OPTIMIZATION OF INDUSTRIAL MANAGEMENT SYSTEMS BY THE SEQUENTIAL UNCONSTRAINED MINIMIZATION TECHNIQUE

by 1264

Keeting Chiecheng Lai

B.S. Chong-Uan College of Science and Technology

Taw-Uan, Formosa, 1965

A Master's Report

Submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

Kansas State University

Manhattan, Kansas

1970

Approved by

Ching-Lai Hwang

Major Professor

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The problems considered in this report are optimization of system reliability of a complex system and optimization of production scheduling and inventory control subject to some linear and/or nonlinear constraints. The optimization method employed is the sequential unconstrained minimization technique (SUMT). This method is considered as one of the simplest and the most efficient methods for solving the constrained nonlinear programming problems.

The purposes of this report are twofold. The first is to present a result of implementing SUMT by a combination of the Hooke and Jeeves pattern search technique [13,14] and a heuristic programming technique [19]. The second is to present results of the optimization study of system reliability of a complex system and production scheduling and inventory control problems by means of the developed technique.

The principle of the sequential unconstrained minimization technique (SUMT) is a transformation of a constrained minimization problem into a sequence of unconstrained minimization problem. This transformation enables us to use well developed unconstrained optimization techniques to solve the constrained problem without inventing a new technique for such a constrained optimization problem. The method was first proposed by Carroll in 1959 [4,5] and further developed by Fiacco and McCormick [8,9,10,11,17]. In 1964, Fiacco and McCormick developed a general algorithm based on SUMT, and in 1965, they proposed a method which is called SUMT without parameters. By using this method, the difficulty of choosing the penalty parameters can be avoided, although there are still some difficulties exist. There is a general computer program provided by McCormick, Mylander and Fiacco called "RAC Computer Program Implementing the Sequential Unconstrained Minimization Technique for Nonlinear Programming." (IBM SHARE number 3189) [17]. In this computer program, the unconstrained minimization technique used is the second order gradient method.

Difficulties which arise from use of the second order gradient method as a unconstrained minimization technique in SUMT becomes predominate in a large size and/or very complex nonlinear problem. The difficulties arise particularly in taking correctly the first order and second order partial derivatives of very complex nonlinear functions which most of practical problems have. Therefore, a new algorithm which using a much simpler direct search technique is very desirable.

For the above reason, a new technique of implementing SUMT by Hooke and Jeeves pattern search technique to be its unconstrained minimization process is suggested [6] and is developed. The procedures are presented in Chapter 3 in details. Hooke and Jeeves pattern search technique [13,14] is different from the gradient method by the decision making process to decide the direction of search. The direction of search in the gradient method is in the steepest decent direction while that of the Hooke and Jeeves pattern search technique is determined by direct comparison of the values of the objective function at two points depart from each other for a finite step. For this reason, when the pattern search is getting close to the boundary of some inequality constraints, it shall frequently go out of the feasible region bounded by inequality constraints, and the search might be terminated at some point near the boundary which might not be the real constrained optimum. A heuristic programming technique was developed by Paviani and Himmelblau [19], which provides a method for applying a sequential simplex pattern search routine [2,3,6a,18] to a constrained problem. The method enables to make turns at the pattern search near the boundary of constraints. This heuristic idea is employed here in order to handle the boundary of inequality constraints [6]. The details of the method are described in Chapter 3 and a general FORTRAN-IV program together with detailed computer diagrams is presented in Appendix.

This newly developed method is utilized to obtain optimum solutions of two examples of production scheduling and inventory control in chapter 4. The first problem is a simple two dimensional problem used for demonstrating the procedure of the algorithm in details and the second problem is a 20-dimensional problem used for demonstrating the capacity and practicability of the technique. Both problems have previously been solved by using the RAC program introduced before [15].

Much has been written about the optimization of the reliability of a system. Usually the increase in the system reliability is due to adding redundancies. Previously, with redundant components in parallel or in series were considered [7, 23, 24, 25, 26]. The problem becomes considerably more difficult when the redundant units of the system cannot be reduced to parallel or series configurations.

In attempting to optimize the reliability of such a complex system a major difficulty is encountered in that the reliability expression is not a separable function and thus cannot be analyzed as a multistage process. Thus another approach is used to solve this type of problem where the reliability is obtained by Bayes' theorem which utilizes conditional probabilities [1]. With this in mind a mathematical model for the nonlinear system reliability subject to constraints is formulated. The nonlinear programming problem of optimizing the system reliability is then solved by SUMT using RAC computer program [17] in Chapter 2.

The same reliability problem is also solved by the newly developed technique and the results are presented in Chapter 5. Far less preparatory work is required and the partial derivatives of objective function and functions of inequality and equality constraints are not needed. By comparing the results with that obtained in Chapter 2, we can conclude that the newly developed technique is workable and much simpler than the original technique mentioned is. Thus the new technique is capable of solving a wide range of practical optimization problems.

REFERENCES

1. Bazovsky, I., Reliability Theory and Practice, Englewood Cliffs, N. J.: Prentice-Hall, 1961.

2. Box, M. J., "A New Method of Constrained Optimization and a Comparison with Other Methods," Computer Journal, Vol. 8, 1965.

3. Carpenter, B. H., and H. C. Sweeny, "Process Improvement with Simplex Self-Directing Evolutionary Operation", Chemical Engineering, July 5, 1965.

4. Carroll, C. W., "An Operations Research Approach to the Economic Optimization of a Kraft Pulping Process," Ph.D. Dissertation, Institute of Paper Chemistry, Appletown, Wisc., 1959.

5. Carroll, C. W., "The Created Response Surface Technique for Optimizing Nonlinear Restrained Systems," Operations Research, 9, 169-184 (1961).

6. Fan, L. T., Private Communication, 1969.

6a. Fan, L. T., C. L. Hwang, and F. A. Tillman, "A Sequential Simplex Pattern Search Solution to Production Planning Problems," AIIE Trans. Vol. 1, No. 3, pp. 267-272 (1969).

7. Fan, L. T., C. S. Wang, F. A. Tillman, and C. L. Hwang, "Optimization of Systems Reliability," IEEE Trans. Reliability, Vol. R-16, pp. 81-86, September 1967.

8. Fiacco, A. V., and G. P. McCormick, "The Sequential Unconstrained Minimization Technique for Nonlinear Programming: A Primal-Dual Method," Management Sci., 10, 360-366 (1964).

9. Fiacco, A. V., and G. P. McCormick, "Computational Algorithm for the Sequential Unconstrained Minimization Technique for Nonlinear Programming," Management Sci., 10, 601-617 (1964).

10. Fiacco, A. V., and G. P. McCormick, "Extension of SUMT for Nonlinear Programming: Equality Constraints and Extrapolation," Management Sci., 12 (11): 816-829 (1966).

11. Fiacco, A. V., and G. P. McCormick, Nonlinear Programming: Sequential Unconstrained Minimization Techniques, Wiley, New York, 1968.

12. Fletcher, R. and M. J. D. Powell, "A Rapidly Convergent Descent Method for Minimization," Computer J., 6, 163-168 (1963).

13. Hooke, R., and T. A. Jeeves, Direct Search Solution of Numerical and Statistical Problems, J. Assoc. Compt. Mach., 8, 212 (1961).

14. Hwang, C. L., L. T. Fan, and S. Kumar, "Hooke and Jeeves Pattern Search Solution to Optimal Production Planning Problems" Report No. 18, Institute for Systems Design and Optimization, Kansas State University, 1969.

15. Hsu, F. T., L. T. Fan and C. L. Hwang, "Sequential Unconstrained Minimization Technique (SUMT) for Optimal Production Planning", paper summitted for publication, 1969.

16. Kowalik, J., and M. R. Osborne, Methods for Unconstrained Optimization Problems, Elsevier, New York, 1968.

17. McCormick, G. P., W. C. Mylander, III., and A. V. Fiacco, "RAC Computer Program Implementing the Sequential Unconstrained Minimization Technique for Nonlinear Programming," SHARE Number 3189.

18. Nelder, J. A., and R. Mead, "A Simplex Method for Function Minimization," Computer Journal, Vol. 7, 1965.

19. Paviani, D. A., and D. M. Himmelblau, "Constrained Nonlinear Optimization by Heuristic Programming", Paper Presented in AIChE meeting, New Orleans, March, 1969.

20. Powell, M. J. D., "An Efficient Method of Finding the Minimum of a Function of Several Variables without Calculating Derivatives," Computer J., 7, 155-162 (1964).

21. Powell, M. J. D., "A Method for Non-Linear Constraints in Minimization Problems," T.P. 310, Atomic Energy Research Establishment, Harwell, England, 1967.

22. Spang, H. A. III, "A Review of Minimization Techniques for Nonlinear Functions," SIAM, Rev., 4, 343-365 (1962).

23. Tillman, F. A., "Optimization of Constrained Reliability Problems with Several Modes of Failures by Integer Programming," IEEE Trans. Reliability, Vol. R-18, pp. 47-53, 1969.

24. Tillman, F. A., C. L. Hwang, L. T. Fan and S. A. Balbale, "Systems Reliability Subject to Multiple Nonlinear Constraints," IEEE Trans. Reliability, Vol. R-17, pp. 153-157, September 1969.

25. Tillman, F. A., C. L. Hwang, L. T. Fan and K. C. Lai, "Optimal Reliability of a Complex System," paper submitted for publication, Institute for Systems Design and Optimization, Kansas State University, 1969.

26. Tillman, F. A., and J. Liittschwager, "Integer Programming Formulation of Constrained Reliability Problems," Management Science, Vol. 13, pp. 887-899, 1967.

CHAPTER 2

OPTIMAL RELIABILITY OF A COMPLEX SYSTEM

2.1 INTRODUCTION

Much has been written about the optimization of the reliability of a system. Usually these problems are concerned with optimizing some objective function subject to constraints where the increase in the system reliability is due to adding redundancies. In previous work, the systems treated usually have redundant components in parallel or in series (4, 14, 15, 16). The problem becomes considerably more difficult when the redundant units of the system cannot be reduced to parallel and series configurations. One such example is shown in Fig. 1. In the system, unit 1 is backed up by a parallel unit 4. There are two equal paths, where each path has unit 2 in series with the stage formed by unit 1 and unit 4. These two equal paths operate in parallel so that if at least one of them is good the output is assured. However, because unit 2 does not have a high degree of reliability, a third unit, unit 3, is inserted into the circuit. Therefore, the following operations are possible: 2-1, 2-4, 3-1, and 3-4, and each operation has two equal paths.

In attempting to optimize the reliability of such a configuration a major difficulty is encountered in that the reliability expression is not a separable function† and thus cannot be analyzed as a multistage process. Thus another approach is used to solve this type of problem where the

---

†A function is separable if $f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} f(x_i)$.

reliability is obtained by Bayes' theorem which utilizes conditional probabilities (1). With this in mind a mathematical model for the non-linear system reliability subject to constraints is formulated. The nonlinear programming problem of optimizing the system reliability is then solved by the sequential unconstrained minimization technique (SUMT) (5, 6, 7, 8). This method appears to be one of the more efficient methods of solving constrained nonlinear optimization problems.

2.2 SYSTEM RELIABILITY USING CONDITIONAL PROBABILITIES

In a complex system where the redundant units cannot be reduced to a parallel or series configuration the reliability is obtained by using Bayes' Theorem involving conditional probabilities, Bazovsky [1].

In solving this problem, a simplified form of Bayes' probability theorem is used. The theorem says that if A is an event which depends on one of two mutually exclusive events B_i and B_j of which one must necessarily occur, then the probability of the occurence of A is given by

$$P(A) = P(A, \text{ given } B_i) \cdot P(B_i) + P(A, \text{ given } B_j) \cdot P(B_j) \tag{1}$$

To put this theorem in the context of a reliability problem, let us denote the event of a system's failure by A and the survival by B_i and the failure by B_j of a component or unit on whose operation the system reliability depends. The probability of system failure P(A), then, equals the probability of system failure given that a specified component in the system is good, P(A, given B_i), times the probability that the component is good, P(B_i), plus the probability of system failure given that the component is bad, P(A, given B_j), times the probability that the component is bad, P(B_j). Thus if K is a component upon whose state, whether good or bad, the system reliability depends, we say that the probability of system failure, P (system failure), is equal to

$$\text{P(System failure given component K is good)} \cdot \text{P(K is good)}$$
$$+ \text{ P(System failure given component K is bad)} \cdot \text{P(K is bad)}. \qquad (2)$$

Let Q_s represent the probability of system failure, R_k the probability that component K is good, and Q_k the probability that component K is bad, then we obtain the usual expression for system unreliability

$$Q_s = Q_s(\text{given K is good}) \cdot R_k + Q_s(\text{given K is bad}) \cdot Q_k. \qquad (3)$$

The system reliability, R_s, is then

$$R_s = 1 - Q_s \qquad (4)$$

Equation (3) now enables us to calculate the reliability of complex systems. To illustrate we will obtain the reliability of the system presented in Fig. 1. Component 3 for K is selected for the key component in equation (3), thus we have the expression for system unreliability

$$Q_s = Q_s(\text{if 3 is good}) \cdot R_3 + Q_s(\text{if 3 is bad}) \cdot Q_3. \qquad (5)$$

If component 3 is good the system can fail if the two paths, which contain unit 2 in series with the stage formed by units 1 and 4 in parallel, fail. With these two paths in parallel, the system's unreliability, given unit 3 is good, is

$$Q_s(\text{if 3 is good}) = [(1-R_1)(1-R_4)]^2. \qquad (6)$$

If on the other hand unit 3 is bad the system will fail only if both parallel paths fail, and the system's unreliability, if 3 is bad, is

$$Q_s(\text{if 3 is bad}) = \left\{1 - R_2[1 - R_2[1 - (1-R_1)(1-R_4)]\right\}^2 \qquad (7)$$

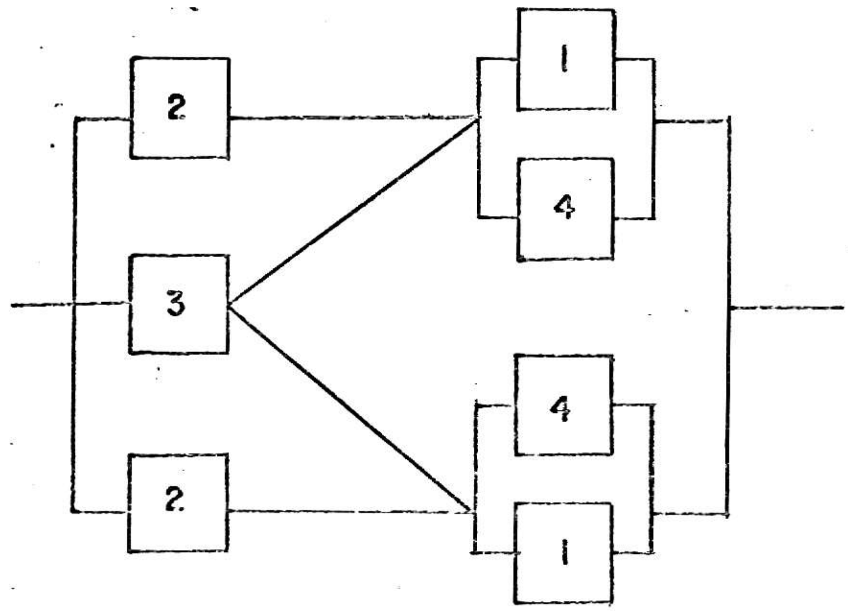

Fig.1. A schematic diagram of a complex system.

where $\{1 - R_2[1 - (1-R_1)(1-R_4)]\}$ is the unreliability of the path which has unit 2 in series with the stage formed by units 1 and 4.

Using equation (5) the unreliability of the system is

$$Q_s = [(1-R_1)(1-R_4)]^2 \cdot R_3 + \{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2 \cdot (1-R_3). \quad (8)$$

The system reliability is given by equation (4).

2.3 FORMULATION OF AN OPTIMIZATION PROBLEM

The problem of maximizing the reliability of the complex system given in Fig. 1 which is subject to a single constraint can be stated as follows:

Maximize

$$\begin{aligned} R_s &= 1 - Q_s \\ &= 1 - R_3[(1-R_1)(1-R_4)]^2 \\ &\quad - (1-R_3)\{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2 \end{aligned} \quad (9)$$

subject to

$$\sum_i C_i \leq C \quad (10)$$

where

$$C_i = K_i R_i^{\alpha_i} \quad (11)$$

The system reliability, R_s, given by equation (9) can be obtained from equations (4) and (8). The constraint given by equation (10) can be interpreted as follows: C_i can represent the weight, the cost, or the volume of each unit or component of the system, and the summation of the weight, the cost, or the volume of the system must be less than C. The weight, cost, or volume of each unit or component of the system is a function of reliability which can be expressed by equation (11), where K_i is a proportionality constant and α_i, the exponential factor, relates C_i and the reliability. Usually α_i is less than one.

The solution of the above constrained nonlinear programming problem can be obtained by the technique which is described in the following section.

2.4 SEQUENTIAL UNCONSTRAINED MINIMIZATION TECHNIQUE (SUMT)

The general nonlinear programming problem with nonlinear inequality constraints is one where x is selected to

$$\left.\begin{array}{l} \quad \text{minimize } f(x) \\ \text{subject to} \\ \quad g_i(x) \geq 0, \qquad i = 1, 2, \ldots, m \end{array}\right\} \tag{12}$$

where x is an n-dimensional column vector $(x_1, x_2, \ldots, x_n)^T$. The superscript T denotes transposition. If the variables are required to be non-negative, such constraints are included in the g_i's. The functions, f(x) and $g_i(x)$, $i = 1, 2, \ldots, m$, can take a linear or nonlinear form.

The following algorithm is presented [5, 6, 7, 8] to solve this problem. First define the function (called the P function)

$$P(x, r_k) = f(x) + r_k \sum_{i=1}^{m} \frac{1}{g_i(x)} \tag{13}$$

where r_k is a positive constant. The subscript k indicates the number of times the P function has been solved. The conditions imposed on the P function are as follows:

(1) r_k, $k = 1, 2, \ldots$, is a positive real number and $r_1 > r_2 > \ldots > r_k > \ldots > 0$. This indicates that $\{r_k\}$ is a strictly monotonic

decreasing sequence and $r_k \to 0$ as $k \to \infty$.

(2) $R^0 = \{x \mid g_i(x) > 0, \ i = 1, 2, \ldots, m\}$ is non-empty. This condition indicates that at least one point must exist within the interior of the feasible region.

(3) The functions $f(x)$, $g_1(x)$, ..., $g_m(x)$ are twice continuously differentiable.

(4) The function $f(x)$ is convex.

(5) The functions $g_1(x)$, ..., $g_m(x)$ are concave.

(6) For every finite M, $\{x \mid f(x) \leq M; x \varepsilon R\}$ is a bounded set, where $R = \{x \mid g_i(x) \geq 0, \ i = 1, 2, \ldots, m\}$.

(7) The function $P(x, r_k) = f(x) + r_k \sum_{i=1}^{1} \frac{1}{g_i(x)}$ is, for each $r > 0$, strictly convex for $x \varepsilon R^0$. This also indicates that either $f(x)$ is strictly convex or one of the functions g_1, ..., g_m is strictly concave.

Practical experience indicates that the problems given by equation (1) can be solved even when these conditions are not met. The three conditions which are absolutely required to obtain any useful results are conditions (1), (2), and (6). Condition (1) guarantees that the sequential minimization of the P function will eventually lead to the solution of minimization of function f(x). Condition (2) eliminates problems with equality constraints. Condition (6) eliminates problems having local minimum at infinite points.

The characteristics of the P function are as follows:

(1) $\lim_{k\to\infty} r_k \sum_{i=1}^{m} \frac{1}{g_i(x)} = 0$,

(2) $\lim_{k\to\infty} f[x(r_k)] = u^*$,

(3) $\lim_{k\to\infty} P[x(r_k), r_k] = u^*$,

(4) $\{f[x(r_k)]\}$ is a monotonically decreasing sequence,

(5) $\{\sum_{i=1}^{m} \frac{1}{g_i(x)}\}$ is a monotonically increasing sequence.

The proofs of these characteristics are presented in detail by Fiacco and McCormick [5, 6, 7, 8].

Intuitive Concept of P Function

The term $r_k \sum_{i=1}^{m} \frac{1}{g_i(x)}$ in the P function of equation (13) can be considered as a penalty factor attached to the objective function f(x). By adding the penalty term, the minimization of the P function will assure a minimum to be in the interior of the inequality constrained region by avoiding crossing the boundaries of the feasible region. Since the feasible boundary is defined by one or more of the $g_i(x) = 0$, $i = 1, \ldots, m$, the value of $r_k \sum_{i=1}^{m} \frac{1}{g_i(x)}$ will approach infinity as the value of x approaches one of the boundary lines. Hence the value of x will tend to remain inside the inequality-constrained region.

The motivation behind this formulation of the P function is the transformation of the original constrained problem into a sequence of unconstrained minimization problems. The desirability of this transformation lies in the fact that numerous methods for minimizing an unconstrained function are known and newer methods are continually being developed [2, 3, 9, 10, 11, 13].

Computational Procedure

The procedure for using SUMT is summarized below [5, 6].

(1) Select the initial value of r_0 arbitrarily or use the formula for the selection r_0, which is available in reference [6].

(2) Select a feasible starting point $x^0 = (x_1^0, x_2^0, \ldots, x_n^0)$. If the feasible point can not be easily obtained, select x^0 arbitrarily. The computer program [12] will minimize the following P function and obtain a feasible point.

$$P(x, r_k) = -g_s(x) + r_k \sum_{t \varepsilon T} \frac{1}{g_t(x)}$$

where $g_s(x^0) \leq 0$ and $T = \{t \mid g_t(x^0) > 0\}$. Note that the constraint function $g_s(x) \geq 0$ is violated.

(3) Minimize the P function for the current value of r_k by using the second-order optimum gradient method.

(4) Check to see if the stopping criterion such as

$$\frac{f[x(r_k)]}{G[x(r_k)]} - 1 < \varepsilon \tag{14}$$

is satisfied. If it is satisfied the solution is optimal; otherwise go to step 5. The dual function, $G[x(r_k)]$, is defined as [5]

$$G[x(r_k)] = f[x(r_k)] - r_k \sum_{i=1}^{m} \frac{1}{g_i[x(r_k)]} \tag{15}$$

(5) Set $k = k+1$ and $r_{k+1} = r_k/C$, where $C > 1$. Repeat the iteration from step 3.

The procedures described above must satisfy two stopping criteria before any meaningful optimal solution can be obtained. The stopping criterion used for terminating the minimization of the P function [Step 3] may be one of the following

$$\text{(i)} \qquad \left|\nabla_x P^T(x,r) \left|\frac{\partial^2 P(x,r)}{\partial x_i \, \partial x_j}\right|^{-1} \nabla_x P(x,r)\right| < \varepsilon' \tag{16a}$$

or

$$\text{(ii)} \qquad \left|\nabla_x P^T(x,r) \left|\frac{\partial^2 P(x,r)}{\partial x_i \, \partial x_j}\right|^{-1} \nabla_x P(x,r)\right| < \frac{P(x,r_{k-1}) - P(x,r_k)}{5} \tag{16b}$$

or

$$\text{(iii)} \qquad \left|\nabla_x P(x,r)\right| < \varepsilon' \tag{16c}$$

The first stopping criterion was used throughout this study with ε' in the range of 10^{-3} to 10^{-5}. The stopping criterion for terminating overall minimization of $f[x(r_k)]$ may take the following form in addition to the form given by equation (14).

$$r_k \sum_{i=1}^{m} \frac{1}{g_i[x(r_k)]} < \varepsilon \tag{17}$$

The first form equation (14), was used in the numerical examples presented in this work with ε generally ranging from 10^{-3} to 10^{-5}. The procedure should not be terminated until both criteria given by equations (14) and (16) are satisfied. If these stopping criteria are not satisfied within a specified time limit, the iterations should be terminated.

We used a computer program entitled "RAC Computer Program Implementing the Sequential Unconstrained Minimization Technique for Nonlinear Programming" which is available for solving the example problems. Its

SHARE number is 3189 [12]. The program is written in FORTRAN IV and can be used on IBM 360. With minor modifications the program can be run on any sufficiently large computer with a FORTRAN compiler.

2.5 A NUMERICAL EXAMPLE

The nonlinear programming problem formulated in the preceding section is restated again and the objective is to maximize

$$R_s = 1 - R_3[(1-R_1)(1-R_4)]^2$$

$$- (1-R_3)\ \{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2 \tag{18}$$

subject to the constraint

$$2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + 2K_4R_4^{\alpha_4} \leq C\ . \tag{19}$$

The constants K_1, K_2, K_3, and K_4, the constraint, C, and the exponential constant α_i, i = 1, 2, 3, 4, are as follows:

$K_1 = 100$, $K_2 = 100$, $K_3 = 200$, $K_4 = 150$,

$C = 800$, $\alpha_i = 0.6$, $i = 1, 2, 3, 4$.

The problem is formulated in SUMT format as follows:

Minimize

$$f(x) = -R_s$$

$$= -1 + R_3[(1-R_1)(1-R_4)]^2 + (1-R_3)\{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2$$

subject to the constraints

$$g_1(x) = C - (2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + K_4R_4^{\alpha_4}) \geq 0$$

$$g_{i+1}(x) = 1 - R_i \geq 0, \qquad i = 1, 2, 3, 4$$

The P function of equation (13) is

$$P(x,r_k) = -1 + R_3[(1-R_1)(1-R_4)]^2 + (1-R_3)\{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2$$

$$+ r_k \left[\frac{1}{C - (2K_1R_1^{\alpha_1}+2K_2R_2^{\alpha_2}+K_3R_3^{\alpha_3}+K_4R_4^{\alpha_4})} + \sum_{i=1}^{4} \left(\frac{1}{1-R_i}\right)\right]$$

The optimal solutions which were obtained by starting from two different points, namely, $[R_1, R_2, R_3, R_4] = [0.7, 0.7, 0.7, 0.7]$ and $[R_1, R_2, R_3, R_4] = [0.6, 0.6, 0.6, 0.6]$ are presented in Table 1. The solutions are almost identical, that is, the optimal system reliability is R_s equal to 0.99996 with the cost of 799.78 for the first starting point and R_s equal to 0.99995 with the cost of 799.28 for the second starting point. Recall that the constraint on the cost is 800. Note that the optimal components reliabilities are almost the same for both starting points. The stopping criterion for terminating the minimization of the P function at each k iteration is $\varepsilon' = 10^{-5}$, and the stopping criterion for terminating the over all minimization of $f[x(r_k)]$ is $\varepsilon = 10^{-4}$. For the first starting point, it required 10 iterations for the P functions with a total of 152 functional values calculated, and for the second point, 11 iterations were required for the P functions with a total of 167 functional values calculated.

TABLE 1. Optimal Solution

Iteration k	Number of functional value calculated	R_1	R_2	R_3	R_4	System Reliability R_s	Cost	Stopping criterion: Stop for each k ε'	Stopping criterion: final stop ε
0		0.7	0.7	0.7	0.7				
10	152	0.9876	0.9936	0.6972	0.6941	0.99996	799.78	10^{-5}	10^{-4}
0		0.6	0.6	0.6	0.6				
11	167	0.9889	0.9921	0.7019	0.6886	0.99995	799.28	10^{-5}	10^{-4}

Tables 2a and 2b present some suboptimal solutions according to different stopping criterions. From these tables, we can see that the number of iterations, k, is dictated by the final stopping criterion, ε, and that the number of functional values calculated for each iteration is dictated by the stopping criterion for each iteration, ε'. The number of iterations, k, increases from 4 for $\varepsilon = 10^{-2}$ to 10 for $\varepsilon = 10^{-4}$, and the number of functional values calculated for each iteration increases from an average of 1 for $\varepsilon' = 10^{-2}$ to an average of 14 for $\varepsilon' = 10^{-4}$. Although the cost for each suboptimum solution is near the cost constraint of 800, the systems reliability and corresponding set of components reliabilities are different for each combination of ε' and ε. The highest system reliability is obtained when the stopping criterions are $\varepsilon' = 10^{-5}$ and $\varepsilon = 10^{-4}$.

Results given in Tables 3a and 3b show that the system reliability, R_s, is monotonically increasing as the iteration k increases. The value of the P function approaches that of the f function ($= -R_s$) as the iterations proceed. Thus the minimization of the P function will eventually lead to the minimization of f function.

2.6 DISCUSSION

This approach provides a practical method for solving a very complex reliability problem. The system may be one where the redundant components cannot be reduced to a parallel or series configuration. The reliability function is obtained by using Bayes' theorem and a mathematical model is formulated for the constrained nonlinear programming problem. The solution of the problem is obtained by the sequential unconstrained minimization technique (SUMT). As is evident from the results obtained

TABLE 2a. Suboptimal solutions according to different stopping criterion

Stopping criterion		Iteration K	Number of functional value calculated	R_1	R_2	R_3	R_4	System Reliability R_s	Cost
Stop for each k ε'	final stop ε								
		0		0.7	0.7	0.7	0.7	0.9584	726.61
10^{-4}	10^{-2}	4	27	0.8186	0.9222	0.7690	0.7843	0.9958	798.02
10^{-4}	10^{-3}	6	105	0.9333	0.9722	0.6937	0.7336	0.9992	798.23
10^{-2}	10^{-4}	9	10	0.8156	0.9147	0.7715	0.7991	0.9957	799.95
10^{-3}	10^{-4}	9	79	0.8091	0.9620	0.7512	0.7876	0.9973	799.94
10^{-4}	10^{-4}	10	140	0.9527	0.9926	0.6850	0.7242	0.99975	799.95
10^{-5}	10^{-4}	10	152	0.9876	0.9936	0.6972	0.6941	0.99996	799.78
10^{-2}	10^{-5}	11	12	0.8156	0.9149	0.7715	0.7991	0.9957	799.99

TABLE 2b. Suboptimal solutions according to different stopping criterion

Stopping criterion		Iteration k	Number of functional value calculated	R_1	R_2	R_3	R_4	System Reliability R_s	Cost
Stop for each k ε'	final stop ε								
		0		0.6	0.6	0.6	0.6		
10^{-4}	10^{-3}	8	146	0.9420	0.9834	0.6861	0.7344	0.9995	799.76
10^{-2}	10^{-4}	10	12	0.7989	0.9139	0.7731	0.8096	0.9955	799.95
10^{-3}	10^{-4}	10	55	0.8068	0.9650	0.7488	0.7888	0.9974	799.94
10^{-4}	10^{-4}	11	174	0.9499	0.9929	0.6808	0.7285	0.99974	799.94
10^{-5}	10^{-4}	11	167	0.9889	0.9921	0.7019	0.6886	0.99995	799.28
10^{-2}	10^{-5}	12	14	0.7990	0.9141	0.7731	0.8096	0.9956	799.99

TABLE 3a. Computer results of a suboptimal results for stopping criterion ($\epsilon' = 10^{-2}$ and $\epsilon = 10^{-5}$)

Iteration k	Number of functional value calculated at each iteration	Value of r_k	R_1	R_2	R_3	R_4	$-P$	$-f$ ($=R_s$)	Cost
0		3000	0.7	0.7	0.7	0.7		0.95480	726.61
1	1	0.01427	0.6877	0.7504	0.6687	0.6723	0.6850	0.9577	721.61
2	2	0.003568	0.7475	0.8223	0.7200	0.7334	0.9019	0.9815	759.10
3	2	0.000892	0.8012	0.8764	0.7594	0.7860	0.9682	0.9924	789.08
4	1	0.000223	0.8139	0.9095	0.7708	0.7980	0.9883	0.9953	798.77
5	1	0.0000558	0.8150	0.9125	0.7713	0.7988	0.9937	0.9956	799.52
6	1	0.00001394	0.8153	0.9138	0.7715	0.7990	0.9951	0.99564	799.78
7	1	0.000003485	0.8155	0.9143	0.7715	0.7991	0.9955	0.99567	799.90
8	1	0.0000008712	0.8156	0.9147	0.7715	0.7991	0.9956	0.99569	799.95
9	1	0.000000218	0.8156	0.9148	0.7715	0.7991	0.9957	0.995702	799.98
10	1	0.000000054	0.81563	0.9149	0.7715	0.7991	0.9957	0.995706	799.99

TABLE 3b. Computer results of a suboptimal results for stopping criterion ($\epsilon' = 10^{-2}$ and $\epsilon = 10^{-5}$)

Iteration k	Number of functional value calculated at each iteration	Value of r_k	R_1	R_2	R_3	R_4	$-P$	$-f$ ($=R_s$)	Cost
0		3000	0.6	0.6	0.6	0.6	0.0	0.8862	786.24
1	1	0.066595	0.5869	0.6330	0.5861	0.5828	-0.2231	0.8888	785.95
2	2	0.01665	0.6556	0.7357	0.6617	0.6669	0.6406	0.9502	712.98
3	2	0.004162	0.72695	0.8136	0.7172	0.7372	0.8886	0.9794	755.56
4	2	0.001041	0.7829	0.8706	0.7587	0.7933	0.9640	0.9916	787.26
5	1	0.0002601	0.7971	0.9085	0.7722	0.8080	0.9870	0.9952	798.62
6	1	0.000065	0.7983	0.9117	0.7729	0.8091	0.9933	0.9954	799.48
7	1	0.0000163	0.7987	0.9130	0.7730	0.8094	0.9950	0.9955	799.76
8	1	0.000004065	0.7988	0.9136	0.7731	0.8095	0.9954	0.9955	799.89
9	1	0.00000102	0.799	0.9139	0.7731	0.8096	0.9955	0.99554	799.95
10	1	0.00000025	0.7989	0.9141	0.7731	0.8096	0.99553	0.99555	799.97
11	1	0.000000064	0.79895	0.9144	0.7731	0.8096	0.99554	0.99552	799.99

in solving the example problem this is an efficient method for solving a difficult problem.

The complex reliability system presented in Fig. 1. can be identified to many practical systems concerning with the space life support systems. One such example is a communication system of a two man space capsule as shown in Fig. 1. The unit 2 represents each of the two microphones of the headsets of each astronaut in the capsule. Unit 3 is a hand microphone which may be picked up by either astronaut. There are two different type of amplifiers in the system with units 1 and 4 respectively. Such a system is identical to that we have studied in this chapter.

Another example is a high pressure oxygen supply system as shown in Fig. 2. The high pressure oxygen in the cabin is supplied through a system of regulators and valves from a high pressure oxygen storage tank. There are two pairs of the sub-systems of check valves, shut-off valves and non-return automatic shut-off valves in the system. The function of these valves is to stop the reverse flow of air from the cabin to the gas tank in case of pressure drop and to close the line supply if there is same sudden pressure drop in header line or the cabin in order to avoid the wastage of the gas.

Each pair of the valve systems consists of two alternative branches. One consists of a non-return automatic emergency shut off valve, and the other consists of a check valve and a shut off valve in series. Any branch of the two pairs (totally four branches) is capable of supplying sufficient gas to the cabin.

There are three alternative pathes between the 0_2 tank and the pairs of valves. The 0_2 can pass through either of the two regulator to

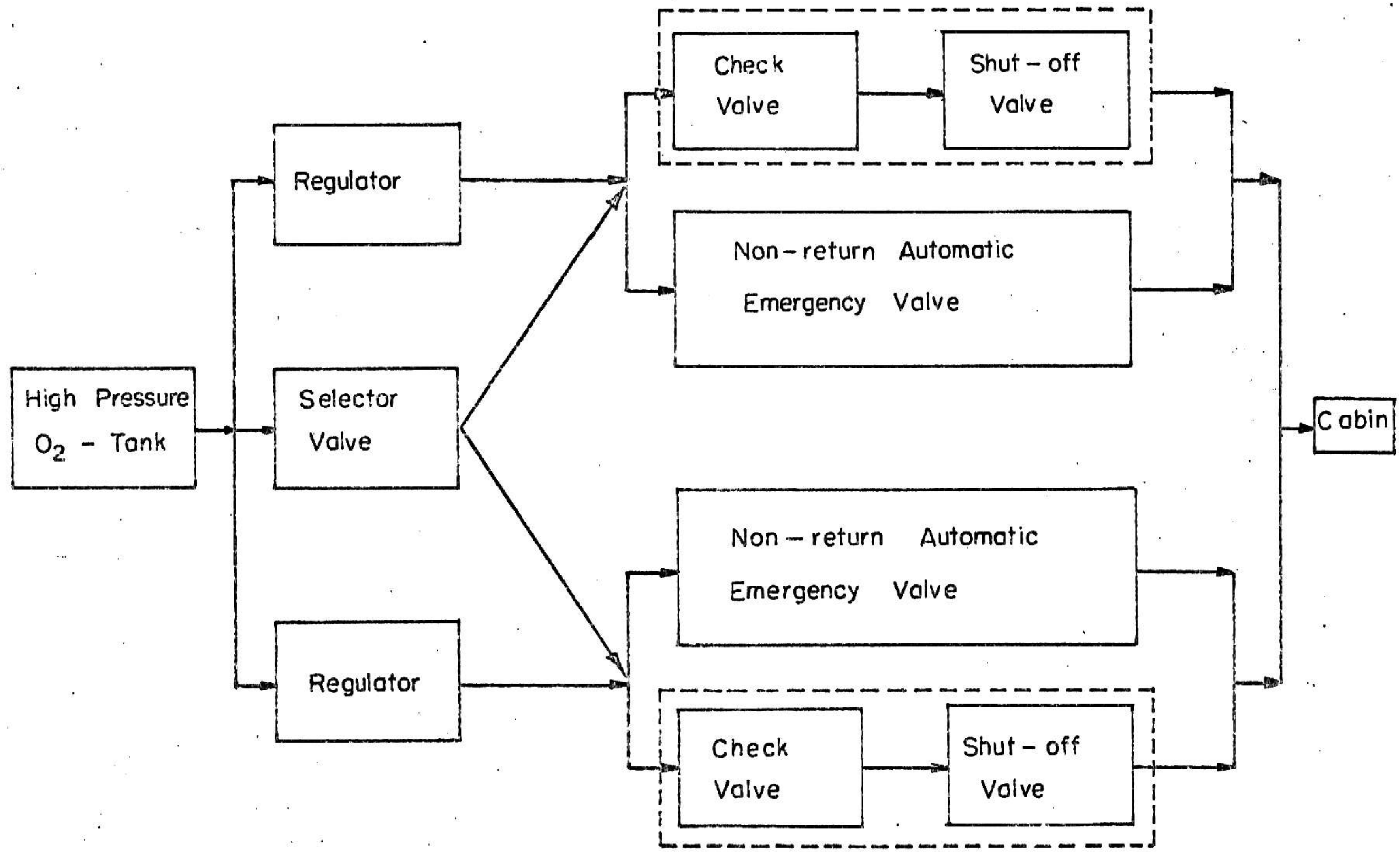

Fig. 2. High pressure O_2 supply system of a spacecraft life support system.

the pair of valves connected to that regulator then supply to the cabin. It also can pass through a selector valve to either of the two pairs of valves then supply to the cabin.

Suppose the reliability of the high pressure O_2 tank can be considered as 1, and denote the reliability for the regulators (they are the same kind of regulators and have the same reliability) by R_2, the reliability for the selector valve by R_3; the reliability for the non-return automatic emergency valve by R_1; and the reliability for the series of check valve and shut-off valve by R_4. Then the system can be reduced to the system presented in Fig. 1 which has been studied in this chapter.

By grouping all the parallel as well as series parts of a complex reliability system into local sub-systems in the whole system and treating them as single components, the system can often be reduced to such configuration that Bayes' theorem of conditional probability shall be able to be employed.

REFERENCES

1. Bozovsky, I., Reliability Theory and Practice, Englewood Cliffs, N.J.: Prentice-Hall, 1961.

2. Davidon, W. C., "Variable Matric Method for Minimization," Research and Development Report ANL-5990 (Revised), Argonne National Laboratory, U.S. Atomic Energy Commission, 1959.

3. Fan, L. T., C. L. Hwang, and F. A. Tillman, "A Sequential Simplex Pattern Search Solution to Production Planning Problems," AIIE Trans. Vol. 1, No. 3, pp. 267-272 (1969).

4. Fan, L. T., C. S. Wang, F. A. Tillman, and C. L. Hwang, "Optimization of Systems Reliability," IEEE Trans. Reliability, Vol. R-16, pp. 81-86, September 1967.

5. Fiacco, A. V., and G. P. McCormick, "The Sequential Unconstrained Minimization Technique for Nonlinear Programming: A Primal-Dual Method," Management Sci., 10, 360-366 (1964).

6. Fiacco, A. V., and G. P. McCormick, "Computational Algorithm for the Sequential Unconstrained Minimization Technique for Nonlinear Programming," Management Sci., 10, 601-617 (1964).

7. Fiacco, A. V., and G. P. McCormick, "Extension of SUMT for Nonlinear Programming: Equality Constraints and Extrapolation," Management Sci., 12 (11): 816-829 (1966).

8. Fiacco, A. V., and G. P. McCormick, Nonlinear Programming: Sequential Unconstrained Minimization Techniques, Wiley, New York, 1968.

9. Fletcher, R. and M. J. D. Powell, "A Rapidly Convergent Descent Method for Minimization," Computer J., 6, 163-168 (1963).

10. Fletcher, R. and C. M. Reeves, "Function Minimization by Conjugate Gradients," Computer J., 7, 149-154 (1964).

11. Kowalik, J., and M. R. Osborne, Methods for Unconstrained Optimization Problems, Elsevier, New York, 1968.

12. McCormick, G. P., W. C. Mylander, III., and A. V. Fiacco, "RAC Computer Program Implementing the Sequential Unconstrained Minimization Technique for Nonlinear Programming," SHARE Number 3189.

13. Powell, M. J. D., "An Efficient Method of Finding the Minimum of a Function of Several Variables without Calculating Derivatives," Computer J., 7, 155-162 (1964).

14. Tillman, F. A., "Optimization of Constrained Reliability Problems with Several Modes of Failures by Integer Programming," IEEE Trans. Reliability, Vol. R-18, pp. 47-53, 1969.

15. Tillman, F. A., C. L. Hwang, L. T. Fan and S. A. Balbale, "Systems Reliability Subject to Multiple Nonlinear Constraints," IEEE Trans. Reliability, Vol. R-17, pp. 153-157, September 1969.

16. Tillman, F. A., and J. Liittschwager, "Integer Programming Formulation of Constrained Reliability Problems," Management Science, Vol. 13, pp. 887-899, 1967.

CHAPTER 3

IMPLEMENTATION OF SEQUENTIAL UNCONSTRAINED MINIMIZATION TECHNIQUE BY HOOKE AND JEEVES PATTERN SEARCH AND HEURISTIC PROGRAMMING

3.1. INTRODUCTION

The general nonlinear programming problem with nonlinear (and/or linear) inequality and/or equality constraints is to choose x to

$$\left.\begin{array}{l} \text{minimize } f(x) \\ \text{subject to} \\ \qquad g_i(x) \geq 0,\ i = 1, 2, \ldots, m \\ \text{and} \\ \qquad h_j(x) = 0,\ j = 1, 2, \ldots, \ell \end{array}\right\} \tag{3.1}$$

where x is an n-dimensional vector $(x_1, x_2, \ldots, x_n)$. To solve this problem, there are a number of techniques developed recently. Among them, a technique which was originally proposed by Carroll [1,2] and further developed by Fiacco and McCormic [3,4,5,6,7] is introduced here.

This technique, known as the sequential unconstrained minimization technique (SUMT), is considered as one of the simplest and most efficient methods for solving the problem given by equation (3.1). The basic scheme of this technique is that a constrained minimization problem is transformed into a sequence of unconstrained minimization problems which can be optimized by any available techniques for solving unconstrained minimization.

The unconstrained minimization technique which is employed here is the well-known Hooke and Jeeves pattern search technique [8,9]. For increasing the efficiency of the method, some modifications have been made.* Among these modifications, a heuristic programming technique [10] is used to handle the inequality constraints of the problem given by equation (3.1).

The method and its computational procedure is illustrated in details in the following sections of this chapter.*

3.2. SEQUENTIAL UNCONSTRAINED MINIMIZATION TECHNIQUE (SUMT)

The SUMT technique for solving the problem given in equation (3.1) is based on the minimization of a function

$$P(x,r_k) = f(x) + r_k \sum_{i=1}^{m} 1/g_i(x) + r_k^{-\frac{1}{2}} \sum_{j=1}^{\ell} h_j^2(x) \tag{3.2}$$

over a strictly monotonic decreasing sequence $\{r_k\}$. Under certain restrictions, the sequence of values of the P function, $P(x,r_k)$, are respectively minimized by a sequence of $\{x(r_k)\}$ over a strictly monotonic decreasing sequence $\{r_k\}$, converges to the constrained optimum values of the original objective function, $f(x)$. The essential requirement is the convexity of the P function.

The intuitive concept of P function is described below:

Since the sequence $\{r_k\}$ is strictly monotonic decreasing, as $r_k \to 0$ the third term of the P function defined in equation (3.2), $r_k^{-\frac{1}{2}} \sum_{j=1}^{\ell} h_j^2(x)$, will approach to ∞ unless $h_j(x) = 0$ for $j = 1,2,\ldots,\ell$. While we are minimizing P function, the formulation of P function in equation (3.2) will force all equality constraints to be zero.

For the second term of the P function, $r_k \sum_{i=1}^{m} 1/g_i(x)$, when we start at a point which is inside the feasible region bounded by inequality

*Developments of this modified method and the computer program for implementing SUMT by the Hooke and Jeeves pattern search technique were not financially supported by any source. The possibility of developing the method and computer program was suggested to the author by Professors L. T. Fan and C. L. Hwang (11).

constraints to minimize the P function, $r_k \sum_{i=1}^{m} 1/g_i(x)$ will approach to infinity as the value of x approaches to one of the boundary of the inequality constraints given by equation (3.1), $g_i(x) \geq 0$. Hence, the value of x will tend to remain inside the inequality-constrained feasible region.

The motivation behind this formulation of P function is the transformation of the original constrained problem into a sequence of unconstrained minimization problem, $\{P(x,r_k)\}$.

The solution to the problem then is to define the P function as shown in equation (3.2) first. To search for the minimum P function value it is started at an arbitrary point which is inside the feasible region bounded by the inequality constraints. After a minimum P function value is reached, the value of r_k is reduced, and a search is repeated again starting from the previous minimum point of the P function. By employing a strictly monotonic decreasing sequence $\{r_k\}$, a monotonic decreasing sequence $\{P_{min}(x,r_k)\}$ inside the feasible region bounded by the inequality constraints is obtained. The equality constraints, $h_j(x) = 0$ for $j = 1,2,\ldots,\ell$, will be satisfied by the nature of the formulation of the P function automatically as r_k tends to zero as explained before.

When $r_k \to 0$, the second term of equation (3.2), $r_k \sum_{i=1}^{m} 1/g_i$ approaches to zero, while the third term, $r_k^{-\frac{1}{2}} \sum_{j=1}^{\ell} h_j^2(x)$, is forced to approach to zero as described before. In other words, as $r_k \to 0$, $P(x,r_k) \to f(x)$, where x is the optimum point which yields the minimum $P(x,r_k)$ and is the optimum point of the problem given in equation (3,1).

Further mathematical proof of the convergence of the method can be seen in reference [3,4,5,6,7].

3.3. COMPUTATIONAL PROCEDURE

The computational procedure for using SUMT with Hooke and Jeeves pattern search technique is summarized below (refer to Fig. 1).

(1) Select a starting point $x^0 = (x_1^0, x_2^0, \ldots, x_n^0)$ and initial values of the penalty coefficient r_k^0, an initial tolerance limit of the violation to constraints, B^0, and the initial step-sizes needed in search processes, d^0.

(2) Select a feasible starting point by minimizing the total weight of violation, if the initial starting point chosen, x^0, is out of the feasible region bounded by the inequality constraints. The total weight of violation, TGH, is defined as [10]

$$TGH = \left(\sum_{t \in T} g_t^2(x^0) + \sum_{s \in R} h_s^2(x^0) \right)^{\frac{1}{2}}$$

where $T = \{t | g_t(x^0) < 0\}$ and $R = \{s | h_s(x^0) \neq 0\}$. Note that TGH includes only the violated constraints.

(3) Define P function as [6,7]

$$P(x, r_k) = f(x) + r_k \sum_i \frac{1}{g_i(x)} + r_k^{-\frac{1}{2}} \sum_j h_j^2(x)$$

where $g_i(x) > 0$, $i = 1, 2, \ldots, m$ are inequality constraints, and $h_j(x) = 0$, $j = 1, 2, \ldots, \ell$ are equality constraints.

(4) Minimize P function by Hooke and Jeeves pattern search technique. After every move during the search, it is checked if the move goes out of the feasible region or not. If the move is out of the

feasible region, got to step 5; if not, after the optimum x is reached for the current $P(x,r_k)$, go to step 6.

(5) Move back to the near-feasible region and then return to step 4. The near-feasible region is defined as the region that all the points in that region satisfy the following condition [10].

$$B - TGH > 0$$

where B is the tolerance limit of violation which is sequentially decreased after every violation to the inequality constraints during the search.

(6) Check if the optimum, $\bar{x}$, obtained in step 4 is inside the feasible region or not. If $\bar{x}$ is feasible, go to step 8, and it it is near-feasible or not feasible, go to step 7.

(7) Move the optimum $\bar{x}$ in the infeasible region into the feasible region along the direction toward the last optimum point, then go to step 8.

(8) Check if a stopping criterion such as

$$\left| \left| \frac{f(x)}{G(x,r_k)} \right| - 1 \right| < \varepsilon$$

is satisfied. The solution is the optimal one if the criterion is satisfied; otherwise, go to step 9. The dual value $G(x,r_k)$, is defined as [6,7]

$$G(x,r_k) = f(x) - r_k \sum_{i=1}^{m} \frac{1}{g_i(x)} + r_k^{-\frac{1}{2}} \sum_{j=1}^{\ell} h_j^2(x)$$

(9) Set $k = k+1$; $r_{k+1} = r_k/C$, where C is a constant and greater than 1; and $d_{k+1} = d^0/(k+1)$, d_{k+1} to be the starting step-sizes; and go

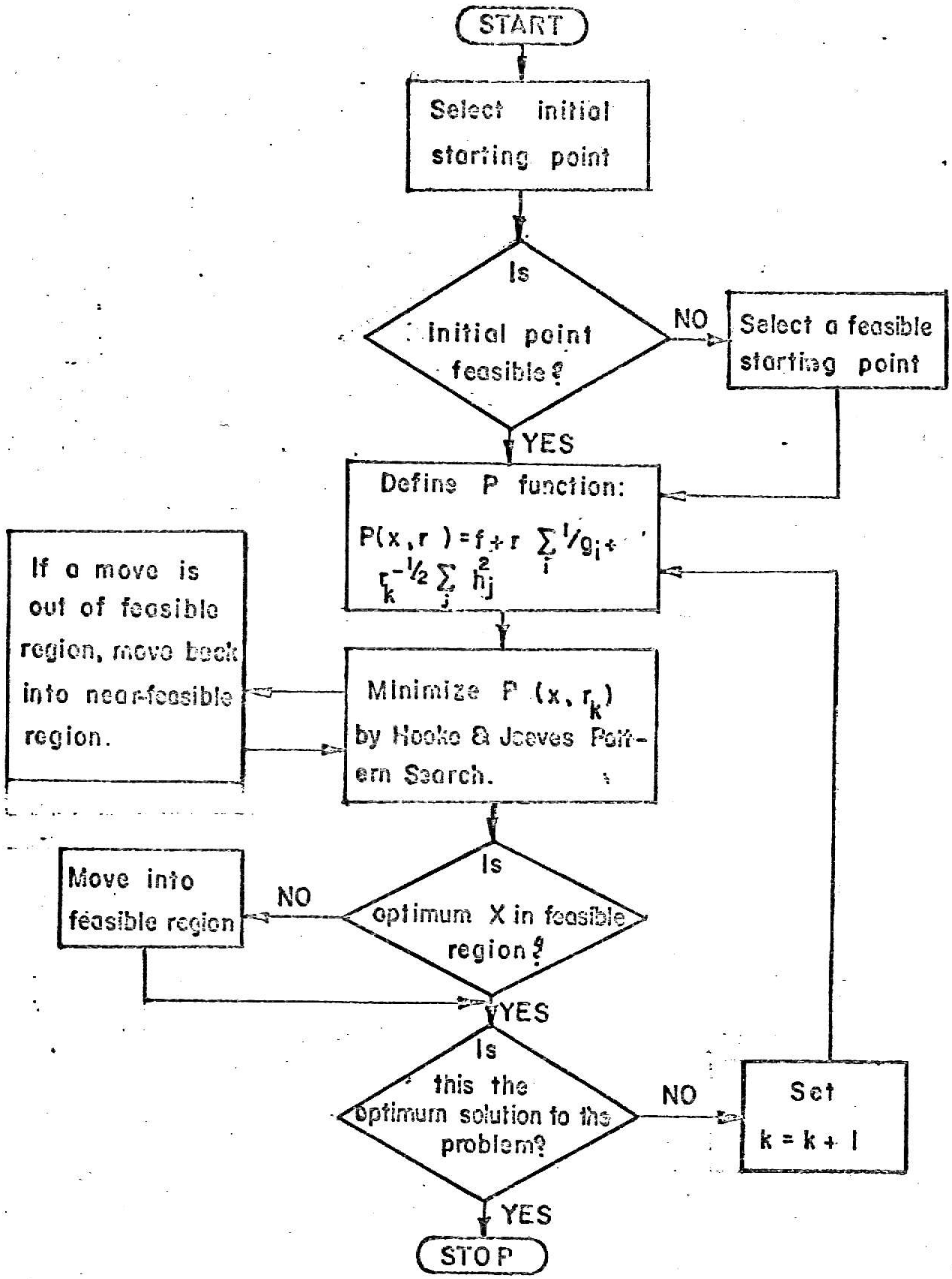

Fig. 1. Descriptive flow diagram for SUMT with Hooke and Jeeves Pattern Search.

back to step 3.

The following sections present in details procedures of each step described above. The basic Hooke and Jeeves pattern search is presented in Section 3.5.

3.4. PROCEDURE FOR SELECTING A FEASIBLE STARTING POINT FROM THE INFEASIBLE INITIAL POINT

The procedure for selecting a feasible starting point when the initial point is out of the feasible region bounded by inequality constraints, $g_i(x) \geq 0$ for $i = 1,2,\ldots, m$, is based on Hooke and Jeeves pattern search technique. For increasing the speed and efficiency of the process, some modifications from the basic Hooke and Jeeves pattern search technique have been made.

Note that in above description of the feasible region only the inequality constraints are included. The violation to equality constraints is not considered here but it is taken into account in the SUMT formulation automatically as explained in Section 3.2 [6,7].

The procedure is summarized below (refer to Fig. 2).

(1) Start at the input initial point, x_0, which is out of the feasible region bounded by the inequality constraints and needs to be moved into the feasible region.

(2) Compute the weight of violation, TGH, at the initial point: [10]

$$TGH = \left\{ \sum_{t \varepsilon T} [g_t(x^0)]^2 + \sum_{s \varepsilon R} [h_s(x^0)]^2 \right\}^{\frac{1}{2}}$$

where $T = \{t | g_t(x^0) < 0\}$ and $R = \{s | h_s(x^0) \neq 0\}$. Note, again, that TGH includes only the violated constraints.

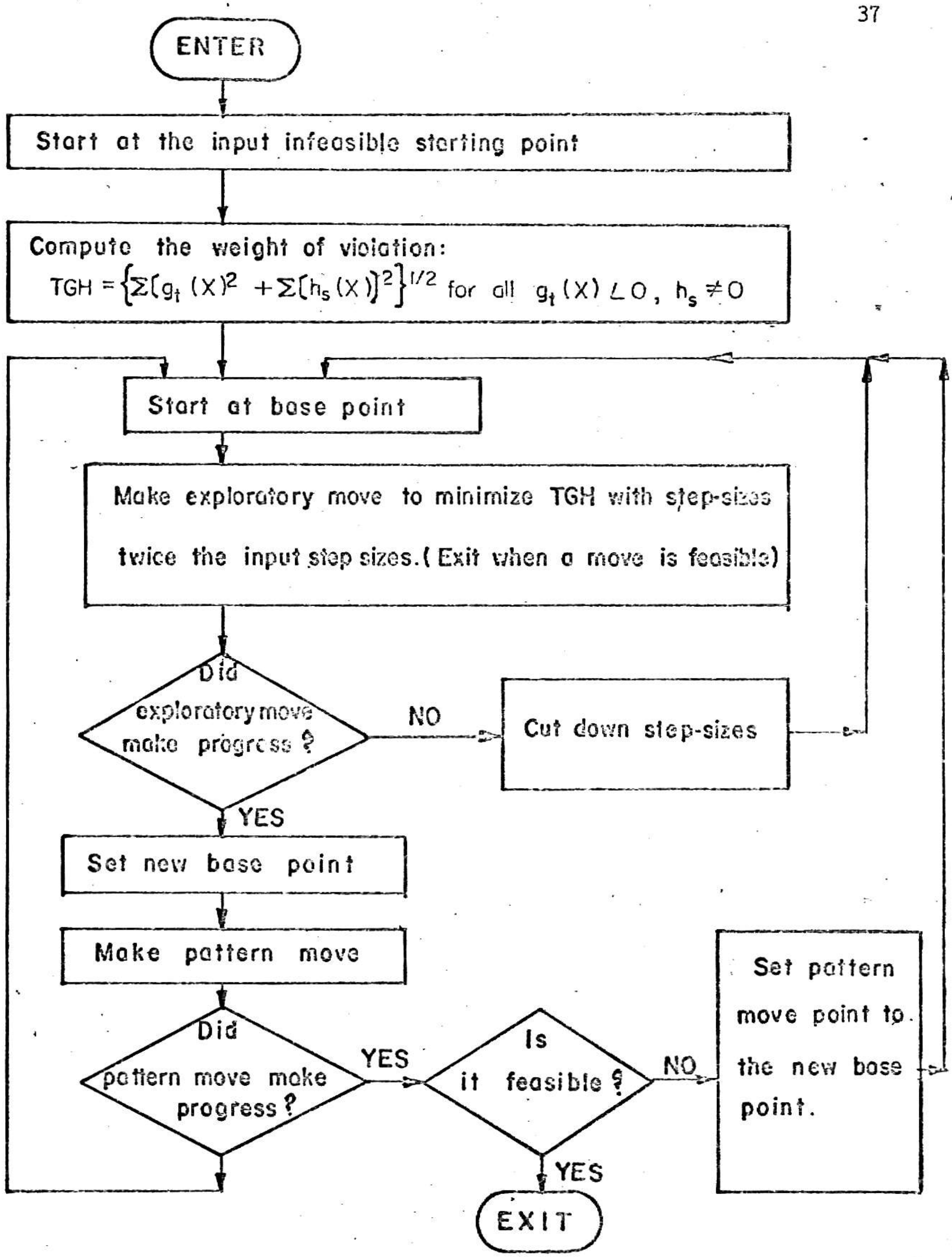

Fig. 2. Descriptive flow diagram for selecting a feasible starting point.

(3) Make an exploratory move to minimize TGH from x^0. Note that, the objective function to be minimized in this step is TGH which has been defined in step 2. For increasing the efficiency of the process, two modifications are made here. First, the starting step-sizes used is twice the input initial starting step-sizes, which is used in minimizing the $P(x,r_k)$ function as described in Section 3.5. Second, after every successful move, the feasibility is checked; whenever a move has reached a point which is inside the feasible region bounded by inequality constraints, the process of selecting a feasible starting point is terminated. And the feasible point obtained is used as the desired feasible starting point.

(4) Check if the exploratory move has made any progress; in the other words, it searches a new point which has a less value of TGH than the base point of the exploratory move does. If it does not, cut down the step-sizes and go back to step 2, if it does, go to step 5.

(5) Convert the exploratory move point to be the new base point; let it be x^0.

(6) Make a pattern move along the line connecting the two base points to a new pattern move point x^p.

(7) Check if x^p has a less value of TGH than x^0 does. Return to step 3 if the answer is negative. If x^p does make progress, check if it is in the feasible region bounded by the inequality constraints. Terminate the process of selecting a feasible starting point and use x^p as the feasible starting point if x^p is feasible. Otherwise, set $x^0 = x^p$ and return to step 3.

3.5. COMPUTATIONAL PROCEDURE FOR MINIMIZING $P(x,r_k)$ FUNCTION BY THE HOOKE AND JEEVES PATTERN SEARCH

The computational procedure for minimizing the $P(x,r_k)$ function is the basic Hooke and Jeeves pattern search technique [8,9]. The method is a sequential search routine for searching a point $x = (x_1, x_2, \ldots, x_n)$ which minimize the function, $P(x,r_k)$. A descriptive flow diagram of the method is given in Fig. 3. The procedure consists of two types of moves: Exploratory and Pattern.

A move is defined as the procedure of going from a given point to the following point. A move is a success if the value of the $P(x,r_k)$ decreases; otherwise, it is a failure. The first type of move is an exploratory move which is designed to explore the local behavior of the function, $P(x,r_k)$. The success or failure of the exploratory move is utilized by combining it into a pattern which indicates a probable direction for a successful move [8,9].

The exploratory move is performed as follows:

(1) Introduce a starting point x with a prescribed step size d_i in each of the independent variables x_i, $i = 1, 2, \ldots, n$.

(2) Compute the function, $P(x,r_k)$, where $x = (x_1, x_2, \ldots, x_n)$. Set $i = 1$.

(3) Compute $P_i(x,r_k)$ at the trial point

$$x = (x_1, x_2, \ldots, x_i + d_i, x_{i+1}, \ldots, x_n).$$

(4) Compare $P_i(x,r_k)$ with $P(x,r_k)$:

(i) If $P_i(x,r_k) < P(x,r_k)$, set $P(x,r_k) = P_i(x,r_k)$, $x = (x_1, x_2, \ldots, x_n) = (x_1, x_2, \ldots, x_i + d_i, \ldots, x_n)$, and $i = i+1$. Consider this trial point as a starting point, and repeat from step 3.

(ii) If $P_i(x,r_k) \geq P(x,r_k)$, set $x = (x_1, x_2, \ldots, x_i - 2d_i, \ldots, x_n)$. Compute $P_i(x,r_k)$, and see if $P_i(x,r_k) < P(x,r_k)$. If this move is a successthe new trial point is retained. Set $P(x,r_k) = P_i(x,r_k)$, $x = (x_1, x_2, \ldots, x_i, \ldots, x_n) = (x_1, x_2, \ldots, x_i - 2d_i, \ldots, x_n)$, and $i = i+1$, and repeat from step 3. If again $P_i(x,r_k) \geq P(x,r_k)$, then the move is a failure and x_i remains unchanged, that is,

$x = (x_1, x_2, \ldots, x_i, \ldots, x_n)$.

Set $i = i+1$ and repeat from step 3.

The point x_B obtained at the end of the exploratory moves, which is reached by repeating step 3 until $i = n$, is defined as a base point. The starting point introduced in step 1 of the exploratory move is a starting base point or point obtained by the pattern move.

The pattern move is designed to utilize the information acquired in the exploratory move, and executes the actual minimization of the function by moving in the direction of the established pattern. The pattern move is a simple step from the current base to the point

$$x = x_B + (x_B - x_B^*)$$

x_B^* is either the starting base point or the preceding base point. Following the pattern move a series of exploratory moves is conducted to further improve the pattern. If the pattern move followed by the exploratory moves brings no improvement, the pattern move is a failure. Then we return to the last base which becomes a starting base and the process is repeated.

If the exploratory moves from any starting base do not yield a point which is better than this base, all the step sizes are reduced and the

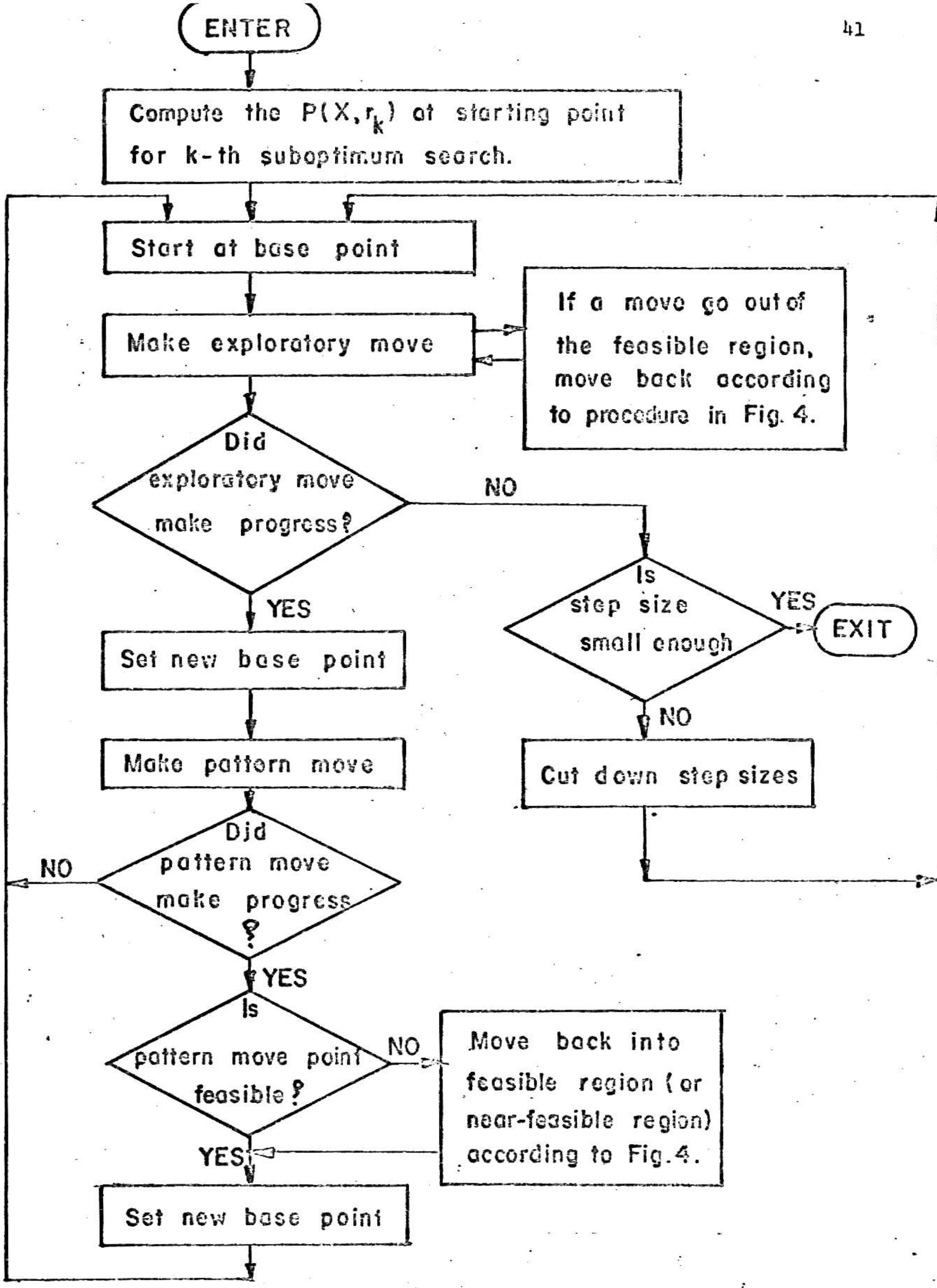

Fig. 3. Descriptive flow diagram for Hooke and Jeeves Pattern Search for minimizing $P(X, r_k)$ function.

moves are repeated. Convergence is assumed when the step sizes, d_i's, have been reduced below predetermined limits.

The following modifications are made so that the above method originally developed for unconstrained minimization shall be able to handle inequality constraints.

(i) During the exploratory moves, every successful move is checked to see if it goes out of the feasible region bounded by the inequality constraints. If it is so, it will be moved back into feasible or near-feasible region according to the procedure described in Section 3.6, and then continue on the regular search routine.

(ii) After making a pattern move, the function, $P(x,r_k)$, is evaluated. Check if the pattern move make progress. If the pattern move makes no progress, return to the base point and make an exploratory move from the base point. If the pattern move makes progress, check if the pattern move point is feasible (subject to the inequality constraints only). Move back into the feasible or the near-feasible region bounded by the inequality constraints according to the procedure described in Section 3.6 if the success pattern move is infeasible.

3.6. PROCEDURE FOR MOVING AN INFEASIBLE POINT INTO THE FEASIBLE OR NEAR-FEASIBLE REGION BOUNDED BY INEQUALITY CONSTRAINTS

The procedure for moving an infeasible point into the feasible or the near-feasible region bounded by the inequality constraints is based on a simplified Hooke and Jeeves pattern search. Since the optimum will be located at somewhere very close to the boundary of the set of constraints for most of the constrained problems, the moving procedure used here consists of small step size exploratory moves only. Pattern moves are not used.

The procedure is summarized below (refer to Fig. 4).

(1) Start at the infeasible point, x, which is to be moved into the feasible or the near-feasible region bounded by inequality constraints.

(2) Compute the weight of violation, TGH, at x,

$$TGH = \left\{ \sum_{t\varepsilon T} [g_t(x)]^2 + \sum_{s\varepsilon R} [h_s(x)]^2 \right\}^{\frac{1}{2}}$$

where $T = \{t | g_t(x) < 0\}$ and $R = \{s | h_s(x) \neq 0\}$.

(3) Decide the tolerance limit, B, which is sequentially decreased, for example 3/4 of the preceding value, after each moving back process. The starting tolerance limit, B^0, for the k-th sub-optimum search is defined as [10]

$$B_k^0 = 0.5 \sum_{i=1}^{n} d_i/n$$

where d_i is the starting step-sizes of the i-th dimension for the k-th sub-optimum search; n is the dimension of the problem. This implies that the starting tolerance limit for the k-th sub-optimum is set to be a half of the average starting step-sizes. After an infeasible point is moved back to the feasible or near-feasible region bounded by inequality constraints, the size of the tolerance limit is decreased.

(4) Check if x is at least in the near-feasible region. If the answer is positive, go to step 7, otherwise, set x as the base point

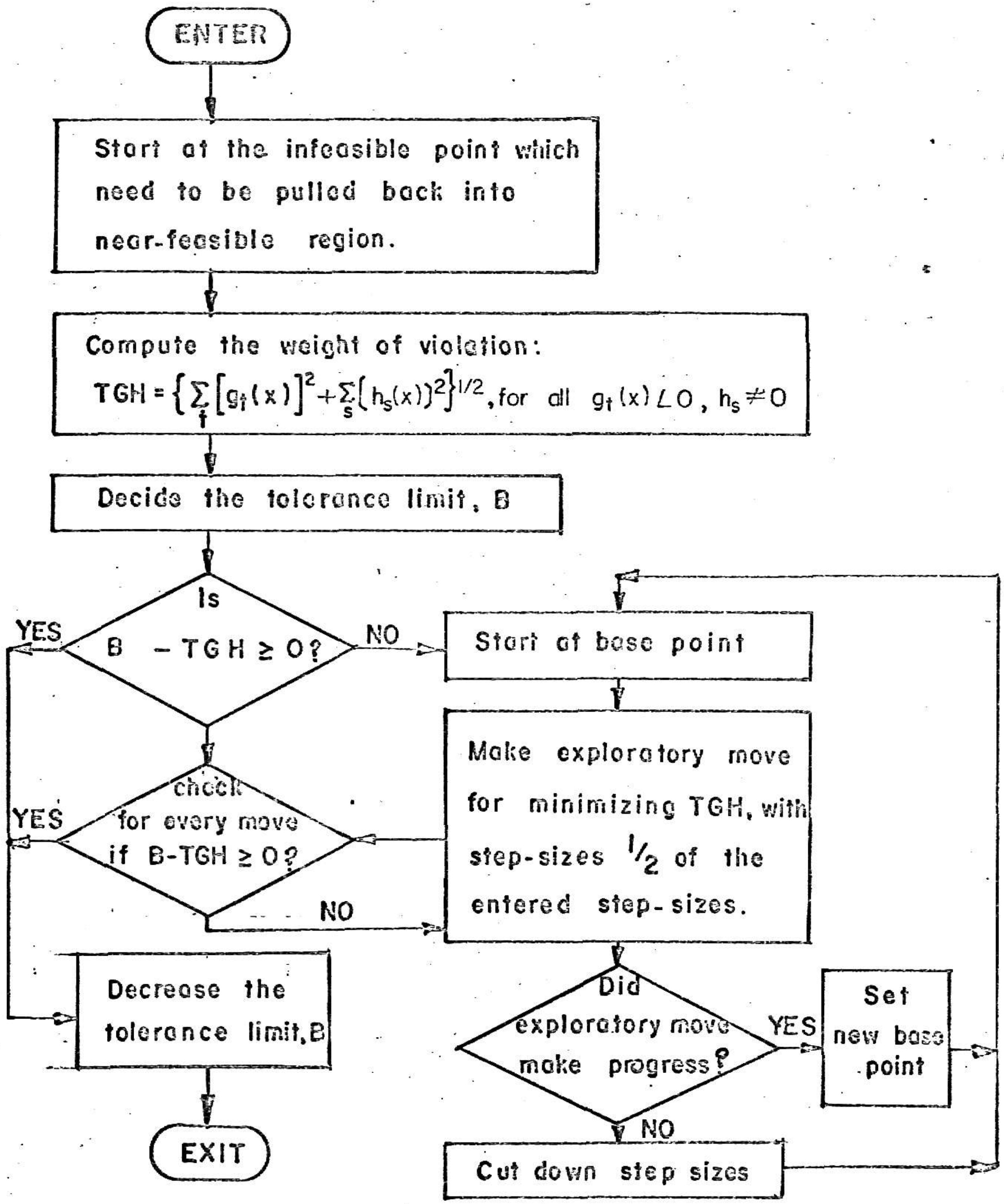

Fig. 4. Descriptive flow diagram for moving an infeasible point back into near feasible region.

and go to step 5. The near-feasible region is defined as the point set $A = \{x | B - TGH \geq 0\}$.

(5) Start at the base point and make an exploratory move for minimizing TGH, with step-sizes one half of the current step-sizes entered to this routine. Whenever a move is feasible or near-feasible, go to step 7; otherwise go to step 6.

(6) Check if the exploratory move makes progress. If the answer is positive, set the exploratory move point to be the new base point and go to step 5. Otherwise, reduce step-sizes then start at the old base point, go to step 5.

(7) Reduce the tolerance limit B which will be used as the starting tolerance limit for next moving back procedure when a preceding move go out of the feasible region again; set the point which satisfies the formula

$$B - TGH \geq 0$$

to be x and terminate the process of moving back procedure.

3.7. PROCEDURE FOR MOVING THE NEAR-FEASIBLE k-TH SUB-OPTIMUM INTO THE FEASIBLE REGION

After the k-th sub-optimum has been reached, it is desirable to have the optimum point in the feasible region subject to all the inequality constraints.

If the optimal point for $P(x,r_k)$ is in the near-feasible region but not in the feasible region, it will be moved back into the feasible region by the following procedure (refer to Fig. 5).

(1) Compute the weight of violation, TGH, at the near-feasible k-th sub-optimum, x_k^0.

$$TGH = \left\{ \sum_{t \varepsilon T} [g_t(x_k^0)]^2 + \sum_{s \varepsilon R} [h_s(x_k^0)]^2 \right\}^{\frac{1}{2}}$$

where $T = \{t | g_t(x_k^0) < 0\}$ and $R = \{s | h_s(x_k^0) \neq 0\}$.

(2) Move x_k^0 toward x_{k-1}^0, the feasible (k-1)-th sub-optimum for a small step δ to obtain a new point $x_k^{0'}$.

(3) Set $x_k^0 = x_k^{0'}$ and check if x_k^0 is feasible. If x_k^0 is not feasible, go to step 2; if x_k^0 is feasible, terminate the process.

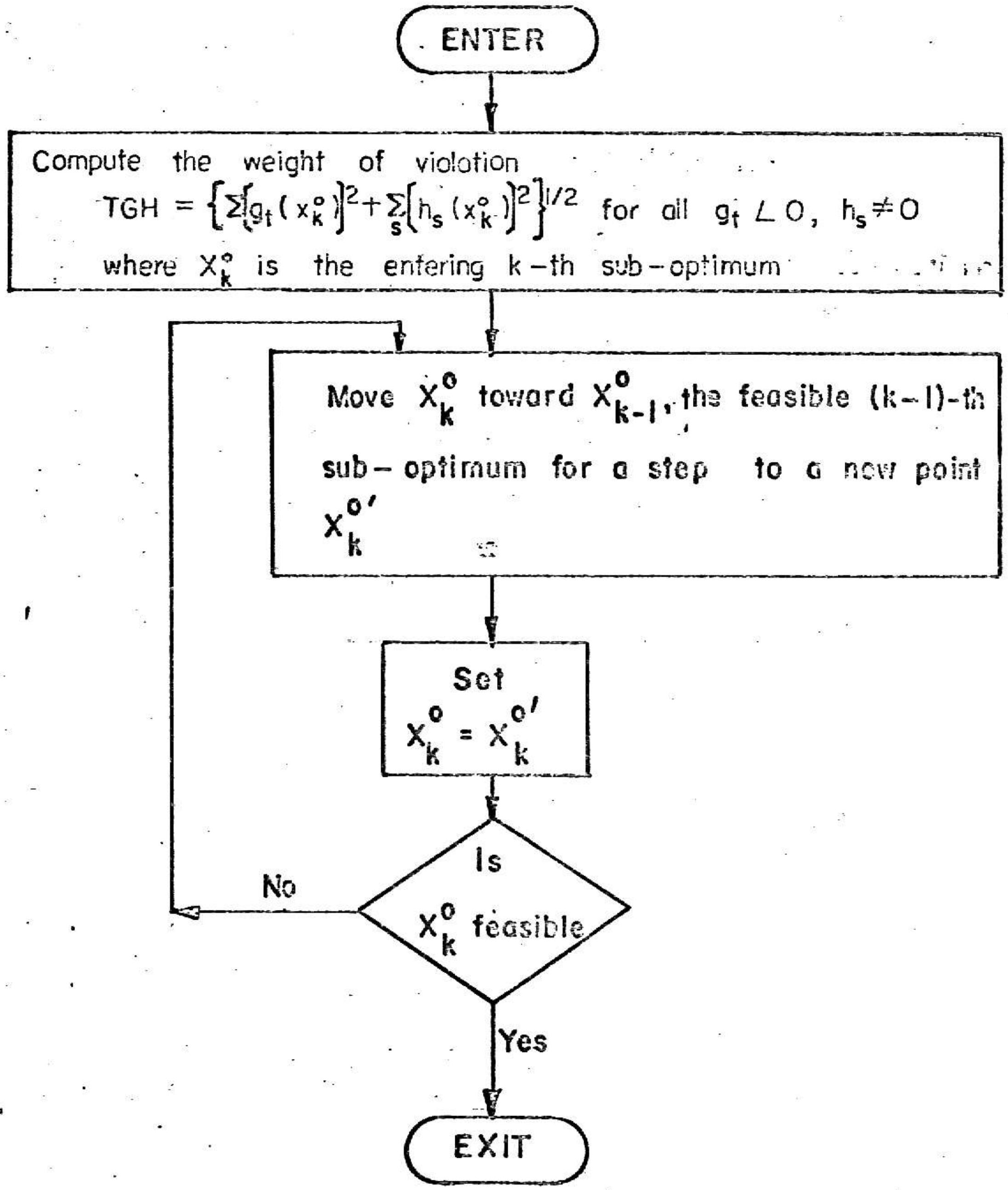

Fig.5. - Descriptive flow diagram for moving the near-feasible k-th sub-optimum into feasible region.

REFERENCES

1. Carroll, C. W., "An Operations Research Approach to the Economic Optimization of a Kraft Pulping Process," Ph.D. Dissertation, Institute of Paper Chemistry, Appletown, Wisc., 1959.

2. Carroll, C. W., "The Created Response Surface Technique for Optimizing Nonlinear Restrained Systems," Operations Research, 9, 169-184 (1961).

3. Fiacco, A. V., and G. P. McCormick, "The Sequential Unconstrained Minimization Technique for Nonlinear Programming: A Primal-Dual Method," Management Sci., 10, 360-366 (1964).

4. Fiacco, A. V., and G. P. McCormick, "Computational Algorithm for the Sequential Unconstrained Minimization Technique for Nonlinear Programming," Management Sci., 10, 601-617 (1964).

5. Fiacco, A. V., and G. P. McCormick, "SUMT without parameters," Systems Research Memorandum No. 121, Technical Institute, Northwestern University, Evanston, Illinois, 1965.

6. Fiacco, A. V., and G. P. McCormick, "Extension of SUMT for Nonlinear Programming: Equality Constraints and Extrapolation," Management Sci., 12 (11): 816-829 (1966).

7. Fiacco, A. V., and G. P. McCormick, Nonlinear Programming: Sequential Unconstrained Minimization Techniques, Wiley, New York, 1968.

8. Hooke, R., and T. A. Jeeves, Direct Search Solution of Numerical and Statistical Problems, J. Assoc. Compt. Mach., 8, 212 (1961).

9. Hwang, C. L., L. T. Fan, and S. Kumar, "Hooke and Jeeves Pattern Search Solution to Optimal Production Planning Problems", Report No. 18, Institute of Systems Design and Optimization, Kansas State University, 1969.

10. Paviari, D. A., and D. M. Himmelblau, "Constrained Nonlinear Optimization by Heuristic Programming," AIChE meeting in New Orleans, March, 1969.

11. Fan, L. T., and C. L. Hwang, Private Communication (1969).

CHAPTER 4

SUMT IMPLEMENTED BY HOOKE AND JEEVES SEARCH TECHNIQUE APPLIED TO PRODUCTION SCHEDULING PROBLEMS

4.1 INTRODUCTION

To illustrate the sequential unconstrained minimization technique (SUMT) implemented by the Hooke and Jeeves pattern search technique, two production scheduling problems, a two dimensional production scheduling problem with four inequality constraints [1, 3] and a twenty dimensional personnel and production planning problem with forty inequality constraints [2, 3, 5], are considered here.

The problems and their solutions are described in the following sections of this chapter.

4.2 A PRODUCTION SCHEDULING AND INVENTORY CONTROL PROBLEM

The problem is to minimize the sum of the production cost and inventory cost subject to the constraints of non-negative inventory and the maximum capacity of machine which produces the desired items. The demand of each period is known and must be satisfied.

The cost for changing the production level and for carrying inventory are given by

$C(\theta_i - \theta_{i-1})^2$ = Cost due to the change in production level from the (i-1)th period to the i-th period,

$D(E - I_i)^2$ = Inventory cost at the i-th period,

where C, D, and E are positive constants. θ_i and I_i are the production level and the inventory level at the i-th period respectively.

The problem is to find $* = (\theta_1^*, \theta_2^*, \ldots, \theta_n^*)$ which minimizes

$$f(\theta) = \sum_{i=1}^{n} [C(\theta_i - \theta_{i-1})^2 + D(E - I_i)^2] \tag{4.1}$$

subject to

$$\left.\begin{array}{ll} I_i = I_{i-1} + \theta_i - Q_i \quad 0, & i = 1, 2, \ldots, n \\ \text{and} & \\ 0 \leq \theta_i \leq M, & i = 1, 2, \ldots, n \end{array}\right\} \tag{4.2}$$

where M is the maximum production capacity. Q_i represents the sales at the i-th period. θ_0 and I_0 are the production level and inventory level at the initial period respectively.

NUMERICAL EXAMPLE 1

For this example, a two period production and inventory system is presented. The optimal decision variable $\theta^* = (\theta_1^*, \theta_2^*)$ will be determined by solving the following problem.

Minimize

$$f(\theta) = C(\theta_1 - \theta_0)^2 + D(E - I_1)^2 + C(\theta_2 - \theta_1)^2 + D(E - I_2)^2 \tag{4.3}$$

subject to

$$\left.\begin{array}{l} g_1(\theta) = I_1 = I_0 + {}_1 - Q_1 \geq 0 \\ g_2(\theta) = I_2 = I_1 + {}_2 - Q_2 \geq 0 \\ g_3(\theta) = M - {}_1 \geq 0 \\ g_4(\theta) = M - {}_2 \geq 0 \end{array}\right\} \tag{4.4}$$

The values of C, D, E, M, θ_0, I_0, and Q_i, i = 1, 2, are given as

$C = 100$, $D = 20$, $E = 10$, $M = 30$,

$\theta_0 = 15$, ${}_0 = 12$, $Q_1 = 30$, $Q_2 = 10$.

To illustrate the procedure the contour lines for equal values of total cost, given by equation (4.3), are shown in Fig. 1. The shaded area represents the feasible region bounded by the inequality constraints given by equation (4.4). The global minimum, $\theta_1^{**} = (\theta_1^{**}, \theta_2^{**}) = (17.82, 18.21)$, of the original unconstrained problem [1] is apparently located outside the feasible region.

The P function of this problem is

$$P(\theta, r_k) = f(\theta) + r_k \sum_{i=1}^{4} \frac{1}{g_i(\theta)}$$

$$= 100(\theta_1 - 15)^2 + 20(28 - \theta_1)^2 + 100(\theta_2 - \theta_1)^2 + 20(38 - \theta_1 - \theta_2)^2$$

$$+ r_k \left(\frac{1}{\theta_1 - 18} + \frac{1}{\theta_1 + \theta_2 - 28} + \frac{1}{30 - \theta_1} + \frac{1}{30 - \theta_2}\right)$$

The step by step procedure of SUMT implemented by the Hooke and Jeeves pattern search technique is as follows:

(1) Let the initial value of r be $r_0 = 3000$. This value of r_0 has been selected arbitrarily.

(2) Let the initial starting point $\theta^0 = (25, 29)$. Note that θ^0 is in the feasible region.

(3) Obtain the optimal solution, $\theta^* = (\theta_1^*, \theta_2^*) = (19.75, 19.00)$, by minimizing the P function for the current value of r. The minimization technique used is the Hooke and Jeeves pattern search technique (details have been discussed in Chapter 3).

(4) Check if the stopping criterion is satisfied. The values of the objective function evaluated at θ^0 and θ^* are $f(\theta^0) = 16,900$ and $f(\theta^*) = 3,418.75$ respectively. It indicates the rapid rate of

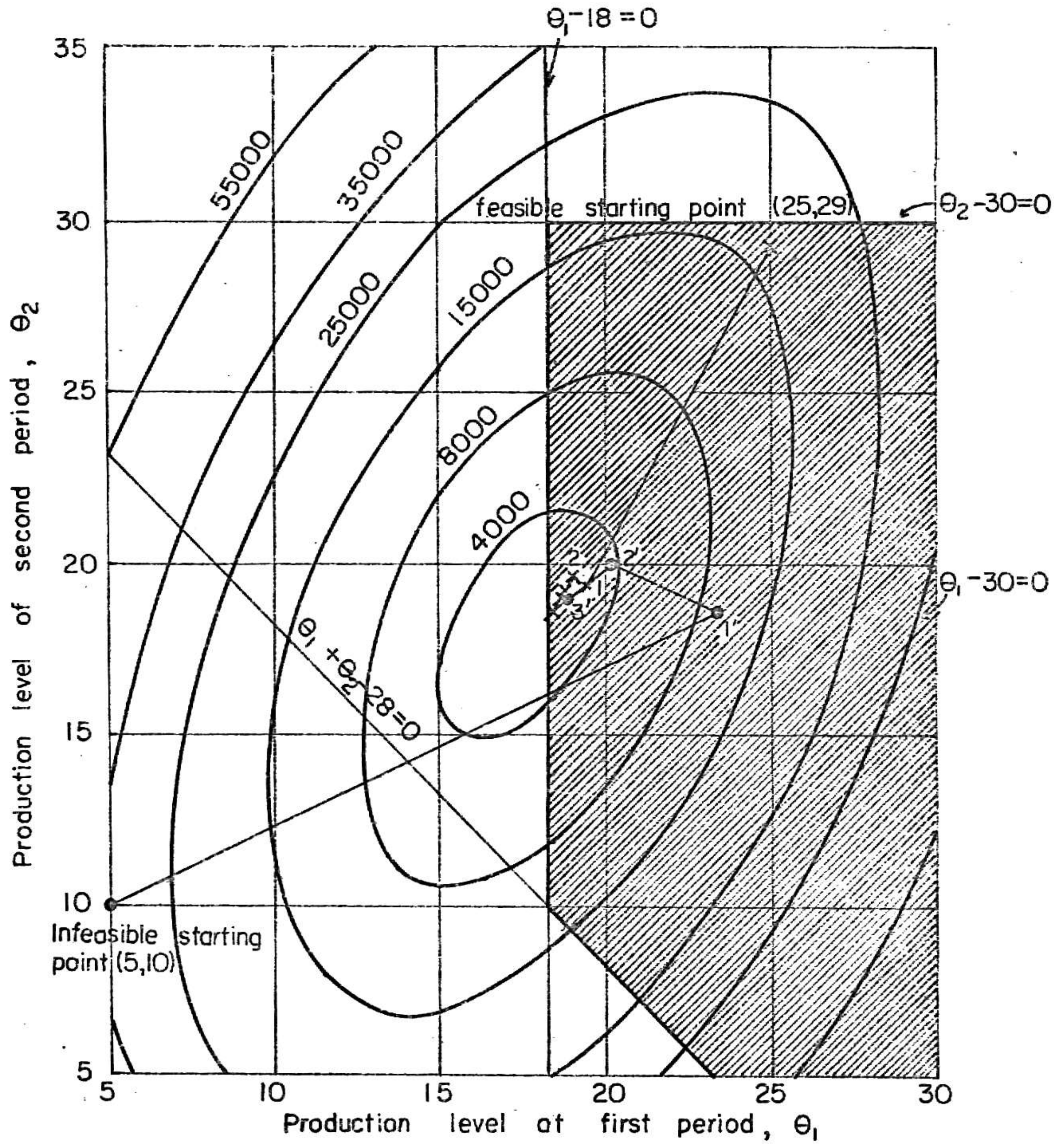

Fig. 1. Production scheduling problem involving two decision variables; contour lines indicate equal quantities of total cost given by equation. (4.3)

convergence at the first iteration. The stopping criterion, $\frac{f(\theta)}{G(\theta)} - 1$, has the value $7.71 > 10^{-4}$. This indicates that more iterations are needed. Iteration will be terminated if

$$\left|\frac{f(\theta)}{G(\theta)} - 1\right| < 10^{-4}.$$

(5) Let $r_1 = r_0/4 = 750$. Return to step 3.

The computational results of the problem are shown in Tables 4.1, and 4.2. Table 4.1 shows the results of starting from a feasible point (25, 29) followed by a series of iterations which converge to the constrained minimum (18.000, 18.350). Table 4.2 shows the results of starting from an infeasible point (5, 10) followed by a series of iterations which also converge to the constrained minimum (18.000, 18.362). The same problem has been solved by employing SUMT with RAC computer program which uses a second order gradient method as the minimization process [4]. The results obtained by these two different programs in Table 4.3a (for starting at $\theta_1 = 25$, $\theta_2 = 29$) and in Table 4.3b (for starting at $\theta_1 = 5$, $\theta_2 = 10$) [3]. These results are identical. It is worth noting again that both the computer programs have self-adjusting procedures to transfer an infeasible starting point to a reasonable feasible starting point before proceeding to iterations [Step 2]. The both cases of starting at two different points have required the same amount of computing time, 1.68 minutes by RAC program and 0.6 minute by the present program, on IBM 360/50. (Both use the WATFOR processor).

Note that in Table 4.2 the iteration makes practically no moves since k = 13. The final stopping criterion is not satisfied at the

Table 4.1 Computer Results of Production And Inventory Problem
[Feasible Starting Point (25,29)]

Number of Iteration k	Value of r	Value of θ_1	θ_2	Value of f	P
0	3000	25	29	16900	21040
1	3000	19.7500	19.0000	3685.00	6243.76
2	750	19.1875	19.5000	3325.94	4168.48
3	187.5	18.6875	18.8333	3100.94	3426.72
4	46.88	18.4063	18.5833	3024.61	3153.36
5	11.72	18.0239	18.3313	2968.75	3387.05
6	2.930	18.1475	18.4222	2980.58	3001.27
7	0.7324	18.0239	18.3414	2968.37	2999.25
8	0.1831	18.0024	18.3261	2966.84	3043.32
9	0.04578	18.0003	18.3261	2966.69	3133.37
10	0.01144	18.0127	18.3594	2967.58	2968.48
11	0.002861	18.9971	18.3551	2967.19	2967.59
12	0.0007153	18.0019	18.3512	2966.83	2967.20
13	0.0001788	18.0004	18.3500	2966.73	2967.18
14	0.0000447	18.0002	18.3499	2966.71	2966.90

Table 4.2 Computer Results of Production and Inventory Problem [Infeasible Starting Point (5, 10)]

Number of Iteration k	Value of r	Value of θ_1	θ_2	Value of f	P
	3000	5.000	10.000	33660	34390
0	3000	23.000	18.000	9580	11090
1	3000	20.000	20.000	3860	6210
2	750	19.250	19.000	3345	4156.12
3	187.5	18.750	19.000	3125	3427.94
4	46.88	18.469	18.500	3041.5	3154.85
5	11.72	18.2437	18.5000	2994.0	3045.44
6	2.93	18.0058	18.3317	2967.06	3477.19
7	0.732	18.0839	18.4150	2973.65	2982.59
8	0.183	18.0189	18.3669	2968.05	2977.78
9	0.0458	18.0122	18.3669	2967.59	2971.35
10	0.01145	18.0065	18.3628	2967.18	2968.94
11	0.002862	18.0021	18.3628	2966.90	2968.29
12	0.000713	18.0005	18.3615	2966.79	2968.35
13	0.000178	18.0000	18.3615	2966.76	2970.67
14	0.0000445	18.0000	18.3615	2966.76	2968.23
15	0.00001112	18.0000	18.3615	2966.76	2967.49
16	0.00000278	18.0000	18.3615	2966.76	∞*
17	0.000000642	18.0000	18.3615	2966.76	2966.76

*On the boundary.

Table 4.3a. Comparison of the Optimal Solutions of the Production Scheduling and Inventory Control Problem [Feasible starting point (25, 29)]

Program	Number of k iterated	θ_1	θ_2	Cost	Stopping criteria: for each k	Stopping criteria: for final ε	Computing time
RAC	0	25	29	16900			
Program	12	18.003	18.335	2966.9		$\varepsilon = 10^{-4}$	1.68 min.
New	0	25	29	16900			
Program	14	18.0002	18.3499	2966.71	INCUT = 3	$\varepsilon = 10^{-4}$	0.6 min.

Table 4.3b. Comparison of the Optimal Solution of the Production Scheduling and Inventory Control Probl

Program	Number of k iterated	θ_1	θ_2	Cost	Stopping criteria: for each k	Stopping criteria: for final ε	Computing time
RAC	0	5	10	33660			
	(feasible starting point)	20.587	15.732			$\varepsilon = 10^{-4}$	1.68 min.
Program	12	18.003	18.335	2966.9			
New	0	5	10	33660			
	(feasible starting point)	23	18	9580	INCUT = 3	$\varepsilon = 10^{-4}$	0.6 min.
Program	17	18.0000	18.3615	2966.76			

end of the 13th iteration. The value of r_k is reduced and the iteration goes to k = 14. Because the value of r_k is decreasing as k increasing, the dual comparison term used in the final stopping criterion is decreasing. The final stopping criterion is finally satisfied at k = 17.

Note that at k = 16, the value of P function is $+\infty$ (for avoiding this critical situation which essentially will cause overflow in computation, a large finite number (,10^{49},) is used to replace $+\infty$). The reason for this is that the sub-optimum point is right on the bounding of an inequality constraint; thus the value of the P function becomes infinity. Recall that the P function is defined as

$$P(x,r_k) = f(x) + r_k \sum_i \frac{1}{g_i(x)}$$

Figure 1 shows the locus of convergence for both the case for feasible starting point and the case for feasible starting point and the case for infeasible starting point.

NUMERICAL EXAMPLE 2

For demonstrating the solution to a problem involving the equality constraints, the above numerical example is modified by adding an equality constraint, namely,

$$h(\theta) = \theta_1 - \theta_2 - 5 = 0$$

This implies that the production level in the first period is five unit larger than that in the second period.

The problem is restated as follows:

Minimize

$$f(\theta) = C(\theta_1 - \theta_0)^2 + D(E - I_1)^2 + C(\theta_2 - \theta_1)^2 + D(E - I_2)^2$$

subject to

$$g_1(\theta) = I_0 + \theta_1 - Q_1 \geq 0$$

$$g_2(\theta) = I_1 + \theta_2 - Q_2 \geq 0$$

$$g_3(\theta) = M - \theta_1 \geq 0$$

$$g_4(\theta) = M - \theta_2 \geq 0$$

$$h(\theta) = \theta_1 - \theta_2 - 5 = 0$$

With the same numerical values given in numerical example 1, the solutions obtained are presented in Tables 4.3c and 4.3d.

During the early iterations, say, from k = 1 to k = 3 or 4, the equality constraint does not play any significant effect to the searches. However, as k increased, the value of r_k approaches to a small numbers, the penalty of violation to the equality constraint becomes significant. The search after k = 4 or 5 in both Table 4.3c and Table 4.3d, as one can see that, the equality constraint is forced to approach to zero. Recall that the formulation of the P-function with equality constraints is defined as

$$P(x,r_k) = f(x) + r_k \sum_i \frac{1}{g_i(x)} + r_k^{-\frac{1}{2}} \sum_j h_j^2(x)$$

As $r_k \to 0$, the penalty to equality constraints h_j's, $r_k^{-\frac{1}{2}} \sum_j h_j^2(x)$, becomes very large. When minimizing the P-function, all the h_j's will be forced to approach to zero.

Table 4.3c Computer Result of the 2-Dimensional Problem with a Equality Constraint [Start at (25, 29)].

Number of Iteration k	Value of r_k	Value of θ_1	θ_2	Value of $f(\theta)$	Value of $P(\theta)$	Value of Constraints $g_1(\theta)$	$g_2(\theta)$	$g_3(\theta)$	$g_4(\theta)$	$h(\theta)$	Value of Computed ε
0	51.29	25	29	16,900	16,980						
1	51.29	18.4375	18.5586	3,032.09	3,167.60	0.4375	8.9961	11.5625	11.4414	-5.121	0.04414
2	12.82	18.2969	18.5586	3,003.00	3,057.58	0.2969	8.8554	11.7031	11.4414	-5.262	0.01320
3	3.205	18.1621	18.4140	2,982.47	3,018.57	0.1621	8.5762	11.838	11.586	-5.252	0.00178
4	0.8013	18.0918	18.3437	2,974.67	3,014.44	0.0918	8.436	11.908	11.656	-5.252	0.00729
5	0.2003	18.0503	18.0503	2,971.60	3,036.53	0.0503	8.3211	11.950	11.729	-5.220	0.0188
6	0.05008	18.0503	18.2708	2,974.82	3,093.27	0.0503	8.227	11.950	11.823	-5.127	0.03767
7	0.00626	18.0503	18.2708	2,987.98	3,295.88	0.0503	8.035	11.950	12.015	-4.935	0.09335
8	0.001138	18.0472	17.3114	3,103.38	3,642.37	0.0472	7.359	11.953	12.689	-4.264	0.1480
9	0.000207	18.235	16.561	3,438.83	4,208.13	0.2347	6.796	11.765	13.439	-3.327	0.1828
10	0.000038	18.462	15.66	4,103.98	4,891.48	0.4621	6.122	11.538	14.340	-2.198	0.1610
11	0.0000068	18.658	14.879	4,910.26	5,479.97	0.65797	5.537	11.342	15.121	-1.221	0.1040
12	0.0000012	18.787	14.387	5,533.93	5,856.30	0.7869	5.173	11.213	15.614	-0.5996	0.05505
13	0.00000023	18.8695	14.143	5,896.71	6.053.39	0.8695	5.012	11.131	15.858	-0.2730	0.0259
14	0.00000004	18.990	14.089	6,081.04	6,148.27	0.990	5.097	11.010	15.893	-0.117	0.0109
15	0.000000008	19.004	14.055	6,159.13	6,189.37	1.004	5.0595	10.996	15.945	-0.05114	0.00208
16	1.4×10^{-9}	19.028	14.048	6,195.14	6,208.08	1.028	5.077	10.972	15.951	-0.0218	0.00208
17	2.5×10^{-10}	19.033	14.043	6,210.13	6,215.95	1.033	5.076	10.967	15.957	-0.0095	0.000932

Table 4.3d Computer Result of the 2-Dimensional Problem with a Equality Constraint [Start at (5, 10)].

Number of Iteration	Value of	Value of		Value of	Value of	Value of Constraints					Computed
k	r_k	θ_1	θ_2	$f(\theta)$	$P(\theta)$	$g_1(\theta)$	$g_2(\theta)$	$g_3(\theta)$	$g_4(\theta)$	$h(\theta)$	ε
0	83.95	5	10	33660	33690						
0	83.95	23	10	24300	24360						
1	83.95	18.500	18.563	3047.97	3242.56	0.5000	9.0625	11.5000	11.4375	-5.0625	0.0661
2	20.99	18.324	18.563	3007.92	3084.64	0.3242	8.8867	11.6758	11.4375	-5.2383	0.02199
3	5.247	18.190	18.453	2986.00	3027.30	0.1895	8.6426	11.8105	11.5469	-5.264	0.00576
4	1.312	18.110	18.383	2976.37	3012.90	0.1104	8.4932	11.8896	11.6172	-5.272	0.00402
5	0.3279	18.063	18.298	2972.32	3025.48	0.0630	8.361	11.937	11.702	-5.235	0.0141
6	0.08198	18.035	18.207	2971.90	3067.71	0.0347	8.241	11.965	11.793	-5.172	0.0297
7	0.0103	18.035	18.207	2980.76	3229.15	0.0346	8.081	11.965	11.954	-5.011	0.07675
8	0.00186	18.019	17.485	3057.14	3519.13	0.0193	7.504	11.981	12.516	-4.465	0.1312
9	0.000339	18.173	16.804	3308.58	4024.71	0.1734	6.9773	11.827	13.196	-3.631	0.1779
10	0.0000616	18.394	15.916	3883.35	4694.25	0.39365	6.30998	11.606	14.084	-2.523	0.17274
11	0.0000112	18.605	15.075	4683.45	5329.99	0.6045	5.6799	11.3955	14.9246	-1.4709	0.1213
12	0.00000204	18.749	14.492	5382.23	5769.16	0.7490	5.2411	11.2510	15.5079	-0.7431	0.0671
13	0.00000037	18.839	14.184	5814.27	6010.20	0.8391	5.0235	11.1609	15.8156	-0.3453	0.0326
14	0.000000067	18.9998	14.131	6043.99	6129.43	0.9998	5.1485	11.000	15.851	-0.149	0.0139
15	0.000000012	19.0498	14.106	6145.96	6183.45	1.0498	5.1639	10.950	15.886	-0.0644	0.00606
16	0.0000000022	19.0585	14.082	6189.96	6206.27	1.059	5.145	10.942	15.914	-0.0277	0.00263
17	0.0000000004	19.0550	14.067	6208.52	6215.63	1.055	5.122	10.945	15.933	-0.01196	0.00114
18	0.00000000007	19.0599	14.065	6216.96	6220.04	1.0599	5.125	10.940	15.935	-0.00514	0.000496

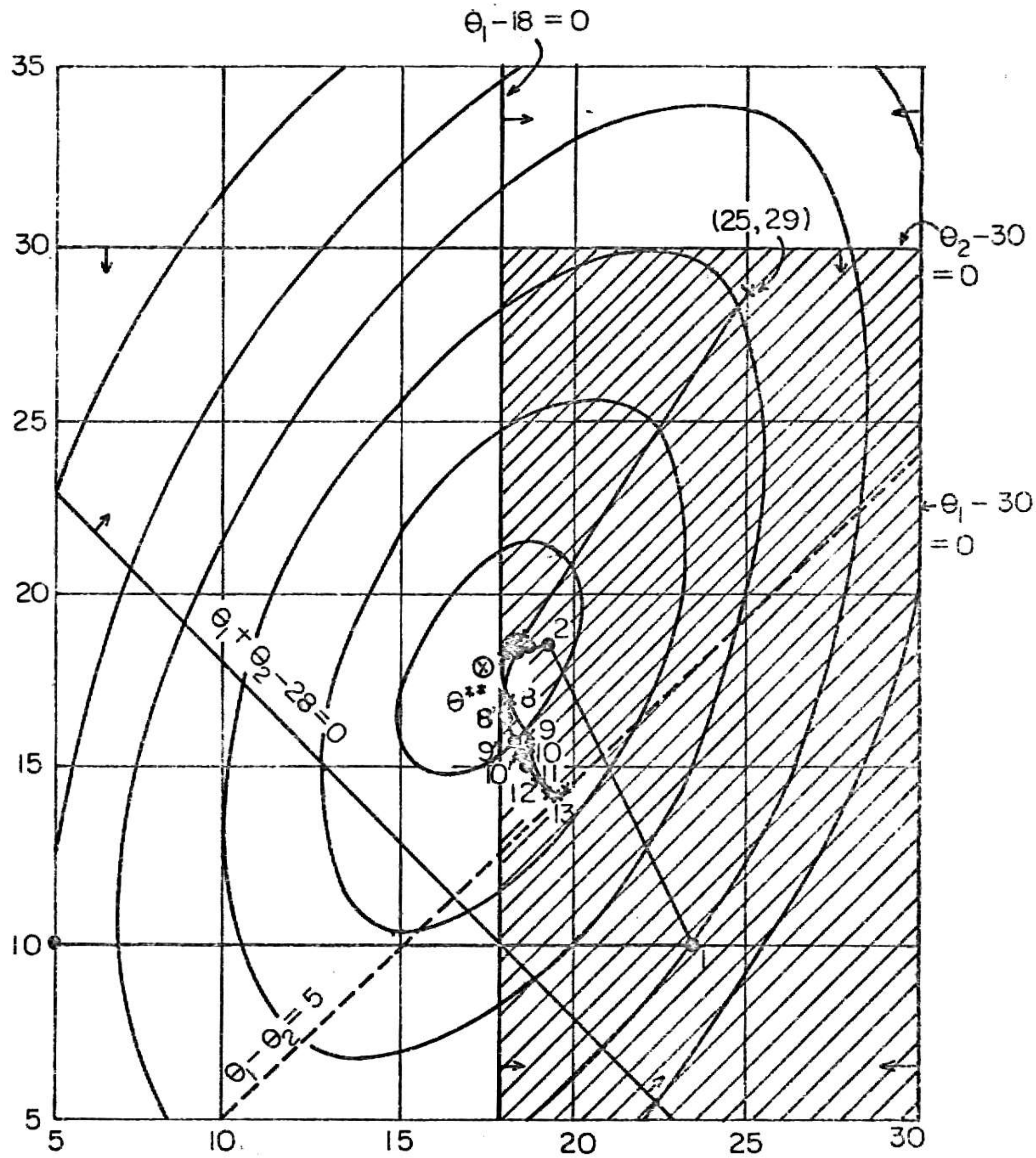

Fig. 4--1a. Production scheduling problem involving two decision variables; contour lines indicate equal quantities of total cost given by equation (4.3).

The optimum of this numerical example is $(\theta_1, \theta_2) = (19.03, 14.04)$. The route of the iterations start from two different starting points, (25,29) and (5,10) is shown in Fig. 4-1a. Note that the selected feasible starting point searched from the infeasible initial point, (5,10), are different for these two numerical examples, one with inequality constraints only, and the other, with inequality constraints and equality constraints.

4.3 A PERSONNEL AND PRODUCTION SCHEDULING PROBLEM

To demonstrate the capability and practical nature of the method, it is employed to obtain the solution of a problem based on the well known model of Holt, Modigliani, Muth and Simon [2]. The problem is to find the optimal operation cost in a paint factory by considering the monthly production and work force level as decision variables in four different sub-costs, namely, the cost of regular payroll, the cost of hiring and firing, the cost of overtime, and the inventory cost. The schematic diagram of the system is shown in Fig. 2. The problem is to minimize the sum of all four different costs over a planning period subject to the constraints of non-negative inventory and non-negative overtime cost. (The main reasons of considering non-negative otertime cost will be discussed later.) The demand of each period is known in advance and must be satisfied.

Let

n = a month in the planning horizon

N = the duration, in months

P_n = production rate at the n-th month

W_n = work force level in the n-th month

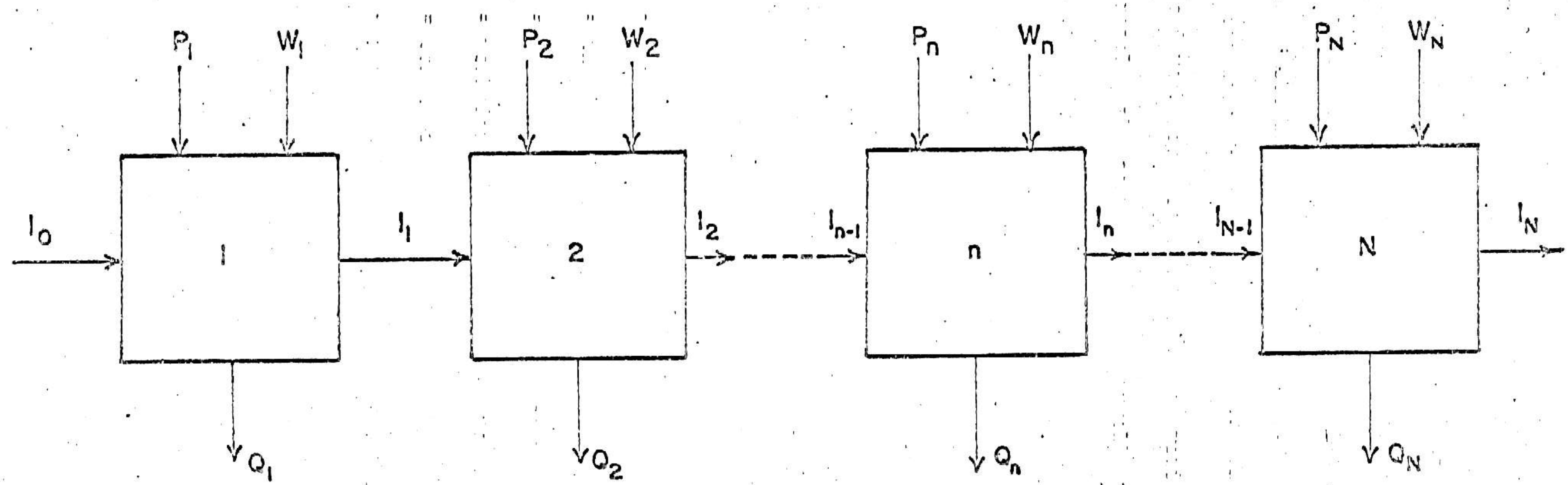

Fig.2. Block diagram for personnel and production scheduling.

Q_n = demand at the n-th month

L_n = inventory level at the end of the n-th month

Inventory level at the end of each month is represented by the recursive relationship among current demand, current production and inventory level of the preceding month.

$$I_n = I_{n-1} + P_n - Q_n, \qquad n = 1, 2, \ldots, N$$

The model considers the following monthly operation cost.

1. Regular payroll cost = $340.0\ W_n$
2. Hiring and lay off cost = $64.3\ (W_n - W_{n-1})^2$
3. Overtime cost = $0.2\ (P_n - 5.67\ W_n)^2 + 51.2\ P_n - 281.0\ W_n$
4. Inventory cost = $0.0825\ (I_n - 320.0)^2$

The system can then be represented by the following mathematical model.

Minimize

$$f(P_1, P_2, \ldots, P_N; W_1, W_2, \ldots, W_N) = \sum_{n=1}^{N} S_n$$

subject to

$$I_n = I_{n-1} + P_n - Q_n \geq 0, \qquad n = 1, 2, \ldots, N-1$$

$$I_N = I_{N-1} + P_N - Q_N \geq I_f$$

and

$$0.2(P_n - 5.67W_n)^2 + 51.2P_n - 281.0W_n \geq 0, \qquad n = 1, 2, \ldots, N$$

where

$$\begin{aligned} S_n = {} & [340.0W_n] + [64.3(W_n - W_{n-1})^2] \\ & + [0.2(P_n - 5.67W_n)^2 + 51.2P_n - 281.0W_n] \\ & + [0.0825(I_n - 320.0)^2] \end{aligned}$$

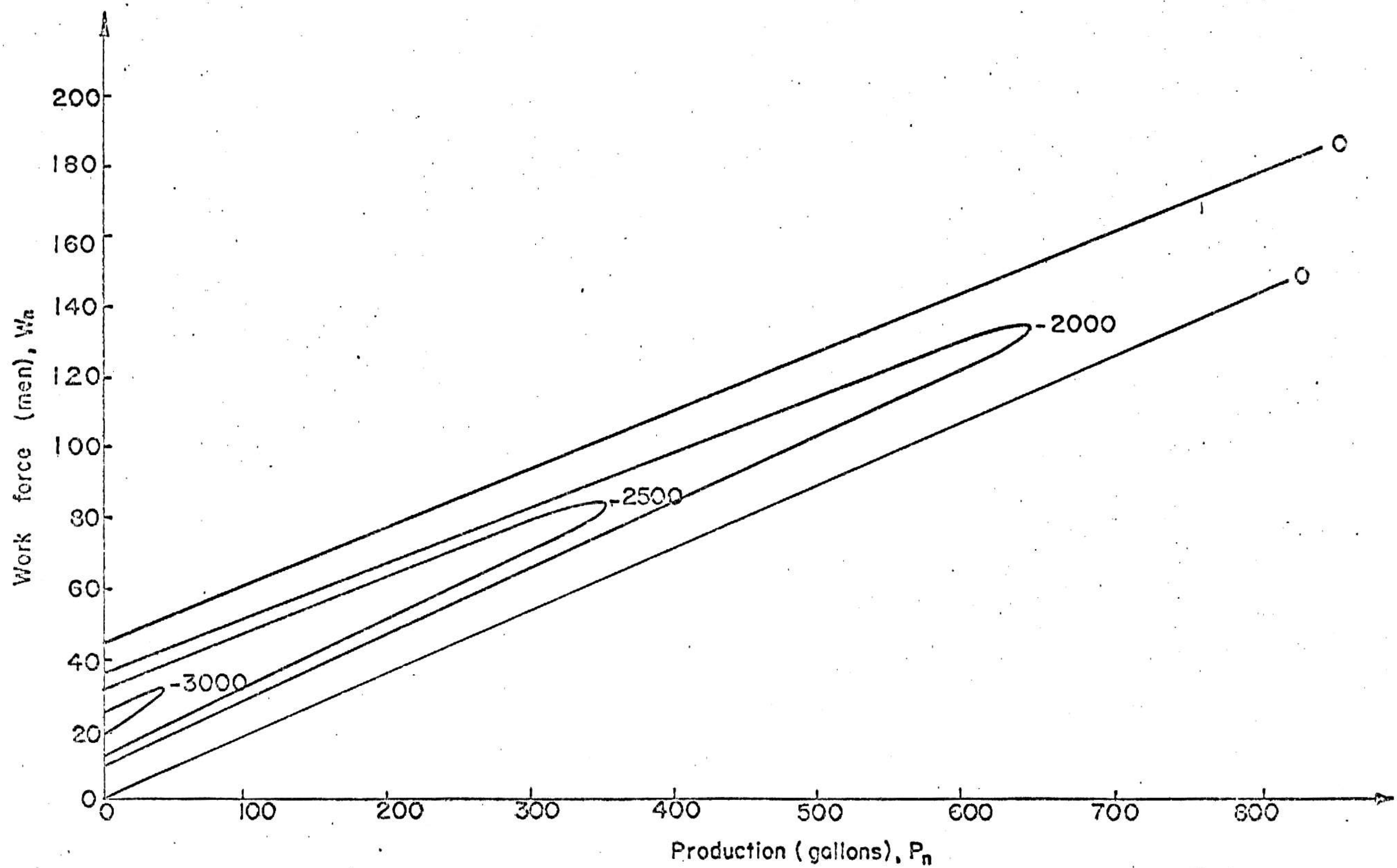

Fig.3. Negative cost contours of the overtime cost equation [5].

The reason of considering the non-negative overtime cost is due to the characteristics of its mathematical formula. Taubert [5] found that minimizing the total costs over the planning period by selecting a certain W_n and P_n combination contributed a negative overtime cost to the objective function. Since the negative cost is illogical in the context of the original paint factory example, a careful examination has been carried out by investigating the overtime cost equation.

$$\text{Overtime cost} = 0.2(P_n - 5.67W_n)^2 + 51.2P_n - 281.0W_n$$

The equation is in quadratic form and can have negative cost contours as plotted in Fig. 3 [3, 5]. Therefore, the constraint of the non-negative overtime cost should be imposed.

A NUMERICAL EXAMPLE

To demonstrate the technique, a numberical example of the model with ten stages is studied.

The numerical data used are as follows:

Demand: $Q_1 = 430,\quad Q_6 = 375,$

$Q_2 = 447,\quad Q_7 = 292,$

$Q_3 = 440,\quad Q_8 = 458,$

$Q_4 = 316,\quad Q_9 = 400,$

$Q_5 = 397,\quad Q_{10}=350.$

The initial inventory, $I_0 = 263$, the inventory for the last month, $I_f = 263$ and the initial work force level $w_0 = 81$.

The starting point is chosen arbitrarily at $x^0 = (P_1^0, W_1^0, P_2^0, W_2^0, P_3^0, W_3^0, P_4^0, W_4^0, P_5^0, P_6^0, W_6^0, P_7^0, W_7^0, P_8^0, W_8^0, P_9^0, W_9^0, P_{10}^0, W_{10}^0) = (500, 90, 500, 90, 500, 90, 500, 90, 500, 90, 500, 90, 500, 90, 500, 90, 500, 90, 500, 90)$. The final result is obtained in 9 iteration (k=9) when the stopping criterion is $\varepsilon = 10^{-5}$ and the starting penalty coefficient is $r_0 = 3.352 \times 10^6$. The value of r_0 is computed by the formula

$$r_0 = (\frac{1}{4}) \cdot \left[\frac{f(x^0)}{\sum_i \frac{1}{g_i(x^0)}} \right] .$$

The results are presented in Table 4.4.

Table 4.5 lists the optimal results obtained by employing the RAC computer program and the present program for comparison. The results obtained by both methods are almost identical.

The computing time is 15.12 minutes for the RAC program and is 8 minutes for the present program; on IBM 360/50 (use the WATFOR processor).

4.4 DISCUSSION AND CONCLUDING REMARKS

The developed technique is a workable technique; and because of its simplicity it can be applied to a wide range of practical problems. The important advantages of this technique over the original available RAC technique are that the new technique does not need to evaluate any derivatives, and requires less computing time.

There is a disadvantage that exploratory moves with small step sizes in the Hooke and Jeaves pattern search may produce the values of P-functional identical in all significant digits for a large numerical value problem. A double precision specification which specifies more significant digits in a computer may be able to overcome this disadvantage.

Table 4.4 Computer Results of Personnel and Production Scheduling (ten-stages)

Number of Iteration k	Value of r	Value of P_1 (W_1)	P_2 (W_2)	P_3 (W_3)	P_4 (W_4)	P_5 (W_5)	P_6 (W_6)	P_7 (W_7)	P_8 (W_8)	P_9 (W_9)	P_{10} (W_{10})	Value of $f(P_n;W_n)$	Value of $P(P_n;W_n)$
0		500 (90)	500 (90)	500 (90)	500 (90)	500 (90)	500 (90)	500 (90)	500 (90)	500 (90)	500 (90)	613,200	766.500
1	3.352×10^6	467.0 (73.7)	415.0 (73.7)	415.0 (73.7)	415.0 (73.7)	415.0 (73.7)	415.0 (73.7)	415.0 (74.5)	415.0 (66.7)	471.0 (66.0)	527.0 (68.2)	303,020	439,884
2	8.381×10^5	487.3 (77.2)	451.0 (73.2)	420.5 (69.4)	392.0 (65.9)	381.0 (63.5)	372.5 (61.8)	367.5 (60.9)	383.0 (60.6)	383.0 (60.4)	378.5 (60.3)	252,845	285,509
3	2.095×10^5	473.3 (77.4)	443.5 (73.8)	416.0 (70.3)	389.0 (67.3)	379.5 (65.0)	371.0 (63.4)	365.5 (62.4)	381.5 (62.0)	378.5 (61.7)	368.5 (61.4)	247.745	258,339
4	5.238×10^4	469.0 (77.7)	442.3 (74.3)	415.3 (71.1)	387.5 (68.2)	378.8 (65.9)	370.0 (64.3)	362.8 (63.1)	379.8 (62.6)	373.0 (62.0)	359.5 (61.7)	245,882	249,436
5	1.31×10^4	468.8 (77.8)	442.3 (74.5)	415.4 (71.4)	385.3 (68.5)	377.6 (66.3)	368.4 (64.7)	361.3 (63.5)	380.5 (62.9)	370.0 (62.2)	352.8 (61.7)	245,076	246,364
6	3.274×10^3	472.3 (77.8)	441.6 (74.5)	414.6 (71.4)	381.9 (68.6)	376.6 (66.5)	367.1 (64.8)	359.8 (63.6)	380.5 (62.9)	369.0 (62.2)	350.0 (61.7)	244,716	245,218
7	8.185×10^2	471.5 (77.9)	443.9 (74.6)	417.1 (71.6)	383.1 (68.7)	375.0 (66.5)	366.4 (64.8)	357.8 (63.5)	379.0 (62.8)	368.3 (62.1)	347.6 (61.6)	244,525	244,747
8	2.046×10^2	469.4 (77.9)	444.1 (74.7)	417.2 (71.6)	383.3 (68.8)	376.1 (66.6)	367.0 (64.8)	357.6 (63.6)	378.3 (62.8)	367.8 (62.1)	346.6 (61.6)	244,432	244,532
9	51.15	468.8 (77.9)	443.8 (74.7)	416.9 (71.6)	383.1 (68.8)	376.1 (66.6)	367.0 (64.8)	357.6 (63.6)	378.3 (62.8)	367.8 (62.1)	346.6 (61.6)	244,375	244,438

Table 4.5. Comparison of the Optimal Solution of the Personnel and Production Planning Problem

Month	RAC Program		New Program	
	P_n	W_n	P_n	W_n
0		81.0		81.0
1	468.6	77.8	468.8	77.9
2	443.0	74.7	443.8	74.7
3	416.4	71.6	416.9	71.6
4	382.2	68.8	383.1	68.8
5	377.7	66.6	376.1	66.6
6	368.3	64.9	367.0	64.8
7	358.8	63.6	357.6	63.6
8	379.4	62.9	378.3	62.8
9	368.0	62.1	367.8	62.1
10	344.8	61.6	346.6	61.6
	Total cost = 244,336		Total cost = 244,375	

REFERENCE

1. Fan, L. T., C. L. Hwang, and F. A. Tillman, "A Sequential Simplex Pattern Search Solution to Production Planning Problems," AIIE Trans. Vol. 1, No. 3 (1969).

2. Holt, C. C., F. Modigliani, J. F. Muth, and H. A. Simon, Planning Production, Inventories, and Work Force, Prentice-Hall, Englewood Cliffs, New Jersey, 1960.

3. Hsu, F. T., L. T. Fan and C. L. Hwang, "Sequential Unconstrained Minimization Technique (SUMT) for Optimal Production Planning", paper summitted for publication, 1969.

4. McCormick, G. P., W. C. Mylander, III., and A. V. Fiacco, "RAC Computer Program Implementing the Sequential Unconstrained Minimization Technique for Nonlinear Programming," SHARE Number 3189.

5. Taubert, W. H., "A Search Decision Rule for the Aggregate Scheduling Problem," Management Science, 14, B-343 (1968).

CHAPTER 5

OPTIMIZATION OF A COMPLEX SYSTEM RELIABILITY

5.1 INTRODUCTION

In this chapter the reliability of a complex system studied in Chapter 2 is again investigated. Two optimization problems associated with this system are considered and the results are compared to the results obtained in Chapter 2.

The first problem is the maximization of system reliability which is identical to the problem studied in Chapter 2. The problem is to (i) find the optimal component reliabilities for the components of the complex system shown in Fig. 1 in Chapter 2. The system reliability is maximized subject to a nonlinear cost function. The second problem is to minimize the cost of the system. (ii) In this problem, constraints of a minimal required system reliability and a minimum component reliability for each component must be satisfied.

The method used to solve the above two problems in this chapter is the method developed in Chapter 3.

The purposes of this chapter are: (1) to demonstrate the usefulness of the method developed in Chapter 3, particularly to the system reliability optimization problems, and (2) to compare the capacity and efficiency of this method with that of the method used in the RAC program.

The optimal solution of the cost minimization problem cannot be obtained by the RAC program, however, an optimal solution can be obtained by employing the new method. The reasons of the particular difficulty in system reliability optimization problem ~~is~~ are explained in section 5.5. For the problem of maximizing the system reliability subject to cost constraint, the same results are obtained by the RAC program and by the new method. The computer time requirement, however, has substantial difference. The RAC program requires over 20 minutes on an IBM 360/50 computer and the new method it requires less than one minute (55 seconds) on the same computer.

5.2 FORMULATIONS OF TWO SYSTEM RELIABILITY PROBLEMS

For the convenience the complex system reliability problem in chapter 2 is briefly summarized below. The diagram which shows the configuration of this system is presented in Fig 1 in chapter 2. In the system, unit 1 is backed up in a parallel by unit 4. There are two equal paths, where each path has unit 2 in series with the stage formed by units 1 and 4. These two equal paths operate in parallel so that if at least one of them is good the output is assured. However, because unit 2 does not have a high degree of reliability, a third unit, unit 3, is inserted into the circuit. Therefore, the following operations are possible: 2-1, 2-4, 3-1, and 3-4, and each operation has two equal paths.

By applying Bayes' theorem of conditional probability, the following expression of the reliability of this system has been derived (see chapter 2).

$$\left.\begin{aligned} &R_s = 1 - Q_s \\ &\text{where} \\ &Q_s = [(1-R_1)(1-R_4)]^2\, R_3 + \{1-R_2[1-(1-R_1)(1-R_4)]\}^2\,(1-R_3) \end{aligned}\right\} \qquad (5.1)$$

The two optimization problems studied with this system can be summarized as follows:

The first problem is to find the optimal component reliability, R_i, which maximize the system reliability, R_s, subject to a maximal cost functional constraint. It can be restated as;

Maximize

$$R_s = 1 - Q_s$$

$$= 1 - R_3 \, [(1-R_1)(1-R_4)]^2$$

$$- (1-R_3)\{1-R_2[1-(1-R_1)(1-R_4)]\}^2$$

subject to

$$\left.\begin{array}{l} \sum\limits_i C_i \leq C \\ \text{where} \\ C_i = K_i R_i^{\alpha_i} \text{, } K_i \text{ and } \alpha_i \text{ are constants} \end{array}\right\} \tag{5.2}$$

The second problem is to find the optimal component reliability, R_i, which minimize the cost function, C. It can be restated as;

Minimize

$$\left.\begin{array}{l} C = \sum\limits_i C_i \\ \text{where} \\ C_i = K_i R_i^{\alpha_i} \text{, } K_i \text{ and } \alpha_i \text{ are constants} \end{array}\right\} \tag{5.3}$$

subject to a minimal system reliability constraint

$$R_s \geq R_{min} \tag{5.4}$$

where R_{min} is a constant minimal system reliability required; and the system reliability, R_s, is given by Equation (5.1).

The cost function C_i in equations (5.2) and (5.3) can represent the weight, cost, or volume of each component of the system, and the summation of C_i then represent the total weight, the total cost, or the total volumn of the system. The weight, cost, or volume of each unit or component of the system is a function of reliability which can be expressed by equations (5.2) and (5.3), where K_i is a proportionality constant and α_i, the exponential factor, relates C_i and the reliability. Usually α_i is less than one.

5.3. THE PROBLEM OF MAXIMIZING SYSTEM RELIABILITY

The numerical example solved in chapter 2 is resolved by the new developed technique. The problem is to find the optimal R_i which maximize

$$R_s = 1 - R_3[(1-R_1)(1-R_4)]^2$$

$$- (1-R_3)\ \{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2 \tag{5.5}$$

subject to the constraint

$$2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + 2K_4R_4^{\alpha_4} \leq C. \tag{5.6}$$

The constants K_1, K_2, K_3, and K_4, the constraint, C, and the exponential constant α_i, i = 1, 2, 3, 4, are follows:

$K_1 = 100,\quad K_2 = 100,\quad K_3 = 200,\quad K_4 = 150,$

$C = 800,\quad \alpha_i = 0.6,\quad i = 1, 2, 3, 4.$

The problem is formulated in SUMT format as follows:

Minimize

$$f(x) = - R_s$$

$$= - 1 + R_3[(1-R_1)(1-R_4)]^2 + (1-R_3)\{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2$$

subject to the constraints

$$g_1(x) = C - (2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + K_4R_4^{\alpha_4}) \geq 0 \tag{5.8}$$

$$g_{i+1}(x) = 1 - R_i \geq 0, \quad i = 1, 2, 3, 4 \tag{5.9}$$

The P function for this problem is

$$P(x,r_k) = f(x) + r_k \sum_i 1/g_i(x)$$

$$= -1 + R_3[(1-R_1)(1-R_4)]^2 + (1-R_3)\{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2$$

$$+ r_k \left(\frac{1}{C - (2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + K_4R_4^{\alpha_4})} + \sum_{i=1}^{4} \left(\frac{1}{1-R_i}\right) \right) \tag{5.10}$$

where x is the row vector of (R_1, R_2, R_3, R_4).

The optimal solutions obtained from two sets of different starting components reliabilities, namely, $[R_1, R_2, R_3, R_4] = [0.7, 0.7, 0.7, 0.7]$ and $[R_1, R_2, R_3, R_4] = [0.6, 0.6, 0.6, 0.6]$, are presented in Table 5.1 together with the corresponding results obtained in Chapter 2. The solutions are almost identical, that is, the optimal system reliability, R_s, of 0.999998 with the cost of 799.733 for the first set of starting components reliabilities, and the optimal system reliability, R_s, of 0.999997 with the cost of 799.908 for the second set of starting components reliabilities are obtained. Recall that the constraint on the cost is 800. The optimal components reliabilities are almost the same for the both starting sets of the starting points. The stopping criterion for terminating the minimization of the P function at each k iteration is that terminating when the number of cut-down step-size operations in the Hooke and Jeeves pattern search is 3, and the final stopping criterion for terminating the problem is $\varepsilon = 10^{-4}$. For the

OVERSIZED DOCUMENT

THE FOLLOWING DOCUMENTS ARE BEING FILMED IN SECTIONS.

THE FOLLOWING IMAGES WILL BE TAKEN FROM LEFT TO RIGHT, TOP TO BOTTOM. SEE EXAMPLE BELOW:

1	2	3
4	5	6
7	8	9

Table 5.1. Comparison of the Optimal Solutions

Program	Number of k iterated	Component reliability R_1	R_2	R_3	R_4
RAC	0	0.7	0.7	0.7	0.7
	10	0.9876	0.9936	0.6972	0.6941
Program	0	0.6	0.6	0.6	0.6
	11	0.9889	0.9921	0.7019	0.6886
New	0	0.7	0.7	0.7	0.7
	12	0.997626	0.998399	0.682652	0.694958
Program	0	0.6	0.6	0.6	0.6
	12	0.997409	0.998117	0.702590	0.682817

of the System Reliability Maximization Problem

System Reliability R_s	Cost	Stopping criteria for each k	Stopping criteria for final ε	Computing time
0.99996	799.78	$\varepsilon'=10^{-5}$	$\varepsilon=10^{-4}$	exceeds 20 min.
0.99995	799.28	$\varepsilon'=10^{-5}$	$\varepsilon=10^{-4}$	exceeds 20 min.
0.999998	799.733	INCUT=3	$\varepsilon=10^{-5}$	(both problems together)
0.999997	799.908	INCUT=3	$\varepsilon=10^{-5}$	90.4 sec.

Table 5.2a. Computer Results of th

[Start at $R_i = 0.$

Iteration k	Times of f-value calculated at each iteration	Value of r_k	R_1	R_2
0		2.214×10^{-2}	0.6	0.6
1	70	2.214×10^{-2}	0.6200	0.7150
2	68	5.535×10^{-3}	0.7900	0.7900
3	59	1.384×10^{-3}	0.8700	0.8700
4	89	3.459×10^{-4}	0.872499	0.91125
5	72	8.648×10^{-5}	0.904999	0.94274
6	174	2.162×10^{-5}	0.944907	0.96815
7	202	5.405×10^{-6}	0.973031	0.98076
8	129	1.351×10^{-6}	0.983415	0.98817
9	110	3.378×10^{-7}	0.989665	0.99263
10	85	8.446×10^{-8}	0.993554	0.99530
11	76	2.111×10^{-8}	0.996045	0.99720
12	60	5.279×10^{-9}	0.997409	0.99811

System Reliability Maximization Problem

, for all i]

	R_3	R_4	$-P$	$-f$ $(= R_s)$	Cost
	0.6	0.6	0.6647	0.8862	662.4
	0.5850	0.6175	0.677501	0.924867	683.298
	0.6600	0.6750	0.88815	0.970493	753.431
	0.7400	0.70833	0.991246	0.991240	776.258
0	0.791250	0.736458	0.986439	0.996132	796.919
9	0.767749	0.722707	0.994712	0.997902	798.981
8	0.730748	0.707252	0.997960	0.999207	798.889
5	0.715331	0.693293	0.999216	0.999740	798.831
3	0.710138	0.688581	0.999691	0.999898	799.273
6	0.704067	0.687510	0.999878	0.999960	799.504
3	0.703067	0.685176	0.999952	0.999984	799.680
3	0.702590	0.683271	0.999981	0.999994	799.730
7	0.702590	0.682817	0.999992	0.999997	799.908

Table 5.2b. Computer Results of the Syst

[Start at R_i = 0.7, for

Iteration k	Times of f-value calculated at each iteration	Value of r_k	R_1	R_2
			0.7	0.7
1	68	1.788×10^{-2}	0.640000	0.73000
2	100	4.471×10^{-3}	0.726250	0.81625
3	64	1.118×10^{-3}	0.816250	0.87625
4	149	2.794×10^{-4}	0.878124	0.92124
5	38	6.986×10^{-5}	0.881124	0.92724
6	126	1.747×10^{-5}	0.911037	0.94932
7	232	4.366×10^{-6}	0.969679	0.98001
8	115	1.092×10^{-6}	0.983835	0.98912
9	94	2.729×10^{-7}	0.990263	0.99327
10	68	6.822×10^{-8}	0.993763	0.99577
11	69	1.706×10^{-8}	0.996263	0.99720
12	69	4.264×10^{-9}	0.997626	0.99839

em Reliability Maximization Problem

all i]

	R_3	R_4	$-P$	$-f$ $(= R_s)$	Cost
	0.7	0.7	0.7161	0.9548	726.6
0	0.610000	0.632500	0.726307	0.936015	695.180
0	0.711250	0.783750	0.906109	0.992678	753.153
0	0.791250	0.783750	0.967770	0.992678	790.870
9	0.766874	0.744843	0.988329	0.996569	797.343
9	0.763874	0.750843	0.994753	0.996927	798.672
8	0.740436	0.736226	0.997489	0.998211	799.632
6	0.692296	0.715007	0.999295	0.999699	799.637
6	0.687223	0.702855	0.999731	0.999911	799.305
7	0.685080	0.698569	0.999894	0.999965	799.322
7	0.684080	0.697069	0.999958	0.999986	799.593
6	0.682652	0.695640	0.999983	0.999994	799.568
9	0.682652	0.694958	0.999993	0.999998	799.733

first set of starting points, it takes 12 iterations for P functions, k = 12, with totally 1192 f-functional values evaluated. And for the second set, 12 iterations for P functions, k = 12, with totally 1194 f-functional values evaluated.

Tables 5-2a and 5-2b present the iteration results converging to the optimal solution. Results given in these tables show that the system reliability, R_s, is monotonically increasing as iteration k increases. The value of P function approaches to that of f function (= $-R_s$) as the iteration proceeds. Thus the minimization of P function will eventually lead us to the minimization of f function.

The values of r_0 used in Tables 5.2a and 5.2b are determined by

$$f(x_0) = r_0 \sum_i \frac{1}{g_i(x_0)} \tag{5.11}$$

where x_0 is the initial point. The basis for of this selection procedure is to render the value of the penalty of the constraints to be approximately the same order of magnitude as the value of the f-function at the starting point in the P-function formulation

$$P(x_0, r_0) = f(x_0) + r_0 \sum_i \frac{1}{g_i(x_0)}$$

The computer time consumed to obtain each set of the solutions presented in Tables 5.2a and 5.2b is 45 seconds respectively on an IBM 360/50 computer by using the Watfor processor. Recall that the same problems solved by the RAC program, as presented in Chapter 2, consumes over 20 minutes on the same computer.

5.4. THE COST FUNCTION MINIMIZATION PROBLEM

The numerical example of this problem studied is restated below. The objective is to find the optimal R_i's which minimize

$$C = 2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + 2K_4R_4^{\alpha_4} \tag{5.11}$$

subject to the constraints

$$R_{min} \leq 1 - R_3[(1-R_1)(1-R_4)]^2 - (1-R_3)\{1-R_2[1 - (1-R_1)(1-R_4)]\}^2 \tag{5.12}$$

$$R_i \geq R_{i,min} \tag{5.13}$$

The numerical values of parameters are

$K_1 = 100,\quad K_2 = 100,\quad K_3 = 200,\quad K_4 = 150$

$C = 800,\quad \alpha_i = 0.6,\quad i = 1,2,3,4.$

$R_{min} = 0.9,\ R_{i,min} = 0.5,\quad i = 1,2,3,4.$

The problem is formulated in SUMT format as follows:

Minimize

$f(x) = C$

$$= 2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + 2K_4R_4^{\alpha_4} \tag{5.14}$$

subject to the constraints

$$g_1(x) = 1 - R_3[(1-R_1)(1-R_4)]^2 - (1-R_3)\{1 - R_2[1 - (1-R_1)(1-R_4)]\}^2 - R_{min} \geq 0 \tag{5.15}$$

$$g_{i+1}(x) = R_i - R_{i,min} \geq 0,\quad i = 1,2,3,4. \tag{5.16}$$

The P function for this problem is

$$P(x,r_k) = f(x) + r_k \sum_i 1/g_i(x)$$

$$= 2K_1R_1^{\alpha_1} + 2K_2R_2^{\alpha_2} + K_3R_3^{\alpha_3} + 2K_4R_4^{\alpha_4}$$

$$+ r_k\left\{\frac{1}{1 - R_3[(1-R_1)(1-R_4)]^2 - (1-R_3)\ 1-R_2[1 - (1-R_1)(1-R_4)\ ^2 - R_{min}}\right.$$

$$\left. + \sum_{i=1}^{4} \left(\frac{1}{R_i - R_{i,min}}\right)\right\} \tag{5.17}$$

where x is the row vector of (R_1, R_2, R_3, R_4).

For this problem, the RAC program failes to satisfy the special rèquirement that the violable non-negativity constraints should never be violated during the search. The results obtained by applying the new developed program is presented in Tables 5.3, 5.4a and 5.4b.

The optimal solutions obtained from two sets of different starting components reliabilities, namely, $[R_1, R_2, R_3, R_4] = [0.6, 0.6, 0.6, 0.6]$ and $[R_1, R_2, R_3, R_4] = [0.7, 0.7, 0.7, 0.7]$ are presented in Table 5.3. The solutions are almost identical, that is, the optimal minimum cost, C, of 642.249 with the system reliability, R_s, of 0.900159 for the first set of starting components reliabilities, and the optimal minimum cost, C, of 642.428 with the system reliability, R_s, of 0.900021 for the second set of starting components reliabilities are obtained. Recall that the constraint on the system reliability is 0.9. The optimal components reliabilities are almost the same for both starting sets. The stopping criterion for terminating minimization of the P function at each iteration is that terminating when the number of cut-down step-size operations is 3. And the final stopping criterion for terminating the problem is $\varepsilon = 10^{-4}$. For the first set of starting points, it takes 12 iterations for P functions, k = 12, with totally 1896 f-functional values calculated. And for the second set, 14 iterations for P functions,

Table 5.3 . Optimal So

Iteration of r_k k	Iteration of f n	Values of Component Reliabil		
		R_1	R_2	R_3
0		0.6	0.6	0.6
12		0.502711	0.834631	0.500438
0		0.7	0.7	0.7
14		0.512435	0.825132	0.500549

lution of the Cost Minimization Problem

ities	System reliability		Stopping criteria	
R_4	R_s	Cost	for each k 'INCUT'	for final ε 'THETA'
0.6	0.8992	662.4	3	10^{-5}
0.502024	0.900159	642.249		
0.7	0.9548	726.6	3	10^{-5}
0.501431	0.900021	642.428		

Table 5.4a. Computer results of the cos

[Start at R_i = 0.6,

Iteration k	Times of f-value calculated at each iteration	Value of r_k	R_1	R_2
			0.6	0.6
0			0.640000	0.640000
1	70	1.470	0.605000	0.645000
2	87	0.3675	0.554999	0.795000
3	67	0.09189	0.541666	0.821666
4	44	0.02297	0.531666	0.806666
5	107	0.005743	0.513666	0.828666
6	101	0.001436	0.507302	0.833211
7	68	0.0003589	0.503969	0.835711
8	52	0.00008973	0.503199	0.834942
9	189	0.00002243	0.502712	0.834632
10	51	0.000005608	0.502712	0.834632
11	1009	0.000001402	0.502711	0.834631
12	51	0.0000003505	0.502711	0.834631

t function minimization problem.

for all i]

R_3	R_4	P	f (= Cost)	R_s
0.6	0.6	828.0	662.4	0.8892*
0.600000	0.600000	981.8	676.9	
0.600000	0.640000	848.140	684.796	0.912690
0.559999	0.580000	704.935	673.866	0.927331
0.519999	0.533333	672.318	658.077	0.916563
0.509999	0.518332	656.939	650.045	0.906742
0.506999	0.507332	649.202	645.493	0.903444
0.503362	0.503696	645.501	643.648	0.901716
0.501696	0.501821	643.660	642.738	0.900850
0.500926	0.501821	642.817	642.395	0.900363
0.500439	0.501494	644.016	642.124	0.900012
0.500439	0.501452	649.370	642.114	0.900001
0.500438	0.501451	665.640	642.113	0.900000**
0.500438	0.502024	642.252	642.249	0.900159

Table 5.4b. Computer results of the co

[Start at R_i = 0.7,

Iteration k	Times of f-value calculated at each iteration	Value of r_k	R_1	R_2
			0.7	0.7
1	81	4.749	0.745000	0.842500
2	102	1.187	0.043749	0.793750
3	99	0.2968	0.581249	0.798749
4	84	0.07421	0.549374	0.806249
5	76	0.01855	0.526874	0.818249
6	91	0.004638	0.515965	0.825749
7	58	0.001159	0.514089	0.825749
8	68	0.0002899	0.513224	0.826325
9	497	0.00007247	0.512688	0.825790
10	358	0.00001812	0.512412	0.825519
11	51	0.000004529	0.512412	0.825519
12	288	0.000001132	0.512433	0.825194
13	1006	0.0000002831	0.512435	0.825132
14	59	0.00000007077	0.512435	0.825132

st function minimization problem.

for all i]

R_3	R_4	P	f (= Cost)	R_s
0.7	0.7	908.3	726.6	0.9548
0.715000	0.670000	890.124	747.532	0.980070
0.610000	0.589374	756.752	694.793	0.946418
0.552499	0.546874	696.952	668.156	0.924405
0.526249	0.524999	668.814	655.247	0.912314
0.512749	0.512999	655.177	648.439	0.905970
0.505931	0.506692	648.484	645.156	0.902995
0.502806	0.503723	645.279	643.667	0.901447
0.501652	0.501704	643.706	642.946	0.900739
0.501116	0.501194	642.998	642.586	0.900259
0.500879	0.501078	642.807	642.443	0.900056
0.500879	0.501978	642.679	642.656	0.900298
0.500560	0.501531	642.481	642.461	0.900065
0.500549	0.501470	642.448	642.438	0.900031
0.500549	0.501431	642.432	642.428	0.900021

END

OF

OVERSIZED

DOCUMENTS

$k = 14$, with totally 2918 f-functional values calculated.

Results given in Tables 5.4a and 5.4b show that the cost of the system, C, is monotonically decreasing as iteration k increases. The value of the P function approaches to that of the f function (=C) as the iteration proceeds. Thus the minimization of the P function will eventually lead us to the minimization of f function.

Again, the values of r_0 are determined from Equation (5.11) as explained in Section 5.3.

The computer time consumed to obtain both sets of the results presented in Tables 5.4a and 5.4b is 75 seconds on an IBM 360/50 computer by using the Watfor processor.

It is worth noting that the starting point $R^o = (R_1, R_2, R_3, R_4) = (0.6, 0.6, 0.6, 0.6)$ in Table 5.4a is in infeasible region. The system reliability given by R^o is 0.8892 which is less than $R_{s,min}$, of 0.9. Therefore, before the P-function minimization routine is started, a new feasible point is searched first. The point (0.64, 0.64, 0.6, 0.6) in the second row of Table 5.4a is thus selected and is used as the feasible starting point to start the minimization procedure. The method used to search this new feasible starting point has been discussed in Chapter 3.

5.5 CONCLUDING REMARKS

From the results presented in this chapter and those in Chapter 4, several concusions can be drawn.

(1) The procedure of selecting the initial value of penalty coefficient, r_0, is valid and convenient. In this procedure the value of the sum of the penalty terms is made approximately the same order of magnitude of the f-function at the initial point, x_0, that is,

$$f(x_0) = r_0 \sum_i \frac{1}{g_i(x_0)} + r_0^{-\frac{1}{2}} \sum_j h_j^2(x_0) \tag{5.18}$$

Equation (5.18) is solved for r_0 and this value is used as the starting r.

(2) The modified Hooke and Jeeves pattern search technique has been proven to be a successful one in the solution of the numerical examples studied in this chapter. As shown in Table 1, the optimal solutions obtained by employing the RAC program and that by the new program developed in the present work are almost identical.

(3) The number of functional values evaluated by applying the new computer program is large. This can be a significant disadvantage, especially when the f-function and/or constraint functions cannot be evaluated in a straight forward manner, for example, the functions are nonlinear differential equations.

(4) The computing time compared in Table 5.1 shows the big difference on the time consumptions by the two different programs for the same system reliability maximization problem. Only 90.4 seconds are needed for obtaining the two solutions by the new computer program developed in this work. While either problem needs to consume over 20 minutes by the RAC program.

(5) The optimal solution for the cost minimization problem can be obtained by the new program while the RAC program fails to give an solution.

(6) There is a difficulty in the optimization of the cost minimization problem mentioned. The feasible region bounded by the given constraints is very narrow and so the constraints is violated frequently. Usually,

in most techniques such as the RAC program and the method developed in Chapter 3, there provided some modification to move a point in the infeasible region back to the feasible region. In maximizing the system reliability problem both programs does not give rise to much difficulty. However, in the minimizing cost problem for this particular character of system reliability optimization problems the RAC program fails to solve the problem. Because the cost function to be minimized is

$$C = 2K_1R_1^{0.6} + 2K_2R_2^{0.6} + K_3R_3^{0.6} + 2K_4R_4^{0.6}$$

where K_1, K_2, K_3 and K_4 are constants and R_i is the component reliability for the ith component which involve $R_i^{0.6}$ terms. When C is minimized, R_i's, essentially, decrease. The non-negativity constraints over component reliabilities are violable. When the non-negativity constraints are violated, the respective $R_i^{0.6}$ is mathematically undefined and so is the cost function.

ACKNOWLEDGEMENTS

The author thanks Dr. C. L. Hwang, and Dr. L. T. Fan for their valuable guidance, encouragement and cooperation in the preparation of this report. He also sincerely acknowledges the support and encouragement provided by Dr. F. A. Tillman and Mr. F. T. Hsu.

This study was partly supported by NASA Grant No. NGR-17-001-034, and by the Air Force Office of Scientific Research, Office of Aerospace Research, United States Air Force, Under Contract F44620-68-0020 (Themis Project).

APPENDIX

COMPUTER PROGRAM FOR IMPLEMENTING SUMT BY HOOKE AND JEEVES PATTERN SEARCH TECHNIQUE

The computer flow chart which illustrates the computational procedure is presented in Fig. 1, 2, 3, 4 and 5; the FORTRAN program symbols, their explanations and corresponding mathematical notations are summarized in Table 1. The computer program listing follows the symbol table.

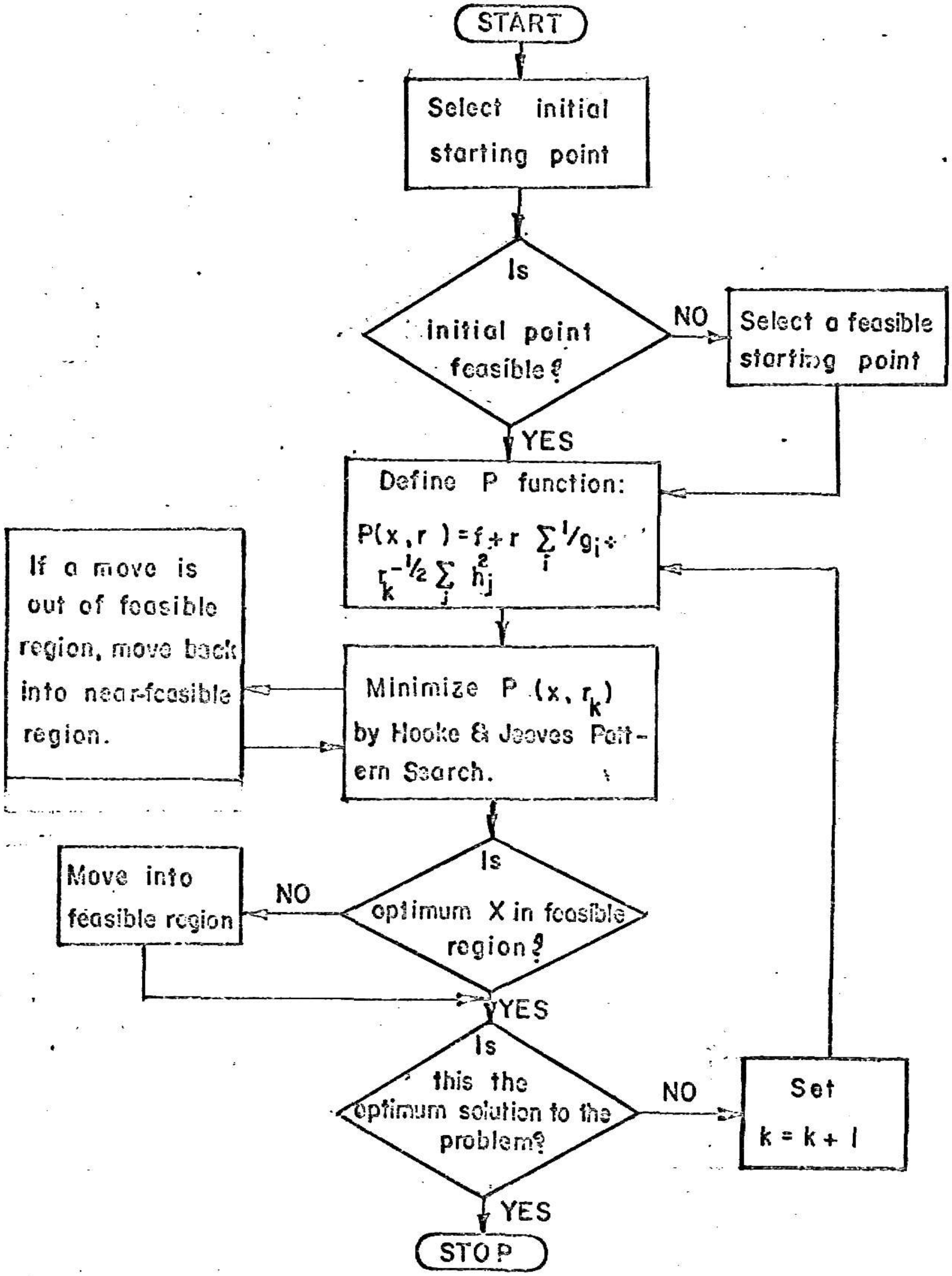

Fig. 1. Descriptive flow diagram for SUMT with Hooke and Jeeves Pattern Search.

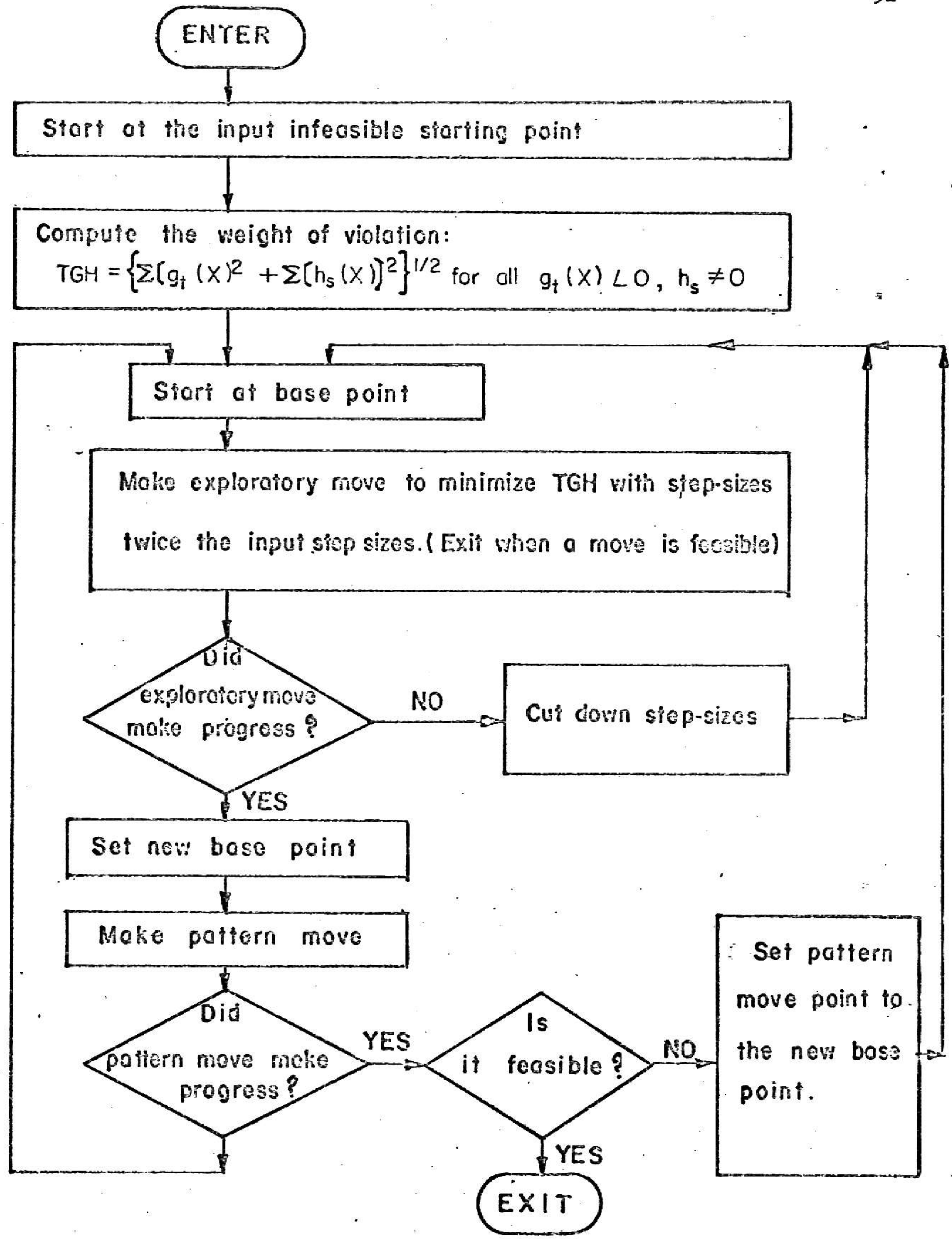

Fig. 2. Descriptive flow diagram for selecting a feasible starting point.

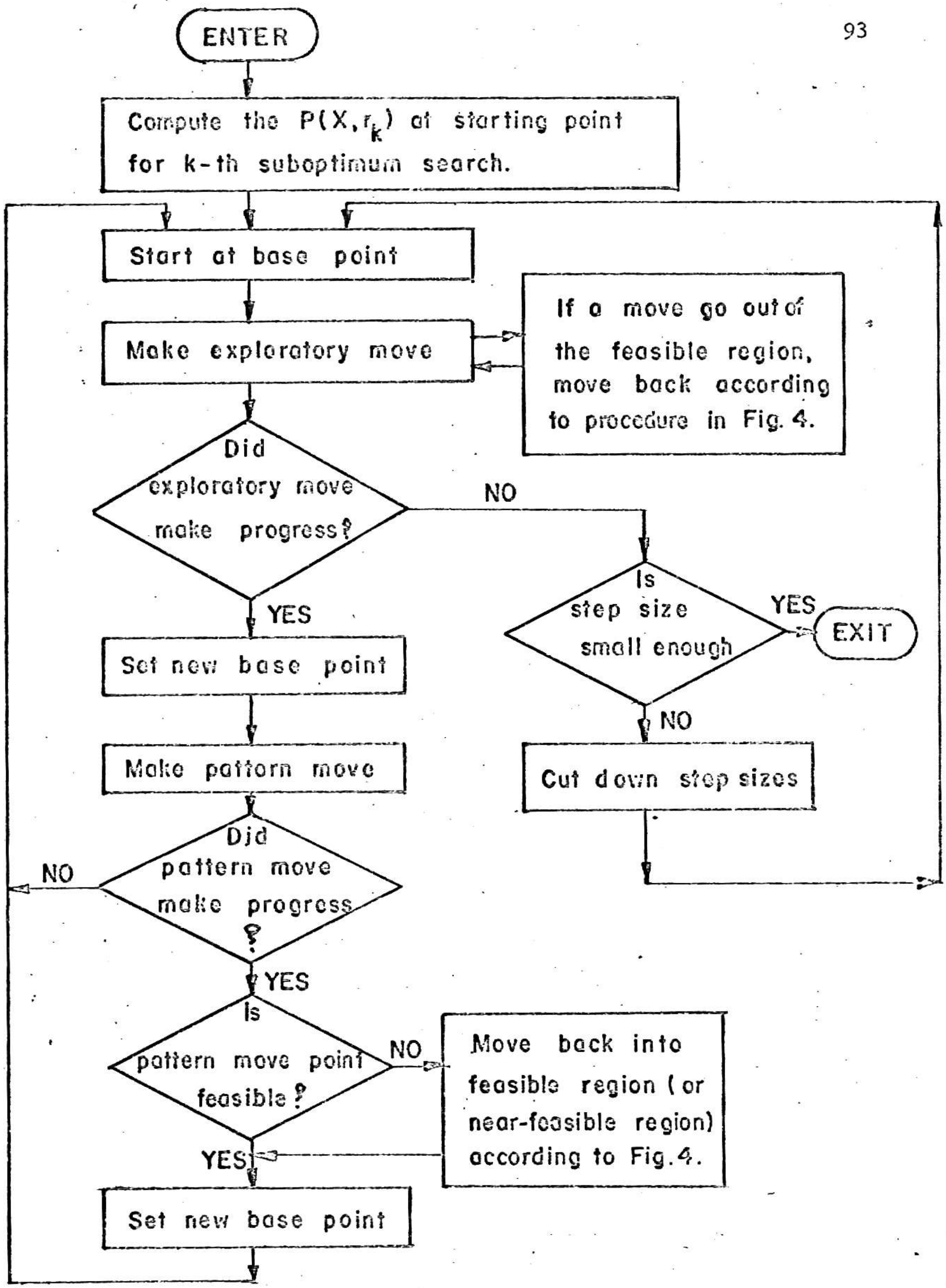

Fig.3. Descriptive flow diagram for Hooke and Jeeves Pattern Search for minimizing $P(X, r_k)$ function.

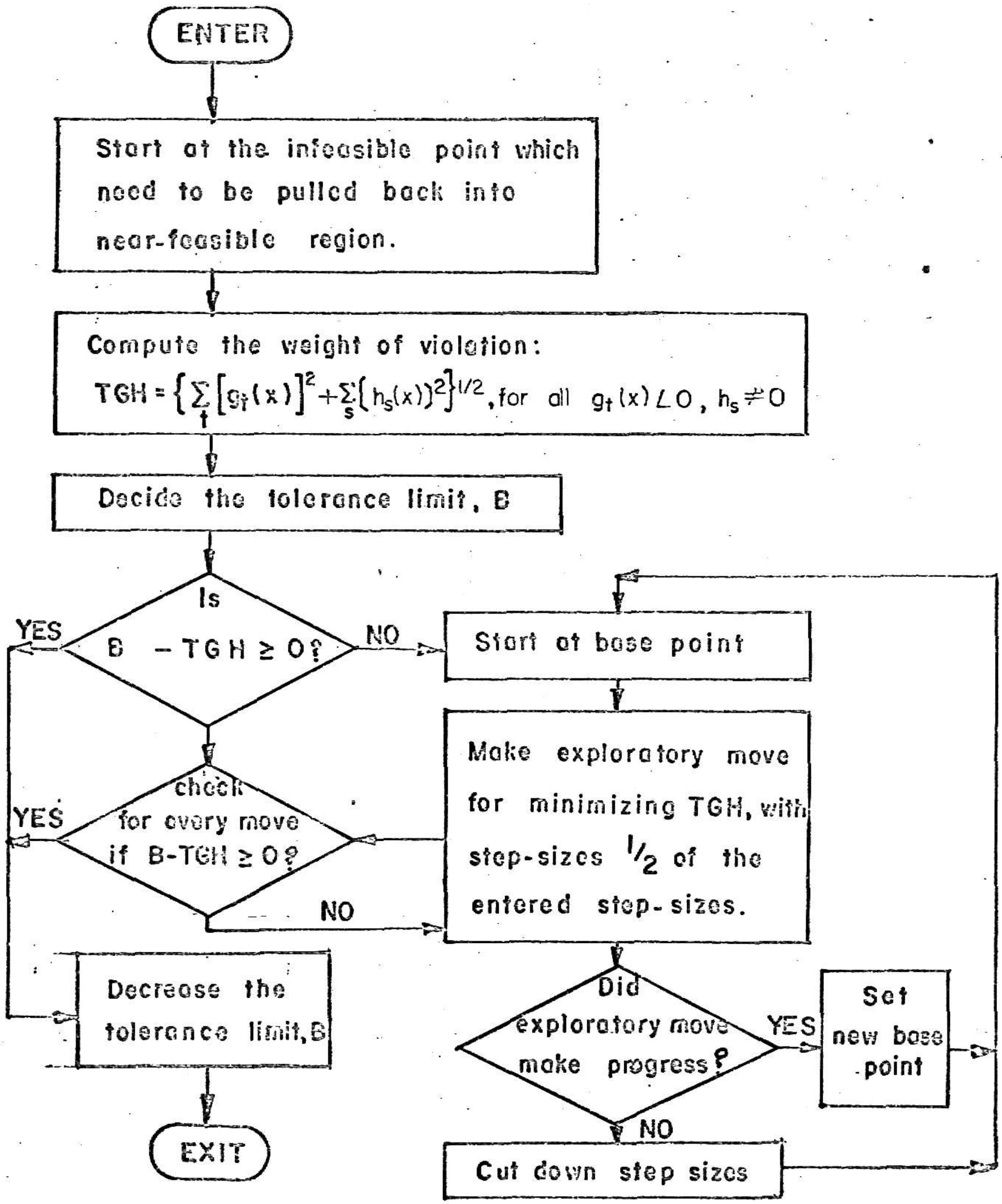

Fig. 4. Descriptive flow diagram for moving an infeasible point back into near feasible region.

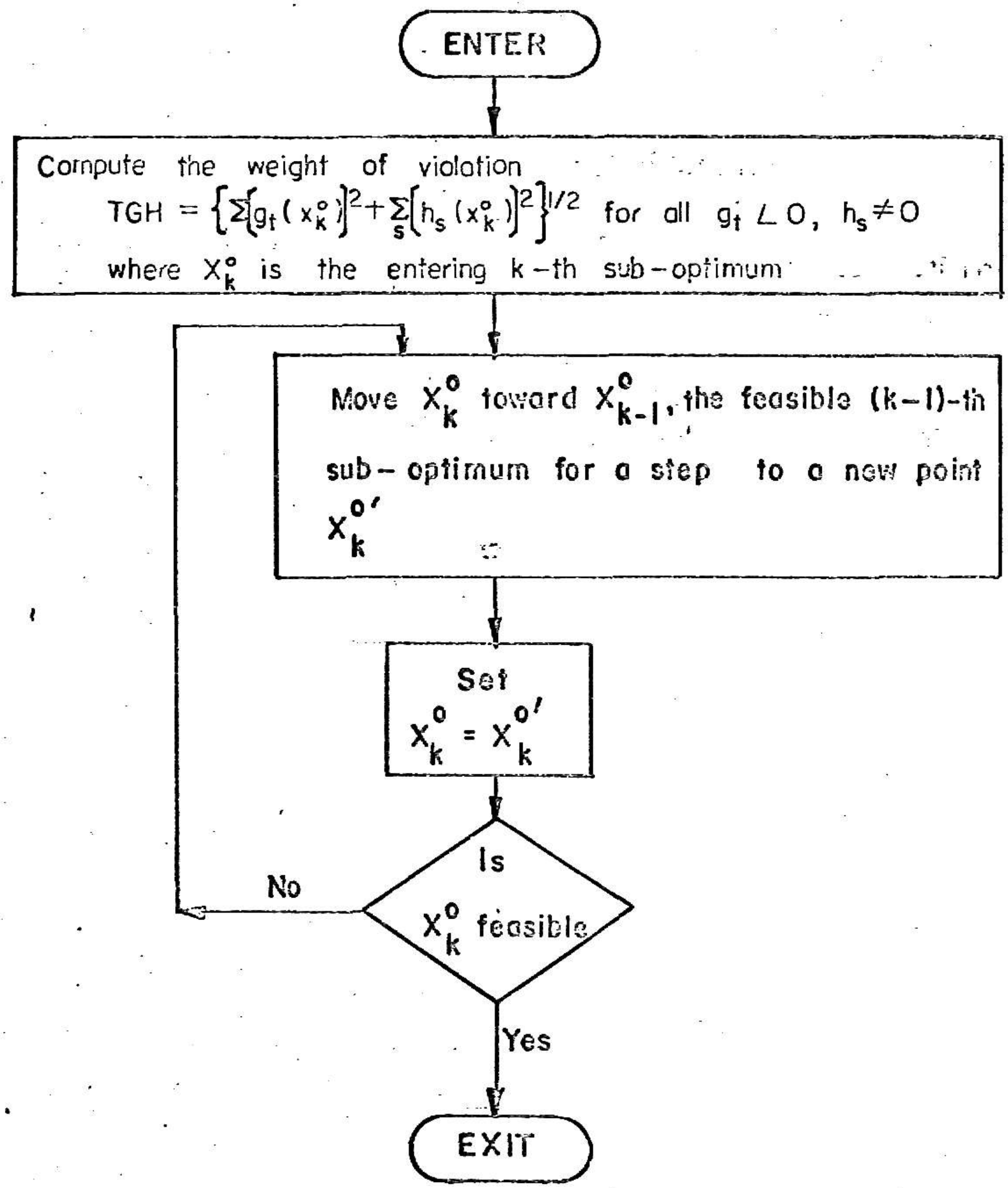

Fig.5. - Descriptive flow diagram for moving the near-feasible k-th sub-optimum into feasible region.

Table 1. Program Symbols and Explanation

Program Symbols	Explanation	Mathematical Symbols
B	tolerence limits for constraint violation	
BX(I)	base point in Hooke and Jeeves pattern search	
D(I)	step-size in Hooke and Jeeves pattern search	d_i
FG(J)	(j)th inequality constraint value at point FX(I)	g_j
FH(K)	(k)th equality constraint value at point FX(I)	h_k
FP	P-function value at point FX(I)	P
FRAC	the fraction of step-sizes used in pulling back infeasible point to the feasible region	
FX(I)	the intermediate suboptimum point during search	
FY	f-function value at point FX(I)	f
FTGH	the intermediate least value of TGH during pulling-back procedure	
G(J)	(j)th inequality constraint value at point X(I)	g_j
H(K)	(k)th equality constraint value at point X(I)	h_k
IB	program control code, IB = 1 means that the point is on the boundary	
ICHECK	program control code, ICHECK = 1 means that ITMAX is exceeded	
ICUT	input option code for initial step-sizes set-up	
IDPM	problem number	
INCUT	stopping criterion for stopping each k-iteration	
ISIZE	input option code for initial step-sizes set-up	
ITER	number of times of calculating f-functional values within a k-iteration	
ITMAX	specified maximum number of calculating f-functional values within each k-iteration	
LOST	program control code, LOST $\neq$ 0 means that some $g_i < 0$	
MG	total number of inequality constraints	m

Table 1. Program Symbols and Explanation (continued)

Program Symbols	Explanation	Mathematical Symbols
MH	total number of equality constraints	ℓ
MAXP	specified maximum number of k-iterations	
N	total number of decision variables	n
NAME1, NAME2, NAME3	three parts of the name of the input problem (6 characters)	
NOBP	number of moves go out of the feasible region	
NOCUT	number of cut-down step-size operations	
NOEXP	number of exploratory moves	
NOIT	total number of times of calculating f-functional values from the very beginning	
NOITB	number of moves iterated in the infeasible region	
NOITP	number of moves iterated in the feasible region	
NOPAT	number of pattern moves	
NOPM	number of input problem sets	
NOPULL	number of times of the operations pulling back suboptimum to the feasible region	
NOR	number of k	k
OX(I)	suboptimum point	x^0
P	P-functional value at point X(1)	P
PB	initial tolerence limit of constraint violation	
PD(1)	initial step-size	d_i^0
PENA1	penalty value to inequality constraints	$r_k \sum_j 1/g_j$
PENA2	penalty value to equality constraints	$r_k^{-\frac{1}{2}} \sum_j h_j^2$
PX(I)	pattern move point in Hooke and Jeeves pattern search	

Table 1. Program Symbols and Explanation (continued)

Program Symbols	Explanation	Mathematical Symbols
PULL	a fraction used to pull back suboptimum to the feasible region	
R	penalty coefficient	r_k
RATIO	reducing rate for reducing R	C
STGH	least value of TGH during searching a feasible starting point procedure	
TGH	weight of violation to constraints	$(\sum_k g_k^2 + \sum_s h_s^2)^{\frac{1}{2}}$
THETA	final stopping criterion	ε
X(I)	a point	x_i
XB(NB)	a point in dulling-back processes	x_i
Y	f-functional value at point X(1)	f
YSTOP	computed value of ε	$\left\lvert \left\lvert \frac{f}{f - r_k \sum_i \frac{1}{g_i} + r_k^{-\frac{1}{2}} \sum_j h_j^2} \right\rvert - 1 \right\rvert$

```
$JOB            KCLAI,TIME=2,LINES=40
C
C          THIS PROGRAM IS FOR OPTIMIZING CONSTRAINED MINIMIZATION PROBLEMS
C     BY A COMBINATIONAL USE OF HOOKE AND JEEVES PATTERN SEARCH TECHNIQUE
C     AND SUMT FORMULATION . WHEN THE SEARCH GETS OUT OF THE FEASIBLE
C     REGION . IT WILL BE PULLED BACK BY A HEURISTIC PROGRAMMING TECHNIQUE ,
C     EXECUTED BY THE SUBROUTINE BACK .
C          THE ORIGINAL IDEALS CAME FROM ..
C               SEARCH TECHNIQUE ... HOOK AND JEEVES .
C               SUMT FORMULATION ... FIACCO AND MCCORMICK .
C               PULL BACK TECHNIQUE ...PAVIANI AND HIMMELBLAU .
C          THE NECESSARY REFERENCE DOCUMENTS CAN BE SEEN IN MY MASTER
C     REPORT .
C                                            K. C. LAI , IE , KSU .
C
C          ***********************************************************
C
C          **INPUT-OUTPUT VARIABLES ...
C               NOPM .. NO. OF SUBPROBLEMS INPUT .
C               NAME1.NAME2,NAME3 .. 3 PARTS OF PROBLEM NAME, USER MAY USE
C                                         ANY 6 CHARACTERS TO NAME THE PROBLEM.
C               N .. NO. OF VARIABLES OF THE PROBLEM .
C               MG .. NO. OF INEQUALITY CONSTRAINTS G(J) .GE. 0. .
C               MH .. NO. OF EQUALITY CONSTRAINTS H(K) .EQ. 0. ,
C               R .. PENALTY COEFFICIENT FOR SUMT FORMULATION .
C                    OPTION --R .LE. 0.0, WILL USE A COMPUTED VALUE .
C               RATIO .. REDUCING RATE FOR R FROM STAGE TO STAGE .
C                        OPTION -- RATIO .LE. 0.0,  WILL USE RATIO=4.0 .
C               ITMAX .. INPUT WITHIN-STAGE ITERATION MAXIMUM NO.
C               INCUT .. STOPPING CRITERION FOR STAGE ITERATION, NO. OF
C                        CUT-DOWN STEP-SIZE OPERATION, USE 2,3 OR 4.
C               THETA .. FINAL STOPPING CRITERION, USE ABOUT 10**(-4).
C               MAXP .. INPUT MAXIMUM NO. OF STAGES , IF EXCEEDED, STOP .
C               X(I) .. (I)TH DIMENSION OF DECISION VARIABLE ,
C               D(I) .. (I)TH DIMENSION OF STEP SIZE .
C               P .. P FUNCTION VALUE .
C               Y .. F FUNCTION VALUE ,
C               YSTOP .. COMPUTED VALUE OF FINAL-STOPPING DETERMINATOR .
C               IDPM .. SEQUENCE NO. OF SUBPROBLEMS OUTPUT .
C               NOR .. NO. OF STAGES UP TO CURRENT STAGE .
```

```
C             B .. TOLELANCE LIMIT FOR VIOLATIONS .
C             FY .. MINIMUM Y GOT SO FAR ,
C             FP .. MINIMUM P GOT SO FAR ,
C             G(J) .. (J)TH INEQUALITY CONSTRAINT VALUE .
C             H(K) .. (K)TH EQUALITY CONSTRAINT VALUE .
C             ITER .. WITHIN STAGE ITERATION NO,
C             NOIT .. CUMULATED ITERATION NO.
C             NOCUT .. NO. OF CUT DOWN STEP-SIZE OPERATION WITHIN STAGE.
C             NOEXP .. NO. OF SUCCESSFUL EXPLORATORY MOVES,
C             NOPAT .. NO. OF SUCCESSFUL PATTERN MOVES.
C             NOB .. NO. OF TIMES OF PULLING BACK PROCEDURE.
C             NOP .. NO. OF SUCCESSFUL MOVES INSIDE FEASUBLE REGION.
C             NOP .. NO. OF SUCCESSFUL MOVES INSIDE FEASIBLE REGION.
C             NOITB .. NO.O
C             NOITB .. NO. OF ITERATIONS OUT OF FEASIBLE REGION.
C
C         ***********************************************************
C
C         **SEQUENCE OF INPUT DECK ...
C             (1) PROBLEM ID CARD .. ONE CARD, FORMAT 1000 .
C                    PARAMETERS -- NOPM,NAME( COMPOSED BY 3 PARTS ),
C                                  N,MG AND MH .
C             (2) PROBLEM ADDITIONAL DATA CARDS .. SPECIFIED IN THE
C                    SUBROUTINE READIN BY USER HIMSELF, ( OPTIONAL ).
C             (3) SUBPROBLEM 1 INITIAL DATA CARDS ..
C                    FIRST  -- ONE CARD, FORMAT 1002 ,
C                              PARAMETERS - R,RATIO,ITMAX,INCUT,THETA
C                                           AND MAXP .
C                    SECOND -- N CARDS, FORMAT 1004 .
C                              PARAMETERS - J,X(I), AND D(I) .
C                               *NOTE -- 1. J IS ONLY FOR USER TO
C                                           CHECK ,
C                                        2. CARDS SHOULD BE IN ORDER
C                                           ( SEQUENCE OF DIMENSION )
C             (4) SUBPROBLEM 2 INITIAL DATA CARDS ,
C                              .
C                              .
C                              .
C             ( ..., UP TO THE LAST SUBPROBLEM INITIAL DATA CARDS ...)
C
```

```
         111.4,11H,  RATIO = E11.4,2H, /5X4HB = E11.4,11H,  INCUT = I4,  11H
         2,  THETA = E11.4,2H .)
     1006 FORMAT(10X2HX(I3,4H) = E14.6,7H,     D(I3,4H) = E14.6,2H .)
     1007 FORMAT(3X75(1H*))
     1008 FORMAT(3X15H**P OPTIMUM.. (I4,1H)              /5X5HFY = E13.6,8H,
         1FP = E13.6,7H,  R = E11.4,10H,  ITER = I5,1H,/5X7HNOIT = I5,9H,  N
         2OB = I4,9H,  NOP = I4,10H,  NOBP = I4/5X8HNOEXP = I4,11H,  NOPAT =
         3 I4,11H,  NOCUT = I4,2H ,/5X8HYSTOP = E13.6,1H,)
     1011 FORMAT(5X/5X16H**CONSTRAINTS ..)
     1012 FORMAT(10X2HG(I3,4H) = E14.6,2H ,)
     1013 FORMAT(10X2HH(I3,4H) = E14.6,2H ,)
     1015 FORMAT(3X46H*****THE ABOVE RESULTS ARE THE FINAL OPTIMUM .)
     1016 FORMAT(3X28H**NO. OF P OPTIMUM EXCEEDED I5,2H .)
     1020 FORMAT(5X//5X47H**SELECTED FEASIBLE STARTING POINT ..          )
     1021 FORMAT(1H1////////////////////5X)
C
C    **READ IN PROBLEM NUMBER, PROBLEM NAME, AND DIMENSIONS .
          READ(1,1000) NOPM,NAME1,NAME2,NAME3,N,MG,MH
          WRITE(3,1021)
          WRITE(3,1001) NAME1,NAME2,NAME3,N,MG,MH,NOPM
          IDPM=1
C    **READ IN ADDITIONAL DATA ( USED FOR ALL SUB-PROBLEMS ),
          CALL READIN(N,MG,MH)
C
C    **READ IN INITIAL PARAMETERS AND STOPPING CRITERIA .
        1 READ(1,1002) R,RATIO,ITMAX,INCUT,THETA,MAXP
          WRITE(3,1003) IDPM
          MP=1
          MULT=1
          NOEXP=0
          NOPAT=0
          NOCUT=0
          NOR=1
          NOBP=0
          NOITP=0
          NOITR=0
          ITER=0
          NOIT=0
          LOST=0
          IB=0
```

```
C          ************************************************************
C
C          **SUBROUTINES NEEDED ...
C               BACK     -- USED TO PULL BACK INFEASIBLE POINT .
C               PENAT    -- USED TO COMPUTE PENALTY TERMS .
C               WEIGH    -- USED TO COMPUTE VIOLATION WEIGHT .
C               READIN   -- A USER SUPPLIED SUBROUTINE, USED TO READ IN
C                           ADDITIONAL DATA NEEDED .
C               OBRES    -- A USER SUPPLIED SUBROUTINE, USED TO COMPUTE
C                           THE OBJECTIVE AND CONSTRAINTS .
C               OUTPUT   -- A USER SUPPLIED SUBROUTINE, USED TO OUTPUT
C                           ADDITIONAL INFORMATION DESIRED .
C
C          ************************************************************
C
C          **DIMENSIONS ...
C               THIS PROGRAM IS DESIGNED FOR N,MG,MH .LE. 20, WHEN
C          THE DIMENSIONS OF N,MG, AND/OR MH EXCEED 20, MAKE PROPURE
C          CHANGES. THE KEY TO CHANGES ARE ...
C               X,FX,PX,BX,OX,D,PD -- N DIMENSIONS
C               G,FG               -- MG DIMENSIONS
C               H,FH               -- MH DIMENSIONS .
C
C          ************************************************************
C
      DIMENSION X(20),FX(20),BX(20),PX(20),OX(20),PD(20),D(20),G(20),
     1FG(20),H(20),FH(20)
      COMMON /BLOGY/ N,MG,MH,ITER,ITMAX,ICHECK,IP,LOST
      COMMON /BLOGB/ NOITP,NOITB,B,D
      COMMON /BLOGR/ O(10)
      COMMON /BLOGO/ FG
 1000 FORMAT(I5,5XA2,A2,A2,3I5)
 1001 FORMAT(1H 30X1H*A2,A2,A2,10H* PROBLEMS/30X20(1H*)///25X14HNO. OF X
     1(I) ..I4,2H ,/25X14HNO. OF G(J) ..I4,2H ,/25X14HNO. OF H(K) ..I4,2
     2H ,//18X17HNO OF PROBLEMS ..I4,2H ,////6X58HPARAMETERS IN OUTPUT .
     3., SEE COMMENT CARDS IN THE PROGRAM. )
 1002 FORMAT(2E15.4,2I5,E15.4,I5)
 1003 FORMAT(1H15X7HPROBLEMI4//////)
 1004 FORMAT(I5,2E15.4)
 1005 FORMAT(20X13HINITIAL POINT/5X4HY = E11.4,7H,  P = E11.4,7H,  R = E
```

```
      IDR=0
      ICHECK=0
      B=0.
      FN=N
C
C   **READ IN INITIAL POINT AND STARTING STEP-SIZES .
      DO 2 I=1,N
      READ(1,1004) J,X(I),D(I)
C   **VARIABLE (J) IS USED FOR CHECKING THE SEQUENCE OF CARDS BY THE
C     USER HIMSELF, AND HAS NO INFFERENCE TO THE PROGRAM ( USER MAY
C     USE ANY INTEGER NUMBER FOR J.
      BX(I)=X(I)
      FX(I)=X(I)
      PD(I)=D(I)
      OX(I)=X(I)
    2 B=B+0.5*D(I)
C   **DECIDE THE STARTING VALUE OF TOLERENCE LIMIT FOR G(J) .LT. 0. .
      B=B/FN
      CALL OBRES(FX,FY,FG,FH)
      CALL WEIGH(STGH,MG,FG)
      ITER=0
   11 CALL PENAT(FG,FH,PENA1,PENA2)
C   **COMPUTE AN INITIAL VALUE OF R WHEN INPUT R VALUE IS .LE. 0. .
      IF(R) 12,12,13
   12 R=ABS(FY/(PENA1+PENA2))
C   **USE RATIO=4.0 WHEN INPUT RATIO VALUE IS .LE. 0. .
   13 IF(RATIO) 14,14,15
   14 RATIO=4.0
   15 FP=FY+R*PENA1+R**(-0.5)*PENA2
      WRITE(3,1005) FY,FP,R,RATIO,B,INCUT,THETA
      WRITE(3,1006) (I,FX(I),I,D(I),I=1,N)
      WRITE(3,1007)
      IF(LOST-2) 50,16,16
C   **SELECT AFFASIBLE STARTING POINT WHEN INPUT INITIAL POINT IS
C     NOT FEASIBLE SUBJECT TO INEQUALITY CONSTRAINTS .
C
C   **MAKE EXPLORATORY MOVE FOR SELECTING A FEASIBLE STARTING POINT .
   16 NOF=0
      DO 28 I=1,N
      FX(I)=X(I)+2.0*D(I)
```

```
      CALL OBRES(FX,FY,FG,FH)
      CALL WEIGH(TGH,MG,FG)
      IF(TGH) 44,44,18
   18 IF(STGH-TGH) 20,20,26
   20 FX(I)=FX(I)-4.0*D(I)
      CALL OBRES(FX,FY,FG,FH)
      CALL WEIGH(TGH,MG,FG)
      IF(TGH) 44,44,22
   22 IF(STGH-TGH) 24,24,26
   24 FX(I)=FX(I)+2.0*D(I)
      NOF=NOF+1
      GO TO 28
   26 STGH=TGH
      X(I)=FX(I)
   28 CONTINUE
C
      IF(NOF-N) 34,30,30
C   **CUT STEP-SIZES FOR SELECTING A FEASIBLE STARTING POINT .
   30 DO 32 I=1,N
   32 D(I)=D(I)*0.5
      GO TO 16
C   **MAKE PATTERN MOVE FOR SELECTING A FEASIBLE STARTING POINT .
   34 DO 36 I=1,N
   36 PX(I)=FX(I)+(FX(I)-X(I))
      CALL OBRES(PX,FY,FG,FH)
      CALL WEIGH(TGH,MG,FG)
      IF(STGH-TGH) 16,16,40
   40 DO 42 I=1,N
      X(I)=PX(I)
   42 FX(I)=PX(I)
      IF(TGH) 44,44,43
   43 STGH=TGH
      GO TO 16
   44 DO 46 I=1,N
      D(I)=PD(I)
   46 PX(I)=FX(I)
      LOST=0
C   **OUTPUT THE MESSAGE OF THE SELECTED FEASIBLE STARTING POINT .
      WRITE(3,1020)
      GO TO 11
```

```
   48 DO 49 I=1,N
   49 X(I)=FX(I)
C   **START TO MINIMIZE THE CURRENT P-FUNCTION .
C
C   **MAKE EXPLORATORY MOVE FOR MINIMIZING THE P-FUNCTION .
   50 NOF=0
      IDIFF=1
      DO 101 I=1,N
      X(I)=FX(I)+D(I)
      LOST=0
      CALL OBRES(X,Y,G,H)
      IF(LOST-1) 62,62,52
   52 IF(Y-FY) 55,68,68
   55 CALL BACK(X,X,Y,G,H)
      NOITB=NOITB+1
      NOBP=NOBP+1
C   **CHECK THE ITMAX IS EXCEEDED OR NOT IN (BACK)( LOST=1 MEANS THE
C     RETURNED POINT IS INFEASIBLE )
      IF(LOST-1) 56,150,56
C   **ADD NOCUT 1 CREDIT TO STOP THE CURRENT STAGE FASTER AFTER
C     EVERY 5 VIOLATIONSMADE WITHIN THE STAGE .
   56 IF(NOBP-5*MULT) 60,57,57
   57 NOCUT=NOCUT+1
      MULT=MULT+1
   60 LOST=0
C   **CHECK THE ITMAX IS EXCEEDED OR NOT IN (BACK)( LOST .NE. 0 MEANS
C     THE ENTERED POINT IS NEAR-FEASIBLE )
   62 IF(ICHECK-1) 64,140,140
   64 CALL PENAT(G,H,PENA1,PENA2)
      P=Y+R*PENA1+R**(-0.5)*PENA2
      IF(P-FP) 88,68,69
   68 X(I)=FX(I)-D(I)
      LOST=0
      CALL OBRES(X,Y,G,H)
      IF(LOST-1) 80,80,70
   70 IF(Y-FY) 73,86,86
   73 CALL BACK(X,X,Y,G,H)
      NOITB=NOITB+1
      NOBP=NOBP+1
C   **CHECK THE ITMAX IS EXCEEDED OR NOT IN (BACK)( LOST=1 MEANS THE
```

```
C       RETURNED POINT IS INFEASIBLE )
        IF(LOST-1) 74,150,74
C     **ADD NOCUT 1 CREDIT TO STOP THE CURRENT STAGE FASTER AFTER
C       EVERY 5 VIOLATIONSMADE WITHIN THE STAGE .
     74 IF(NOBP-5*MULT) 78,75,75
     75 NOCUT=NOCUT+1
        MULT=MULT+1
     78 LOST=0
C     **CHECK THE ITMAX IS EXCEEDED OR NOT IN (BACK)( LOST .NE. 0 MEANS
C       THE ENTERED POINT IS NEAR-FEASIBLE )
     80 IF(ICHECK-1) 82,140,140
     82 CALL PENAT(G,H,PENA1,PENA2)
        P=Y+R*PENA1+R**(-0.5)*PENA2
        IF(P-FP) 88,86,86
     86 X(I)=FX(I)
        NOF=NOF+1
        GO TO 99
     88 FY=Y
        FP=P
        NOITP=NOITP+1
        FX(I)=X(I)
        IF(MG) 94,94,90
     90 DO 92 JJ=1,MG
     92 FG(JJ)=G(JJ)
     94 IF(MH) 99,99,96
     96 DO 98 KK=1,MH
     98 FH(KK)=H(KK)
C
C     **CHECK THE STAGE STOPPING CRITERION IS SATISFYED OR NOT .
     99 IF(NOCUT-INCUT) 100,150,150
    100 IF(ICHECK-1) 101,150,101
    101 CONTINUE
    102 IF(NOF-N) 108,104,104
C
C     **CUT STEP-SIZES FOR MINIMIZING THE P-FUNCTION .
    104 DO 106 I=1,N
    106 D(I)=0.5*D(I)
        NOCUT=NOCUT+1
        IDIFF=IDIFF+1
        IF(IDIFF-INCUT) 50,107,107
```

```
  107 INCUT=INCUT*2
      GO TO 50
  108 NOEXP=NOEXP+1
C
C   **MAKE PATTERN MOVE FOR MINIMIZING THE P-FUNCTION .
      DO 110 I=1,N
      PX(I)=FX(I)+(FX(I)-BX(I))
  110 BX(I)=FX(I)
      LOST=0
      CALL OBRES(PX,Y,G,H)
      IF(LOST-1) 124,124,112
  112 IF(Y-FY) 114,50,50
  114 CALL BACK(PX,X,Y,G,H)
      NOITB=NOITB+1
      NOBP=NOBP+1
C   **CHECK THE ITMAX IS EXCEEDED OR NOT IN (BACK)( LOST=1 MEANS THE
C     RETURNED POINT IS INFEASIBLE )
      IF(LOST-1) 115,150,115
C
C   **ADD NOCUT 1 CREDIT TO STOP THE CURRENT STAGE FASTER AFTER
C     EVERY 5 VIOLATIONSMADE WITHIN THE STAGE .
  115 IF(NOBP-5*MULT) 120,116,116
  116 NOCUT=NOCUT+1
      MULT=MULT+1
  120 LOST=0
C   **CHECK THE ITMAX IS EXCEEDED OR NOT IN (BACK)( LOST .NE. 0 MEANS
C     THE ENTERED POINT IS NEAR-FEASIBLE )
  122 IF(ICHECK-1) 124,140,140
  124 CALL PENAT(G,H,PENA1,PENA2)
      P=Y+R*PENA1+R**(-0.5)*PENA2
      IF(P-FP) 128,48,48
  128 NOPAT=NOPAT+1
      NOITP=NOITP+1
      DO 129 II=1,N
  129 FX(II)=PX(II)
  130 IF(MG) 133,133,131
  131 DO 132 J=1,MG
  132 FG(J)=G(J)
  133 IF(MH) 136,136,134
  134 DO 135 K=1,MH
```

```
  135 FH(K)=H(K)
  136 FY=Y
      FP=P
C
C  **CHECK THE STAGE STOPPING CRITERION IS SATISFYED OR NOT .
      IF(NOCUT-INCUT) 138,150,150
  138 IF(ICHECK-1) 50,150,150
C
C  **CHECK THE ITMAX EXCEEDED POINT( WHEN IT IS RETURNED FROM BACK )
C     IS BETTER OR NOT AND SET PROPER STAGE-OPTIMUM .
  140 CALL OBRES(X,Y,G,H)
      CALL PENAT(G,H,PENA1,PENA2)
      P=Y+R*PENA1+R**(-0.5)*PENA2
      IF(P-FP) 142,150,150
  142 DO 144 I=1,N
  144 FX(I)=X(I)
      GO TO 130
C  **SET THE SUB-OPTIMUM GOT BEFORE ENTERED TO BACK BE THE
C     STAGE-OPTIMUM .
  150 NOPULL=0
      IPULL5=0
      PULL=0.63
      CALL WEIGH(TGH,MG,FG)
      IF(TGH) 170,170,162
  160 CALL WEIGH(TGH,MG,FG)
C
C  **CHECK THE STAGE OPTIMUM IS FEASIBLE OR NOT .
      IF(TGH) 170,170,162
C  **PULL BACK THE INFEASIBLE STAGE-OPTIMUM INTO THE FEASIBLE REGION
  162 DO 163 I=1,N
  163 FX(I)=PULL*(FX(I)-OX(I))+OX(I)
      NOPULL=NOPULL+1
      CALL OBRES(FX,FY,FG,FH)
      NOITB=NOITB+1
      IF(NOPULL-5) 168,164,164
  164 NOPULL=0
      IPULL5=IPULL5+1
      PULL=PULL/2.0
      IF(IPULL5-4) 160,160,165
  165 DO 166 I=1,N
```

```
  166 FX(I)=OX(I)
C   **CHECK THE POINT IS ON THE BOUNDARY OR NOT .
  168 IF(IB-1) 160,162,162
  170 LOST=0
      CALL PENAT(FG,FH,PENA1,PENA2)
      FP=FY+R*PENA1+R**(-0.5)*PENA2
  208 NOIT=NOIT+ITER
      YSTOP=ABS( FY/(FY-R*PENA1+R**(-0.5)*PENA2))
      YSTOP=ABS( YSTOP-1.0)
      WRITE(3,1008) NOR,FY,FP,R,ITER,NOIT,NOBP,NOITP,NOITB,NOEXP,
     1NOPAT,NOCUT,YSTOP
      WRITE(3,1006) (I,FX(I),I,D(I),I=1,N)
      WRITE(3,1011)
  215 WRITE(3,1012) (J,FG(J),J=1,MG)
  216 IF(MH) 218,218,217
  217 WRITE(3,1013) (K,FH(K),K=1,MH)
  218 CALL OUTPUT(N,MG,MH)
      WRITE(3,1007)
      IF(YSTOP-THETA) 226,226,220
C   **CHECK THE MAXP IS EXCEEDED OR NOT .
  220 IF(NOR-MAXP) 221,227,227
C   **STORE LAST SUB-OPTIMUM POINT .
  221 DO 222 I=1,N
  222 OX(I)=FX(I)
C   **SHIFT TO THE NEXT STAGE SEARCH .
      R=R/RATIO
      FP=FY+R*PENA1+R**(-0.5)*PENA2
      NOR=NOR+1
      IF(NOR-5*MP) 224,224,223
  223 INCUT=INCUT+1
      MP=MP+1
      IDR=IDR+5
  224 NOBP=0
      MULT=1
      NOITB=0
      NOITP=0
      ICHECK=0
      NOEXP=0
      NOPAT=0
      NOCUT=0
```

```
      ITER=0
      IB=0
      FNOR=NOR+IDR
C
      B=0.
      DO 225 I=1,N
      D(I)=PD(I)/FNOR
  225 B=B+D(I)
      B=B/FN
      GO TO 50
  226 WRITE(3,1015)
      GO TO 228
  227 WRITE(3,1016) MAXP
  228 IDPM=IDPM+1
      IF(IDPM-NOPM) 1,1,230
  230 STOP
      END
```

```
      SUBROUTINE BACK(XB,X,Y,G,H)
C
C        THIS SUBROUTINE PULLS INFEASIBLE POINTS BACK INTO THE
C     FEASIBLE OR NEAR-FEASIBLE REGION .
C
C        **DEFINITION ..
C              FEASIBLE .. ALL G(I) .GE. 0. ,
C              NEAR-FEASIBLE,. (B-TGH) .GE. 0. .
C
      DIMENSION XB(20),X(20),G(20),H(20),D(20)
      COMMON /BLOGY/ N,MG,MH,ITER,ITMAX,ICHECK,IB,LOST
      COMMON /BLOGB/ NOITP,NOITB,B,D
      CALL WEIGH(TGH,MG,G)
      IF(TGH) 8,8,4
C   **DECREASE THE VALUE OF B IN RETURN .
    4 IF(B-TGH) 12,12,6
    6 IF(0.70*B-TGH) 10,8,8
    8 B=0.75*B
   10 LOST=0
      RETURN
   12 FTGH=TGH
C
C   **MAKE EXPLORATORY MOVE FOR MINIMIZING TGH .
   22 NOF=0
      DO 38 NB=1,N
      XB(NB)=XB(NB)-0.5*D(NB)
      CALL OBRES(XB,Y,G,H)
      CALL WEIGH(TGH,MG,G)
      IF(TGH) 24,24,26
   24 NOITP=NOITP+1
   25 LOST=0
      GO TO 46
   26 NOITB=NOITB+1
      IF(ICHECK-1) 27,45,45
   27 IF(TGH-FTGH) 28,32,32
   28 FTGH=TGH
      IF(B-TGH) 38,38,25
C
   32 XB(NB)=XB(NB)+D(NB)
      CALL OBRES(XB,Y,G,H)
```

```
      CALL WEIGH(TGH,MG,G)
      IF(TGH) 24,24,34
   34 NOITB=NOITB+1
      IF(ICHECK-1) 35,45,45
   35 IF(TGH-FTGH) 28,36,36
   36 XB(NB)=XB(NB)-0.5*D(NB)
      NOF=NOF+1
   38 CONTINUE
      IF(NOF-N) 22,42,42
C
C    **CUT STEP-SIZES FOR MINIMIZING TGH .
   42 DO 44 IR=1,N
      D(IR)=0.5*D(IR)
   44 CONTINUE
      GO TO 22
   45 LOST=1
C
C    **SET BASE POINT TO RETURN .
   46 DO 50 NB=1,N
      D(NB)=D(NB)*0.5
   50 X(NB)=XB(NB)
C    **DECREASE THE VALUE OF B IN RETURN .
      IF(0.7*B-TGH) 60,58,58
   58 B=0.75*B
   60 RETURN
      END
```

```
      SUBROUTINE PENAT(G,H,PENA1,PENA2)
C
C     THIS SUBROUTINE COMPUTES THE PENALTY TERMS FOR SUMT FORMULATION .
C        PENA1 FOR INEQUALITY CONSTRAINTS .
C        PENA2 FOR EQUALITY CONSTRAINTS .
C
      DIMENSION G(20),H(20)
      COMMON /BLOGY/ N,MG,MH,ITER,ITMAX,ICHECK.IB.LOST
      PENA1=0.
      PENA2=0.
      IF(MG) 5,5,1
    1 DO 4 I=1.MG
      IF(G(I)) 4,2,4
C   **SET G(I)=0.1E-48  WHEN G(I)=0. ( ON THE BOUNDARY )
    2 G(I)=0.1E-48
    4 PENA1=PENA1+ABS(1.00/G(I))
    5 IF(MH) 10,10,6
    6 DO 9 K=1,MH
      IF(H(K)) 9,8,9
    8 PENA2=PENA2+H(K)**2
    9 CONTINUE
   10 RETURN
      END
```

```
      SUBROUTINE WEIGH(TGH,MG,G)
C
C     THIS SUBROUTINE COMPUTES THE TOTAL WEIGHT OF VIOLATION
C     TO THE INEQUALITY CONSTRAINTS .
C
      DIMENSION G(20)
      TGH=0.
      IF(MG) 4,4,1
    1 DO 3 IR=1,MG
      IF(G(IR)) 2,3,3
    2 TGH=TGH+G(IR)**2
    3 CONTINUE
    4 TGH=TGH**0.5
      RETURN
      END
```

```
356          SUBROUTINE READIN(N,MG,MH)
       C     THIS SUBROUTINE IS FOR READ IN ADDITIONAL DATA .
       C     USER SUPPLIES HIS OWN READ STATEMENT AND FORMAT .
       C     ARGUMENTS N,MG,MH ARE NUMBERS OF VARIABLES,OF INEQUALITY CONSTRAINTS
       C                       AND OF EQUALITY CONSTRAINTS .
       C     COMMON /BLOGR/ ....... STATEMENT IS FOR TRANSFER DATA USE .
       C
357          COMMON /BLOGR/ Q(10)
358          RETURN
359          END
```

```
      SUBROUTINE OUTPUT(N,MG,MH)
C        THIS SUBROUTINE IS FOR USER TO PRINT OUT ADDITIONAL INFORMATION
C     WANTED. ARGUMENTS N,MG,MH ARE NUMBERS OF VARIABLES,OF INEQUALITY
C     CONSTRAINTS,AND OF EQUALITY CONSTRAINTS .
C        COMMON /BLOGO/,...., IS FOR TRANSFER NEEDED DATA IN MAIN TO
C     THE SUBROUTINE OUTPUT .
C        USER SUPLLIES ALL NECESSARY FORMATS .
C
      DIMENSION G(20)
      COMMON /BLOGO/ G
      RETURN
      END
```

```
      SUBROUTINE OBRES(X,Y,G,H)
C
C     THIS SUBROUTINE COMPUTES OBJ. AND CONSTRAINT VALUES .
C     USER SHOULD SUPPLY ALL NECESSARY STATEMENTS IN THE FORM ..
C        Y=......, FUNCTION OF X(I) , FOR OBJECTIVE FUNCTION .
C        G(J)=......, J FROM 1 TO MG , FOR CONSTRAINTS G(J) .GT. 0.0  .
C        H(K)=......, K FROM 1 TO MH , FOR CONSTRAINTS H(K) .EQ. 0.0  .
C     INSERT THESE STATEMENTS IN THE BLOCK BELOW LINED BY ********** .
C
      DIMENSION X(20),G(20),H(20),Q(10)
      COMMON /BLOGY/ N,MG,MH,ITER,ITMAX,ICHECK,IB,LOST
      COMMON /BLOGR/ Q
  100 FORMAT(3X25H**THE ITERATION EXCEEDED I5,1H.)
C
C  ***************************************************************
C   **NOTE.. STATEMENT NUMBERS 1,2,3,4,5,6,7,8,100 HAVE BEEN USED.
C
C
C  ***************************************************************
      ITER=ITER+1
      IF(ITER-ITMAX) 3,1,2
C   **OUTPUT THE MESSAGE OF ITMAX EXCEEDED.
    1 PUNCH 100,ITMAX
    2 ICHECK=1
C   **CHECK FOR THE VIOLATION TO INEQUALITY CONSTRAINTS.
    3 IB=0
      IF(MG) 8,8,4
    4 DO 7 I=1,MG
      IF(G(I)) 5,6,7
    5 LOST=2
      GO TO 7
    6 IB=1
    7 CONTINUE
    8 RETURN
      END
```

OPTIMIZATION OF INDUSTRIAL MANAGEMENT SYSTEMS BY THE SEQUENTIAL UNCONSTRAINED MINIMIZATION TECHNIQUE

by

KEETING CHIECHENG LAI

B.S. Chong-Uan College of Science and Technology

Taw-Uan, Formosa, 1965

AN ABSTRACT OF A MASTER'S REPORT

Submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

Kansas State University

Manhattan, Kansas

1970

The problems considered in this report are optimization of system reliability of a complex system and optimization of production scheduling and inventory control subject to some linear and/or nonlinear constraints. The optimization method employed is the sequential unconstrained minimization technique (SUMT).

The purposes of this report are twofold. The first is to present a result of implementing SUMT by a combination of the Hooke and Jeeves pattern search technique and a heuristic programming technique. The second is to present results of employing the developed technique to the optimization of system reliability of a complex system and production scheduling and inventory control problems.

There is a general computer program available entitled "RAC Computer program Implementing the Sequential Unconstrained Minimization Technique for Nonlinear Programming". In This computer program, the unconstrained minimization technique used is the second order gradient method. Difficulties which arise from use of the second order gradient method as a unconstrained minimization technique in SUMT becomes predominate in a large size and/or very complex nonlinear problem. The difficulties arised particularly in taking correctly the first order and second order partial derivatives of complex nonlinear functions which most of practical problems have. Therefore, a new algorithm which using a much simpler direct search technique is very desirable. For this reason, a new technique of implementing SUMT by the Hooke and Jeeves pattern search technique to be its unconstrained minimization has been developed.

This newly developed method is utilized to obtain the optimum solutions of two examples of production scheduling and inventory control problems. The first problem is a simple two dimensional problem used for demonstrating the procedure of the algorithm in details and the second problem is a 20-dimensional problem used for demonstrating the capacity and practicability of the technique.

The problem of optimizing a system reliability becomes considerably more difficult when the redundant units of the system cannot be reduced to pure parallel or series configurations. In such a complex system the system reliability is obtained by Bayes' theorem which utilizes conditional probabilities. A mathematical model for the nonlinear system reliability subject to constraints is formulated. The nonlinear programming problem of optimizing the system reliability is then solved by SUMT using RAC computer program and by the newly developed technique and the results are compared.