Inspection Allocation in Manufacturing Systems Using Stochastic Search Techniques

N. Viswanadham, *Fellow, IEEE*, Shashi M. Sharma, and Mukesh Taneja

Abstract—**Quality is the hallmark of a competitive product. It is necessary to use inspection stations to check product quality and process performance. In this paper, we are concerned with the problem of location of inspection stations in a multistage manufacturing system. We present two stochastic search algorithms for solving this problem, one based on Simulated Annealing and the other on Genetic Algorithms. These algorithms are developed to determine the location of inspection stations resulting in a minimum expected total cost in a multistage manufacturing system. The total cost includes inspection, processing and scrapping cost at each stage of the production process. A penalty cost is also included in it to account for a defective item which is not detected by the inspection scheme. A set of test examples are solved using these algorithms. We also compare performance of these two algorithms.**

I. Introduction

QUALITY is the hallmark of a competitive product. Consumers reject products that are of inferior quality and they shun companies who are perceived to provide products with less than acceptable quality. A company cannot survive in the world marketplace without providing a high quality product. Pressures for improvement have become intense. The result is a heightened interest in strategic quality management at many companies and growing recognition of strategic importance of quality.

"Quality" is not a universal descriptor that has a unique definition under all circumstances. Garvin [1] has described five approaches to defining quality: transcendent, product-based, user-based, manufacturing-based and value-based.

According to the transcendent view, quality is synonymous with "innate excellence". It is both absolute and universally recognizable, a mark of uncompromising standards and high achievement. Product-based definition views quality as a precise and measurable variable. Differences in quality thus reflect differences in the quantity of some ingredient or attribute processed by a product. According to the user-based view, the quality "lies in the eyes of the beholder". Individual customers are assumed to have different wants or needs, and the goods that best satisfy their requirements are the ones they regard as having the highest quality. Manufacturing-based definitions focus on the supply side of the equation and are primarily concerned with engineering and manufacturing practices. These identify quality as "conformance to requirements". Improvement in quality leads to lower cost, for preventing defects is viewed as less expensive than repairing or reworking them. Value-based view defines quality in terms of costs and prices. A quality product is one that provides performance or conformance at an acceptable price or cost.

According to Juran [2], cost of achieving a given level of quality can be divided into avoidable and unavoidable costs. Unavoidable cost includes costs of prevention-inspection, sampling and sorting. Avoidable cost includes costs of defects and failures, scrapped material, labor hours required for rework and repair, complaint processing and financial losses resulting from unsatisfied customers.

In this paper, we are concerned with one very important component of the unavoidable quality cost, i.e. the inspection cost. In spite of the best process control methods used, it is essential to use inspection stations to check product quality and process performance. We consider both the serial and nonserial manufacturing systems. In a serial multistage manufacturing system, raw material is transformed into the final product in a series of discrete manufacturing stages. In a nonserial system certain manufacturing stages may involve joining the results of previous stages. If a manufacturing operation is not performed properly, some of the product units may become nonconforming. The introduction of inspection stations into the manufacturing process entails additional costs and this results in an increase in the total cost. Careful analysis is required to determine the location of inspection stations that justify the additional cost.

Inspection allocation models are formulated with the objective of determining the number and location of inspection stations which will minimize the expected total cost per unit produced. The total cost includes some or all of the following: inspection cost, diagnosis and repair cost, loss resulting from removal of a unit perceived to be nonconforming less any applicable salvage value and penalty associated with shipping a nonconforming unit. Possible *constraints* on the above problem are based on an accepted outgoing quality level (AOQL) and/or on a limit on the maximum number of inspection stations that may be used.

Manuscript received April 24, 1994; revised December 28, 1994. This work was carried out under the Indo-U.S. collaborative project "Modeling, Analysis and Control of Discrete Event Systems", ONR Grant N0014-93-1017.

N. Viswanadham and S. M. Sharma are with the Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560 012, India (e-mail: vishu@bhaskara.csa.iisc.ernet.in).

M. Taneja is with the Department of Industrial Engineering and Operations Research, Columbia University, New York, NY 10027 USA (e-mail: taneja@ieor.columbia.edu).

Publisher Item Identifier S 1083-4427(96)01396-3.

Ignoring the possibility that inspection procedures could conceivably be integrated into the manufacturing operation itself, 2^N possible inspection location alternatives exist for a N stage manufacturing system. For all but the modest N, identification of the minimum cost inspection allocation plan by a complete enumeration of combinations becomes prohibitive. Various solution approaches have been employed for solving this problem. These approaches can be classified in two categories: exact and approximate methods. The dynamic programming [3]–[5], and integer programming [6] techniques fall in the first category. Though these techniques yield optimal solution, they are computationally very expensive. The computational effort further goes up if the optimization problem is a constrained one.

The second category consist of approximation techniques which yield a nearly optimal solution at a considerably lower computational effort. Among the various approximation methods, two types of stochastic search techniques which are promising are: Genetic Algorithms (GA's) and Simulated Annealing (SA). Both are modeled on processes found in nature (natural evolution and thermodynamics). Both these techniques can process cost functions possessing quite arbitrary degrees of nonlinearities, discontinuities and stochasticity. These techniques are similar in the sense that they achieve their power by demanding that the problems be mapped onto their own particular representation in order to be solved. These algorithms perform very well if a fairly natural mapping exists. However the advantage of using the genetic algorithms is that they are suitable for processing by massively parallel machines.

In this paper, we design these two algorithms for locating the inspection stations in a multistage manufacturing system. It is found that the inspection allocation problem maps fairly well into the representation required by these two algorithms. We consider both the serial and nonserial manufacturing systems. The paper is organized as follows. The problem formulation is presented in Section II. In Section III, we present an overview of genetic and simulated annealing algorithms. Experimental results are presented in Section IV and conclusions are drawn in Section V.

II. Problem Formulation

A. Serial Multi-Stage System

A serial multi-stage manufacturing process is shown in Fig. 1. There are N discrete manufacturing stages through which the work in process is routed in a fixed sequence. Each stage of the manufacturing process receives as input a batch or stream of identically processed items, which contains some mix of conforming and nonconforming items. The option is available to locate an inspection center before each manufacturing stage. If extra-high quality is required, repeated inspections may be performed. The objective is to find the number of repeated inspections at each stage as well as the number of such inspection stations such that the total cost of the manufacturing process is minimized. We make the following assumptions:

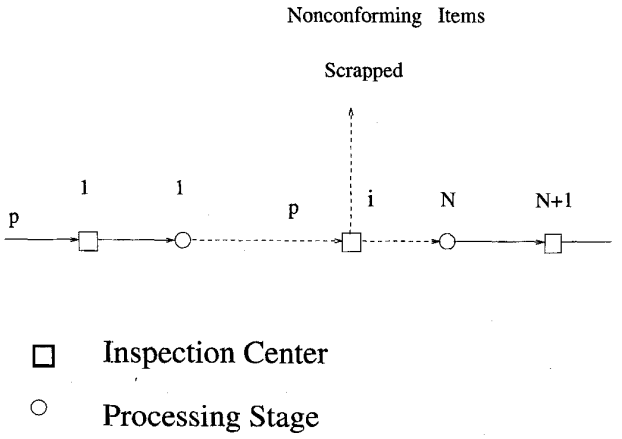


Fig. 1. Serial manufacturing process.

1) The probability of being conforming is known for each item entering the manufacturing line.
2) An inspection operation may involve errors of two types. It may reject a conforming item or accept a nonconforming item. These are referred to as type I and type II errors respectively. The probabilities of these errors are known at each stage. Also, these are constant for each *inspection level* at a particular stage. Here, an inspection level for a stage refers to the number of repeated inspections performed at that stage.
3) All the nonconforming items are scrapped.
4) The inspection costs, scrapping costs and manufacturing costs are known for each stage in the manufacturing line.
5) If a nonconforming item reaches the customer, a penalty cost is incurred. The penalty cost is taken into account after the last stage in Fig. 1.

In order to specify the cost elements, we consider the system in Fig. 1 with repeated inspections at each stage of the manufacturing system. If an item is conforming and is accepted at all the repeated inspections, then the total cost would be equal to the cost of inspection plus the cost of manufacturing that item. If the item is nonconforming and is not rejected at any of the repeated inspections, the total cost would be equal to the sum of the inspection cost, penalty cost and the manufacturing cost. If an item is rejected at some inspection station, then the total cost would be equal to the sum of the inspection cost, scrapping cost and manufacturing cost.

Let p_i be the probability that any single item is conforming at stage i. If we choose to inspect at stage i, an inspection cost of n_i occurs for each item inspected. Here, one or more measurements (for testing) are taken for an item and a decision has to be made whether to accept or reject the item. The accuracy of these measurements determines the prevalence of type I (false reject) and type II (false accept) errors. Let α_i be the probability of type I error. Thus if an item is conforming, it will be rejected with probability α_i during an inspection process at stage i. Similarly, let β_i be the probability of type II error i.e. a nonconforming item will be accepted with probability β_i. The number of repeated inspections performed at stage i is denoted by x_i. The probability of accepting an item

after x_i inspections at stage i is denoted by $A_i(p_i, x_i)$. Then

$$A_i(p_i, x_i) = A_{i-1}(p_{i-1}, x_{i-1})\tilde{A}_i(p_i, x_i). \quad (1)$$

Here,

$$\tilde{A}_i(p_i, x_i) = p_i(1-\alpha_i)^{x_i} + (1-p_i)\beta_i^{x_i}. \quad (2)$$

Let s_i be the scrapping cost at stage i respectively. The scrapping cost represents the income generated by selling nonconforming units as scrap or lower grade products. This is treated as a negative cost. The unit manufacturing cost of an item at stage i is denoted by c_i. A manufacturing stage may cause nonconformity in an item during the manufacturing. The probability that a conforming item becomes nonconforming through stage i is denoted by f_i. The objective function to be minimized at the ith stage is the expected total cost, given that the input has probability p_i of being conforming. The total expected cost $T_i(p_i, x_i)$ is equal to the manufacturing cost at stage i when $x_i = 0$. If $x_i > 0$ then $T_i(p_i, x_i)$ is the sum of three types of costs: inspection cost, scrapping cost and manufacturing cost. The expected inspection cost at the ith stage is given by

$$I_i(p_i, x_i) = A_{i-1}(p_{i-1}, x_{i-1})N_i(p_i, x_i). \quad (3)$$

Here, $N_i(p_i, x_i)$ is given by

$$\begin{aligned} N_i(p_i, x_i) = {} & n_i x_i (1-p_i)\beta_i^{x_i} + n_i x_i p_i (1-\alpha_i)^{x_i} \\ & + n_i(1-p_i)\sum_{k=1}^{x_i} k(1-\beta_i)\beta_i^{k-1} \\ & + n_i p_i \sum_{k=1}^{x_i} k\alpha_i(1-\alpha_i)^{k-1}. \end{aligned} \quad (4)$$

Here, the four terms correspond to undetected nonconforming item, accepted conforming item, detected nonconforming item and rejected conforming item respectively. The summation in the third term represents the expected number of inspections for rejection of nonconforming item. The summation in the fourth term represents the expected number of inspections for the rejection of conforming items. Equation (4) is valid for $1 \le i \le N+1$. If $\alpha_i \neq 0$ and $\beta_i \neq 1$, then (4) can be simplified to

$$N_i(p_i, x_i) = q\frac{n_i p_i(1-(1-\alpha_i)^{x_i})}{\alpha_i} + \frac{n_i(1-p_i)(1-\beta_i^{x_i})}{1-\beta_i}. \quad (5)$$

The probability that an item is rejected is given by

$$\tilde{D}_i(p_i, x_i) = p_i\sum_{k=1}^{x_i}\alpha_i(1-\alpha_i)^{k-1} + (1-p_i)\sum_{k=1}^{x_i}(1-\beta_i)\beta_i^{k-1}. \quad (6)$$

Equation (6) can be simplified to

$$\tilde{D}_i(p_i, x_i) = p_i(1-(1-\alpha_i)^{x_i}) + (1-p_i)(1-\beta_i^{x_i}). \quad (7)$$

Let $M_i(p_i, x_i)$ be the modified probability that an item is conforming after x_i inspections. Then,

$$M_i(p_i, x_i) = \frac{p_i(1-\alpha_i)^{x_i}}{\tilde{A}_i(p_i, x_i)}. \quad (8)$$

The expected scrapping cost is given by

$$S_i(p_i, x_i) = s_i A_{i-1}(p_{i-1}, x_{i-1})\tilde{D}_i(p_i, x_i). \quad (9)$$

The manufacturing cost at the ith stage is given by

$$P_i(p_i, x_i) = c_i A_i(p_i, x_i). \quad (10)$$

Equations (1)–(10) are valid for $x_i > 0$. The total cost incurred at the ith stage is given by

$$T_i(p_i, x_i) = \begin{cases} P_i(p_i, x_i) & \text{if } x_i = 0 \\ I_i(p_i, x_i) + S_i(p_i, x_i) + P_i(p_i, x_i) & \text{if } x_i > 0 \end{cases}. \quad (11)$$

The probability of the output stream of the ith stage is given by

$$p_{i+1} = \begin{cases} p_i(1-f_i) & \text{if } x_i = 0 \\ M_i(p_i, x_i)(1-f_i) & \text{if } x_i > 0 \end{cases}. \quad (12)$$

Once the manufacturing process is completed, all items are supplied to the customers. Let d be the cost of shipping a defective item to a customer site. It includes cost of a field repair, cost of the analysis & repair of the defective item that comes back to the plant and a cost measuring the customer's loss of good will. It represents the penalty cost occurred in the whole manufacturing process. The revenue obtained from selling an item is denoted by v. The net income generated by selling that item is given by

$$G(p_{N+2}) = A_{N+2}[p_{N+2}(d+v) - d]. \quad (13)$$

Let the vector $\bar{x} = (x_1, \ldots, x_{N+1})^T$ specify the decisions taken in the manufacturing line. It is to be noted that x_i can take only nonnegative integer values. Given the input probability p_1 and the decision vector $\bar{x}$, the total cost can be computed as

$$T(p_1, \bar{x}) = \sum_{i=1}^{i=N+1} T_i(p_i, x_i) - G(p_{N+2}). \quad (14)$$

Here, T_i and $G(p_{N+2})$ are defined in Eqs. (11) and (13) respectively. The objective is to find a vector $\bar{x}$ such that the total cost in Eq. (14) is minimized for a given p_1. Thus the above optimization problem is defined as

$$\begin{aligned} \text{P1:} \quad & \min T(p_1, \bar{x}) \\ & \text{subject to } x_i \ge 0 \quad \forall i \in [1, N+1]. \end{aligned} \quad (15)$$

Depending on the nature of process technology and economic limitations, it may be necessary to incorporate some additional constraints into the above formulation. These constraints can be of the following types

1) Maximum number of inspection stations that can be used.
2) Minimum acceptable outgoing quality level required.
3) Maximum number of repeated inspections allowed at each stage.

If all of the above constraints are present, then the constrained problem becomes

$$\begin{aligned} \text{P2:} \quad & \min T(p_1, \bar{x}) \\ & \text{subject to } L \le L_{\max}, \\ & p_{N+2} \ge p^* \text{ and} \\ & 0 \le x_i \le z_i. \end{aligned} \tag{16}$$

Here, L is the actual number of inspection stations located and $L_{\max}$ is the maximum number of inspection stations that can be used. The acceptable outgoing quality level (AOQL) is denoted by p^*. The vector $\bar{z} = [z_1, \ldots, z_{N+1}]$ denotes the maximum number of repetitive inspections allowed.

Similarly, if we assume that all the rejected items are reworked, then the corresponding equations are listed in (17). Here we assume that a reworked item is always good and it reenters the manufacturing line.

$$\begin{aligned} I_i(p_i, x_i) &= N_i(p_i, x_i) \\ M_i(p_i, x_i) &= 1 - (1 - p_i)\beta_i^{x_i} \\ R_i(p_i, x_i) &= r_i \tilde{D}_i(p_i, x_i) \\ P_i(p_i, x_i) &= c_i. \end{aligned} \tag{17}$$

Here, r_i denotes the reworking cost at stage i and $R_i(p_i, x_i)$ denotes the expected reworking cost. Now, we discuss some special cases of the above problem.

Case 1. Assumptions:

1) Repetitive inspections are not allowed. Thus we can inspect only once at a stage.
2) All the rejected items are scrapped.

Thus,

$$\begin{aligned} x_i &= 0 \text{ or } 1 \quad \forall i \in [1, N+1] \\ r_i &= 0 \qquad \forall i \in [1, N+1]. \end{aligned} \tag{18}$$

In this case, a presence of an inspection center at stage i is represented by $x_i = 1$ and an absence is represented by $x_i = 0$. Equations (2), (4), (7)–(9) reduce to the following

$$\begin{aligned} \tilde{A}_i(p_i) &= p_i(1 - \alpha_i) + (1 - p_i)\beta_i \\ N_i(p_i) &= n_i \\ \tilde{D}_i(p_i) &= p_i\alpha_i + (1 - p_i)(1 - \beta_i) \\ M_i(p_i) &= \frac{p_i(1 - \alpha_i)}{\tilde{A}_i(p_i)} \\ S_i(p_i) &= s_i A_{i-1}(p_{i-1})\tilde{D}_i(p_i). \end{aligned} \tag{19}$$

The cost incurred at the ith stage is given by

$$T_i(p_i, x_i) = \begin{cases} P_i(p_i) & \text{if } x_i = 0 \\ I_i(p_i) + S_i(p_i) + P_i(p_i) & \text{if } x_i = 1 \end{cases}. \tag{20}$$

The probability of the output stream of the ith stage is given by

$$p_{i+1} = \begin{cases} p_i(1 - f_i) & \text{if } x_i = 0 \\ M_i(p_i)(1 - f_i) & \text{if } x_i = 1 \end{cases}. \tag{21}$$

The above two expressions can be rewritten as

$$T_i(p_i, x_i) = P_i(p_i)(1 - x_i) + [I_i(p_i) + S_i(p_i) + P_i(p_i)]x_i \tag{22}$$

and

$$p_{i+1} = p_i(1 - f_i)(1 - x_i) + M_i(p_i)(1 - f_i)x_i. \tag{23}$$

Eppen and Hurst [3] have discussed the unconstrained problem (P1), under the above assumptions. They formulate a recursive objective function and solve that using dynamic programming.

Case 2. Assumptions:

1) Repetitive inspections are not allowed.
2) All the rejected items are reworked.

Thus

$$\begin{aligned} x_i &= 0 \text{ or } 1 \quad \forall i \in [1, N+1] \\ s_i &= 0 \qquad \forall i \in [1, N+1]. \end{aligned} \tag{24}$$

Corresponding equations are given by

$$\begin{aligned} I_i(p_i) &= n_i \\ R_i(p_i) &= r_i \tilde{D}_i(p_i) \\ S_i(p_i) &= 0. \end{aligned} \tag{25}$$

The cost incurred at the ith stage is given by

$$T_i(p_i, x_i) = \begin{cases} P_i(p_i) & \text{if } x_i = 0 \\ I_i(p_i) + R_i(p_i) + P_i(p_i) & \text{if } x_i = 1 \end{cases}. \tag{26}$$

Case 3. Assumptions:

1) Repetitive inspections are allowed.
2) All the rejected items are scrapped.
3) Probability of type I error is zero.

Thus,

$$\begin{aligned} r_i &= 0 \quad \forall i \in [1, N+1] \\ \alpha_i &= 0 \quad \forall i \in [1, N+1]. \end{aligned} \tag{27}$$

Now, (2), (7) and (8) reduce to the following:

$$\begin{aligned} \tilde{A}_i(p_i, x_i) &= p_i + (1 - p_i)\beta_i^{x_i} \\ \tilde{D}_i(p_i, x_i) &= (1 - p_i)(1 - \beta_i^{x_i}) \\ M_i(p_i, x_i) &= \frac{p_i}{\tilde{A}_i(p_i, x_i)}. \end{aligned} \tag{28}$$

The cost incurred at the ith stage is given by

$$T_i(p_i, x_i) = \begin{cases} P_i(p_i, x_i) & \text{if } x_i = 0 \\ I_i(p_i, x_i) + S_i(p_i, x_i) & \text{if } x_i > 0 \\ \quad + P_i(p_i, x_i) & \end{cases}. \tag{29}$$

B. Nonserial Multistage System

We consider a special class of nonserial manufacturing system shown in Fig. 2. The inspection allocation problem for this configuration has been solved using dynamic programming in Garcia-Diaz [4]. We use the stochastic search techniques namely genetic algorithm and simulated annealing for solving this problem. At each stage in Fig. 2, there are exactly two inputs. A basic module of this configuration is shown in Fig. 3.

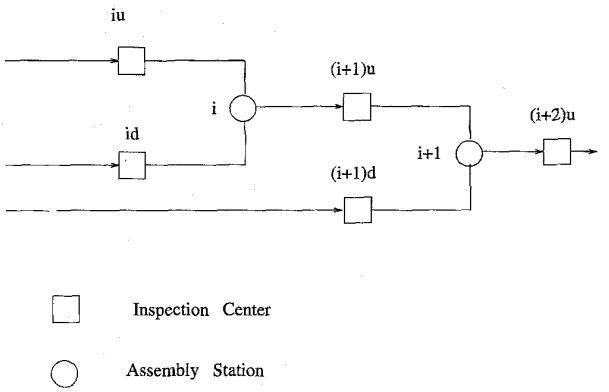


Fig. 2. Nonserial manufacturing process.

The inspection stations at the upper and lower branch of stage i are denoted by iu and id respectively.

Notations:

r_{iu} (r_{id})	rework cost at inspection station iu (id)
n_{iu} (n_{id})	inspection cost at inspection station iu (id)
p_{iu} (p_{id})	p(conforming) at the input of iu (id)
p'_{iu} (p'_{id})	p(conforming) at the output of iu (id)
$p_{(i+1)u}$	p(conforming) at the output of stage i
x_{iu} (x_{id})	number of repeated inspections at iu (id)
α_{iu} (α_{id})	p(Type I error) at iu (id)
β_{iu} (β_{id})	p(Type II error) at iu (id)
I_{iu} (I_{id})	expected inspection cost at iu (id)
R_{iu} (R_{id})	expected reworking cost at iu (id)
P_i	assembly cost at stage i

We assume that all the rejected items are reworked. The inspection cost, reworking cost, manufacturing cost and the modified probability after inspection are given by (17) with subscript i changed to iu and id for the two inspection stations in Fig. 3. The probability after inspection (p'_{iu}) at center iu is equal to M_{iu}. If there is no inspection, then it is same as p_{iu} only. The probability of an item being conforming at the output is given by

$$p_{(i+1)u} = (1 - f_i) + p'_{iu} + p'_{id} - (1 - f_i)p'_{iu} - (1 - f_i)p'_{id} - p'_{iu}p'_{id} + p'_{iu}p'_{id}(1 - f_i). \quad (30)$$

The total cost for the ith basic module is given by

$$T_i = I_{iu} + I_{id} + R_{iu} + R_{id} + P_i. \quad (31)$$

Here, one or more of the above terms will be equal to zero, in case an item is not inspected at a stage. The total cost for a N stage system is given by

$$T = \sum_{i=1}^{i=N+1} T_i - G(p_{N+2}). \quad (32)$$

Here, we assume that the $(N+1)$th stage act as a dummy stage with $I_{(N+1)d} = R_{(N+1)d} = P_{(N+1)} = 0$ and $G(p_{N+2})$ is as defined in (13). The output probability of this stage is given by

$$p_{(N+2)} = p'_{(N+1)u} + (1 - f_{(N+1)}) - p'_{(N+1)u}(1 - f_{(N+1)}). \quad (33)$$

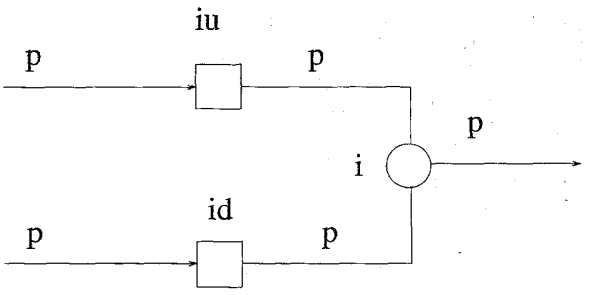


Fig. 3. Basic module of nonserial process.

The optimization problem is to minimize the total cost given in (32). The next Section presents a brief overview of genetic algorithms and simulated annealing.

III. Stochastic Search Techniques

In this Section, we discuss about two types of stochastic search techniques: genetic algorithms and simulated annealing. Both techniques have been applied to problems that are difficult and important. Both are modeled on processes found in nature (natural evolution and thermodynamics). These techniques achieve their power by demanding that problems be mapped onto their own particular representation in order to be solved. If a fairly natural mapping exists, impressive robust performance results. We show that the inspection allocation problem maps fairly well into the representation required by these techniques.

A. Genetic Algorithms

Genetic Algorithms (GA's) are search algorithms based on the mechanics of natural selection and natural genetics. They combine the concept of survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some innovative flair of human search. GA's are different from other optimization methods in the following ways [7]

1) GA's work with strings of characters representing the parameter set and not with the parameters themselves.
2) GA's maintain a population of potential solutions. Each solution is represented by a string of binary bits. Conventional search algorithms consider a single point at a time.
3) GA's use only the objective function information and not their derivatives etc.
4) GA's use probabilistic transition rules in contrast to the deterministic transition rules used by most of the other methods.

GA's require the natural parameter set of the optimization problem to be coded as a finite-length string over some finite alphabet. The parameters in our implementation are coded as strings of 0's and 1's. In the case, when only a single inspection is allowed, each parameter x_i is represented by a one bit string. A '0' indicates absence of an inspection station at the ith stage and a '1' indicates presence of an inspection center. Thus a solution is represented by a $(N+1)$

bit string $x_1, \ldots, x_{N+1}$, where each x_i can take value either 0 or 1. When repeated inspections are allowed, each of these parameters can be coded as a finite length string of 0's and 1's.

GA's consider many points in the search space simultaneously and therefore have a reduced chance of converging to a local optima. GA's evaluate each point independently in the search space and combine qualities from these points to produce a improved population. They require only objective function information to evaluate a point in the search space. As GA's do not require problem-specific information, they are more flexible than most of the other search methods. Finally, they use random information efficiently in their exploitation of prior knowledge to rapidly locate near-optimal solutions.

The Mechanics of a GA: A simple GA is compose of the following three operators

1) Reproduction
2) Crossover
3) Mutation

These operators are applied to successive population to generate new population. Reproduction is a process in which strings are copied according to their fitness values. This means that a string with a higher fitness value has a higher probability of contributing one or more offsprings in the next generation. This operator is implemented by using roulette wheel strategy, histogram estimation, Elitist expected value strategy, etc. A detailed description of these can be found in [7].

The crossover operator combines the features of two selected strings to form two new solution strings. This is accomplished in two steps. First, an integer position k along the string is selected at random between positions 1 and the string length less one. Two new strings are created by swapping all bits between positions $k+1$ and the string length. This operator is applied with a given crossover probability. It introduces new solution strings in the population and searches for better strings. The mutation operator forms a new string by flipping some of the bits in a string and is performed with a specified mutation rate. This operator plays a secondary role in a simple GA and is used to introduce new bits in a solution string.

Mutation rate, crossover rate and population size constitute an important control parameter set for a GA. It has been acknowledged that these parameters can have a significant impact on its performance and that the theory behind this technology gives little guidance for their proper selection. If the crossover rate is high, more new strings will be introduced in the population. A high mutation rate will drive the genetic search to a random search, whereas a low mutation rate will result in loss of some important characteristics in a population. Normally, a low mutation rate and a high crossover rate is selected for all the generations. GA's with a low size population are prune to premature convergence. This happens as a superior individual (one with high fitness value) may take over other individuals in a population. But if the population size is large, it slows down the convergence rate of the algorithm and increases the number of objective function evaluations. A suitable combination of these parameters has to be found for the GA to work well on a given problem.

```
Step 1.Get an initial configuration.
Step 2.Get an initial temperature T > 0.
Step 3.
      3.1   For 1 ≤ i ≤ L
            3.1.1 Pick any random neighbor Ŝ of S.
            3.1.2 Let Δ = cost(Ŝ) - cost(S).
            3.3.3 If Δ < 0 then S = Ŝ
            3.1.4 If Δ ≥ 0 then set S = Ŝ with
                  probability exp(-Δ/T)
      3.2   Set T = α.T
Step 4 Return S
```

Fig. 4. The simulated annealing algorithm.

GA's have been applied to many complex optimization problems and it is shown that these are competent in finding optimal or near optimal solutions [7] and [8]. Application of GA's to constrained problems require some special methods. Three different approaches exist for this purpose. The first two involves transforming potential solutions of the problem into a form suitable for the GA and then using penalty functions or applying "decoders" or "repair" algorithm. The third approach involves modifying the GA to suit the problem by using new data structures and new genetic operators. Details of these methods can be found in [7] and [9]. The first method, namely the *Exterior penalty method*, has been used successively in a number of problems. With this method, whenever a constraint is violated, the unconstrained objective function value is penalized by an amount related to a function of the constraint violation. We use this method for solving the constrained inspection allocation problem.

B. Simulated Annealing

Simulated annealing is a stochastic computational technique derived from statistical mechanics for finding near globally-minimum cost solutions to difficult optimization problems. Kirkpatrick *et al.* [10] were the first to propose and demonstrate the application of annealing techniques from statistical physics to problems of combinatorial optimization.The name of the algorithm derives from an analogy between solving optimization problems and annealing of solids as proposed by Metropolis *et al.* [12]. Computational experiments with algorithm on a variety of notorious combinatorial optimization problems are reported on Aarts and Van Laarhoven [12], Aarts and Korst [13] and Collins *et al.* [14]. A description of the simulated annealing algorithm appears in Fig. III-B of [14].

The choices the designer of a simulated annealing has to make can be classified into two categories

- Problem specific
- Generic

The problem specific parameters are

1) *Configuration:* The state of the system is represented in this case by a binary valued vector $S = \{s_1, s_2 \ldots s_n\}$ where s_i is 1 if there is an inspection center at stage i otherwise it 0.
2) *Neighborhood of the Configuration:* The neighborhood of S is the set of states to which the system can move from S with nonzero probability. For our problem the

neighborhood of state S is the set $\mathcal{S}$ of configuration which are at unit hamming distance from S.

3) *Cost of the Configuration:* The cost of the configuration is the measure of the goodness of the configuration. We have already defined an appropriate cost function for our problem in the previous Section.
4) *Initial Configuration:* The initial configuration is chosen picking up a random configuration from the set of all possible configurations.

The generic parameters of the simulated annealing are

- Initial value of the temperature T_0
- Decreasing factor α
- Stopping criterion
- Length of the iteration at each value of temperature

Starting from an initial configuration the simulated annealing algorithm generates at random a new configuration from the neighborhood of the original configuration. If the change represents a reduction in the value of the objective function then the transition to the new configuration is accepted. If the change represents an increase in the objective function, then the transition to the new configuration is accepted with a specified probability. The acceptance probability function usually takes the form $\exp(-\Delta/T)$ where T is the control parameter. This provides a mechanism which enables an SA algorithm to avoid becoming trapped in a local minimum in its search for the global minimum.

There are four generic parameters which must be specified namely the initial temperature T_0, the stopping criterion, the length of the iterator L and a rule specifying how the temperature is reduced. A choice of these parameters is referred to as cooling schedule. The initial temperature T_0 is chosen such that a cost increasing transition occurs with probability P_0. The length of the iteration is chosen in the order of the size of the neighborhood. The temperature is decremented by the following rule

$$T_{k+1} = \alpha T_k. \tag{34}$$

Typical value of α lies between 0.85 to 0.95.

The last detail that must be filled is the stopping criterion which will determine when the system is frozen. In our implementation of SA the execution is terminated when the optimal value is reached or if there is no improvement in the optimal value for a number of temperature reduction stages.

In Section IV, we discuss the results obtained by applying genetic algorithm and simulated annealing on a set of test problems. We consider some special cases of the serial and nonserial manufacturing systems and present the results for them.

IV. Experimental Study

The GA in its standard form assumes that the problem is a maximization problem and the objective function takes only positive values on its domain. Thus minimization problem in (15) is converted to the following maximization form

$$\begin{aligned} \text{P1}': \quad & \max T'(p_1, \bar{x}) \\ & \text{subject to } x_i \geq 0 \quad \forall i \in [1, N+1]. \end{aligned} \tag{35}$$

TABLE I
Test Problem Statistics

i	$1-f_i$	α_i	β_i	n_i	s_i	c_i
1	0.998	0.01	0.04	0.2	-1	4
2	0.985	0.01	0.04	0.5	-1	5
3	0.989	0.02	0.06	0.6	-3	20
4	0.992	0.03	0.07	0.8	-5	8
5	0.987	0.03	0.08	1.0	-8	5
6	1.000	0.04	0.08	1.1	-12	0

TABLE II
Inspection Allocation Policy for P1

p_1	Inspection Policy
0.40	110000
0.60	101000
0.70	101000
0.80	100000
0.90	100000
0.95	100000

Here, $T' = -T + C$ and C is a large constant such that T' remains positive on its domain. Throughout our implementation, we use the Elitist expected value strategy for implementation of the reproduction operator of the GA. The crossover and mutation rates for the GA are taken to be 0.70 and 0.015 respectively.

First, we consider the single inspection case where it is possible to inspect at the most once at a stage. Thus x_i can take value either 0 or 1. In this case, each solution in the GA is represented by a $(N+1)$ bit string $x_1, \ldots, x_{N+1}$. For example, a solution string 0101 means to place inspection stations after the first and the third manufacturing stage in a three stage system (fourth stage acts as a dummy stage). As the constraint in problem P1′ is automatically taken into account in the structure of a solution string of GA, this problem is converted to the following unconstrained problem

$$\text{P1}^u: \ \max \ T'(p_1, \bar{x}) \tag{36}$$

Cases 1 and of Section II-A come into this class of problems. To demonstrate characteristics of the solution, a five stage problem has been solved. The data for this problem is given in Table I. A sixth stage is added to this table which acts as a dummy stage. The values for d and v are assumed to be 50 and 125 respectively. The experimental results are presented for the case 1 of Section II-A. The inspection policies obtained at different values of the incoming probabilities are shown in Table II. Table III gives the performance statistics of GA on a set of five test problems.

As mentioned earlier, we have used the penalty methods for solving the constrained problems. The constrained problem (P2) in our case has inequality constraints. Whenever a candidate solution violates a constraint, the corresponding objective function is penalized by an amount related to a function of the constraint violation. In other words, a constrained problem is transformed to an unconstrained problem by associating a penalty with all constraint violations and the penalties

TABLE III
PERFORMANCE OF GA ON A TEST SET FOR P1

Problem No.	No. of Stages	Population Size	Number of Generations
1	6	4	9
2	10	6	72
3	16	8	186
4	20	8	568
5	30	10	1020

TABLE IV
INSPECTION ALLOCATION POLICY WITH CONSTRAINT ON $L_{\max}$

p_1	Inspection Policy
0.60	100000
0.70	100000
0.80	100000
0.90	100000

are included in the function evaluation. However, though the function evaluation is well defined, there is no accepted methodology for combining it with the penalty. Davis [15] discusses this problem listing disadvantages of using high, moderate or light penalties. In our implementation, we square the violation of constraint. For example in P2, whenever we find that $L > L_{\max}$, we penalize the objective function by an amount proportional to $(L - L_{\max})^2$, i.e. we evaluate the following

$$T' - P * (L - L_{\max})^2. \tag{37}$$

Here, P is a positive constant. Rest of the algorithm proceeds as usual. Same technique is used to deal with other constraints. We consider the earlier problem (Table I) again. The inspection allocation policies generated for the unconstrained problem P1 were given in Table II. Now, we restrict the number of maximum inspection stations that can be located to one (i.e. $L_{\max} = 1$). The new inspection policies are presented in Table IV.

The advantage of these stochastic search techniques is that the number of computation can be dramatically reduced compared to the exhaustive search while it yields quite good approximation to the optimal solution. This is clearly seen from Table III. We solve these problems with simulated annealing algorithm using similar decoding scheme. The results for the SA algorithm are similar to the GA. The parameters for the SA algorithm are $T_0 = 50, \alpha = 0.95$ and $L = 50$. We present the performance of these two algorithms in Table V. The GA performs better than the SA for small to medium size problems but the SA takes over for large problems.

V. CONCLUSION

We have considered an inspection allocation problem for a serial and a special nonserial multistage manufacturing system. The emphasis of this paper has been on the development of stochastic search techniques for solving the inspection allocation problem in manufacturing systems. We have designed simulated annealing and genetic algorithms for this purpose. The advantage of these techniques is that the number of computations can be dramatically reduced compared to the exhaustive search while yielding quite good approximation to the optimal solution. We have used exterior penalty methods for solving the constrained problem. Experimental results are presented on a set of test problems.

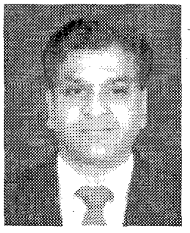
TABLE V
PERFORMANCE GA VERSUS SA

No. of Stages	CPU time (sec) (GA)	CPU time (sec) (SA)
5	1.667e-02	2.344e-01
10	3.499e-02	4.667e-01
15	1.098e-01	5.887e-01
20	9.106e-01	8.998e-01
25	1.650e+00	1.636e+00

REFERENCES

[1] D. A. Garvin, *Managing Quality*. New York: Free Press, 1988.
[2] J. M. Juran, *Quality Control Handbook*. New York: McGraw-Hill, 1951.
[3] G. D. Eppen and E. G. Hurst, "Optimal location of inspection stations in a multistage production process," *Manag. Sci.*, vol. 20, pp. 1194–1200, 1974.
[4] A. Garcia-Diaz, J. W. Foster, and M. Bonyuet, "Dynamic programming analysis of multistage inspection systems," *IIE Trans.*, vol. 16, pp. 115–125, June 1984.
[5] G. F. Linsay and A. B. Bishop, "Allocation of screening inspection effort–A dynamic programming approach," *Manag. Sci.*, vol. 10, pp. 342–352, 1967.
[6] B. Yum and E. D. McDowell, "Optimal inspection policies in a serial production system including scrap rework and repair: AN milp approach," *Int. J. Prod. Res.*, vol. 25, no. 10, pp. 1451–1464, 1987.
[7] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
[8] K. A. DeJong and W. M. Spears, "Using genetic algorithm to solve NP-Complete problems," in *Int. Conf. on Genetic Algorithm*, 1989.
[9] Z. Michalewicz and C. Z. Zanikow, "Handling constraints in genetic algorithms," in *Int. Conf. on Genetic Algorithm*, 1989.
[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 342–352, 1983.
[11] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, and A. H. Teller, "Equation of state calculations by fast computing machines," *J. Chemical Phys.*, vol. 21, no. 6, pp. 1087–1092, 1953.
[12] E. H. L. Aaarts and P. J. M. V. Laarhoven, "Simulated annealing: A pedestrian review of the theory and applications," in *Pattern Recognition Theory and Applications*, P. A. Devijer and J. Kittler, Eds., *NASI series on Computer and System Science*. Berlin: Springer-Verlag, 1987, vol. 30, pp. 179–192.
[13] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machine: A Stochastic Approach to Combinatorial Optimization with Neural Computing*. New York: Wiley, 1989.
[14] R. E. Collins and K. H. Stratmann, "Simulated annealing: An annotated bibliography," *Amer. J. Mathemat. Manag. Sci.*, vol. 8, no. 3, pp. 207–209, 1988.
[15] L. Davis, *Genetic Algorithms and Simulated Annealing*. London: Pitman, 1987.
[16] P. B. Chevalier and L. M. Wein, "Inspection for circuit board assembly," Preprint.
[17] A. V. Feigenbaum, *Total Quality Control*. New York: McGraw-Hill, 1956.
[18] D. S. Johnson, C. R. Aargon, L. A. McGeoch, and C. Schevon, "Optimization by simulated annealing: An experimental evaluation, Part-i, Graph Partitioning," *Oper. Res.*, vol. 37, pp. 865–892, 1989.
[19] M. H. Peters and W. W. Williams, "Location of quality inspection stations: An experimental assessment of five normative heuristics," *Decision Sci.*, vol. 15, pp. 389–408, 1984.

[20] T. Raz, "A survey of models for allocating inspection effort in multistage production systems," *J. Quality Contr.*, vol. 18, pp. 239–247, 1986.
[21] M. Taneja, S. M. Sharma, and N. Viswanadham, "Location of quality control stations in manufacturing systems: A simulated annealing approach," *System Practise*, vol. 7, no. 4, pp. 367–380, 1994.
[22] M. Taneja and N. Viswanadham, "Inspection allocation in manufacturing systems: A genetic algorithm approach," in *IEEE Conf. on Robotics and Automation*, 1994.

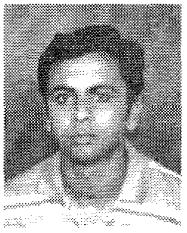


N. Viswanadham (SM'86–F'93) Ph.D. degree in 1970 from the Indian Institute of Science, Bangalore, India. He is now a TataChem Professor and chair of the Department of Computer Science and Automation, IISc. He has held visiting appointments at several North American universities. He was a GE Research Fellow at the corporate research center during 1989. His current research interests are in the areas of modeling, control and management of competitive manufacturing systems and software quality and reliability.

He is the author of several journal articles and conference papers. He is a joint author of two textbooks: *Reliability in Computer and Control Systems* (North-Holland, 1987) and *Performance Modeling of Automated Manufacturing Systems* (Prentice-Hall, 1992). He is co-editor of four other books. He currently is editor of *Sadhana: Academy Proceedings in Engineering Sciences*. He also is Associate Editor of the *Journal of Franklin Institute*, the *Journal of Manufacturing Systems*, the *International Journal on Information Technology, Intelligent and Robotic Systems*, and the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION. He was Associate Editor-at-Large for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, in 1990 and 1991.

He is a Fellow of the Indian National Science Academy, the Indian Academy of Sciences, and the Indian National Academy of Engineering.

Shashi M. Sharma is a Ph.D. student in the Department of Computer Science and Automation, Indian Institute of Science. He received the B.Tech. degree from the Institute of Technology, Benares Hindu University, and the M.E. degree from the Indian Institute of Science. His current interests are neural networks and combinatorial optimization.



Mukesh Taneja was born in Jaipur, India in 1968. He received the B.E. degree in electrical and electronics engineering from the Birla Institute of Technology and Science, Pilani, India, in 1989, and the M.E. degree in systems science and automation from the Indian Institute of Science, Bangalore, in 1993.

Currently, he is pursuing doctoral studies in the Department of Industrial Engineering and Operations Research, Columbia University, New York, NY. His research interests include stochastic processes, queuing theory and parallel computation. He has also worked as a Software Enginner at Tata Steel, Jamshedpur, India, and Motorola, Bangalore.