 \leq j \leq [m/3]^+)$.

ii. (As for Group 2) Properties 1', 2 and 5 show to generate

$$\tilde{S}_k \cdot \tilde{S}_\ell \quad (\ell = \lambda_{m-j,1},\ldots, \lambda_{m-j+1,1}{}^{-1})$$

for all $k = 1,\ldots, \lambda_{[(m+1)/2]^+,1}{}^{-1}$, where $j = u(S_k)$ $(1 \leq j \leq [(m+1)/2]^+)$.

If m is even, Property 1' and 6 show furthermore to generate

$$\tilde{S}_k \cdot \tilde{S}_\ell \quad (\ell = k, \ldots, \lambda_{[m/2]^+ + 1,1} - 1)$$

for each $k = \lambda_{[m/2]^+,1}, \ldots, \lambda_{[m/2]^+ + 1,1} - 1$, where $j = u(\tilde{S}_k) = [m/2]^+$.

By using i and ii, we can systematically generate all possible e.s. structures for m units. A systematic procedure is given in Section 2.3.2.

2.4.2 Procedure for Generation of Structures

We have generated and designated all possible e.s. structures for 1,2,3,4 units by S_i (i = 1,...,9), as seen in Fig. 2.4. In this section, a systematic procedure is presented for generating and simultaneously designating each of all possible e.s. structures for 5,...,M units by S_i (i = 10,...,N). The generating technique in Procedure A is based on d) in Section 2.4.1.2, and the generated structures are designated by S_i according to the rule mentioned in Section 2.4.1.2.

Procedure A begins with m = 5, but can begin with m = 2.

Procedure A

Step 1: Set m = 5, $\lambda_{1,1} = 1$, $\lambda_{2,1} = 2$, $\lambda_{3,1} = 3$, $\lambda_{4,1} = 5$, $\lambda_{4,2} = 9$, $\lambda_{5,1} = 10$. (When we begin with m = 2, change this step as follows: set m = 2, $\lambda_{1,1} = 1$, $\lambda_{2,1} = 2$, and go to Step 5)

Step 2: DO iv j = 1, $[m/3]^+$, 1

(This definition is the same as DO-CONTINUE statement in FORTRAN)

DO iv k = $\lambda_{j,1}$, $\lambda_{j+1,1} - 1$, 1

i. Generate the e.s. structures

$$\tilde{S}_k \cdot S_\ell \quad (\ell = \lambda_{m-j,k}, \ldots, \lambda_{m-j+1,1}-1),$$

and designate the structures by S_i so that

$$i = \lambda_{m,k} + (\ell - \lambda_{m-j,k}).$$

ii. Generate the e.s. structures

$$\tilde{S}_k \cdot \tilde{S}_\ell \quad (\ell = \lambda_{m-j,1}, \ldots, \lambda_{m-j+1,1}-1),$$

and designate the structures by S_i so that

$$i = \lambda_{m,k} + (\lambda_{m-j+1,1} - \lambda_{m-j,k}) + (\ell - \lambda_{m-j,k}).$$

iii. Calculate

$$\lambda_{m,k+1} = \lambda_{m,k} + (\lambda_{m-j+1,1} - \lambda_{m-j,k}) + (\lambda_{m-j+1,1} - \lambda_{m-j,1}).$$

iv. CONTINUE

Step 3: If $[m/3]^+ + 1 > [(m-1)/2]^+$, then go to Step 5. Otherwise go to Step 4.

Step 4: DO iii $j = [m/3]^+ + 1, [(m-1)/2]^+, 1$

DO iii $k = \lambda_{j,1}, \lambda_{j+1,1}-1, 1$

i. Generate the e.s. structures

$$\tilde{S}_k \cdot \tilde{S}_\ell \quad (\ell = \lambda_{m-j,1}, \ldots, \lambda_{m-j+1,1}-1),$$

and designate the structures by S_i so that

$$i = \lambda_{m,k} + (\ell - \lambda_{m-j,1}).$$

ii. Calculate

$$\lambda_{m,k+1} = \lambda_{m,k} + (\lambda_{m-j+1,1} - \lambda_{m-j,1}).$$

iii. CONTINUE

Step 5: If m is odd, then go to Step 6. Otherwise, set $j = [m/2]^+$ and do the following:

DO iii $k = \lambda_{j,1}, \lambda_{j+1,1}-1, 1$

i. Generate the e.s. structures

$$\tilde{S}_k \cdot \tilde{S}_\ell \quad (\ell = k, \ldots, \lambda_{m-j+1,1}-1),$$

and designate the structures by S_i so that

$$i = \lambda_{m,k} + (\ell - k).$$

ii. Calculate

$$\lambda_{m,k+1} = \lambda_{m,k} + (\lambda_{m-j+1,1} - k).$$

iii. CONTINUE

Step 6: If $m \neq M$, then reset $m \leftarrow m + 1$ and set $\lambda_{m+1,1} = \lambda_{m,k+1}$ and return to Step 2. Otherwise, stop; we now have all possible e.s. structures for 5,...,M units, i.e., all possible s-p

structures for 1,...,M units.

An illustration (m = 8) of Procedure A is shown in Appendix C.

Numbers of all possible e.s. structures for 1, ..., 30 units are shown in Table 2.2, which are obtained from [R1, Table I]. We have made sure that numbers of all possible e.s. structures generated for 1,...,15 units by Procedure A, are exactly the same as ones in Table 2.2.

Table 2.2 shows that the number of all possible e.s. structures increases exponentially with increasing number of units. Hence Procedure A will soon become impractical with incresing number of units. However Procedures B and C presented in the succeeding sections are applicable to the cases of relatively large numbers of units.

2.5. PROCEDURE OF DETERMINING A PARTICULAR STRUCTURE

We may be faced with a problem of determining a particular structure $S_{i'}$ or $\tilde{S}_{i'}$, when we have a structure number i' but do not know what s-p structures correspond to i'. In fact, we will be faced in Section 2.7 with the problem of determining the structures S_{134}. In such case, if we use Procedure A to determine the structure $S_{i'}$ or $\tilde{S}_{i'}$, we must generate all the structures S_{10}, S_{11},..., $S_{i'}$ (the structures S_1,..., S_9 have been generated in Fig. 2.4). This is quite cumbersome, and becomes impractical for structures having more than about 10 units (see Table 2.2).

In this section, a more efficient procedure is presented for determining a particular structure. By using Table 2.1, we can determine a particular structure $S_{i'}$ or $\tilde{S}_{i'}$ ($10 \leq i' \leq 405$). For

Table 2.2

Number of All Possible E.S. Structures
(obtained from [R1, Table I])

m	n_m	N_m (= $\lambda_{m+1,1}$ -1)
1	1	1
2	1	2
3	2	4
4	5	9
5	12	21
6	33	54
7	90	144
8	261	405
9	766	1,171
10	2,312	3,483
11	7,068	10,551
12	21,965	32,516
13	68,954	101,470
14	218,751	320,221
15	699,534	1,019,755
16	2,253,676	3,273,431
17	7,305,788	10,579,219
18	23,816,743	34,395,962
19	78,023,602	112,419,564
20	256,738,751	369,158,315
21	848,152,860	1,217,311,175
22	2,811,996,972	4,029,308,147
23	9,353,366,564	13,382,674,711
24	31,204,088,381	44,586,763,092
25	104,384,620,070	148,971,383,162
26	350,064,856,815	499,036,239,977
27	1,176,693,361,956	1,675,729,601,933
28	3,963,752,002,320	5,639,481,604,253
29	13,378,623,786,680	19,018,105,390,933
30	45,239,588,651,121	64,257,694,042,054

example, the structure S_{134} is determined as follows:

1. Find $S_{134} = \tilde{S}_2 \cdot \tilde{S}_{21}$ from the row m = 7, j = 2, k = 2, and then $S_{21} = \tilde{S}_2 \cdot \tilde{S}_4$ from the row m = 5, j = 2, k = 2; i.e., $S_{134} = \tilde{S}_2 \cdot \widetilde{\tilde{S}_2 \cdot \tilde{S}_4}$.
2. Because we know the structures $\tilde{S}_2$, $\tilde{S}_4$ (see Fig. 2.4), we can easily determine the structure S_{134} (see Fig. 2.5).

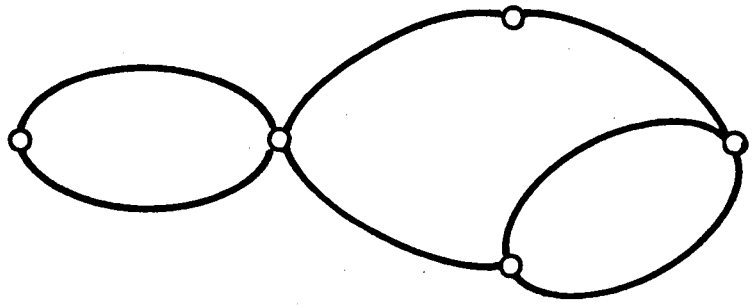


Fig. 2.5 Structure S_{134}.

Generally, if we have the data about m, j, k, $\lambda_{m,k}$ for all $m \leq M$ (the data are obtainable by using Procedure A modified to produce only the value of $\lambda_{m,k}$), Procedure B determines a particular structure $S_{i'}$ or $\tilde{S}_{i'}$, $(u(S_{i'}) = u(\tilde{S}_{i'}) \leq M)$.

Procedure B

Step 1: Set $i = i'$, $K = \phi$, where ϕ is null set.

Step 2: Do the following:

i. Find m such that

$$\lambda_{m,1} \leq i < \lambda_{m+1,1} .$$

ii. Find k such that

$$k = \max\{\kappa | \ \lambda_{m,\kappa} \leq i\},$$

and obtain j corresponding to k.

(For example, when i = 333, we obtain m = 8, k = 2, j = 2; when i = 54, we obtain m = 6, k = 4, j = 3)

Step 3: Find which case j corresponds to: a), b) or c) below.

a) $1 \leq j \leq [m/3]^{+}$. Calculate

$$z = \lambda_{m,k} + \lambda_{m-j+1,1} - \lambda_{m-j,k} .$$

If i < z, then set

$$\ell = \lambda_{m-j+1,1} + i - \lambda_{m,k} ,$$

and go to Step 4. Otherwise, set

$$\ell = \lambda_{m-j,1} + i - z,$$

and go to Step 5. (For example, when i = 333, i < z and $\ell = 54$)

b) $[m/3]^{+} < j \leq [(m-1)/2]^{+}$. Set

$$\ell = \lambda_{m-j,1} + i - \lambda_{m,k} ,$$

and go to Step 5.

c) $[(m-1)/2]^{+} < j$. Set

$$\ell = k + i - \lambda_{m,k} ,$$

and go to Step 5. (when $i = 54$, $\ell = 4$)

Step 4: Write the data i, k, 0, ℓ (0 means S_{ℓ}) and go to Step 6. (When $i = 333$, the data obtained are 333, 2, 0, 54; i.e., we have $S_{333} = \tilde{S}_2 \cdot S_{54}$)

Step 5: Write the data i, k, 1, ℓ (1 means $\tilde{S}_{\ell}$) and go to Step 6. (when $i = 54$, the data obtained are 54, 4, 1, 4; i.e., we have $S_{54} = \tilde{S}_4 \cdot \tilde{S}_4$)

Step 6: Check if $k > 9$ and if $\ell > 9$ (because we have the structures $S_1, S_2, \ldots, S_9$ in Fig. 2.4).

a) If both k and $\ell > 9$, then reset $i \leftarrow \ell$ and $K \leftarrow K \cup \{k\}$, and return to Step 2.

b) If either k or $\ell > 9$, then reset $i \leftarrow \max\{k,\ell\}$ and return to Step 2. (When $i = 333$, $i \leftarrow 54$)

c) If both k and $\ell \leq 9$, then check if $K \neq \phi$. If $K \neq \phi$, then take an element from K to become i (at the same time, i is removed from K) and return to Step 2. If $K = \phi$, then stop. (When $i = 54$, $K = \phi$)

We can make the structure $S_{i'}$ or $\tilde{S}_{i'}$ by using the output data of Procedure B. For example, the data obtained for S_{333} are as follows:

i'	k'	0 or 1	ℓ'
333	2	0	54
54	4	1	4

These data mean : $S_{333} = \tilde{S}_2 \cdot S_{54}$ and $S_{54} = \tilde{S}_4 \cdot \tilde{S}_4$; i.e., $S_{333} = \tilde{S}_2 \cdot \tilde{S}_4 \cdot \tilde{S}_4$.

Therefore we can obtain the structure designated by S_{333} by using Fig. 2.4.

2.6 RELIABILITY CALCULATION

The reliabilities and the failure probabilities of structures S_i ($=\tilde{S}_k \cdot S_\ell$ or $\tilde{S}_k \cdot \tilde{S}_\ell$) and $\tilde{S}_i$ can be calculated as follows:

a) In the case of $S_i = \tilde{S}_k \cdot S_\ell$, the failure probabilities of S_i and $\tilde{S}_i$ are

$$\begin{aligned} Q_i^1 &= (1 - \tilde{P}_k^1) \, Q_\ell^1 , \\ P_i^0 &= (1 - \tilde{Q}_k^0) \, P_\ell^0 , \\ \tilde{P}_i^1 &= (1 - Q_k^1) \, \tilde{P}_\ell^1 , \\ \tilde{Q}_i^0 &= (1 - P_k^0) \, \tilde{Q}_\ell^0 . \end{aligned} \qquad (2.17)$$

b) In the case of $S_i = \tilde{S}_k \cdot \tilde{S}_\ell$,

$$
\begin{aligned}
Q_i^1 &= (1 - \tilde{P}_k^1) (1 - \tilde{P}_\ell^1) , \\
P_i^0 &= (1 - \tilde{Q}_k^0) (1 - \tilde{Q}_\ell^0) , \\
\tilde{P}_i^1 &= (1 - Q_k^1) (1 - Q_\ell^1) , \\
\tilde{Q}_i^0 &= (1 - P_k^0) (1 - P_\ell^0) .
\end{aligned}
\tag{2.18}
$$

c) The reliabilities of S_i and $\tilde{S}_i$ are

$$
\begin{aligned}
R_i &= P_i^0 - Q_i^1 , \\
\tilde{R}_i &= \tilde{P}_i^1 - \tilde{Q}_i^0 .
\end{aligned}
\tag{2.19}
$$

Procedure C, which is a modification of Procedure A, shows how to calculate the reliabilities of all possible s-p structures for 1,..., M units. This begins with m = 2.

Procedure C

Step 1: Set m = 2, $Q_1^1 = q^1$, $P_1^0 = 1 - q^0$, $\tilde{P}_1^1 = 1 - q^1$, $\tilde{Q}_1^0 = q^0$, $\lambda_{1,1} = 1$, $\lambda_{2,1} = 2$, i = 1, and go to Step 5.

Step 2: DO vi j = 1, $[m/3]^+$, 1

DO vi k = $\lambda_{j,1}$, $\lambda_{j+1,1}-1$, 1

DO ii ℓ = $\lambda_{m-j,k}$, $\lambda_{m-j+1,1}-1$, 1

i. Reset i ← i + 1 and calculate Q_i^1, P_i^0, $\tilde{P}_i^1$, $\tilde{Q}_i^0$ by using (2.17), and then calculate R_i, $\tilde{R}_i$ by using (2.19).

ii. CONTINUE

DO iv $\ell = \lambda_{m-j,1}, \lambda_{m-j+1,1}^{-1}, 1$

iii. Reset $i \leftarrow i + 1$ and calculate Q_i^1, P_i^0, $\tilde{P}_i^1$, $\tilde{Q}_i^0$ by using (2.18), and then calculate R_i, $\tilde{R}_i$ by using (2.19).

iv. CONTINUE

v. Set $\lambda_{m,k} = i + 1$.

vi. CONTINUE

Step 3: If $[m/3]^+ + 1 > [(m-1)/2]^+$, then go to Step 5. Otherwise go to Step 4.

Step 4: DO iv $j = [m/3]^+ + 1, [(m-1)/2]^+, 1$

DO ii $\ell = \lambda_{m-j,1}, \lambda_{m-j+1,1}^{-1}, 1$

DO iv $k = \lambda_{j,1}, \lambda_{j+1,1}^{-1}, 1$

i. Reset $i \leftarrow i + 1$ and calculate Q_i^1, P_i^0, $\tilde{P}_i^1$, $\tilde{Q}_i^0$ by using (2.18), and then calculate R_i, $\tilde{R}_i$ by using (2.19).

ii. CONTINUE

iii. Set $\lambda_{m,k+1} = i + 1$.

iv. CONTINUE

Step 5: If m is odd, then go to Step 6. Otherwise, set $j = [m/2]^+$ and do the following thing:

DO iv $k = \lambda_{j,1}, \lambda_{j+1,1}^{-1}, 1$

DO ii $\ell = k, \lambda_{m-j+1,1}^{-1}, 1$

i. Reset $i \leftarrow i + 1$ and calculate Q_i^1, P_i^0, $\tilde{P}_i^1$, $\tilde{Q}_i^0$ by using (2.18), and then calculate R_i, $\tilde{R}_i$ by using (2.19).

ii. CONTINUE

iii. Set $\lambda_{m,k+1} = i + 1$.

iv. CONTINUE

Step 6: If $m \neq M$, then reset $m \leftarrow m + 1$ and set $\lambda_{m+1,1} = \lambda_{m,k+1}$ and return to Step 2. Otherwise, stop; we have the failure probabilities and the reliabilities of S_i, $\tilde{S}_i$ for all $i = 1, \ldots, N$.

The result that Procedure C solved an example, $q^1 = 0.01$, $q^0 = 0.05$, $M = 7$, is shown in Table 2.3. Table 2.3 shows that, in this case, a) for series redundant structures (S_2, S_3, S_5, S_{10}, S_{22}, S_{55}), the reliabilities decrease with increasing number of redundant units, like 0.9024, 0.8145, ..., 0.6983, b) for parallel redundant structures ($\tilde{S}_2$, $\tilde{S}_3$, $\tilde{S}_5$, $\tilde{S}_{10}$, $\tilde{S}_{22}$, $\tilde{S}_{55}$), the reliability of structure $\tilde{S}_2$ is maximum, and reliabilities of structures having 3 or more i.i.d. units decrease with increasing number of redundant units, c) for general series-parallel redundant structures, the maximum of reliabilities of all possible s-p structures for each value of m increases monotonically. In the optimization problems formulated by Tillman [T7], he uses series or parallel redundancy, but a), b) and c) show that it is much better to use series-parallel redundancy.

Table 2.3

Reliabilities and Failure Probabilities of All Possible S-P Structures for 1,...,7 Units ($q^1 = 0.01$, $q^0 = 0.05$)

i	R_i	Q_i^1	Q_i^0	$\tilde{R}_i$	$\tilde{Q}_i^1$	$\tilde{Q}_i^0$
(m = 1)						
1	.9400	.100E-01	.500E-01	.9400	.100E-01	.500E-01
(m = 2)						
2	.9024	.100E-03	.975E-01	.9776*	.199E-01	.250E-02
(m = 3)						
3	.8574	.100E-05	.143E+00	.9702	.297E-01	.125E-03
4	.9474	.199E-03	.524E-01	.9850*	.101E-01	.488E-02
(m = 4)						
5	.8145	.100E-07	.185E+00	.9606	.394E-01	.625E-05
6	.9002	.199E-05	.998E-01	.9798	.200E-01	.244E-03
7	.9496	.297E-03	.501E-01	.9829	.100E-01	.713E-02
8	.9453	.101E-03	.546E-01	.9872	.102E-01	.262E-02
9	.9946*	.396E-03	.499E-02	.9903	.200E-03	.951E-02
(m = 5)						
10	.7738	.100E-09	.226E+00	.9510	.490E-01	.313E-06
11	.8552	.199E-07	.145E+00	.9702	.298E-01	.122E-04
12	.9024	.297E-05	.976E-01	.9797	.199E-01	.357E-03
13	.8981	.101E-05	.102E+00	.9798	.201E-01	.131E-03
14	.9453	.396E-05	.547E-01	.9893	.102E-01	.475E-03
15	.9496	.394E-03	.500E-01	.9807	.100E-01	.927E-02
16	.9496	.200E-03	.502E-01	.9850	.100E-01	.499E-02
17	.9431	.100E-03	.568E-01	.9872	.103E-01	.251E-02
18	.9474	.102E-03	.525E-01	.9872	.101E-01	.273E-02
19	.9410	.200E-05	.590E-01	.9894	.104E-01	.250E-03
20	.9968*	.591E-03	.262E-02	.9860	.101E-03	.139E-01
21	.9924	.201E-03	.736E-02	.9946	.299E-03	.511E-02
(m = 6)						
22	.7351	.100E-11	.265E+00	.9415	.585E-01	.156E-07
23	.8125	.199E-09	.188E+00	.9605	.395E-01	.609E-06
24	.8573	.297E-07	.143E+00	.9703	.297E-01	.178E-04
25	.8532	.101E-07	.147E+00	.9701	.299E-01	.655E-05
26	.8980	.396E-07	.102E+00	.9799	.201E-01	.238E-04
27	.9025	.394E-05	.975E-01	.9796	.199E-01	.464E-03
28	.9023	.200E-05	.977E-01	.9798	.199E-01	.249E-03
29	.8961	.100E-05	.104E+00	.9797	.202E-01	.125E-03
30	.9001	.102E-05	.999E-01	.9799	.200E-01	.137E-03

Table 2.3 (Continued)

i	R_i	Q_i^1	Q_i^0	$\tilde{R}_i$	$\tilde{Q}_i^1$	$\tilde{Q}_i^0$
31	.8939	.200E-07	.106E+00	.9797	.203E-01	.125E-04
32	.9475	.591E-05	.525E-01	.9892	.101E-01	.695E-03
33	.9430	.201E-05	.570E-01	.9894	.103E-01	.255E-03
34	.9495	.490E-03	.500E-01	.9787	.100E-01	.113E-01
35	.9497	.298E-03	.500E-01	.9828	.100E-01	.724E-02
36	.9495	.199E-03	.503E-01	.9851	.100E-01	.488E-02
37	.9497	.201E-03	.501E-01	.9849	.100E-01	.509E-02
38	.9494	.102E-03	.505E-01	.9873	.100E-01	.274E-02
39	.9411	.100E-03	.588E-01	.9871	.104E-01	.250E-02
40	.9452	.100E-03	.547E-01	.9873	.102E-01	.251E-02
41	.9475	.103E-03	.524E-01	.9871	.101E-01	.284E-02
42	.9473	.101E-03	.526E-01	.9873	.101E-01	.262E-02
43	.9497	.104E-03	.502E-01	.9870	.100E-01	.295E-02
44	.9368	.101E-05	.632E-01	.9893	.106E-01	.131E-03
45	.9451	.299E-05	.549E-01	.9894	.102E-01	.368E-03
46	.9925	.788E-05	.748E-02	.9988	.300E-03	.927E-03
47	.9967	.784E-03	.251E-02	.9818	.100E-03	.181E-01
48	.9969	.398E-03	.274E-02	.9902	.102E-03	.973E-02
49	.9902	.199E-03	.961E-02	.9947	.396E-03	.489E-02
50	.9947	.203E-03	.511E-02	.9945	.201E-03	.533E-02
51	.9880	.398E-05	.120E-01	.9990*	.496E-03	.487E-03
52	.9989	.882E-03	.250E-03	.9797	.200E-05	.203E-01
53	.9947	.300E-03	.500E-02	.9923	.200E-03	.747E-02
54	.9902	.102E-03	.973E-02	.9969	.398E-03	.274E-02
(m = 7)						
55	.6983	.100E-13	.302E+00	.9321	.679E-01	.781E-09
56	.7718	.199E-11	.228E+00	.9509	.491E-01	.305E-07
57	.8144	.297E-09	.186E+00	.9606	.394E-01	.891E-06
58	.8105	.101E-09	.189E+00	.9604	.396E-01	.327E-06
59	.8531	.396E-09	.147E+00	.9701	.299E-01	.119E-05
60	.8574	.394E-07	.143E+00	.9703	.297E-01	.232E-04
61	.8572	.200E-07	.143E+00	.9703	.297E-01	.125E-04
62	.8513	.100E-07	.149E+00	.9700	.300E-01	.626E-05
63	.8551	.102E-07	.145E+00	.9702	.298E-01	.683E-05
64	.8492	.200E-09	.151E+00	.9699	.301E-01	.624E-06
65	.9001	.591E-07	.999E-01	.9800	.200E-01	.348E-04
66	.8959	.201E-07	.104E+00	.9798	.202E-01	.128E-04
67	.9025	.490E-05	.975E-01	.9795	.200E-01	.566E-03
68	.9025	.298E-05	.975E-01	.9797	.199E-01	.362E-03
69	.9022	.200E-05	.978E-01	.9799	.199E-01	.244E-03
70	.9024	.201E-05	.976E-01	.9798	.199E-01	.255E-03

Table 2.3 (Continued)

i	R_i	Q_i^1	Q_i^0	$\tilde{R}_i$	$\tilde{Q}_i^1$	$\tilde{Q}_i^0$
71	.9021	.102E-05	.979E-01	.9800	.199E-01	.137E-03
72	.8941	.100E-05	.106E+00	.9796	.203E-01	.125E-03
73	.8980	.100E-05	.102E+00	.9798	.201E-01	.126E-03
74	.9002	.103E-05	.998E-01	.9799	.200E-01	.142E-03
75	.9000	.101E-05	.100E+00	.9799	.200E-01	.131E-03
76	.9023	.104E-05	.977E-01	.9800	.200E-01	.148E-03
77	.8900	.101E-07	.110E+00	.9795	.205E-01	.656E-05
78	.8979	.299E-07	.102E+00	.9799	.201E-01	.184E-04
79	.9429	.788E-07	.571E-01	.9897	.103E-01	.463E-04
80	.9476	.784E-05	.524E-01	.9890	.101E-01	.904E-03
81	.9474	.398E-05	.526E-01	.9894	.101E-01	.486E-03
82	.9409	.199E-05	.591E-01	.9894	.104E-01	.244E-03
83	.9451	.203E-05	.549E-01	.9895	.102E-01	.266E-03
84	.9386	.398E-07	.614E-01	.9895	.105E-01	.243E-04
85	.9498	.882E-05	.502E-01	.9890	.100E-01	.102E-02
86	.9452	.300E-05	.547E-01	.9894	.102E-01	.373E-03
87	.9408	.102E-05	.592E-01	.9895	.104E-01	.137E-03
88	.9494	.585E-03	.500E-01	.9768	.100E-01	.132E-01
89	.9496	.395E-03	.500E-01	.9806	.100E-01	.938E-02
90	.9497	.297E-03	.500E-01	.9829	.100E-01	.714E-02
91	.9496	.299E-03	.500E-01	.9827	.100E-01	.734E-02
92	.9498	.201E-03	.500E-01	.9849	.100E-01	.510E-02
93	.9494	.199E-03	.504E-01	.9851	.100E-01	.488E-02
94	.9496	.199E-03	.502E-01	.9851	.100E-01	.489E-02
95	.9496	.202E-03	.502E-01	.9848	.100E-01	.520E-02
96	.9496	.200E-03	.501E-01	.9850	.100E-01	.499E-02
97	.9498	.203E-03	.500E-01	.9847	.100E-01	.530E-02
98	.9492	.101E-03	.507E-01	.9874	.100E-01	.262E-02
99	.9497	.103E-03	.502E-01	.9871	.100E-01	.285E-02
100	.9392	.100E-03	.607E-01	.9870	.105E-01	.250E-02
101	.9430	.100E-03	.569E-01	.9872	.103E-01	.250E-02
102	.9452	.100E-03	.546E-01	.9873	.102E-01	.252E-02
103	.9451	.100E-03	.548E-01	.9873	.102E-01	.251E-02
104	.9473	.100E-03	.526E-01	.9874	.101E-01	.252E-02
105	.9475	.104E-03	.524E-01	.9870	.101E-01	.294E-02
106	.9475	.102E-03	.524E-01	.9872	.101E-01	.274E-02
107	.9472	.101E-03	.527E-01	.9873	.101E-01	.262E-02
108	.9474	.101E-03	.525E-01	.9873	.101E-01	.263E-02
109	.9471	.100E-03	.528E-01	.9874	.101E-01	.251E-02
110	.9498	.106E-03	.501E-01	.9863	.100E-01	.316E-02

Table 2.3 (Continued)

i	R_i	Q_i^1	Q_i^0	$\tilde{R}_i$	$\tilde{Q}_i^1$	$\tilde{Q}_i^0$
111	.9495	.102E-03	.503E-01	.9873	.100E-01	.274E-02
112	.9491	.300E-05	.509E-01	.9896	.100E-01	.374E-03
113	.9328	.100E-05	.672E-01	.9891	.108E-01	.125E-03
114	.9408	.102E-05	.592E-01	.9895	.104E-01	.137E-03
115	.9453	.397E-05	.546E-01	.9893	.102E-01	.481E-03
116	.9449	.201E-05	.551E-01	.9895	.102E-01	.256E-03
117	.9495	.496E-05	.505E-01	.9894	.100E-01	.599E-03
118	.9307	.200E-07	.693E-01	.9891	.109E-01	.125E-04
119	.9429	.200E-05	.571E-01	.9897	.103E-01	.250E-03
120	.9474	.398E-05	.526E-01	.9894	.101E-01	.486E-03
121	.9949	.118E-04	.512E-02	.9984	.201E-03	.136E-02
122	.9901	.400E-05	.984E-02	.9991	.399E-03	.498E-03
123	.9965	.975E-03	.250E-02	.9778	.100E-03	.221E-01
124	.9969	.593E-03	.251E-02	.9858	.100E-03	.141E-01
125	.9967	.396E-03	.286E-02	.9904	.103E-03	.952E-02
126	.9970	.400E-03	.263E-02	.9900	.101E-03	.994E-02
127	.9968	.203E-03	.297E-02	.9946	.104E-03	.534E-02
128	.9880	.199E-03	.118E-01	.9946	.494E-03	.488E-02
129	.9923	.199E-03	.748E-02	.9948	.300E-03	.490E-02
130	.9948	.205E-03	.500E-02	.9943	.200E-03	.554E-02
131	.9946	.201E-03	.522E-02	.9947	.202E-03	.512E-02
132	.9970	.207E-03	.275E-02	.9941	.102E-03	.576E-02
133	.9836	.201E-05	.164E-01	.9991	.691E-03	.256E-03
134	.9924	.595E-05	.759E-02	.9990	.301E-03	.718E-03
135	.9987	.117E-02	.131E-03	.9735	.101E-05	.265E-01
136	.9990	.594E-03	.369E-03	.9858	.299E-05	.142E-01
137	.9924	.297E-03	.726E-02	.9926	.298E-03	.715E-02
138	.9970	.303E-03	.274E-02	.9921	.102E-03	.779E-02
139	.9904	.594E-05	.963E-02	.9989	.397E-03	.712E-03
140	.9947	.398E-03	.488E-02	.9901	.199E-03	.972E-02
141	.9947	.202E-03	.512E-02	.9946	.201E-03	.522E-02
142	.9879	.101E-03	.120E-01	.9969	.496E-03	.262E-02
143	.9924	.103E-03	.748E-02	.9968	.300E-03	.286E-02
144	.9857	.202E-05	.143E-01	.9991*	.595E-03	.262E-03

* maximum reliability for each value of m

2.7 AN OPTIMIZATION PROBLEM

Consider the following problem: find the most reliable s-p redundant structure composed of minimum number of valves in the examples in Section 2.2 (output 1 is open, output 0 is closed) which have the failure probabilities $q^1 = 0.01$, $q^0 = 0.05$, subject to

$$Q_i^1 \text{ (or } \tilde{Q}_i^1) \leq q^1, \tag{2.20}$$

$$Q_i^0 \text{ (or } \tilde{Q}_i^0) \leq q^0, \tag{2.21}$$

$$Q_i^0 / Q_i^1 \text{ (or } \tilde{Q}_i^0 / \tilde{Q}_i^1) \geq 1000, \tag{2.22}$$

$$R_i \text{ (or } \tilde{R}_i) \geq 0.99, \tag{2.23}$$

where (2.22) is a constraint for fail-safe: the failure '0' is known to be much safer than the failure '1'. We can solve this problem by applying Procedure C. The structures S_{122}, S_{134} and S_{139} satisfy (2.20)$\sim$(2.23) and S_{134} is optimal; see Table 2.3. The structure designated as S_{134} is determined by using Table 2.1 or Procedure B; see Fig. 2.5. The reliability and the failure probabilities are $R_{134} = 0.9924$, $Q_{134}^1 = 0.59\times10^{-5}$ and $Q_{134}^0 = 0.76\times10^{-2}$.

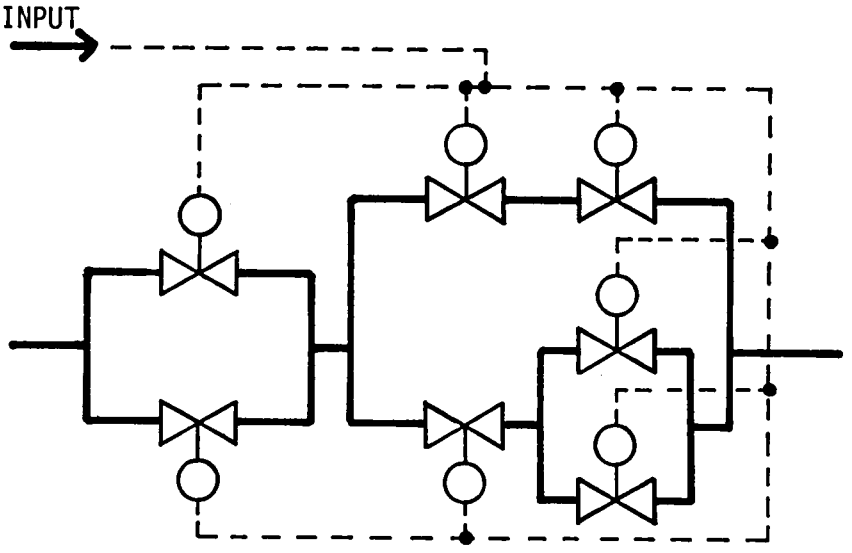


Fig. 2.6 Real structure optimized.

Chapter 3

A HEURISTIC METHOD FOR A SINGLE OBJECTIVE PROBLEM

3.1 INTRODUCTION

In most theoretical reliability problems the two basic methods of improving the reliability of a system are:

(i) increase the reliability of each component,
(ii) add redundant components (in parallel, stand-by, etc.).

In this chapter, we consider the case where either of the two ways can be used at each stage to improve the system reliability. An optimization problem is formulated that maximizes the system reliability where the optimal candidate is selected for each stage without violating the constraints.

Many methods have been presented for solving the optimal redundancy allocation problem. They are classified into two groups:

(i) exact methods that require a considerable amount of computation time,
(ii) approximate methods that provide an approximate, often exact, solution but can solve within a reasonble amount of computation time.

The proposed method falls into the latter group.

In the present method, the solution is obtained by repeatedly using a more reliable candidate at the stage that has the greatest value of the 'weighted sensitivity function' without violating any of the constraints. The weighted sensitivity function for each stage is the product of a quantity obtained as a function of the objective function and the constraints. The balance between these two quantities is controlled by a balancing coefficient, α. The best solution is selected from among a number of solutions obtained by using multiple values of α.

The present method and the methods presented in [A1,G10,M7,M9,S2, S5] were used to solve randomly generated problems. The problems involve determining the optimal level of reliability with parallel redundancies under linear constraints. The effectiveness of the methods are measured by comparing the average 'relative error', the 'maximum relative error', and the 'optimality rate' obtained from each method with the exact solutions.

3.2 PROBLEM FORMULATION

Consider an n-stage series system where the components are statistically independent. In each stage, redundant components can be added (in parallel, stand-by, or k-out-of-n:G, etc.), or a more reliable component can be used to improve the system reliability. The following assumptions are made.

(i) The candidate x_i is less reliable than the candidate $x_i + 1$.
(ii) x_i does not consume more of any resource than $x_i + 1$.
(iii) The function $g_{ji}(x_i)$ are separable functions.

The problem can be formally stated as follows:

$$\text{Problem A:} \quad \text{maximize} \quad R(X) = \prod_{i=1}^{n} R_i(x_i)$$

$$\text{subject to} \quad \sum_{i=1}^{n} g_{ji}(x_i) \leq b_j \quad (j = 1,\ldots,m),$$

$$1 \leq x_i \leq \bar{x}_i,$$

$$x_i \text{ integer},$$

where $R_i(x_i)$ and $g_{ji}(x_i)$ are monotone nondecreasing.

Problem A can be reformulated as follows:

$$\text{Problem B:} \quad \text{maximize} \quad \ln R(X) = \sum_{i=1}^{n} \sum_{k=1}^{x_i} \Delta f_i(k)$$

$$\text{subject to} \quad \sum_{i=1}^{n} \sum_{k=1}^{x_i} \Delta g_{ji}(k) \leq b_j \quad (j = 1,\ldots,m),$$

$$1 \leq x_i \leq \bar{x}_i,$$

$$x_i \text{ integer},$$

where

$$\Delta f_i(k) = \begin{cases} \ln R_i(1) & (k = 1), \\ \ln R_i(k) - \ln R_i(k-1) & (k \geq 2), \end{cases}$$

$$\Delta g_{ji}(k) = \begin{cases} g_{ji}(1) & (k = 1), \\ g_{ji}(k) - g_{ji}(k-1) & (k \geq 2), \end{cases}$$

$$\Delta f_i(k) \geq 0 \quad \text{for any } i \text{ and } k,$$

$$\Delta g_{ji}(k) \geq 0 \quad \text{for any } j, i, \text{ and } k.$$

3.3 COMPUTATIONAL PROCEDURE

The present procedure begins with an initial current solution X^* (in the most cases, = (1,1,...,1)). The following steps summarize the present method:

Step 1: Set $X^c = X^*$, where X^c is the current solution, and set

$$L_{+1} = \{1,2,\ldots,n\},$$

where L_{+1} is the set of all stages whose reliabilities can be increased more.

Step 2: Calculate b_j^c for all $j \in \{1,\ldots,m\}$:

$$b_j^c = b_j - \sum_{i=1}^{n} \sum_{k=1}^{x_i^c} \Delta g_{ji}(k). \tag{3.1}$$

Step 3: Calculate Δx_i for all $i \in L_{+1}$:

$$\Delta x_i = \min_{j \in \{1,\ldots,m\}} \{b_j^c / \Delta g_{ji}(x_i^c + 1)\}. \tag{3.2}$$

Step 4: Let $L_{+1} = \{i \mid \Delta x_i \geq 1\}$.

Step 5: If L_{+1} is empty, then stop. A solution has been obtained. Otherwise continue to search over k such that

$$s_{i^*} = \max_{i \in L_{+1}} \{s_i\}, \tag{3.3}$$

where s_i is weighted sensitivity function such as

$$s_i = \Delta f_i(x_i^c + 1)[(1 - \alpha)\min_{k \in L_{+1}} \{\Delta x_k\} + \alpha \Delta x_i]. \tag{3.4}$$

Step 6: If $x^c_{i*} = \bar{x}_{i*}$, then x^c_{i*} is the final number used in stage i*; exclude i* from L_{+1} and return to Step 5. If $x^c_{i*} < \bar{x}_{i*}$, then reset $x^c_{i*} \leftarrow x^c_{i*} + 1$ and $b^c_j \leftarrow b^c_j - g_{ji*}(x^c_{i*})$ for all j, and return to Step 3.

The above procedure requires a balancing coefficient α to be given. The solutions for a set of α(probably α = 0, 0.1, 0.2,..., 1.0; $1/\alpha$ = 0.9, 0.6, 0.3) should be obtained. The best solution among the solution for the given set of α's is the final solution.

3.4 NUMERICAL EXAMPLE

Consider the system composed of 3 stages operating in series. The system reliability is increased by choosing a more reliable component out of four candidates at stage 1, adding redundant components in parallel at stage 2, and using 2-out-of-(x_3+1):G configuration at stage 3. The problem is to maximize the system reliability

$$R(X) = \prod_{i=1}^{3} R_i(x_i),$$

$$R_1(x_1) = 0.88, 0.92, 0.98, 0.99, \quad \text{for } x_1 = 1,2,3,4, \text{ respectivly},$$

$$R_2(x_2) = 1 - (1 - 0.81)^{x_2},$$

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

subject to the constraints

$$g_1 = 4\exp\{\frac{0.02}{1 - R_1(x_1)}\} + 5x_2 + 2(x_3 + 1) \leq 45,$$

$$g_2 = 5 + e^{x_1/8} + 3(x_2 + e^{x_2/4}) + 5(x_3 + 1 + e^{x_3/4}) \leq 70,$$

$$g_3 = 10 + 8x_2 e^{x_2/4} + 6x_3 e^{x_3/4} \leq 240.$$

This problem is illustrated in Table 3.1 according to the formulation of Problem B. The current solutions at $\alpha = 0.5$ are obtained as shown in Table 3.2. The procedure was reapplied with 14 α-values. The solu-

Table 3.1

Input Data of the Example

OBJECT FUNCTION						
i	$\Delta f_i(1)$	$\Delta f_i(2)$	$\Delta f_i(3)$	$\Delta f_i(4)$	$\Delta f_i(5)$	$\Delta f_i(6)$
1	-0.127833	0.044452	0.063178	0.010153		
2	-0.210721	0.173953	0.029885	0.005579	0.001057	0.000200
3	-0.522729	0.378436	0.103187	0.029623	0.008356	0.002295
CONSTRAINT 1, b_1= 45.0						
i	$\Delta g_{1i}(1)$	$\Delta g_{1i}(2)$	$\Delta g_{1i}(3)$	$\Delta g_{1i}(4)$	$\Delta g_{1i}(5)$	$\Delta g_{1i}(6)$
1	4.725440	0.410659	5.737030	18.68309		
2	5.000000	5.000000	5.000000	5.000000	5.000000	5.000000
3	4.000000	2.000000	2.000000	2.000000	2.000000	2.000000
CONSTRAINT 2, b_2= 70.0						
i	$\Delta g_{2i}(1)$	$\Delta g_{2i}(2)$	$\Delta g_{2i}(3)$	$\Delta g_{2i}(4)$	$\Delta g_{2i}(5)$	$\Delta g_{2i}(6)$
1	6.133148	0.150877	0.170966	0.193729		
2	3.750000	7.196260	4.404739	4.803848	5.316177	5.973984
3	16.42014	6.823471	7.341400	8.006409	8.860291	9.956741
CONSTRAINT 3, b_3=240.0						
i	$\Delta g_{3i}(1)$	$\Delta g_{3i}(2)$	$\Delta g_{3i}(3)$	$\Delta g_{3i}(4)$	$\Delta g_{3i}(5)$	$\Delta g_{3i}(6)$
1	10.00000	0.000001	0.000001	0.000001		
2	10.27220	16.10736	24.42845	36.17702	52.62871	75.50735
3	7.704150	12.08051	18.32135	27.13277	39.47153	56.63051

Table 3.2

Current Solutions at $\alpha = 0.5$

$\text{Max.}S_i$	Current Solution Stage 1	Stage 2	Stage 3
	1	1	1
2.360687	1	1	2
1.696050	2	1	2
0.878904	2	2	2
0.415004	2	2	3
0.207915	3	2	3
0.089280	3	3	3
0.065152	3	3	4
0.009056	3	3	5

tions obtained for $\alpha = 0, 0.1, 0.2, \ldots, 0.7$ are identical:

$X = (3, 3, 5)$

$R(X) = 0.97024.$

The solutions for $\alpha = 0.8, 0.9, 1.0$, or $1/\alpha = 0.9, 0.6, 0.3$ are, too, identical:

$X = (3, 4, 4)$

$R(X) = 0.96755.$

The solution (3,3,5) is the exact optimal solution.

3.5 COMPUTATIONAL EXPERIENCE

In this section we evaluate the present method. Seven problem sets were solved, each consisting of 20 parallel redundancy allocation problems having the same numbers of stages and linear constraints. The parallel redundancy allocation problem is as follows:

$$\text{Maximize} \quad R(X) = \prod_{i=1}^{n} (1 - q_i^{x_i})$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_{ji} x_i \leq b_j \quad (j = 1,\ldots,m),$$

$$x_i \geq 1,$$

$$x_i \text{ integer.}$$

In each problem set, the coefficients of the constraints were randomly generated.

The present method was repeated with $\alpha = 0, 0.1, 0.2, \ldots, 1.0$ and $1/\alpha = 0.9, 0.6, 0.3$ for each of the test problems. Table 3.3 shows the average 'relative error' and 'optimality rate'. The relative error is defined as

$$\frac{R(X^{opt}) - R(X^*)}{1 - R(X^{opt})},$$

where X^* is a solution obtained and X^{opt} is the exact optimal solution which is obtained by method [N4] or dynamic programming. The optimality rate is the ratio of the number of problems to which optimal solutions are obtained for the 20 problems. Table 3.3 suggests that, when the size of problem is small, the procedure yields a fairly good

Table 3.3 Quality of Solutions Obtained with Each of 14 α-values

		Problem size (n x m)						
		5 x 3	8 x 3	10 x 1	10 x 3	10 x 8	20 x 1	50 x 1
α = 0.0	A	0.00549	0.06266	0.03430	0.02995	0.03829	0.08185	0.05169
	O	19/20	10/20	12/20	12/20	13/20	0/20	0/20
α = 0.1	A	0.00549	0.03688	0.02795	0.01488	0.01319	0.04237	0.03617
	O	19/20	13/20	13/20	14/20	15/20	1/20	0/20
α = 0.2	A	0.00549	0.03929	0.01696	0.00857	0.00381	0.02956	0.01891
	O	19/20	13/20	15/20	14/20	16/20	4/20	0/20
α = 0.3	A	0.00000	0.03030	0.01658	0.00857	0.00398	0.01417	0.01178
	O	20/20	15/20	15/20	14/20	15/20	7/20	1/20
α = 0.4	A	0.00000	0.02479	0.00075	0.00627	0.00398	0.01148	0.00572
	O	20/20	15/20	18/20	15/20	15/20	10/20	1/20
α = 0.5	A	0.00000	0.01898	0.00075	0.00783	0.00455	0.00483	0.00245
	O	20/20	16/20	18/20	14/20	14/20	13/20	1/20
α = 0.6	A	0.00302	0.01898	0.00371	0.01510	0.00455	0.00082	0.00116
	O	19/20	16/20	16/20	13/20	14/20	17/20	6/20
α = 0.7	A	0.00302	0.02644	0.00560	0.02243	0.00393	0.00082	0.00078
	O	19/20	15/20	15/20	10/20	14/20	17/20	7/20
α = 0.8	A	0.00302	0.03490	0.00797	0.02537	0.00683	0.00159	0.00058
	O	19/20	15/20	14/20	10/20	15/20	14/20	11/20
α = 0.9	A	0.02404	0.02598	0.01843	0.03183	0.01175	0.00259	0.00085
	O	17/20	16/20	14/20	10/20	13/20	12/20	11/20
α = 1.0	A	0.02404	0.02598	0.03082	0.03642	0.01631	0.00310	0.00107
	O	17/20	16/20	13/20	10/20	13/20	11/20	9/20
1/α = 0.9	A	0.03294	0.02792	0.03082	0.03642	0.01631	0.00686	0.00159
	O	16/20	15/20	13/20	10/20	13/20	9/20	6/20
1/α = 0.6	A	0.05954	0.02697	0.05880	0.04623	0.02664	0.02796	0.00951
	O	12/20	15/20	7/20	9/20	12/20	3/20	1/20
1/α = 0.3	A	0.14871	0.06103	0.07202	0.06032	0.03333	0.10918	0.02339
	O	12/20	11/20	5/20	6/20	8/20	1/20	0/20

A : Average relative error
M : Maximum relative error

Table 3.4 Quality of the Best Solutions for 14 α-values

		Problem size (n x m)						
		5 x 3	8 x 3	10 x 1	10 x 3	10 x 8	20 x 1	50 x 1
The best of multi. sol.	A	0.00000	0.00000	0.00025	0.00010	0.00010	0.00020	0.00006
	M	0.00000	0.00000	0.00504	0.00206	0.00206	0.00395	0.00116
	O	20/20	20/20	19/20	19/20	19/20	18/20	17/20

A : Average relative error
M : Maximum relative error
O : Optimality rate

result with any value of α. The range of α within which the procedure yields a good solution becomes narrower with larger problems. Furthermore, Table 3.3 suggests that the value of α yielding the best solution becomes larger, which is in keeping with the number of variables.

Table 3.4 shows the average and maximum relative errors and the optimality rate for the best solutions obtained with the 14 α-values. A comparison of Tables 3.3 and 3.4 shows that in order to improve the quality of the solution, it is very useful to obtain multiple solutions and then select the best from these. With this approach, the computation time becomes longer than the other methods. However the computation time is 3.1 seconds on a FACOM 230/75 digital computer even for a problem that has 30 stages, 3 constraints, and is solved with 14 α-values to yield a final solution, which allocate 117 components in total.

3.6 COMPARISON WITH OTHER METHODS

We use here 'forward methods' as a general name for the present method and the methods [A1,B5,G10,M7,M9,S2,S5]. The forward methods are similar except for the difference in sensitivity function s_i. The basic idea of forward methods was proposed first by Mine [M7].

(i) The Mine method [M7] have been presented for solving single-constrained Problem A_1 in Section 1.5. Its sensitivity function is

$$s_i = \frac{1}{a_i} \ln\left(\frac{1 - q_i^{x_i+1}}{1 - q_i^{x_i}}\right). \tag{3.5}$$

Exchanging (3.4) in the present procedure to (3.5), we have the procedure for Mine's method.

(ii) Barlow et al. [B5] generated the Mine method to solve multi-constrained Problem A_1. The sensitivity function is

$$s_i = \frac{1}{\sum_{j=1}^{m} \lambda_j a_{ji}} \ln\left(\frac{1 - q_i^{x_i+1}}{1 - q_i^{x_i}}\right). \tag{3.6}$$

where $\sum_{j=1}^{m} \lambda_j = 1$. The procedure is reapplied in succession for each vectors $(\lambda_1,\ldots,\lambda_m)$, varing the λ_j by some fixed increment until all choices from $(1,0,\ldots,0)$ to $(0,\ldots,0,1)$ have been exhausted. The best solution among the solutions for the set of vectors $(\lambda_1,\ldots,\lambda_m)$ is the final solution. Hence the total computation increases exponentially with respect to the number of constraints.

(iii) The Sasaki method [S2], which is one for solving multi-constrained Problem A_1, consists of two parts: procedure for obtaining a forward solution and one for revising the solution. The procedure for a forward solution, which belongs to forward methods, have a sensitivity function such as

$$s_i = q_i^{x_i}. \tag{3.7}$$

This procedure is approximately equal to the Mine method obtained by removing $1/a_i$ from (3.5), because when $q_i << 1$

$$\ln\left(\frac{1 - q_i^{x_i+1}}{1 - q_i^{x_i}}\right) \approx \ln(1 + q_i^{x_i}). \tag{3.8}$$

The procedure for revising the forward solution is complex and becomes impractical with increasing number of variables.

(iv) Ghare et al. [G10] solved multi-constrained Problem A_1. Their procedure, too, consists of two parts: procedure for obtaining a forward solution and one for revising the solution. The procedure for a forward solution have a sensitivity function:

$$s_{ji} = \frac{b'_j}{a_{ji}} \ln\left(\frac{1 - q_i^{x_i+1}}{1 - q_i^{x_i}}\right), \tag{3.9}$$

$$s_{i*} = \min_{j=1,\ldots,m} \max_{ie L_{+1}} \{s_{ji}\}. \tag{3.10}$$

However the procedure does not agree with its computer program [T2] because of programming errors: 'min' and 'max' in (3.10) are programed in reverse order. In most cases, the computer program, which is quite different from the theory discussed in [G10], itself yields a better solution than the procedure based on their theory.

(v) Sharma et al. [S5] solved Problem A_3 in Section 1.5. Their method is just the same as the Sasaki procedure for a forward solution.

(vi) Misra [M9] present a method for solving multi-constrained Problem A_1. The flow chart in [M9] have a defect in stopping criterion: the computer program coded according to the flow chart does not stop in most cases, besides single-constrained problem. We modify his method to make the best use of his original idea as follows: using the sensitivity such as

$$s_i = \frac{1}{a_{ji}} \left(\frac{q_i^{x_i} - q_i^{x_i+1}}{1 - q_i^{x_i}}\right), \tag{3.11}$$

we obtain a solution for each of constraints j, and then

select the best one. Note that, when $q_i << 1$,

$$\frac{q_i^{x_i} - q_i^{x_i+1}}{1 - q_i^{x_i}} \approx q_i^{x_i} . \tag{3.12}$$

(vii) Aggarwal et al. [A1] solve Problem A_3. Their method use the sensitivity

$$s_i = \frac{q_i^{x_i} - q_i^{x_i+1}}{\prod_{j=m}^{m} (g_{ji}(x_i + 1) - g_{ji}(x_i))} , \tag{3.13}$$

where when $q_i << 1$,

$$q_i^{x_i} - q_i^{x_i+1} \approx q_i^{x_i} . \tag{3.14}$$

The method have the defect that, in the cases of multi-constrained problems, it often treats non-active constraints more strongly than active constraints. In general, the method produces poorer solutions as the number of constraints increases.

The Mine method is equal to the present method at $\alpha = 1.0$ for single-constrained Problem A_1. The computer program [T2] except for revising part is equal to the present method at $\alpha = 1.0$ for Problem A_1. The Misra method is approximately equal to the present procedure at $\alpha = 1.0$ for single-constrained Problem A_1. The Sasaki method and Sharma et al. method are approximately equal to the present procedure at $\alpha = 0$. The Aggalwal et al. method is approximately equal to the present procedure at $\alpha = 1.0$ for single-constrained Problem A_3.

In Table 3.5, the computational results for the other forward methods are indicated. The problem solved are the same as ones in Section 3.5. The comparison of Tables 3.4 and 3.5 suggests that the present method is the best one.

Table 3.5

Quality of Solutions Produced by the Other Methods

		Problem size (n × m)						
		5 × 3	8 × 3	10× 1	10× 3	10× 8	20× 1	50× 1
Mine	A			0.03082			0.00310	0.00107
	M			0.61649			0.06210	0.02133
	O			13/20			11/20	9/20
Sasaki*	A	0.00549	0.06266	0.03430	0.02995	0.03829	0.08185	0.05169
	M	0.10976	1.25326	0.68594	0.59902	0.76579	1.63696	1.03389
Sharma et al.	O	19/20	10/20	12/20	12/20	13/20	0/20	0/20
Ghare et al.**	A	0.02102	0.02598	0.03082	0.02942	0.01183	0.00227	0.00107
	M	0.42043	0.51962	0.61649	0.58845	0.23653	0.04541	0.02133
	O	18/20	16/20	13/20	12/20	14/20	13/20	9/20
Misra	A	0.00410	0.01808	0.03082	0.06192	0.00417	0.00310	0.00107
	M	0.08207	0.36159	0.61649	1.23834	0.08334	0.06210	0.02133
	O	18/20	16/20	13/20	8/20	17/20	11/20	9/20
Aggarwal et al.	A	0.22623	0.55135	0.03082	0.61009	3.53407	0.00310	0.00107
	M	1.10876	3.25551	0.61649	2.15883	10.23024	0.06210	0.02133
	O	7/20	2/20	13/20	2/20	0/20	11/20	9/20

A: Average relative error

M: Maximum relative error

O: Optimality rate

* except for revising part

** computer program in [T2]

Chapter 4

AN EXACT METHOD FOR A SINGLE OBJECTIVE PROBLEM

4.1 INTRODUCTION

Many reliability optimization problems are pure-integer non-linear programming problems, e.g., [B2,F2,H5,I2,M6,M17,N3,T7]. Some techniques guarantee exact optimality: i.e., the ones using dynamic programming and integer programming. The former techniques become impractical for problems having more that three functional constraints. In the latter techniques, an increase in the number of constraints affects very little the size of problem, but the computation time increases exponentially with respect to the number of variables. In general the latter techniques are superior to the former when the problem has multi-constraints.

The techniques using integer programming are as follows:

(i) Tillman, e.g., [T7], adopted the Gomory algorithm, after reformulating the problem into a 0-1 linear programming problem.

(ii) Misra [M8,M7] adopted the Lawler-Bell algorithm [L1], after reformulating into a 0-1 nonlinear programming problem.

(iii) Gen (Hyun), e.g., [G6], adopted the Geoffrion implicit

enumeration algorithm, after reformulating into a 0-1 linear programming problem having the same number of variables as Tillman's but fewer constraints.

(iv) Henin [H4] used the branch-and-bound method, without transforming the variables into 0-1 variables.

(v) McLeavy [M2,M3] modified Ghare-Taylor's method [G10] so as to adhere strictly to branch-and-bound principle.

Techniques (i)-(iii) can not solve a large problem (limited to somewhat less than 10 variables), mainly because the numbers of constraints and variables increase when 0-1 algorithms are used. Section 4.4 shows that (i)-(iii) appear to transform ineffectively an 'easy solvable problem' into a '0-1 problem' which is much more difficult to solve.

Techniques (iv) and (v) are designed for solving the linear constraint redundancy allocation problem, which has been dealt with by a number of authors using various techniques since 1956.

The purpose of this chapter is to solve reliability allocation problems having multiple nonlinear constraints regardless of separability: techniques in [G6,H4,M2,M3,T7] can not solve nonseparable problems. An efficient method is given which is based on branch-and-bound [G2,G9] and is designed for dealing with integer variables as they are, without increasing the number of variables or constraints. The method is developed by applying separation and relaxation techniques and considers the character specific to a given problem.

Three types of numerical examples are given including an illustration of the computations.

4.2 PROBLEM

The following integer nonlinear programming problem is treated.

$$\text{Problem (P):} \quad \text{maximize (or minimize)} \quad f(X)$$

$$\text{subject to} \quad g_j(X) \leq b_j \quad (j = 1,\ldots,m),$$

$$X = (x_1,\ldots,x_n),$$

$$\underline{c}_i \leq x_i \leq \bar{c}_i;\ x_i \text{ integer},$$

where

$f, g_1,\ldots, g_s$ monotone nondecreasing functions in each x_i,

$g_{s+1}, \ldots, g_m$ monotone nonincreasing functions in each x_i.

For maximization, this is the problem of maximizing system reliability or availability subject to several nonlinear resource constraints. For minimization, this is the problem of minimizing a system resource (cost, volume, weight, etc.) subject to nonlinear constraints including minimum reliability and/or availability. Some functions might be nonseparable; e.g., the system might have a nonseries configuration.

The problem becomes much easier ot solve when its functions are separable, viz.,

$$f(X) = \sum_{i=1}^{n} f_i(x_i), \tag{4.1}$$

$$g_j(X) = \sum_{i=1}^{n} g_{ji}(x_i) \quad (j = 1,\ldots,M). \tag{4.2}$$

4.3 COMPUTATIONAL PROCEDURE

The present method is based on branch-and-bound and uses relaxation and separation techniques. These are briefly explained here; further information can be found in [G2,G9]. An example is shown in Section 4.4 and Appendix D.

4.3.1 Enumeration Tree

A tree enumerates all feasible solutions of Problem (P); see Fig. 1. A vertex v_κ is the predecessor of the vertices emanating from v_κ, which in turn are successors of its predecessor v_κ. Each vertex belongs to a specified level.

Each edge from v_κ to its immediate successors corresponds to a restriction of fixing a variable x_k to one allowable integer value, i.e.,

$$x_k = \alpha_k \ , \quad \underline{x}_k^1 \leq \alpha_k \leq \bar{x}_k^1 \ , \tag{4.3}$$

where $\underline{x}_k^1$ and $\underline{x}_k^1$ are minimum and maximum integer values (lower and upper bounds) of x_k allowed at v_κ by the constraints, respectively, see (4.7), (4.8).

Each vertex corresponds to a problem; v_0 corresponds to the original problem (P), and generally the subproblem corresponding to v_κ is:

Subproblem (P_κ): maximize (or minmize) $f(X)$

$$\text{subject to} \quad g_j(X) \leq b_j \quad (j = 1,\ldots,m)$$

$$x_i = \alpha_i \quad (i \in L^r),$$

$$\underline{x}_i \leq x_i \leq \bar{x}_i;\ x_i \text{ integer} \quad (i \in L^f),$$

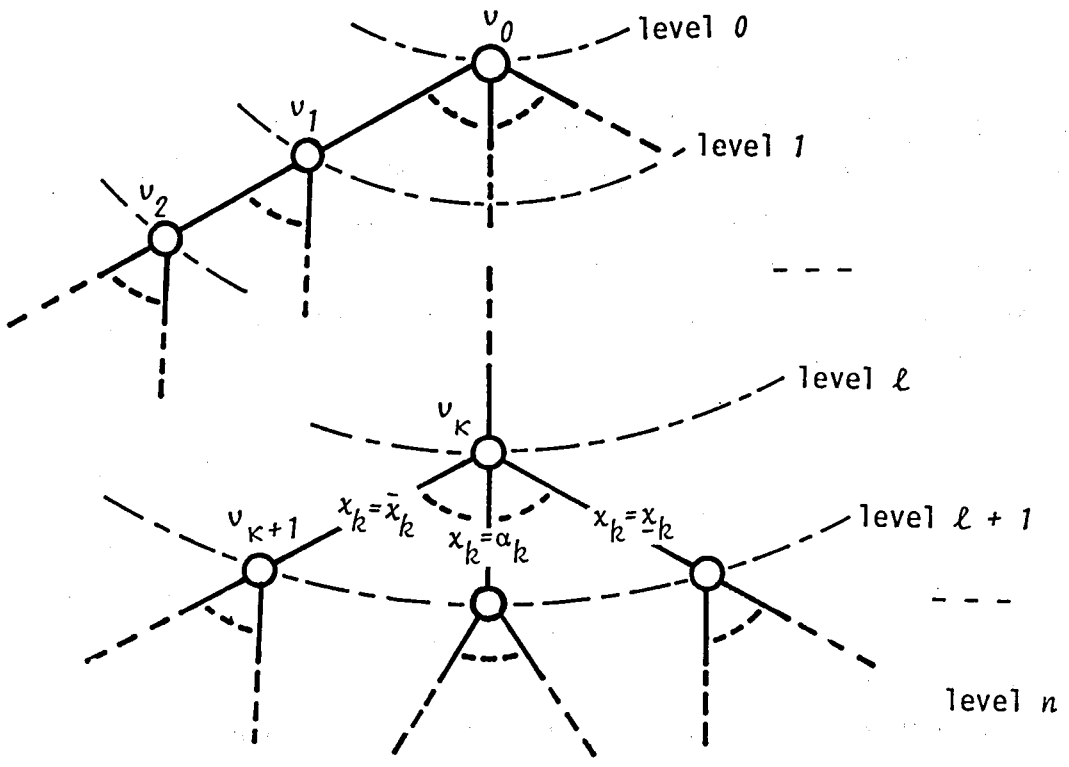


Fig. 4.1 Enumeration tree

where

L^r set of indices of fixed variables restricted by the edges along the unique path from v_0 to v_K,

L^f set of indices of the other variables, i.e., free variables.

Let $\underline{x}_i = \underline{c}_i$, $\bar{x}_i = \bar{c}_i$ $(i \in L^f)$ here (in the present procedure, $\underline{x}_i$ and $\bar{x}_i$ are exchanged for stronger bounds for x_i, as seen in Section 4.3.5). The subproblems at level n (terminal subproblems) have no free variables and each of them corresponds to one feasible solution.

4.3.2 Relaxation

Relaxation techniques are used to obtain the upper (or lower) value of f(X) for subproblem (P_K).

Any subproblem (P_K) can be relaxed with respect to its constraints, resulting in a new problem (P_K^r). The only requirement for (P_K^r) to be a valid relaxation of (P_K) is

$$F(P_K) \subseteq F(P_K^r), \tag{4.4}$$

where $F(\cdot)$ is the set of feasible solutions of the problem $(\cdot)$. This implies that for maximization (or minimization),

$$w(P_K) \leq w(P_K^r) \quad (\text{or } w(P_K) \geq w(P_K^r)), \tag{4.5}$$

where $w(\cdot)$ is the optimal value of the objective function in problem $(\cdot)$.

Therefore if

$$f^* \geq w(P_K^r) \quad (\text{or } f^* \leq w(P_K^r)), \tag{4.6}$$

where f^* is the value of the current incumbent (any solution available by a heuristic method, e.g., [N3], or occasionally even $(\underline{c}_1,\ldots,\underline{c}_n)$ (or $(\bar{c}_1,\ldots,\bar{c}_n)$) is used as the first incumbent), then it is certainly good enough to dismiss (P_K) from further consideration. Because (P_K^r) is used for solving (P_K) as mentioned above, (P_K^r) must be clearly easier to solve than (P_K), and $w(P_K^r)$ ought to be as close to $w(P_K)$ as possible.

The first relaxation to consider is to drop all integral requirements on the variables. This is the most popular for integer linear programming, but insufficient for integer nonlinear programming, since the resulting problem becomes a multi-constrained nonlinear programming problem which is yet difficult to solve.

Three practical relaxation techniques are presented.

(i) If all constraints except a specified constraint are omitted, the problem becomes far easier to solve. Problem (P_K) is relaxed into the following problem.

Problem (RP1):

$$\min_{j=1,\ldots,s} \max \quad (\text{or } \max_{j=s+1,\ldots,m} \min)$$

$$\{f(X) \mid g_j(X) \leq b_j,\ x_i = \alpha_i\ (i \in L^r),\ x_i \text{ real } (i \in L^f)\}.$$

(ii) For the separable function case, only. It relaxes (P_K) into the following linear programming problem by linearizing everything.

Problem (RP2): maximize (or minimize)

$$\sum_{i \in L^f} \sum_{t=0}^{\Delta_i} \xi_{it} f_i(\underline{x}_i + t) + \sum_{i \in L^r} f_i(\alpha_i)$$

subject to

$$\sum_{i \in L^f} \sum_{t=0}^{\Delta_i} \xi_{it} g_{ji}(\underline{x}_i + t) \leq b_j - \sum_{i \in L^r} g_{ji}(\alpha_i)$$

$$(j = 1, \ldots, m), \quad \sum_{t=0}^{\Delta_i} \xi_{it} = 1 \quad (i \in L^f).$$

$$\xi_{it} \geq 0 \quad (i \in L^f),$$

where $\Delta_i = \bar{x}_i - \underline{x}_i$.

(iii) Maximize the objective function subject to simple bounds on the free variables.

Problem (RP3): maximize (or minimize) $f(X)$

subject to $x_i = \alpha_i \quad (i \in L^r)$,

$$\underline{x}_i \leq x_i \leq \bar{x}_i; \; x_i \text{ integer } (i \in L^f).$$

This problem is quite easy to solve.

The efficiency or each relaxation varies with the nature of the problem. RP1 is useful for linear-constraint parallel redundant problems; RP2 is useful for nonlinear-constraint separable problems; RP3 is useful for nonseparable problems. If the problem has a few variables, RP3 is appropriate in most cases. In general, the easier (P_K^r) is to solve, the greater is the gap between the original and relaxed problem.

4.3.3 Bounds for Variables

Suppose that the enumeration is at v_κ. Current upper and lower bounds for a free variable x_k ($k \in L^f$) are obtained as follows.

Let $\bar{x}_k^1$, $\underline{x}_k^1$ and $\bar{x}_k^2$ be

$$\bar{x}_k^1 = \max\{x_k \in I \mid g_j(X) \leq b_j \quad (j = 1,\ldots,s),$$
$$x_i = \alpha_i \ (i \in L^r),\ x_i = \underline{x}_i \ (i \in L^f,\ \neq k)\}, \tag{4.7}$$

$$\underline{x}_k^1 = \min\{x_k \in I \mid g_j(X) \leq b_j \quad (j = s+1,\ldots,m),$$
$$x_i = \alpha_i \ (i \in L^r),\ x_i = \bar{x}_i \ (i \in L^f),\ \neq k)\}, \tag{4.8}$$

$$\bar{x}_k^2 = \min\{x_k \in I \mid g_j(X) \leq b_j \quad (j = s+1,\ldots,m),$$
$$x_i = \alpha_i \ (i \in L^r),\ x_i = \underline{x}_i \ (i \in L^f,\ \neq k)\}. \tag{4.9}$$

where I is the set of nonnegative integers. Then we obtain current upper and lower bounds; for maximization,

$$\left.\begin{aligned} \bar{x}_k^c &= \max \\ \underline{x}_k^c &= \min \end{aligned}\right\} \{\beta \in I \mid w(P_j^r(x_k = \beta)) > f^*,\ \underline{x}_k^1 \leq \beta \leq \bar{x}_k^1\}, \tag{4.10}$$

where $P_\kappa^r(x_k = \beta)$ is Problem (P_κ^r) with the restriction $x_k = \beta$, i.e., Problem (P_κ^r) that is set as $k \in L^r$ and $\alpha_k = \beta$: and for minimization,

$$\left.\begin{aligned} \bar{x}_k^c &= \max \\ \underline{x}_k^c &= \min \end{aligned}\right\} \{\beta \in I \mid w(P_j^r(x_k = \beta)) < f^*,$$
$$\underline{x}_k^1 \leq \beta \leq \min\{\bar{x}_k^1,\ \bar{x}_k^2\}\}. \tag{4.11}$$

4.3.4 Separation

Restrictions corresponding to the edges from v_κ to its immediate successors determine subproblems into which Problem (P_κ) is separated. The variable x_k fixed by the restrictions is selected out of the set L^f. The manner of this selection greatly influences the efficiency of enumeration.

It is desirable to increase the number of vertices dismissed from consideration, owing to (4.6). Relaxed problems in the higher level are easier to solve, since they have fewer variables. Hence, it is appropriate to select fixed variables such that the number of vertices belonging to the lower level is as small as possible.

Considering the above, we fix variables beforehand in the order of increasing difference between both bounds for variables at v_0, as seen in Section 4.3.5 (Step 3). The efficiency of this separation is mentioned in Section 4. Cabot [C1] has presented such a sepatation rule for the knapsack problem. McLeavey [M3] used Cabot ordering for a problem having linear constraints. Our separation technique extends their linear case to the nonlinear case.

4.3.5 Overall Computational Procedure

For maximization (or minimization) problem, the overall computational procedure is

Step 1: Determine the relaxation technique to be used: set

$$f^* = f(X^*),$$

where X^* is an initial incumbent, i.e., $(\underline{c}_1,\underline{c}_2,\ldots,\underline{c}_n)$ (or $(\bar{c}_1,\bar{c}_2,\ldots,\bar{c}_n)$) or a nearoptimal solution obtained by a heuristic method, e.g., [N3]. Let $L^r = \phi$, i.e., $L^f = \{1,2,\ldots,n\}$,

and $\underline{x}_i = \underline{c}_i$, $\bar{x}_i = \bar{c}_i$ for all $i \in L^f$.

Step 2: Calculate $\bar{x}_i^1$ for all $i \in L^f$ using (4.7) and reset $\bar{x}_i^1 \to \bar{x}_i$ for all i. Next calculate $\underline{x}_i^1$ for all i using (4.8) and reset $\underline{x}_i^1 \to \underline{x}_i$ for all i; or, in addition, calculate $\bar{x}_i^2$ for all i using (4.9) and reset $\min\{\bar{x}_i^1, \bar{x}_i^2\} \to \bar{x}_i$ for all i. Last calculate $\bar{x}_i^c$, $\underline{x}_i^c$ for all i using (4.10) or (4.11), and reset $\bar{x}_i^c \to \bar{x}_i$, $\underline{x}_i^c \to \underline{x}_i$ for all i.

Step 3: Let $\{(1), (2), \ldots, (\ell), \ldots, (n)\}$ be a permutation of $\{1, 2, \ldots, i, \ldots, n\}$ such that

$$\bar{x}_{(1)} - \underline{x}_{(1)} \leq \bar{x}_{(2)} - \underline{x}_{(2)} \leq \cdots \leq \bar{x}_{(n)} - \underline{x}_{(n)}.$$

And set $\ell = 1$.

Step 4: Set $\bar{x}_{(\ell)}^c \to x_{(\ell)}$ (or $\underline{x}_{(\ell)}^c \to x_{(\ell)}$), and reset $\{(1),(2),\ldots,(\ell)\} \to L^r$, $\ell + 1 \to \ell$. If $\ell < n$, go to Step 5. Otherwise, go to Step 6.

Step 5: Calculate $\bar{x}_{(\ell)}^c$, $\underline{x}_{(\ell)}^c$ using (4.7)-(4.11). If $\bar{x}_{(\ell)}^c$, $\underline{x}_{(\ell)}^c$ exist, go to Step 4. Otherwise go to Step 8.

Step 6: Calculate $\bar{x}_{(n)}^1$, $\underline{x}_{(n)}^1$, and set $\bar{x}_{(n)}^1 \to x_{(n)}$ (or $\underline{x}_{(n)}^1 \to x_{(n)}$).

Step 7: If $f(X)$ obtained is not greater (or not less) than f^*, then go to Step 8. Otherwise, record X as a new incumbent X^* and reset $f(X) \to f^*$, and go to Step 8.

Step 8: Find

$$\hat{\ell} = \max\{\ell' \mid x_{(\ell')} > \underline{x}_{(\ell')}^c, \ \ell' = 1,2,\ldots,\ell-1\};$$

or invert the inequality and substitute $\bar{x}^c_{(\ell')}$ for $x^c_{(\ell')}$. If $\hat{\ell}$ exists, reset $\hat{\ell} + 1 \to \ell$, $x_{(\hat{\ell})} - 1 \to x_{(\hat{\ell})}$ (or change -1 to +1). If $\ell < n$, go to Step 5, and if $\ell = n$, go to Step 6. If $\hat{\ell}$ does not exist, then stop; the present incumbent x^* is optimal.

For the separable problem in Section 4.2, record the following values at each level ℓ,

$$f^{\ell} = \sum_{i=1}^{\ell} f_{(i)}(x_{(i)}), \tag{4.12}$$

$$g_j^{\ell} = \sum_{i=1}^{\ell} g_{j(i)}(x_{(i)}) \quad (j = 1,\ldots,m). \tag{4.13}$$

The relations between two levels ℓ and $\ell + 1$ can be written as

$$f^{\ell+1} = f^{\ell} + f_{(\ell+1)}(x_{(\ell+1)}), \tag{4.14}$$

$$g_j^{\ell+1} = g_j^{\ell} + g_{j(\ell+1)}(x_{(\ell+1)}) \quad (j = 1,\ldots,m). \tag{4.15}$$

With use of the recorded values and these relations we can shorten the computation time.

Some reliability problems will not have the monotonicity assumed in Problem (P) but in most cases the feasible domain of solution is restricted mainly by monotone constraints, thus they may be treated by extending the method for solving (P).

4.4 NUMERICAL EXAMPLES AND DISCUSSIONS

Three types of numerical examples are considered in this section.

4.4.1 Example 1

A problem of Tillman [T7] is an example of minimizing the cost of a system having series redundant components subject to several separable nonlinear constraints while maintaining an acceptable level of reliability, where each component has several modes of failure.

When Tillman [T7] solved this example, he used wrong data on the objective function and the system reliability constraint [T7, Fig. 1], yielding a solution different from the exact one of this example. The solution in [T7] is the exact solution of a different problem. References [G7,G8] reported that a new optimal solution was found, by comparing a solution produced by Gen et al. with a solution by Tillman's method [T7] had solved a different problem.

Example 1: minimize $f(X) = 5\,\ell n(x_1+1) + x_2^2 + x_3\, e^{x_3}$

subject to

$$g_1(X) = (x_1 + 3)^2 + x_2^2 + (x_3 + 2)^2 \leq 65,$$

$$g_2(X) = \ell n[(1 - 0.01)^{x_1+1} - 0.04^{x_1+1} - 0.05^{x_1+1} - 0.02^{x_1+1}]$$
$$+ \ell n[(1 - 0.02)^{x_2+1} - 0.08^{x_2+1} - 0.02^{x_2+1} - 0.05^{x_2+1}]$$
$$+ \ell n[(1 - 0.08)^{x_3+1} - 0.01^{x_3+1} - 0.05^{x_3+1} - 0.01^{x_3+1}]$$
$$\geq -0.301,$$

$$g_3(X) = 20(x_1 + e^{-x_1}) + 20(x_2 + e^{-x_2}) + 20(x_3 + e^{-x_3}) \geq 120,$$

$$g_4(X) = 20\, x_1\, e^{-x_1/4} + 20\, x_2\, e^{-x_2/4} + 20\, x_3\, e^{-x_3/4} \geq 50.$$

$g_2(X)$ is not monotonic, but becomes so, if $c_1 = 1$ and $c_2 = 1$; the solution satisfying the constraint $g_2(X) \geq -0.301$ does not exist when $x_1 = 0$ or $x_2 = 0$. We can rewrite this problem as follows, corresponding to Problem (P).

$$\text{minimize} \quad f(X) = \sum_{i=1}^{3} f_i(x_i)$$

$$\text{subject to} \quad \sum_{i=1}^{3} g_{ji}(x_i) \leq b_j \quad (j = 1,\ldots,4),$$

$$1 \leq x_1 < \infty,$$

$$1 \leq x_2 < \infty,$$

$$0 \leq x_3 < \infty,$$

$$x_i \text{ integer},$$

where the data of the functions are given in Table 4.1.

The present method provides the exact solution easily with hand-calculation (pencil and paper), using RP3 as a relaxation technique and adopting $f^* \to \infty$ as an initial value. Following the procedure (see Appendix A), we get Table 4.2 and Fig. 4.2. Six feasible solutions are enumerated; the procedure produces the exact solution (3,2,0), and then $f(X) = 10.93$.

Gen et al. [G7,G8] have reported that [G6,M8,T7] have transformed Example 1 with size 3 × 4('the number of variables' × 'the number of

constraints') into the 0-1 linear programming problem with size 12 × 8 for [G6], size 15 × 16 for [T7], and into the 0-1 nonlinear programming problem with size 7 × 4 for [M8].

Gen and Tillman have enumerated several tens of solutions, which include non-feasible solutions. Misra has enumerated 15 solutions. It is reported that Gen required 1.9 seconds of CPU time and Tillman required 14.9 sec both on the digital computer NEAC 2200/500, and Misra required 8.9 sec on NEAC 2200/550.

Table 4.1

Input Data of Example 1

x_i	f_1	f_2	f_3	g_{11}	g_{12}	g_{13}
0	0	0	0	9	0	4
1	3.47	1	2.72	16	1	9
2	5.49	4	14.78	25	4	16
3	6.93	9	60.26	36	9	25
4	8.05	16	218.39	49	16	36
5	8.96	25	742.07	64	25	49

x_i	g_{21}	g_{22}	g_{23}	$g_{3i}(i=1,2,3)$	$g_{4i}(i=1,2,3)$
0	.1278	.1863	.1625	-20.0	0.0
1	.0247	.0501	.1700	-27.4	-15.6
2	.0303	.0613	.2503	-47.7	-24.3
3	.0402	.0809	.3335	-61.0	-28.3
4	.0503	.1010	.4169	-80.4	-29.4
5	.0603	.1212	.5008	-100.1	-28.7

$b_1=65$, $b_2=0.301$, $b_3=-120$, $b_4=-50$

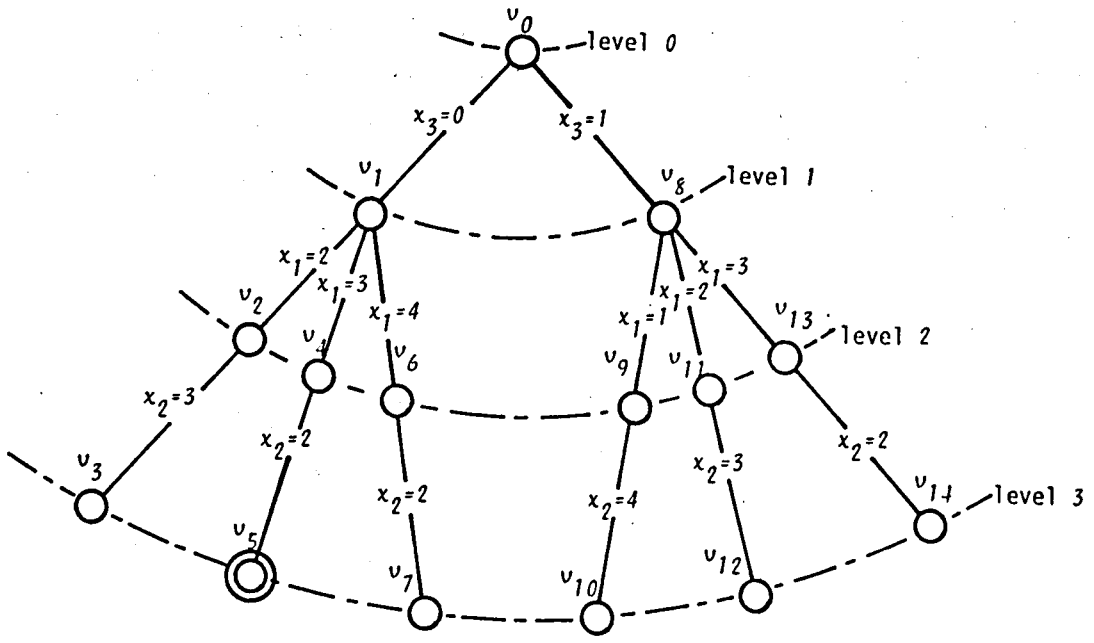


Fig. 4.2 Tree for Example 1

Table 4.2

Enumerated Solutions for Example 1

x_3	$\bar{x}_1^1$	$\underline{x}_1^1$	$\bar{x}_1^2$	$\bar{x}_1^c$	$\underline{x}_1^c$	x_1	$\bar{x}_2^1$	$\underline{x}_2^1$	x_2	f
0	4	2	4	4	2	2	4	3	3	14.49
						3	3	2	2	10.93(Opt.Sol.)
						4	3	2	2	12.05
1	4	1	4	3	1	1	4	4	4	22.19
						2	3	3	3	17.21
						3	3	2	2	13.65

4.4.2 Example 2

Misra [M17] improved on the method [M8] with an appropriate adoption of initial solution. Then he presented the following example of maximizing the reliability of a system having 1-out-of-m:G redundancy in stage 1, stand-by redundancy in stages 2 and 3, and 2-out-of-m:G redundancy in stage 4, subject to a linear constraint.

Example 2: maximize

$$\ln R_s(X) = \ln(1 - 0.2^{x_1}) + \ln[0.6 \sum_{j=0}^{x_2-1} \frac{(-\ln 0.6)^j}{j!}]$$

$$+ \ln[0.7 \sum_{j=0}^{x_3-1} \frac{(-\ln 0.7)^j}{j!}] + \ln[\sum_{j=2}^{x_4} \binom{x_4}{j} (0.75)^j (1-0.75)^{x_4-j}]$$

subject to $2.5x_1 + 2.5x_2 + 3.5x_3 + 3.0x_4 \leq b$,

$1 \leq x_1, x_2, x_3$,

$2 \leq x_4$,

x_1, x_2, x_3, x_4 integers.

For b = 23,25,27,30,33,36, Misra [M17] produced exact solutions by using the Lawler-Bell algorithm [L1] with an initial solution (1,1,1,2), see Table 4.3. The present method produced exact solutions, using RP3 as a relaxation technique and adopting X* = (1,1,1,2). Table 4.4 shows that the number of enumerated solutions of [M17] is about 10 times that of the present method, in fact exceeding the number of feasible solutions. For enumerating each solution, the method [M17] requires more complex calculation than the present method. Thus the present method is more efficient than Misra's. The results using good initial

Table 4.3

Exact solutions for Example 2

Example	b	Optimal reliability	Optimal solution	Resources used
2-a	23	0.5140	2,2,1,3	22.5
2-b	25	0.5811	1,2,2,3	23.5
2-c	27	0.6973	2,2,2,3	26.0
2-d	30	0.7845	2,2,2,4	29.0
2-e	33	0.8522	2,3,2,4	31.5
2-f	36	0.8922	2,3,3,4	35.0

Table 4.4

Comparison between the Number of Solutions Enumerated by the Present Method and that by Misra's Method

Example	Number of feasible solutions	Number of enumerated solutions	
		Present method	Misra's method
2-a	27	10 + 0*	82**
2-b	44	13 + 2	163
2-c	71	9 + 2	217
2-d	131	20 + 2	276
2-e	224	26 + 5	342
2-f	355	38 + 7	479

* 'number of enumerated complete solutions' + 'number of enumerated partial solutions'

** the data in [M17]

solutions are shown in Table 4.5. This table shows almost the same tendency as that described above. Generally speaking, the Lawler-Bell algorithm is barely suitable for problems whose feasible domain of solution is restricted mainly by monotone constraints.

Table 4.5

Comparison between the Number of Solutions Enumerated by the Present Method and that by Misra's method (with good initial solutions)

Example	Initial reliability	Number of enumerated solutions	
		Present method	Misra's method
2-b	0.5811	12 + 2*	84**
2-c	0.6599	4 + 1	81
2-d	0.7654	8 + 2	60
2-e	0.8437	4 + 4	79
2-f	0.8788	9 + 3	88

* 'number of enumerated complete solutions' + 'numberof enumerated partial solutions'

** the data in [M17]

4.4.3 Example 3

This is the problem of maximizing reliability of an n-stage system through 1-out-of-m:G redundancy subject to multiple linear constraints:

$$\text{maximize} \quad f(x) = \prod_{i=1}^{n} \{1 - (1 - r_i)^{x_i}\}$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_{ji}x_i \leq b_j \quad (j = 1,\ldots,s),$$

$$x_i \geq 1 \text{ integer.}$$

Example 3-a [L3] has 15 stages and 2 constraints (see Table 4.6); *Example* 3-b has 30 stages and 3 constraints (see Table 4.7).

It is effective to use as an initial incumbent a solution obtained by the approximate method [N3] and to adopt RP1. The upper value of the relaxation problem can easily be calculated by using an approximate equation (see Appendix E) as follows:

$$\min_{j=1,\ldots,m} \{f(X) \mid x_i = \alpha_i \ (i \in L^r),\ x_i = \gamma_{ji} \ (i \in L^f)\}, \tag{4.16}$$

$$q_i = 1 - r_i \ , \tag{4.17}$$

$$z_{ji} = - a_{ji}/\ln q_i \ , \tag{4.18}$$

$$b'_j = b_j - \sum_{i \in L^r} a_{ji}\alpha_i \ , \tag{4.19}$$

$$\gamma_{ji} = \frac{z_{ji}}{a_{ji}} \ln\{1 + \frac{1}{z_{ji}} \exp\left(\frac{b'_j + \sum_{i \in L^f} z_{ji} \ln z_{ji}}{\sum_{i \in L^f} z_{ji}} \right)\}. \tag{4.20}$$

This value is very close to the optimal value of the problem (P_K) in which the integrality constraints are dropped.

The total execution time ('time spent on finding an initial incumbent by using method [N3]' + 'time spent on finding the exact solution by using the present method') on FACOM 230/75 digital computer was 4.9 (=0.7 + 4.2) sec for Example 3-a, and was 109 (= 3 + 106) sec for Example 3-b. The exact solution for Example 3-a is

Table 4.7

Data of Example 3-a

Stage No.	i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reliability	r_i	.90	.75	.65	.80	.85	.93	.78	.66	.78	.91	.79	.77	.67	.79	.67
Cost	a_{1i}	5	4	9	7	7	5	6	9	4	5	6	7	9	8	6
Weight	a_{2i}	8	9	6	7	8	8	9	6	7	8	9	7	6	5	7

$b_1=400$, $b_2=414$

Table 4.6

Data of Example 3-b

Stage No.	i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Reliability	r_i	.90	.75	.65	.80	.85	.93	.78	.66	.78	.91	.79	.77	.67	.79	.67
		.94	.73	.79	.68	.98	.90	.86	.95	.92	.83	.97	.89	.99	.88	.98
Cost	a_{1i}	5	4	9	7	7	5	6	9	4	5	6	7	9	8	6
		4	3	9	7	4	9	8	6	3	4	5	7	6	8	7
Weight	a_{2i}	8	9	6	7	8	8	9	6	7	8	9	7	6	5	7
		8	4	9	3	9	5	3	4	5	2	6	1	10	7	6
Volume	a_{3i}	2	4	10	1	5	5	4	8	8	10	7	3	1	2	4
		12	6	5	4	3	5	9	2	5	7	8	6	3	12	5

$b_1=750$, $b_2=850$, $b_3=900$

$$X = (3, 4, 6, 4, 3, 2, 4, 5, 4, 2, 3, 4, 5, 4, 5),$$

$$f(X) = 0.9456134$$

(Luus' approximate method [L3] did not produce the exact solution for Example 3-a). The exact solution for Example 3-b is

$$X = (3, 5, 6, 4, 4, 3, 5, 6, 5, 3, 4, 5, 5, 4, 6, 3, 6, 4,$$
$$6, 2, 3, 4, 3, 3, 4, 2, 3, 2, 3, 2),$$

$$f(X) = 0.9697964.$$

When one wants to evaluate the efficiency of a specific integer programming method, the rate of increase in times as problem size increases is particularly interesting, since the computation time increases exponentially. The execution time for Example 3-b is 22.4 times that for Example 3-a. This is really quite small, if we notice that the number of feasible solutions of example 3-b is more than $5^{15} = 3 \times 10^{10}$ times that of Example 3-a. McLeavey [M3] has indicated that, for his method, the average execution time for problems with size 16×2 is 100 times that for problems with size 7×2.

The efficiency of a branch-and-bound method is mainly influenced by bounds for variables produced by the bounding technique. The bounds produced by using the relaxation RP3 are almost the same as the bounds used in Henin's [H5] of McLeavey's method [M3,M4]. RP1, which has been used in Example 3-a, produces much better bounds than RP3; this is true particularly when the number of variables increases.

Tillman's [T7], Misra's [M8] and Gen's [G6] methods are unable to yield the exact solution for a large problem such as Example 3-a and 3-b. McLeavey's method can not solve the nonlinear constraint problem such as Example 1. The present procedure, however, can efficiently provide exact solutions for various problems by selecting an appropri-

ate relaxation technique and adopting a good initial incumbent.

To evaluate the present separation technique, the separation was carried out for Example 3-a by fixing variables in the reverse order. Then the procedure required 27.9 sec of total execution time, that is, the case of the reverse order spent 5.8 times the execution time of the right order.

Chapter 5

A METHOD FOR A MULTIOBJECTIVE PROBLEM

5.1 INTRODUCTION

Many papers have treated reliability optimization problems with a single objective function, all variables of which are integers. These problems are formulated by selecting the most important or appropriate property as an objective function from multiple properties (criteria or objectives), e.g., unavailability, unreliability, cost, weight, volume, and by treating the other properties as the constraints. Evans [E1] points out that the regions around optimum points are important. This suggests that it is best to achieve a best balance among all properties rather than to optimize one property, i.e., find a best (best-compromise) alternative for system designers (decision makers) in the feasible region of trade-offs among all properties. Multiobjective programming is a way to do this.

There have been more than 100 papers, dealing with multiobjective programming problems, and at least 20 different solution techniques have been proposed [C2]. However, few papers have treated multiobjective programming problems including integer variables. Sakawa [S1] and Inagaki et al. [I1] deal with multiobjective mixed-integer programming problems but use solution techniques for continuous variables. Bitran [B10] treats a linear multiobjective programming problem with zero-one variables. No paper treats multiobjective nonlinear pure-integer programming problems, and no method for solving

them exists.

Multiobjective pure-integer programming problems differ essentially from multiobjective programming problems with continuous variables. The former problems usually have finite number of Pareto-optimal (noninferior or undominated) solutions (defined in Section 5.3.2). But the latter problems, however small the problems are, have infinite number of Pareto-optimal solutions, except for the special case that the problem has no feasible solution or only one Pareto-optimal solution.

The present method for solving multiobjective nonlinear pure-integer programming problems consists of three techniques:

(i) Narrowing a given feasible region if the region is too wide (Section 5.2).

(ii) Obtaining exactly the set of Pareto-optimal solutions (Section 5.3).

(iii) Selecting a best-compromise solution for system designers from the set of Pareto-optimal solutions (Section 5.4).

Technique (i) is based on Evans' view mentioned above. System designers find an acceptable solution X^a, which is a final solution obtained by conventional methods, and narrow the feasible region such that it is arround X^a, since a best-compromise solution is probably around X^a. Technique (ii) has two parts:

a) Generating all feasible solutions.

b) Obtainig the set of Pareto-optimal solutions from the set of feasible solutions.

For a, we modify the Nakagawa et al. method [N4] based on branch-and-bound so as to generate all feasible solutions. For b, we obtain the set of Pareto-optimal solutions by comparing two feasible solutions

in all combinations. Technique (iii) is simple and uses paired-comparisons [D1].

The present method has an interesting merit that an increase in the number of objective functions only slightly affects the size of problem, in contrast to the fact that vector optimization problems with continuous variables which require much more computation time with increasing number of objective functions.

5.2 PROBLEM AND TECHNIQUE FOR NARROWING A FEASIBLE REGION

Consider a system having m properties (e.g., unavailability, unreliability, cost, weight, volume). All properties are assumed to be better when they are smaller (e.g., use unreliability rather than reliability).

In the field of reliability optimization, it is desirable to treat all properties as objective functions, since by trading off all properties system designers can find a 'best'-compromise alternative in a given feasible region; 'best' means best for the system designers.

We consider the following vector optimization problem where all important properties are in the objective functions.

$$\begin{aligned} \text{Problem A:} \quad & \text{minimize} && f(x) \\ & \text{subject to} && f_j(x) \leq \bar{b}_j \quad (j \in M^{sub}), \\ & && x = (x_1, \ldots, x_n) \\ & && \underline{c}_i \leq x_i \leq \bar{c}_i \quad \text{integer}, \end{aligned}$$

where

$f_j(X)$ $\begin{cases} \text{monotone nondecreasing function for } j=1,\ldots,s, \\ \text{monotone nonincreasing function for } j=s+1,\ldots,t, \\ \text{non-monotone function for } j=t+1,\ldots,m. \end{cases}$

$f(X)$ $\{f_1(X),\ldots,f_m(X)\}$.

$\bar{b}_j$ upper bound of property $f_j(X)$; $j \in M^{sub}$.

M^{sub} a subset of M.

M $\{1,\ldots,M\}$.

In real-world problems, a feasible region of the problem is often too wide and includes too many Pareto-optimal solutions (defined in Section 5.3.2). In such case, system designers can narrow the feasible region reasonably by finding an 'acceptable' feasible solution X^a of problem A. A solution is 'acceptable' if so judged by system designers from their experience and the analysis of data.

An acceptable solution X^a can analytically be obtained by considering appropriate scalar optimization problems. Two examples are given.

(i) System designers make a single objective problem by selecting one property as an objective function and using some properties as constraints. The exact or approximate solution $\hat{X}$ of this problem can be obtained by using an existing method, e.g., [N3,N4]. If $\hat{X}$ is not acceptable for system designers, they make some other single objective problem after reconsidering the valures $f_1(\hat{X}),\ldots,f_j(\hat{X})$, and solve it again. This is repeated until they get an acceptable solution.

(ii) It is well known that trade offs among two properties can be done easily by using dynamic programming (DP). System desingers select the most important two properties and find an acceptable solution X^a by DP.

An acceptable solution x^a is equivalent to a best alternative which system designers have been conventionally finding.

To obtain a narrowed feasible region including a best-compromise solution, which probably exists around x^a, system designers set new constraints b_j by giving values d_j based on their experiences and data;

$$b_j = f_j(x^a) + d_j \qquad (j \in M), \tag{5.1}$$

where $0 \leq d_j \leq \bar{b}_j - f_j(x^a)$ for $j \in M^{sub}$, and $0 \leq d_j$ for $j \in M$, $\notin M^{sub}$. The obtained feasible region

$$X = \{x \mid f_j(X) \leq b_j \ (j \in M) \text{ and } x_i \in \{\underline{c}_i, \underline{c}_i+1, \ldots, \bar{c}_i\} \ (i \in N)\}, \tag{5.2}$$

is a region around x^a, as illustrated in Fig. 5.1, where $N = \{1, \ldots, n\}$.

The following vector optimization problem is obtained, including all properties not only as the objective function but also as constraints.

Problem B: minimize $f(X)$

subject to $X \in X$.

5.3 TECHNIQUE FOR OBTAINING THE SET OF PARETO-OPTIMAL SOLUTIONS

The method for solving problem A or B, has two parts: generating the set of feasible solutions, and finding the set of Pareto-optimal solutions from the feasible solution set. The feasible solution set can be obtained by using branch-and-bound, which enumerates all feasible solutions explicitly or implicitly. By comparing two feasible solution in all combinations, we can easily

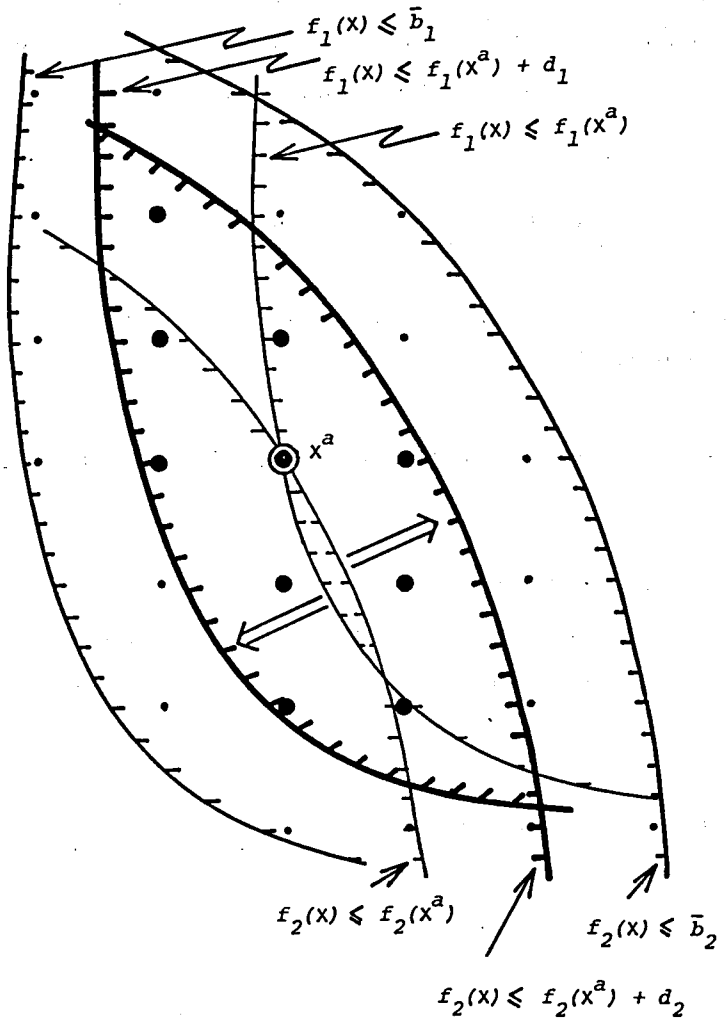


Fig. 5.1 A narrowed feasible region.
● means feasible solution $x \in X$.

get the set of Pareto-optimal solutions.

5.3.1 Procedure for Feasible Solutions

Consider the following problem, which is common to Problems A and B.

(Feasible Region Problem): Find the set of the feasible solutions

$$\text{subject to} \quad f_j(X) \leq b_j \quad (j \in M^1 \cup M^2 \cup M^3),$$

$$\underline{c}_i \leq x_i \leq \bar{c}_i \quad \text{integer},$$

where M^1, M^2, M^3 are the sets of indices of monotone nondecreasing, monotone nonincreasing and nonmonotone function respectively; i.e., $M^1 \cup M^2 \cup M^3 = M^{sub}$ or M.

We modify the procedure in section 4.3.5 to generate all feasible solutions; further information can be found in Chapter 4 (especially Section 4.3.1). The procedure is as follows:

(Procedure 1)

Step 1: Set $L^r = \phi$ and $L^f = N$, where

$$L^r \cup L^f = N \,. \tag{5.3}$$

And set $\underline{c}_k \rightarrow \underline{x}_k$, $\bar{c}_k \rightarrow \bar{x}_k$ for all $k \in L^f$.

Step 2: Calculate $\bar{x}_k^c$ for all $k \in L^f$ using (5.4) and reset $\bar{x}_k^c \rightarrow \bar{x}_k$.

$$\bar{x}_k^c = \max \{x_k \in \{\underline{x}_k, \underline{x}_k+1, \ldots, \bar{x}_k\} \mid f_j(X) \leq b_j \ (j \in M^1),$$

$$x_i = \underline{x}_i \ (i \in L^f, \neq k)\}. \quad (5.4)$$

Next calculate $\underline{x}_k^c$ for all $k \in L^f$ using (5.5) and reset $\underline{x}_k^c \to \underline{x}_k$.

$$\underline{x}_k^c = \min \{x_k \in \{\underline{x}_k, \underline{x}_k+1, \ldots, \bar{x}_k^c\} \mid f_j(X) \leq b_j \ (j \in M^2),$$

$$x_i = \bar{x}_i \ (i \in L^f, \neq k)\}. \quad (5.5)$$

Step 3: Let $\{(1),(2),\ldots,(\ell),\ldots,(n)\}$ be a permutation of $\{1,2,\ldots,i,\ldots,n\}$ such that

$$\bar{x}_{(1)} - \underline{x}_{(1)} \leq \bar{x}_{(2)} - \underline{x}_{(2)} \leq \cdots \leq \bar{x}_{(n)} - \underline{x}_{(n)}. \quad (5.6)$$

And set $1 \to \ell$.

Step 4: Set $\bar{x}_{(\ell)}^c \to x_{(\ell)}$ and reset $L^r = \{(1),(2),\ldots,(\ell)\}$, i.e., $L^f = \{(\ell+1),\ldots,(n)\}$. If $\ell = n$, go to Step 6. Otherwise, set $\ell+1 \to \ell$ and go to Step 5.

Step 5: Calculate $\bar{x}_{(\ell)}^c$ and $\underline{x}_{(\ell)}^c$ using (5.4) and (5.5), respectively. If $\bar{x}_{(\ell)}^c$ and $\underline{x}_{(\ell)}^c$ exist, return to Step 4. Otherwise go to Step 7.

Step 6: If some of the obtained solutions $(x_1,x_2,\ldots,\bar{x}_{(n)}^c,\ldots,x_n)$, $(x_1,x_2,\ldots,\bar{x}_{(n)}^c-1,\ldots,x_n),\ldots,(x_1,x_2,\ldots,\underline{x}_{(n)}^c,\ldots,x_n)$ satisfy the constraints $f_j(X) \leq b_j \ (j \in M^3)$, then record the solutions as being feasible.

Step 7: Find ℓ^* such that

$$\ell^* = \max \{\ell' \mid x_{(\ell')} > \underline{x}^c_{(\ell')} \quad (\ell'=1,2,\ldots,\ell-1)\}. \tag{5.7}$$

If ℓ^* exists, reset $\ell^*+1 \rightarrow \ell$, $x_{(\ell^*)}-1 \rightarrow x_{(\ell^*)}$ and return to Step 5. If ℓ^* does not exist, then stop.

5.3.2 Procedure for Pareto-Optimal Solutions

A feasible solution x^k is Pareto-optimal if there does not exist a feasible solution x^ℓ such that

$$f_j(x^\ell) \leq f_j(x^k), \quad \text{for all } j \in M \tag{5.8}$$

and

$$f_j(x^\ell) \neq f_j(x^k), \quad \text{at least for one } j \in M. \tag{5.9}$$

For example if we compare two feasible solutions x^1, x^2, then if x^1 is inferior to x^2 in all $f_j(x)$, x^1 is not Pareto-optimal. Comparing two feasible solutions in all combinations, we can obtain the set of Pareto-optimal solutions. The maximum number of comparisons is $r(r-1)/2$, where r is the number of feasible solutions. A computer oriented procedure for obtaining Pareto-optimal solutions is as follows:

(Procedure 2)

Step 1: Rearrange $\{f_1(x),\ldots,f_m(x)\}$ in order of importance of properties. Let $\{f_{(1)}(x),\ldots,f_{(m)}(x)\}$ be the permutation obtained. This step becomes useful when system designers select a best-compromise solution by using Procedure 3.

Step 2: Set $X = \{x^1, x^2, \ldots, x^r\}$, where $x^1, \ldots, x^r$ are feasible solutions obtained in Section 5.3.1. Rearrange $\{x^1, x^2, \ldots, x^r\}$ in order of increasing value of $f_{(1)}(x)$. If $f_{(1)}(x^\mu) = f_{(1)}(x^\nu)$ ($\mu, \nu \in \{1, \ldots, r\}$), find j' such that $f_{(j')}(x^\mu) \neq f_{(j')}(x^\nu)$ and arrange x^μ, x^ν in order of incresing value of $f_{(j')}(x)$. When $f_{(j)}(x^\mu) = f_{(j)}(x^\nu)$ for all $j \in M$, remove either x^μ or x^ν from X. Let $X^p = \{x^{(1)}, x^{(2)}, \ldots, x^{(r')}\}$ be the permutation obtained. Set $1 \to k$, $1 \to \ell$.

Step 3: Reset $\ell+1 \to \ell$. If $\ell > r'$, go to Step 5. Otherwise go to Step 4.

Step 4: If $x^{(\ell)} \notin X^p$, return to Step 3. Otherwise calculate $h(x^{(k)}, x^{(\ell)})$, where

$$\text{sign}\{y\} = \begin{cases} 1, & y \geq 0 \\ -1, & y < 0 \end{cases} \tag{5.10}$$

$$h(x^{(k)}, x^{(\ell)}) = \sum_{j \in \{2, \ldots, m\}} \text{sign}\{f_{(j)}(x^{(\ell)}) - f_{(j)}(x^{(k)})\}. \tag{5.11}$$

When $h(x^{(k)}, x^{(\ell)}) = m - 1$, remove $x^{(\ell)}$ from X^p, since $x^{(\ell)}$ is not Pareto-optimal. Otherwise continue. Return to Step 3.

Step 5: Reset $k+1 \to k$. If $k \neq r'$, go to Step 6. If $k = r'$, then stop. The current X^p is the set of Pareto-optimal solutions.

Step 6: If $x^{(k)} \in X^p$, then return to Step 5. Otherwise reset $k \to \ell$ and return to Step 3.

5.4 TECHNIQUE FOR SELECTING A BEST-COMPROMISE SOLUTION

We can obtain the set of Pareto-optimal solutions by numerical calculation. However it is impossible to select a best-compromise solution from the Pareto-optimal solution set without information about the system designers' preferences.

Many methods, e.g., [G13,R3], exist for selecting a best alternative from multiple alternatives or fixing the rank of multiple alternatives. Most of them can be used for selecting a best-compromise solution from the Pareto-optimal solution set. Roughly speaking, the methods divide into two groups: methods with quantification of preferences among all properties and ones without quantification. Which method is better depends on the experience of system designers, kinds of properties, etc. In real-world decision problems, explicit quantification of preferences among properties is usually very difficult. In such cases, experienced system designers use their knowledge and the results of data analysis to judge which of two alternatives is preferable. If so, a reasonable result can obtain by judging items in pairs. This is called paired comparison, and is practical for decision making.

To select a best-compromise solution from a Pareto-optimal solution set, we consider a simple method, which is a kind of paired comparison. For simplicity, the following two assumptions are made:

(i) The system designers can determine which of two Pareto-optimal solutions $\tilde{x}^k$, $\tilde{x}^\ell$ is preferable; i.e., they can answer the following question: "At the point

$$\max_{t=k,\ell}\{f_{(1)}(\tilde{x}^t)\},\ \max_{t=k,\ell}\{f_{(2)}(\tilde{x}^t)\},\ldots,\ \max_{t=k,\ell}\{f_{(m)}(\tilde{x}^t)\}, \tag{5.12}$$

which do you prefer: A. Decreasing each value of property

such as

$$j \in \{j \in M \mid f_{(j)}(\tilde{x}^k) < f_{(j)}(\tilde{x}^\ell)\} \tag{5.13}$$

to $f_{(j)}(\tilde{x}^k)$, or B. Decreasing each value of property such as

$$j \in \{j \in M \mid f_{(j)}(\tilde{x}^\ell) < f_{(j)}(\tilde{x}^k)\} \tag{5.14}$$

to $f_{(j)}(\tilde{x}^\ell)$?". For example, when $f_{(1)}(\tilde{x}^k) = 0.1$ (unreliability), $f_{(2)}(\tilde{x}^k) = 90$ (weight), $f_{(3)}(\tilde{x}^k) = 70$ (cost), $f_{(1)}(\tilde{x}^\ell) = 0.2$, $f_{(2)}(\tilde{x}^\ell) = 80$, $f_{(2)}(\tilde{x}^\ell) = 60$, the question is "At the point that unreliability, weight and cost are 0.2, 90, and 70, respectively, which is preferable for you, decreasing unreliability to 0.1 or decreasing cost and weight to 80 and 60, respectively?" (see Fig. 5.2).

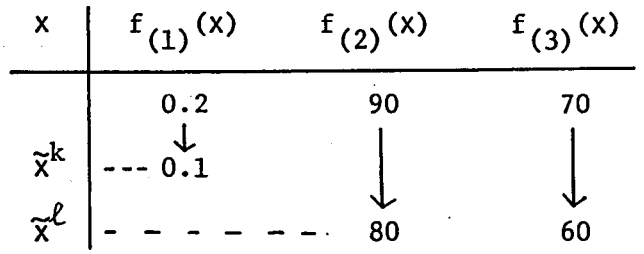


Fig. 5.2. An example of question.

(ii) The effect of order of comparisons, which comes from the system designers' training, fatigue, adaption, etc., is negligible.

A simple procedure for selecting a best-compromise solution from the set of Pareto-optimal solutions is:

(Procedure 3)

Step 1: Let $\tilde{x}^1$, $\tilde{x}^2$,..., $\tilde{x}^p$ be Pareto-optimal solutions obtained in Procedure 2. Set d = 1, k = 2.

Step 2: Let system designers compare $\tilde{x}^d$ and $\tilde{x}^k$, then let the preferable one be $\tilde{x}^d$.

Step 3: If k < p, then reset k + 1 → k and return to Step 2. Otherwise the current $\tilde{x}^d$ is the result.

The resulting $\tilde{x}^d$ can not yet be declared the best-compromise, the decision might be inconsisted. The inconsistency is so called circular triads (see Fig. 5.3).

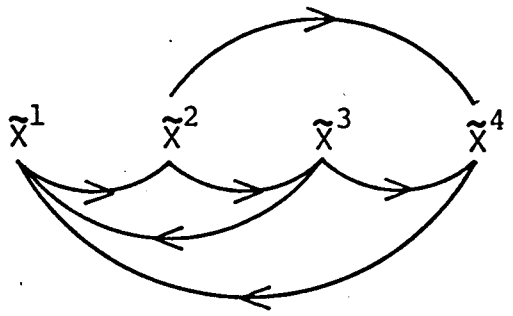


Fig. 5.3 An example including two circular triads. The arrow → means "is not preferred to".

Whether circular triads including $\tilde{x}^d$ exist can be checked by comparing $\tilde{x}^d$ with $\tilde{x}^1$, $\tilde{x}^2$,..., $\tilde{x}^{d-1}$, unless $\tilde{x}^d$ is a Pareto-optimal solution. If $\tilde{x}^d$ is preferable to any of $\tilde{x}^1$, $\tilde{x}^2$,..., $\tilde{x}^{d-1}$, then $\tilde{x}^d$ is a best-compromise solution. Otherwise circular triads exist, then find all circular triads, reexamine the alternatives forming the circular

triads in detail and select an alternative as a best-compromise solution.

5.4 EXAMPLE

Consider a 3-stage system having four properties; reliability, weight, volume, and cost. The system reliability is increased by choosing a more reliable component out of 4 candidates at stage 1, adding redundant components in parallel at stage 2, and usig 2-out-of-(x_3+1):G configuration at Stage 3. The system reliability is

$$R(X) = \prod_{i=1}^{3} R_i(x_i) \ , \tag{5.15}$$

where

$$R_1(x_1) = 0.88,\ 0.92,\ 0.98,\ 0.99 \quad \text{for } x_1 = 1,2,3,4, \text{ respectively,} \tag{5.16}$$

$$R_2(x_2) = 1 - (1 - 0.81)^{x_2}, \tag{5.17}$$

$$R_3(x_3) = \sum_{k=2}^{x_3+1} \binom{x_3+1}{k} 0.77^k \ (1 - 0.77)^{x_3+1-k} . \tag{5.18}$$

The system weight, volume, and cost are:

$$W(X) = 5 + e^{x_1/8} + 3(x_2 + e^{x_2/4}) + 5(x_3 + 1 + e^{x_3/4}) \leq 84, \tag{5.19}$$

$$V(X) = 10 + 8\, x_2\, e^{x_2/4} + 6\, x_3\, e^{x_3/4} \leq 300, \tag{5.20}$$

$$C(X) = 4 \exp\left(\frac{0.02}{1 - R_1(x_1)}\right) + 5x_2 + 2(x_3 + 1). \qquad (5.21)$$

A vector optimization problem is

$$\begin{aligned} \text{minimize} \quad & f(X) = \{1-R(X),\ W(X),\ V(X),\ C(X)\} \\ \text{subject to} \quad & W(X) \leq 84, \\ & V(X) \leq 300, \\ & 1 \leq x_1 \leq 4,\ 1 \leq x_2,\ 1 \leq x_3, \\ & x_1,\ x_2,\ x_3 \text{ integers.} \end{aligned}$$

As this problem is quite small, we need not narrow the given feasible region, but could solve this problem directly by using procedures 1, 2, 3. However for this example, we try to narrow the feasible region.

We assume that to make a single objective problem, system designers give the following appropriate constraints,

$$\begin{aligned} W(X) &\leq 70, \\ V(X) &\leq 240. \end{aligned} \qquad (5.22)$$

A single objective problem is as follows:

$$\begin{aligned} \text{maximize} \quad & R(X) \\ \text{subject to} \quad & W(X) \leq 70, \\ & V(X) \leq 240, \\ & 1 \leq x_1 \leq 4,\ 1 \leq x_2,\ 1 \leq x_3, \\ & x_1,\ x_2,\ x_3 \text{ integers.} \end{aligned}$$

The exact optimal solution of this problem is obtained by an exact method in Chapter 4 or approximate method in Chapter 3, The solution $X = (3,3,5)$, is assumed to be acceptable for the system designers,i.e.,

$$x^a = (3,3,5). \tag{5.23}$$

Then referring to the constraints (5.22) and values

$$\begin{aligned} 1 - R(x^a) &= 0.02976, \\ W(x^a) &= 69.26, \\ V(x^a) &= 165.52, \\ C(x^a) &= 37.87, \end{aligned} \tag{5.24}$$

the system designers offer new constraints 0.1, 84, 260, 54 for R, W, V, C, respectively. A multiobjective programming problem obtained, which is formulated as problem B, is:

$$\text{minimize} \quad f(x) = \{f_1(x), f_2(x), f_3(x), f_4(x)\}$$

$$\text{subject to} \quad f_1(x) = \sum_{i=1}^{3} f_{1i}(x_i) \leq 84,$$

$$f_2(x) = \sum_{i=1}^{3} f_{2i}(x_i) \leq 260,$$

$$f_3(x) = \sum_{i=1}^{3} f_{3i}(x_i) \leq 54,$$

$$f_4(x) = \sum_{i=1}^{3} f_{4i}(x_i) \leq 0.10536 \ (= -\ell n\ 0.9),$$

$$1 \leq x_1 \leq 4, \ 1 \leq x_2, \ 1 \leq x_3,$$

$$x_1, x_2, x_3 \text{ integers},$$

where $f_1(X) = W(X)$, $f_2(X) = V(X)$, $f_3(X) = C(X)$, and $f_4(X) = -\ell n\ R(X)$, and $s = 3$, $t = 4$, $m = 4$. This problem is illustrated in Table 5.1.

Table 5.2 illustrates Procedure 1: the permutation {1, 3, 2} is obtained in Step 3, i.e., the variables are fixed in the order x_1, x_3, x_2, and the feasible solutions are generated in the order 1,2,...,27; further information is in Chapter 4.

Table 5.1

Input Data

CRITERION FOR WEIGHT

i	$f_{1i}(1)$	$f_{1i}(2)$	$f_{1i}(3)$	$f_{1i}(4)$	$f_{1i}(5)$	$f_{1i}(6)$
1	6.133	6.284	6.455	6.649		
2	3.750	10.946	15.351	21.155	25.471	31.445
3	16.420	23.244	30.585	38.592	47.452	57.409

CRITERION FOR VOLUME

i	$f_{2i}(1)$	$f_{2i}(2)$	$f_{2i}(3)$	$f_{2i}(4)$	$f_{2i}(5)$	$f_{2i}(6)$
1	10.000	10.000	10.000	10.000		
2	10.272	26.380	50.808	86.985	139.614	215.121
3	7.704	19.785	38.106	65.239	104.710	161.341

CRITERION FOR COST

i	$f_{3i}(1)$	$f_{3i}(2)$	$f_{3i}(3)$	$f_{3i}(4)$	$f_{3i}(5)$	$f_{3i}(6)$
1	4.725	5.136	10.873	29.556		
2	5.000	10.000	15.000	20.000	25.000	30.000
3	4.000	6.000	8.000	10.000	12.000	14.000

CRITERION FOR RELIABILITY

i	$f_{4i}(1)$	$f_{4i}(2)$	$f_{4i}(3)$	$f_{4i}(4)$	$f_{4i}(5)$	$f_{4i}(6)$
1	-0.127833	-0.083381	-0.020203	-0.010050		
2	-0.210721	-0.036768	-0.006883	-0.001304	-0.000247	-0.000047
3	-0.522729	-0.144293	-0.411060	-0.011483	-0.003127	-0.000832

Table 5.2

Feasible Solutions

	x_1, x_3, x_2	$f_1(x)$	$f_2(x)$	$f_3(x)$	$f_4(x)$
1	4, 6, 2	75.00	197.72	53.56	-0.04765
2	4, 5, 2	65.05	141.09	51.56	-0.04994
3	4, 4, 2	56.19	101.62	49.56	-0.05830
4	4, 3, 3	52.58	98.91	52.56	-0.05804
5	4, 3, 2	48.18	74.49	47.56	-0.08792
6	3, 6, 3	79.21	222.15	39.87	-0.02792
7	3, 6, 2	74.81	197.72	34.87	-0.05780
8	3, 5, 5	79.38	254.32	47.87	-0.02358
9	3, 5, 4	74.06	201.70	42.87	-0.02463
10	3, 5, 3	69.26	165.52	37.87	-0.03021
11	3, 5, 2	64.85	141.09	32.87	-0.06010
12	3, 4, 5	70.51	214.85	45.87	-0.03193
13	3, 4, 4	65.20	162.22	40.87	-0.03299
14	3, 4, 3	60.39	126.05	35.87	-0.03857
15	3, 4, 2	55.99	101.62	30.87	-0.06845
16	3, 3, 5	62.51	187.72	43.87	-0.06156
17	3, 3, 4	57.19	135.09	38.87	-0.06261
18	3, 3, 3	52.40	98.91	33.87	-0.06819
19	3, 3, 2	47.99	74.49	28.87	-0.09808
20	2, 6, 4	83.84	258.33	39.14	-0.08552
21	2, 6, 3	79.04	222.15	34.14	-0.09110
22	2, 5, 5	79.21	254.32	42.14	-0.08675
23	2, 5, 4	73.89	201.70	37.14	-0.08781
24	2, 5, 3	69.09	165.52	32.14	-0.09339
25	2, 4, 5	70.35	214.85	40.14	-0.09511
26	2, 4, 4	65.03	162.22	35.14	-0.09617
27	2, 4, 3	60.23	126.05	30.14	-0.10175

Table 5.3

Pareto-Optimal Solutions

i	$\tilde{x}^i$	$-\ln R(\tilde{x}^i)$	$(1-R(\tilde{x}^i))$	$W(\tilde{x}^i)$	$V(\tilde{x}^i)$	$C(\tilde{x}^i)$
1	(3,5,5) *+	−0.02358	(0.0233)	79.38	254.32	47.87
2	(3,4,5) *+	−0.02463	(0.0243)	74.06	201.70	42.87
3	(3,3,6) +	−0.02792	(0.0275)	79.21	222.15	39.87
4	(3,3,5) *+	−0.03021	(0.0298)	69.26	165.52	37.87
5	(3,4,4) *	−0.03299	(0.0325)	65.20	162.22	40.87
6	(3,3,4) *+	−0.03857	(0.0378)	60.39	126.05	35.87
7	(3,2,6) +	−0.05780	(0.0562)	74.81	197.72	34.87
8	(4,3,3) *	−0.05804	(0.0564)	52.58	98.91	52.56
9	(4,2,4)	−0.05830	(0.0566)	56.19	101.62	49.56
10	(3,2,5) +	−0.06010	(0.0583)	64.85	141.09	32.87
11	(3,4,3)	−0.06261	(0.0607)	57.19	135.09	38.87
12	(3,3,3) *	−0.06819	(0.0659)	52.40	98.91	33.87
13	(3,2,4) +	−0.06845	(0.0662)	55.99	101.62	30.87
14	(4,2,3) *	−0.08792	(0.0842)	48.18	74.49	47.56
15	(3,2,3) *+	−0.09808	(0.0934)	47.99	74.49	28.87

* Pareto-optimal solutions when C, both C and W, or both C and V are removed.

\+ Pareto-optimal solutions when both W and V are removed. When W or V is removed, Pareto-optimal soluitons do not vary.

Table 5.3 shows the set of Pareto-optimal solutions (the properties are assumed to be important in the order R, W, V, C), and also Pareto-optimal solutions in the cases that one or two of W, V, C are removed from the objective function vector. This suggests that if the feasible region does not vary, the number of Pareto-optimal solutions are nondecreasing with increasing number of objective functions, the proof of which is easy.

The system designers can obtain a best-compromise alternative by using Table 5.3 and Procedure 3. If $\tilde{x}^4$ is selected by Procedure 3, as shown in Fig. 5.4, and $\tilde{x}^4$ is preferable to $\tilde{x}^1$ and $\tilde{x}^3$, then $\tilde{x}^4$ is a best-compromise solution of the example.

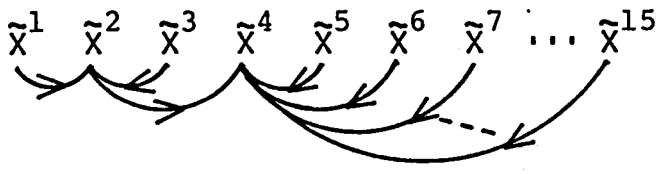


Fig. 5.4 Preference flow.

APPENDIX A

PROOF FOR RULE B IN CHAPTER 2

Let

$$k' = i'_1,$$

$$k'' = i''_1,$$

$$S^*_{\ell'} = \tilde{S}_{i'_2} \cdot \ldots \cdot \tilde{S}_{i'_p},$$

$$S^*_{\ell''} = \tilde{S}_{i''_2} \cdot \ldots \cdot \tilde{S}_{i''_q},$$

and compare Rule B with Rule A, then we have:

a) Case 1 in Rule B is clearly equivalent to the case of $t = 1$ in Rule A.

b) Case 2 in Rule B is equivalent to the case of $t \geq 2$, $p > 2$, $q > 2$ in Rule A because: since e.s. structures $S_{\ell'}$ and $S_{\ell''}$ have been designated according to Rule A, the condition $t \geq 2$ is equivalent to $\ell' < \ell''$.

c) Case 3 in Rule 3 is equivalent to the case of $t = 2$, $p > 2$, $q = 2$ in Rule A because: since we have $u(\tilde{S}_{i'_2}) < u(\tilde{S}_{\ell''}) = \ell''$ $(= i''_2)$ from $u(S_{\ell'}) = u(S_{\ell''})$, it is always true that $i'_2 < \ell'$ $(= i''_2)$.

d) Case 4 in Rule B is clearly equivalent to the case of $t = 2$, $p =$

2, q = 2.

e) Cases 1,...,4 in Rule B clearly include all cases in Rule A.

a) ~ e) show that Rule B is equivalent to Rule A.

APPENDIX B

PROOFS OF SOME PROPERTIES IN CHAPTER 2

1) Proof of Property 1: Without loss of generality, let

$$S_i = \tilde{S}_k \cdot S^*_\ell \quad (k < \ell). \tag{B1}$$

Then from $k < \ell$, we have that $u(\tilde{S}_k) \leq u(S^*_\ell)$. Hence $u(\tilde{S}_k) \leq [m/2]^+$.

2) Proof of Property 3: Without loss of generality, let

$$S_\ell = \tilde{S}_{\ell_1} \cdot \tilde{S}_{\ell_2} \cdot \ldots \cdot \tilde{S}_{\ell_p} \quad (1 \leq \ell_1 \leq \ell_2 \leq \ldots \leq \ell_p \leq \quad , p \geq 2), \tag{B2}$$

then from (2.10) we have

$$k \leq \ell_1 \leq \ell_2 \leq \ldots \leq \ell_p, \tag{B3}$$

that is,

$$u(\tilde{S}_k) \leq u(\tilde{S}_{\ell_1}) \leq \ldots \leq u(\tilde{S}_{\ell_p}). \tag{B4}$$

Hence we have

$$2 \times u(\tilde{S}_k) \leq p \times u(\tilde{S}_k) \leq u(S_\ell), \tag{B5}$$

because $p \geq 2$ and $\sum_{t=1}^{p} u(\tilde{S}_{\ell_t}) = u(S_\ell)$. Combining (2.10) and (B5) yields

$$3 \times u(\tilde{S}_k) \leq m, \tag{B6}$$

so that

$$u(\tilde{S}_k) \leq [m/3]^+. \tag{B7}$$

By combining (B7) and (2.15), we obtain $1 \leq k \leq \lambda_{[m/3]^+ + 1, 1}^{-1}$.

3) Proof of Property 5: We have

$$1 \leq k \leq \lambda_{[(m+1)/2]^+,1}^{-1} = N_{[(m-1)/2]^+}, \tag{B8}$$

that is,

$$u(\tilde{S}_k) \leq [(m-1)/2]^+. \tag{B9}$$

From (B9) we have

$$u(\tilde{S}_k) \leq m - 1 - u(\tilde{S}_k), \tag{B10}$$

that is

$$u(\tilde{S}_k) \leq m - 1 - j. \tag{B11}$$

Combining (2.15) and (B11) yields

$$k \leq \lambda_{m-j,1} - 1. \tag{B12}$$

This shows that Property 5 is proper.

APPENDIX C

AN ILLUSTRATION OF PROCEDURE A IN CHAPTER 2

In order to illustrate Procedure A, suppose that we now have $m = 8$ at Step 2; in other words, we have been already generated and designated all possible s-p structures for 1, ..., 7 units, and we have obtained the values of $\lambda_{m,k}$ for all $m \leq 7$ and the value of $\lambda_{8,1}$, which are shown in Table 2.1.

An illustration ($m = 8$) of Procedure A is as follows:

Step 2: ($m = 8$) We have $j = 1, 2$, and then $k = 1$ when $j = 1$ and $k = 2$ when $j = 2$.

When $j = 1$, $k = 1$, we have:

i. The obtained structures, which have been designated, are

$$S_{145} = \tilde{S}_1 \cdot S_{55},\ S_{146} = \tilde{S}_1 \cdot S_{56},\ \ldots,\ S_{234} = \tilde{S}_1 \cdot S_{144}.$$

ii. $S_{235} = \tilde{S}_1 \cdot \tilde{S}_{55},\ S_{236} = \tilde{S}_1 \cdot \tilde{S}_{56},\ \ldots,\ S_{324} = \tilde{S}_1 \cdot \tilde{S}_{144}.$

iii. $\lambda_{8,2} = 367.$

When j = 2, k = 2, we have:

i. $S_{325} = \tilde{S}_2 \cdot S_{46}$, $S_{326} = \tilde{S}_2 \cdot S_{47}$, ..., $S_{333} = \tilde{S}_2 \cdot S_{54}$.

ii. $S_{334} = \tilde{S}_2 \cdot \tilde{S}_{22}$, $S_{335} = \tilde{S}_2 \cdot \tilde{S}_{23}$, ..., $S_{366} = \tilde{S}_2 \cdot \tilde{S}_{54}$.

iii. $\lambda_{8,3} = 367$.

Step 3: $[8/3]^+ + 1 = [(8 - 1)/2]^+ = 3$.

Step 4: We have j = 3 and k = 3, 4.

When j = 3, k = 3, we have:

i. $S_{367} = \tilde{S}_3 \cdot \tilde{S}_{10}$, ..., $S_{378} = \tilde{S}_3 \cdot \tilde{S}_{21}$.

ii. $\lambda_{8,4} = 379$

When j = 3, k = 4, we have:

i. $S_{379} = \tilde{S}_4 \cdot \tilde{S}_{10}$, ..., $S_{390} = \tilde{S}_3 \cdot \tilde{S}_{21}$.

ii. $\lambda_{8,5} = 391$.

Step 5: We have j = 4, k = 5, 6, 7, 8, 9.

i. (k=5) $S_{391} = \tilde{S}_5 \cdot \tilde{S}_5$, ..., $S_{395} = \tilde{S}_5 \cdot \tilde{S}_9$.

ii. $\lambda_{8,6} = 396$.

i. (k=6) $S_{396} = \tilde{S}_6 \cdot \tilde{S}_6$, ..., $S_{399} = \tilde{S}_6 \cdot \tilde{S}_9$.

ii. $\lambda_{8,7} = 400$.

i. (k=7) $S_{400} = \tilde{S}_7 \cdot \tilde{S}_7$, ..., $S_{402} = \tilde{S}_7 \cdot \tilde{S}_9$.

ii. $\lambda_{8,8} = 403.$

i. (k=8) $S_{403} = \tilde{S}_8 \cdot \tilde{S}_8$, $S_{404} = \tilde{S}_8 \cdot \tilde{S}_9$.

ii. $\lambda_{8,9} = 405.$

i. (k=9) $S_{405} = \tilde{S}_9 \cdot \tilde{S}_9$.

ii. $\lambda_{8,10} = 406.$

Step 6: If $M \neq 8$, then reset $m \leftarrow 9$ and set $\lambda_{9,1} = \lambda_{8,10} = 406$ and return to Step 2. If $m = 8$, then stop.

Appendix D

DETAILS OF EXAMPLE 1 IN CHAPTER 4

1. (Step 1) Relaxation RP3 is used, and set $f^* \leftarrow \infty$, $X^* \leftarrow (\infty,\infty,\infty)$, $L^f = \{1,2,3\}$.
2. (Step 2) We obtain $\bar{x}^1 = (4,4,1)$ using (4.7) and reset $\bar{x} \leftarrow \bar{x}^1$. Next $\underline{x}^1 = (1,1,0)$ is obtained using (4.8) and reset $\underline{x} \leftarrow \underline{x}^1$. Thirdly $\bar{x}^2 = (>5,>5,4)$ is obtained using (4.9), and reset $\bar{x} \leftarrow (\min\{\bar{x}_1^1,\bar{x}_1^2\},\min\{\bar{x}_2^1,\bar{x}_2^2\},\min\{\bar{x}_3^1,\bar{x}_3^2\}) = (4,4,1)$. Finally we obtained $\bar{x}^c = (4,4,1)$ and $\underline{x}^c = (1,1,0)$ using (4.11), and reset $\bar{x} \leftarrow \bar{x}^c$ and $\underline{x} \leftarrow \underline{x}^c$.
3. (Step 3) A permutation $\{3,1,2\}$ is obtained, i.e., (1) = 3, (2) = 1, (3) = 2. Set $\ell = 1$.
4. (Step 4) Set $x_{(1)}$ $(=x_3) = 0$ and reset $L^r = \{3\}$, $\ell = 2$.
5. (Step 5) $\bar{x}^1_{(2)}$ $(=\bar{x}^1_1) = 4$, $\underline{x}^1_{(2)} = 2$, $\bar{x}^2_{(2)} = 4$, then $\bar{x}^c_{(2)} = 2$, $\underline{x}^c_{(2)} = 2$.
6. (Step 4) Set $x_{(2)} = 2$, and reset $L^r = \{3,1\}$, $\ell = 3$ $(=n)$.
7. (Step 6) $\bar{x}^1_{(3)}$ $(=\bar{x}^1_2) = 4$, $\underline{x}^1_{(3)} = 3$, $x_{(3)} = 3$.
8. (Step 7) $X^* = (2,3,0)$, $f^* = 14.49$.

9. (Step 8) $\ell = 2$, $\ell = 3$, $x_{(2)}$ $(=x_{(\ell)}) = 3$, $L^r = \{3,1\}$.

	(Step 6)			(Step 7)		(Step 8)			
	$\overset{-1}{x}_{(3)}$	$\underset{=}{x}^{1}_{(3)}$	$x_{(3)}$	x^*	f^*	ℓ	ℓ	$x_{(2)}$	L^r
10. (Step 6-8)	3	2	2	(3,2,0)	10.93	2	3	4	{3,1}
11. (Step 6-8)	3	2	2	Same	Same	1	2	1	{3}

	(Step 5)					(Step 4)		
	$\overset{-1}{x}_{(2)}$	$\underset{=}{x}^{1}_{(2)}$	$\overset{-2}{x}_{(2)}$	$\overset{-c}{x}_{(2)}$	$\underset{=}{x}^{c}_{(2)}$	$x_{(2)}$	L^r	ℓ
12. (Step 5,4)	4	1	4	3	1	1	{3,1}	3

	(Step 6)			(Step 7)		(Step 8)			
	$\overset{-1}{x}_{(3)}$	$\underset{=}{x}^{1}_{(3)}$	$x_{(3)}$	x^*	f^*	ℓ	ℓ	$x_{(2)}$	L^r
13. (Step 6-8)	4	4	4	(3,2,0)	10.93	2	3	2	{3,1}
14. (Step 6-8)	3	3	3	Same	Same	2	3	3	{3,1}
15. (Step 6-8)	3	2	2	Same	Same	Stop			

Appendix E

DERIVATION OF EQUATION (4.16)

Problem: Maximize $f(X) = \prod_{i=1}^{n} (1 - q_i^{x_i})$

subject to one constraint

$$g_j(X) = \sum_{i=1}^{n} a_{ji} x_i \leq b_j' .$$

The Lagrangian function for this problem is

$$\phi(X,\lambda) = \ln f(X) + \lambda(g_j(X) - b_j'). \tag{E1}$$

Therefore the conditions for stationary point are

$$\partial\phi(X,\lambda)/\partial x_i = 0 \quad (i = 1,\ldots,n), \tag{E2}$$

$$\partial\phi(X,\lambda)/\partial\lambda = 0. \tag{E3}$$

From (E2), we have

$$x_i = (z_{ji}/a_{ji})\ln(1 + t/z_{ji}) \quad (i = 1,\ldots,n), \tag{E4}$$

or

$$x_i = (z_{ji}/a_{ji})\{\ln t - \ln z_{ji} - \ln(1 - q_i^{x_i})\} \quad (i = 1,\ldots,n), \qquad (E5)$$

where $t = -1/\lambda$ and $z_{ji} = -a_{ji}/\ln q_i$. Substituting (E5) into (E3) results in (E6).

$$t = \exp[\{b'_j + \sum_{i=1}^{n} z_{ji}\ln z_{ji} + \sum_{i=1}^{n} z_{ji}\ln(1 - q_i^{x_i})\}/(\sum_{i=1}^{n} z_{ji})]. \qquad (E6)$$

Letting $1 - q_i^{x_i} \simeq 1$, we obtain

$$t \simeq t' = \exp\{(b'_j + \sum_{i=1}^{n} z_{ji}\ln z_{ji})/(\sum_{i=1}^{n} z_{ji})\}, \qquad (E7)$$

where $t \leq t'$. From the objective function, (E4) and (E7), we can finally obtain (4.16), i.e., the upper value of the problem.

REFERENCES

[A1] K. K. Aggarwal, J. S. Gupta, K. B. Misra, A New Heuristic Criterion for Solving a Redundancy Optimization Problem, IEEE Trans. Reliability, Vol. R-24, pp. 86-87, Apr. 1975.

[A2] K. K. Aggarwal, Redundancy Optimization in General System, IEEE Trans. Reliability, Vol. R-25, pp. 330-332, Dec. 1976.

[A3] O. G. Alekseyev, V. I. Iskushev, Algorithm for Optimal Reserve (Redundancy) or an Apparatus, Eng. Cybernectics, No. 3, pp. 55-61, May-June 1964.

[A4] O. G. Alekseyev, A Problem in Optimal Redundancy, Tech. Cybernetics, No. 1, 1967. Rep. JPRS-40629, TT-67-31273, Joint Publications Research Service, Apr. 14, pp. 61-68, 1967.

[B1] S. K. Banerjee, K. Rajamani, Optimization of System Reliability Using a Parametric Approach, IEEE Trans. Reliability, Vol. R-22, pp.35-39, Apr. 1973.

[B2] S. K. Banerjee, K. Rajamani, S. S. Deshpande, Optimal Redundancy Allocation for Non Series-Parallel Networks, IEEE Trans. Reliability, Vol. R-25, pp. 115-117, June 1976.

[B3] R. E. Barlow, L. Hunter, Criteria for Determining Optimum Redundancy, IRE Trans. Reliability and Quality Control, pp. 73-77, 1960.

[B4] R. E. Barlow, L. C. Hunter, F. Proschan, Optimal Redundancy When

Components Are Subject to Two Kinds of Failure, J. Soc. Indust. Appl. Math., Vol. 11, pp. 64-73, Mar. 1963.

[B5] R. E. Barlow, F. Proschan, Mathematical Theory of Reliability, John Wiley & Sons, 1965.

[B6] R. E. Barlow, F. Proschan, Statistical Theory of Reliability and Life Testing, Holt, Rinehart and Winston, 1975.

[B7] P. W. Becker, B. Jarkler, A Systematic Procedure for the Generation of Cost-Minimized Designs, IEEE Trans. Reliability, Vol. R-21, pp. 41-56, Feb. 1972.

[B8] R. E. Bellman, S. E. Dreyfus, Dynamic Programming and Reliability of Multicomponent Devices, JORSA, Vol. 6, pp. 200-206, May-Apr. 1958.

[B9] D. D. Bien, Optimum Allocation of Redundancy among Subsystems Connected in Series, NASA TN D-7164, Mar. 1973.

[B10] G. R. Bitran, Linear Multiple Objective Programs with Zero-One Variables, Math. Prog., Vol. 13, pp. 121-139, 1977.

[B11] G. Black, F. Proschan, On Optimal Redundancy, JORSA, Vol. 7, pp. 581-588, 1959.

[B12] G. R. Burdick, D. M. Rasmuson, S. L. Derby, A Risk-Based Approach to Advanced Reactor Design, IEEE Trans. Reliability, Vol. R-26, pp. 198-202, Aug. 1977.

[B13] R. M. Burton, G. T. Howard, Optimal System Reliability for a Mixed Series and Parallel Structure, J. Math. Analy. and Appl., Vol. 28, pp. 370-382, 1969.

[B14] R. M. Burton, G. T. Howard, Optimal Design for System Reliability and Maintainability, IEEE Trans. Reliability, Vol. R-20, pp. 51-60, May 1971.

[C1] A. V. Cabot, An Enumeration Algorithm for Knapsack Problems, JORSA, Vol. 18, pp. 306-311, Mar-Apr 1970.

[C2] J. L. Cohon, D. H. Marks, A Review and Evaluation of Multi-objective Programing Techniques, Water Resources Research, Vol.

11, pp. 208-220, Apr. 1975.

[D1] H. A. David, The Method of Paired Comparisons, Griffin, 1963.

[D2] M. J. DiToro, Reliability Criterion for Constrained Systems, IRE Trans. Reliability and Quality Control, Vol. RQC-8, pp. 1-6, Sep. 1956.

[E1] R. A. Evans, Reliability Optimization, Proc. NATO Advanced Study Institute on Generic Technique in Systems Reliability Assessment, July 1973,(Nordhoff, Netherlands, pp. 117-123, 1975).

[E2] H. Everett, III, Generalized Lagrange Multiplier Method for Solving Problem of Optimum Allocation of Resources, JORSA, Vol. 11, pp. 399-417, May-June 1963.

[F1] A. J. Federowicz, M. Mazumdar, Using of Geometric Programming to Maximize Reliability Achieved by Redundancy, JORSA, Vol. 16, pp.948-954, Sep.-Oct. 1968.

[F2] D. E. Fuffe, W. W. Hines, N. K. Lee, System Reliability Allocation and a Computational Algorithm, IEEE Trans. Reliability, Vol. R-17, pp. 64-69, June 1968.

[G1] S. L. Gandhi, E. J. Henley, Optimal Availability of a Complex System, Proc. NATO Advanced Study Institute on Generic Techniques in Systems Reliability Assessment, July 1973, (Nordhoff, Netherlands, 1975).

[G2] R. S. Garfinkel, G. L. Nemhause, Integer Programming, John Wiley & Sons, 1972.

[G3] M. Gen (K. N. Hyun), H. Okuno, Optimization of the redundant Allocation and Selection Problems in a System Reliability, Keiei-Kogaku, Vol. 18, pp. 132-147, July 1974 (in Japanese, The Operations Research Society of Japan).

[G4] M. Gen (K. N. Hyun), M. Sasaki, Optimal Allocation of A System Availability, Trans. IECE of Japan, Vol. 58-D, pp. 9-16, Jan.

1975 (in Japanese).

[G5] M. Gen (K. N. Hyun), Optimization by Zero-One Programming of a System Reliability with Several Failure-Modes, Trans. IECE of Japan, Vol. 58-D, pp. 373-380, July 1975.

[G6] M. Gen (K. N. Hyun), Reliability Optimization by 0-1 Programming for a System with Several Failure Modes, IEEE Trans. Reliability, Vol. R-24, pp. 206-210, Aug. 1975.

[G7] M. Gen (K. N. Hyun), H. Okuno, S. Shinofuji, An Optimizing Method in System Reliability with Failure-Modes by Implicit Enumeration Algorithm, JORSJ, Vol. 19, pp. 99-116, June 1976.

[G8] M. Gen (K. N. Hyun), Optimization by Lawler-Bell Algorithm of Life-Support System Reliability with Failure-Modes, Trans. IECE of Japan, Vol. 59-D, pp. 157-164, Mar. 1976.

[G9] A. M. Geoffrion, R. E. Marsten, Integer Programming Algorithms : A Frame Work and State-of-the-Art Survey, Management Science, Vol. 18, pp 465-491, May 1972.

[G10] P. M. Ghare, R. E. Taylor, Optimal Redundancy for Reliability in Series System, JORSA, Vol. 17, pp. 838-847, Sep. 1969.

[G11] Б. В. Гнеденко, Ю. К. Беляев, А. Д. Соловьев, Математические Методьы в Теории Надежности, Наука, Москва, 1965.

[G12] R. Gordon, Optimum Component Redundancy for Maximum System Reliability, JORSA, Vol. 5, pp. 229-243, Apr. 1957.

[G13] J. P. Guilford, Psycometric Methods, McGraw-Hill, 2nd ed, 1954.

[H1] Ir. R. N. v. Hees, Ir. H. W. v. d. Meerendonk, Optimal Reliability of Parallel Multi-Component Systems, Opreations Research Quarterly, Vol. 12, pp. 16-26, May 1961.

[H2] G. G. Hendrix, A. C. Stedry, The Elementary Redundancy Optimization Problem: A Case Study in Probabilistic Multiple-Goal Programming, JORSA, Vol. 22, pp. 639-653, 1974.

[H3] J. E. Hendry, D. F. Rudd, J. D. Seader, Synthesis in the Design of Chemical Processes, AIChE J., Vol. 19, pp. 1-15, Jan 1973.

[H4] C. Henin, An Algorithm for Maximizing Reliability through System Redundancy, Carnegie-Mellon Univ., Management Sci. Res. Rep., 1970 (or Ad723056).

[H5] C. Henin, Double Failure and Other Related Problems in Standby Redundancy, IEEE Trans. Reliability, Vol. R-21, pp. 35-40, Feb. 1972.

[H6] E. J. Henley, S. L. Gandhi, Process Reliability Analysis, AIChE J., Vol. 21, pp. 677-686, July 1975.

[H7] D. P. Herron, Optimizing Trade-Offs of Reliability vs Weight, IEEE Trans. Reliability, Vol. R-12, pp. 50-54, Dec. 1963.

[H8] C. L. Hwang, L. T. Fan, F. A. Tillman, S. Kumar, Optimization of Life Support System Reliability by an Integer Programming Method, AIIE Trans., Vol. 3, pp. 229-238, Sep. 1971.

[I1] T. Inagaki, K. Inoue, H. Akashi, Interactive Optimization of System Reliability under Multiple Objectives, to appear in IEEE Trans. Reliability.

[I2] K. Inoue, S. L. Gandhi, E. J. Henley, Optimal Reliability Design of Process Systems, IEEE Trans. Reliability, Vol. R-23, pp. 29-33, Apr. 1974.

[I3] L. K. Isayev, N. A. Mamed-Zade, Optimization of the Reliability of a Redundant System, Tech. Cybernetics, No.1, 1967. Rep. JPRS-40629, TT-67-31273, Joint Publication Research Service, pp. 57-60, Apr. 1967.

[J1] P. A. Jensen, The Design of Multiple-Line Redundancy Networks, IEEE Trans. Reliability, Vol. R-18, pp. 39-44, May 1969.

[J2] P. A. Jensen, Optimization of Series-Parallel-Series Networks, JORSA, Vol. 18, pp. 471-482, May-June 1970.

[K1] J. D. Kettelle, Least-Cost Allocation of Reliability Investment, JORSA, Vol. 10, pp. 249-265, Mar.-Apr. 1967.

[K2] C. F. King, D. F. Rudd, Design and Maintenance of Economically

Failure-Tolerant Processes, AIChE J., Vol. 18, pp. 257-269, Mar. 1972.

[K3] P. J. Kolesar, Linear Programming and the Reliability of Multi-Component Systems, Naval Research Logistics Quarterly, Vol. 15, pp. 317-327, Sep. 1967.

[K4] J. Kondo, The Optimum Systems Design - An Allocation of Redundancy to Obtain the Maximum Reliability at the Minimum Weight, Space Craft Systems, Vol. 1, Michal Lunc, ed., Gordon and Breach Science Publ., pp. 315-321, 1966.

[K5] A. G. Korman, On Optimal Reserving of an Apparatus, Tech. Cybernetics, No. 4, 1964, Rep. JPRS-27297, TT-64-51553, Joint Publications Research Service, Nov. 1964, pp. 33-50.

[L1] E. L. Lawler, M. D. Bell, A Method of Solving Discrete Optimization Problems, JORSA, Vol. 14, pp. 1098-1112, Nov.-Dec. 1966.

[L2] B. P. Lientz, A Stchastic Method of Allocation of Components to Maximize Reliability, IEEE Trans. Reliability, Vol. R-23, pp. 91-97, June 1974.

[L3] R. Luus, Optimization of System Reliability by a New Nonlinear Integer Programming Procedure, IEEE Trans. Reliability, Vol. R-24, pp. 14-16, Apr. 1975.

[M1] R. E. Marsten, T. L. Morin, A Hybrid Approach to Discrete Mathematical Programming, Math. Prog., Vol. 14, pp. 21-40, 1978.

[M2] D. W. McLeavey , On an Algorithm of Ghare and Taylor, JORSA, Vol. 21, pp. 1315-1318, 1973.

[M3] D. W. McLeavey, Numerical Investigation of Parallel Redundancy in Series Systems, JORSA, Vol. 22, pp. 1110-1117, Sep.-Oct. 1974.

[M4] D. W. McLeavey, J. A. McLeavey, Optimization of System Reliability by Branch-and-Bound, IEEE Trans. Reliability, Vol. R-25, pp. 327-329, Dec. 1976.

[M5] W. S. Meisel, P. C. H. Schaeffer, Reliability in Digital Systems with Asymmetrical Failure Modes, IEEE Trans. Reliability, Vol.

R-18, pp. 74-75, May 1969.

[M6] M. Messinger, M. Shooman, Technique for Optimum Spares Allocation: A Tutorial Review, IEEE Trans. Reliability, Vol. R-19, pp. 156-166, Nov. 1970.

[M7] H. Mine, Reliability of Physical System, Trans. Internat. Symp. Circuit and Information Theory, pp. 138-151, May 1959.

[M8] K. B. Misra, A Method of Solving Redundancy Optimization Problems, IEEE Trans. Reliability, Vol. R-20, pp. 117-120, Aug. 1971.

[M9] K. B. Misra, A Simple Approach for Constrained Redundancy Optimization Problems, IEEE Trans. Reliability, Vol. R-21, pp. 30-34, Feb. 1972.

[M10] K. B. Misra, Reliability Optimization of a Series-Parallel System, IEEE Trans. Reliability, Vol. R-21, pp. 230-238, Nov. 1972.

[M11] K. B. Misra, J. Sharma, A New Geometric Programming Formulation for a Reliability Problem, Inter. J. of Control, Vol. 18, No. 3, pp. 497-503, 1973.

[M12] K. B. Misra, C. E. Carter, Redundancy Allocation in a System with Many Stages, Microelectronics and Reliability, Vol. 12, pp. 223-228, June 1973.

[M13] K. B. Misra, J. Sharma, Reliability Optimization of a System by Zero-One Programming, Microelectronics and Reliability, Vol. 12, pp. 229-233, June 1973.

[M14] K. B. Misra, A Method for Redundancy Allocation, Microelectronics and Reliability, Vol. 12, pp. 389-393, Oct. 1973.

[M15] K. B. Misra, Reliability Optimization through Sequential Simplex Search, Int. J. Control, Vol. 18, pp. 173-183, 1973.

[M16] K. B. Misra, J. Sharma, Reliability Optimization with Integer Constraint Coefficients, Microelectronics and Reliability, Vol. 12, pp. 431-433, Nov. 1973.

[M17] K. B. Misra, Optimum Reliability Design of a System Containing

Mixed Redundancies, IEEE Trans. PAS, Vol. PAS-94, pp. 983-993, May-June 1975.

[M18] K. B. Misra, On Optimal Reliability Design: A Review, IFAC for 6th World Conference, Boston, Mass. USA, pp. 3.4,1-3.4,10, Aug. 1975.

[M19] K. Mizukami, Optimum Redundancy for Maximum System Reliability by the Method of Convex and Integer Programming, JORSA, Vol. 16, pp. 392-406, Mar.-Apr. 1968.

[M20] J. M. Mogg, Constrained Optimum Test Configuration for Reliability Acceptance Tests Incorporating Environmental Stresses, IEEE Trans. Reliability, Vol. R-24, pp. 211-213, Aug. 1975.

[M21] E. F. Moore, C. E. Shannon, Reliable Circuits Using Less Reliable Relays, J. Franklin Inst., Vol. 262, pp. 191-208, Sep. 1956; also Vol. 262, pp. 281-297, Oct. 1956.

[M22] T. L. Morin, R. E. Marsten, An Algorithm for Nonlinear Knapsack Problems, Management Sci., Vol. 22, pp. 1147-1158, June 1976.

[M23] T. L. Morin, R. E. Marsten, Branch-and-Bound Strategies for Dynamic Programming, JORSA, Vol. 24, pp. 611-627, July-Aug 1976.

[M24] D. F. Morison, The Optimum Allocation of Spare Components in System, Technometrics, Vol. 3, pp. 399-406, 1961.

[M25] F. Moskowitz, J. B. McLean, Some Reliability Aspects of System Design, IRE Trans. Reliability and Quality Control, Vol. PGRQC-8, pp. 7-35, Sep. 1956.

[M26] B. L. Myers, N. L. Enrich, Algorithmic Optimization of System Reliability, Annual Technical Conference Trans. of American Society for Quality Control, pp. 455-460, 1968.

[N1] Y. Nakagawa, K. Nakashima, Y. Hattori, A Heuristic Method for Solving Optimal Redundancy Allocation Problems, Reports of Professional Group on Reliability of IECE, Vol. R75-2, pp. 11-19, Apr. 1975 (in Japanese).

[N2] Y. Nakagawa, K. Nakashima, Y. Hattori, A Method for Solving

Optimal Redundancy Allocation Problems by the Enumeration of Promising Solutions, Reports of Professional Group on Reliability of IECE, R75-3, pp. 20-31, Apr. 1975 (in Japanese).

[N3] Y. Nakagawa, K. Nakashima, A Heuristic Method for Determinig Optimal Reliability Allocation, IEEE Trans. Reliability, Vol. R-26, pp. 156-161, Aug. 1977.

[N4] Y. Nakagawa, K. Nakashima, Y. Hattori, Optimal Reliability Allocation by Branch-and-Bound Technique, IEEE Trans. Reliability, Vol. R-27, Apr. 1978.

[N5] Y. Nakagawa, Y. Hattori, Optimal Design of High Reliable System: From the Point of View of Mathemetical Programming, Communications of the Operations Research Society of Japan (Operations Research), Vol. 23, pp.536-543, Sep. 1978 (in Japanese).

[N6] Y. Nakagawa, Y. Hattori, Design of Reliability System with Multiple Properties and Integer Variables, to appear in IEEE Trans. Reliability, Vol. R-28, Apr. 1979.

[N7] Y. Nakagawa, Y. Hattori, Reliability of All Possible Series-Parallel Structures of M I.I.D. Units with Two Failure Modes, to appear in IEEE Trans. Reliability.

[N8] Y. Nakagawa, Y. Hattori, Branching-and-Fathoming Method for Muliobjective Nonlinear Pure-Integer Programming Problems, Preprints of 1978-Oct. Meeting of the Operations Research Society of Japan, D-3, pp. 138-139, Oct. 1978.

[N9] Y. Nakagawa, S. Matushita, Branching-and-Fathoming Method for Knapsack Problems, Preprints of 1978-Oct. Meeting of the Operations Research Sociaty of Japan, D-4, pp. 140-141, Oct. 1978.

[N10] Y. Nakagawa, S. Miyazaki, Branching-and-Fathoming Method for Nonlinear Knapsack Problems having 1 or 2 Active Constraints, Preprints of 1978-Oct. Meeting of the Operations Research Society of Japan, D-5, pp. 142-143, Oct. 1978.

[N11] K. Nakashima, K. Yamato, Optimal Design of a Series-Parallel System with Time-Dependent Reliability, IEEE Trans. Reliability,

Vol. R-26, pp. 119-120, June 1977.

[N12] C. P. Nerman and N. M. Bonhomme, Optimal Inspection and Repair Schedules for Multicomponent Systems, IEEE Trans. Syst., Man, and Cybern., Vol. SMC-4, pp. 58-65, Jan. 1974.

[O1] V. K. Opfermann, Die Optimierung der Systemzuverlassigkeit Durch Komponentenredundanz mit Hilfe des Discreten Maximum-Prinzips, Unternehmens Forschung, Vol. 15, p. 15, 1971.

[P1] F. Proschan, Redundancy for Reliability Improvement, in Statistical Theory of Reliability (ed.) M. Zelen, Makison, The University of Wisconsin Press, pp. 55-70, 1964.

[P2] F. Proschan, T. A. Bray, Optimum Redundancy under Multiple Constraints, JORSA, Vol. 13, pp. 800-814, Sep.-Oct. 1965.

[R1] J. Riordan, C. E. Shannon, The Number of Two-Terminal Series-Parallel Networds, J. Math. and Phys., Vol. 21, pp. 83-93, 1942.

[R2] D. F. Rudd, Reliability Theory in Chemical System Design, I&EC Fundamental, Vol. 1, pp. 138-143, May 1962.

[R3] B. Roy, Problems and Methods with Multiple Objective Functions, Math. Progr., Vol. 1, pp.239-266, 1971.

[S1] M. Sakawa, Multiobjective Reliability and Redundancy Optimization of a Series-Parallel System by the Surrogate Worth Trade-Off Method, Microelectronics and Reliability, Vol. 17, pp. 465-467, 1978.

[S2] M. Sasaki, A Simplified Method of Obtaining Highest System Reliability, Proc. Eighth National Symposium on Reliability and Quality Control, pp. 489-502, 1962.

[S3] M. Sasaki, An Easy Allotment Method Achieving Maximum System Reliability, Pro. Ninth National Symposium on Reliability and Quality Control, pp. 109-124, 1963.

[S4] M. Sasaki, S. Kaburaki, S. Ynagi, System Availability and Opti-

mal Spare Units, IEEE Trans. Reliability, Vol. R-26, pp. 182-188, Aug. 1977.

[S5] J. Sharma, K. V. Venkateswaran, A Direct Method for Maximizing the System Reliability, IEEE Trans. Reliability, Vol. R-20, pp. 256-259, Nov. 1971.

[S6] H. V. K. Shetty, D. P. S. Gupta, Optimization of Redundant Systems: A Non-Linear Programming Approach, NATO Advanced Study Institute on Generic Technique in Systems Reliability Assessment, July 1973 (Nordhoff, Netherlands, pp. 337-341, 1975).

[S7] M. L. Shooman, Probabilistic Reliability: An Engineering Approach, McGraw-Hill, 1968.

[T1] S. Takasaki, M. Gen, H. Okuno, Optimization in a System Reliability with Incomplete FDS by Zero-One Programming, Trans. IECE of Japan, Vol. 60-D, pp. 515-522, July 1977 (in Japanese).

[T2] R. E. Taylor, Optimal Resource allocation for Reliability in Series System, Virginia Polytechnic Institute, PhD Dissertation, 1970.

[T3] R. B. Thakkar, R. C. Hughes, Aerospace Power Systems Maximizing Reliability with respect to Weight, IEEE Trans. Aerospace, Vol. 2, pp. 528-534, Apr. 1965.

[T4] F. A. Tillman, Integer Programming Solutions to Reliability Optimization Problems subject to Linear and Nonlinear Separable Restraints, State University of Iowa, PhD Dissertation, 1965.

[T5] F. A. Tillman, Integer Programming Solutions to Constrained Reliability Optimization Problems, Trans. of Twentieth Annual Technical Conference, American Society for Quality Control, paper number 66-174, pp. 676-693, 1966.

[T6] F. A. Tillman, J. M. Littschwager, Integer Programming Formulation of Constrained Reliability Problems, Management Sci., Vol. 13, pp. 887-899, July 1967.

[T7] F. A. Tillman, C. L. Hwang, L. T. Fan, S. A. Balbale, Systems

Reliability subject to Multiple Nonlinear Constraints, IEEE Trans. Reliability, Vol. R-17, pp. 153-157, Sep. 1968.

[T7] F. A. Tillman, Optimization by Integer Programming of Constrained Reliability Problems with Several Modes of Failure, IEEE Trans. Reliability, Vol. R-18, pp. 47-53, May 1969.

[T8] F. A. Tillman, C. L. Hwang, L. T. Fan, K. C. Lai, Optimal Reliability of a Complex System, IEEE Trans. Reliability, Vol. R-19, pp. 95-100, Aug. 1970.

[T9] F. A. Tillman, C. L. Hwang, W. Kuo, Optimization Techniques for System Reliability with Redundancy - A Review, IEEE Trans. Reliability, Vol. R-26, pp. 148-155, Aug. 1977.

[W1] L. R. Webster, Choosing Optimum System Configurations, Proc. Tenth National Symposium on Reliability & Quality Control, pp. 345-359, 1964.

[W2] L. R. Webster, Optimum System Reliability and Cost Effectiveness, Proc. 1967 Annual Symposium on Reliability, pp. 489-500, 1967.

[W3] J. Weisman, A. G. Holzman, The Integer Programming Problem in Nuclear Power Plant Optimization, Trans. Am. Nuc. Soc., Vol. 12, pp. 141, 1969.

[W4] C. F. Woodhouse, Optimal Redundancy Allocation by Dynamic Programming, IEEE Trans. Reliability, Vol. R-21, pp. 60-62, Feb. 1972.