2002-01-01

Solving a multistage partial inspection problem using genetic algorithms

Heredia-Langner, Alejandro; Montgomery, D.C.; Carlyle, W M.

Taylor & Francis

Journal Name: International Journal of Production Research; Journal Volume: 40; Journal Issue: 8; Other Information: PBD: 1 Jan 2002

http://hdl.handle.net/10945/60989

INT. J. PROD. RES., 2002, VOL. 40, NO. 8, 1923–1940

Taylor & Francis
Taylor & Francis Group

Solving a multistage partial inspection problem using genetic algorithms

A. HEREDIA-LANGNER†, D. C. MONTGOMERY†* and W. M. CARLYLE†

Traditionally, the multistage inspection problem has been formulated as consisting of a decision schedule where some manufacturing stages receive full inspection and the rest none. Dynamic programming and heuristic methods (like local search) are the most commonly used solution techniques. A highly constrained multistage inspection problem is presented where all stages must receive partial rectifying inspection and it is solved using a real-valued genetic algorithm. This solution technique can handle multiple objectives and quality constraints effectively.

1. Introduction

In many types of manufacturing processes, a product usually undergoes a series of operations that progressively alter the nature of the incoming material until it reaches the consumer in the form of a finished good. Under normal operating conditions, each processing stage will produce a proportion of items that fail to meet the necessary requirements imposed by further manufacturing operations, the consumer, government regulations or some combination of these. Inspection of processed lots after every major manufacturing stage may be necessary to verify that a specific quality level is being maintained. This type of inspection is vital to minimize the cost of further processing-flawed materials and that incurred by customer dissatisfaction.

The inspection of goods to monitor the average quality level of a manufacturing system is subject to misclassification errors and a number of cost and quality constraints. These characteristics make this a complex problem that cannot be optimally solved in a reasonable amount of time except in very simple cases. Even when a solution method that does not guarantee optimality is employed, serious limitations exist in the type of problem that can be efficiently solved with any particular technique. An approach that may work very well for some version of the problem is not necessarily the best when a different set of conditions are imposed on it.

Genetic algorithms (GA) are well suited for solving problems of the type just described. These algorithms impose very small demands on the structure of the objective function and thrive in multimodal and even discontinuous environments. With the proper encoding, they can manipulate integer and continuous variables simultaneously and they handle linear and non-linear constraints trivially. In many areas, it is still assumed that a binary encoding of a potential solution is required for the use of a GA. In fact, any encoding that effectively matches the features of the

Revision received June 2001.

† Industrial Engineering Department, Arizona State University, Tempe, AZ 85287-5906, USA.

* To whom correspondence should be addressed. e-mail: doug.montgomery@asu.edu

International Journal of Production Research ISSN 0020–7543 print/ISSN 1366–588X online
http://www.tandf.co.uk/journals
DOI: 10.1080/00207540210123337

problem at hand can be used. Employing the actual value of the decision variables during a run has the advantage of avoiding the need to spend computing resources in the encoding-decoding process. Here, a potential inspection schedule is the encoding employed by the algorithm. The algorithm works with solutions that have immediate intuitive meaning and are relatively simple.

Another advantage of the GA is the possibility of obtaining good solutions even when radical alterations are performed on the statement of the problem. Our examples will show that a change in the number of constraints or the number of objective functions does not fundamentally alter the solution methodology or the behaviour of the algorithm.

2. Description of the process

A common inspection technique starts with a lot of size N that is representative of current manufacturing conditions. Then, a randomly selected sample of size n (usually much smaller than N) is screened to determine its quality level. This pre-specified quality level may measure performance under normal operating conditions, concentration of active ingredients, adequacy of dimensions, appearance or some other characteristic of interest. If the number of defective items in the sample, d, is less than or equal to a predetermined number, c, the lot is deemed acceptable and released to the next manufacturing step or to the final consumer. However, if the number of rejected units exceeds c, this is taken as evidence that an unacceptable proportion of non-conforming goods has been produced. *The entire* lot is then examined and all the non-conforming items are repaired or replaced with acceptable ones. This procedure is referred to as *rectifying inspection*.

For the purposes here, it will be assumed that N is very large (essentially infinite), n is small (in comparison with N) and the selection of inspected items is made at random. Under these conditions, the probability of accepting any given lot, that is, the probability that the inspected sample contains c or less defective items, is accurately found using a binomial distribution:

$$Pa = P(d \leq c) = \sum_{d=0}^{c} \frac{n!}{d!(n-d)!} \cdot p^d (1-p)^{n-d}. \tag{1}$$

where p is the proportion of defective items produced under current operating conditions, assumed to be known and constant. If it is not possible to guarantee the condition of relative size ($n \ll N$) the *hypergeometric* distribution can be used instead of the binomial. For practical purposes, the binomial approach can be used as long as $n < N/10$. The analysis in this paper is not affected by the choice of distribution although the examples presented were obtained employing the binomial function.

The parameters n and c are chosen so as to ensure that *on average*, the process maintains a nominal level of quality and that any individual lot with such quality level or better has a low probability of being rejected by the inspector. For more details regarding the appropriate choice of n and c, see Montgomery (2001, especially ch. 14).

The *average outgoing quality* for the process described is given by:

$$\text{AOQ} = \frac{Pa \cdot p \cdot (N-n)}{N} \tag{2}$$

and the *average total inspection* is:

$$\text{ATI} = n + (1 - Pa) \cdot (N - n). \tag{3}$$

It should be emphasized that these equations represent values that can be expected when the number of screened lots is large.

Quality policy for the process under consideration in this paper requires inspection of the product after every major manufacturing stage. If the process is operating under stable conditions, the quality of the product will be Q_i ($i = 1, 2, \ldots, m$, where m is the number of stages) or better after every stage.

The following assumptions are considered to hold for the analysis of the problem:

- Lot size, N_i, sample size n_i (selected at random from the lot) and the proportion of defective items currently produced on stage i have values that make it reasonable to employ the binomial distribution approach described earlier. This condition holds for all the stages of the process.
- The only link between manufacturing stages is the quality level of the product delivered from one to the next. Moreover, the nature and proportion of the defects is such that the number of non-conforming items can be algebraically added from stage to stage. For example, the proportion of non-conforming items out of stage 2 is equal to AOQ_1 (proportion of defectives out of stage one) plus the proportion of defectives produced in stage two alone. This indicates that the proportion of defective items throughout the process remains relatively small.
- The inspector can make two types of errors when examining the product: rejection of a good item (Type I) and acceptance of a non-conforming one (Type II). The rates of misspecification remain constant for every inspector although they need not be identical from stage to stage (for practical purposes these rates are conservative estimates of the variable error proportions likely to appear in any real process). The cost of inspection at every stage is directly proportional to the total number of screened items.
- The fractions of defective items produced at every stage, p_i^{o}, are known with reasonable accuracy and remain constant. These quantities are independent from each other and need not be identical. It is understood that the current proportion of defective items is acceptable (or unavoidable) and that the partial inspection strategy serves as protection against an unusually large amount of defective items that could have been produced by accident or by changes in processing conditions, suppliers or other process variations.

The problem under investigation is how to allocate n_i and c_i at every stage so that under rectifying inspection:

- the cost of inspection is minimized;
- the lot under inspection has a high probability of being accepted if the proportion of defectives in the current stage remains at its nominal value. Otherwise, partial inspection is of limited practical value since it would force the user to inspect too many lots. To use screening as a tool to achieve a specific quality goal, see the references discussed elsewhere in this paper;

- all other constraints (quality levels or inspection load size at specific stages, for example) are satisfied;
- it is understood that, realistically, not all possible combinations of ni and ci can be exhaustively analysed. This would require a computing effort that increases exponentially with the number of process stages.

It is also of interest to develop a solution procedure that can be employed even if changes in the values of the process parameters occur so that suitable modifications to the sampling procedure can be made when necessary.

3. Notation and statement of the problem

Before the problem can be formally stated, it is necessary to introduce the notation and formulae that will be used throughout the rest of the analysis.

Assumed to be known or estimated:

N_i lot size in stage i. For the examples presented in this paper, this parameter was held close to 28,800 for all stages,
a_i manufacturing cost per item in stage i,
x_i cost of repairing or replacing a defective item in stage i,
b_i inspection cost per item in stage i,
α_i Type I inspection error for stage i. This is the probability that a good item is labelled defective,
β_i Type II inspection error for stage i. This is the probability that a defective item is accepted as good, and
p_i^{o} proportion of defective items inherent to stage i.

The probability that an item will be rejected in stage i is: $R_i = p_i(1-\beta_i) + (1-p_i)\alpha_i$; the probability that an item is accepted is of course $A_i = 1 - R_i$.

Decision variables:

n_i sample size, the number of items to be inspected in stage i. The initial solutions for the examples presented in this paper were randomly generated using an integer, uniform distribution with values between 500 and 2500, and
c_i maximum number of defective items that can be accepted in a given sample, initially with values between 20 and 200 for all stages.

During the execution of the procedure, it is possible for the algorithm to expand the search beyond the ranges of the decision variables mentioned above.

Computation of quality constraints and the objective function:
The average outgoing quality out of stage 1 is:

$$\mathrm{AOQ}_1 = [n_1 p_1^{\mathrm{o}}\beta_1 + p_1^{\mathrm{o}}(N_1 - n_1)Pa_1 + p_1^{\mathrm{o}}(N_1 - n_1)(1 - Pa_1)\beta_1]/N_1(1 - R_1),$$

with

$$Pa_1 = \sum_{d=0}^{c1} \frac{n_1!}{d!(n_1 - d)!} \cdot R_1^d (1 - R_1)^{n_1 - d}.$$

For all other stages:

$$\mathrm{AOQ}_i = [n_i(p_i^{\circ} + \mathrm{AOQ}_{i-1})\beta_i + (p_i^{\circ} + \mathrm{AOQ}_{i-1})(N_i - n_i)Pa_i + (p_i^{\circ} + \mathrm{AOQ}_{i-1})(N_i - n_i)(1 - Pa_i)\beta_i]/N_i(1 - R_i)$$

with

$$Pa_i = \sum_{d=0}^{ci} \frac{n_1!}{d!(n_1 - d)!} \cdot R_1^d (1 - R_1)^{n_1 - d}.$$

$p_i = (p_i^{\circ} + \mathrm{AOQ}_{i-1})$, actual proportion of defective items out of stage i, with $\mathrm{AOQ}_0 = 0$.

The apparent proportion of defective items in stage i is:

$$P(\mathrm{def})_i = p_i - p_i\beta_i + (1 - p_i)\alpha_i.$$

The average total inspection is:

$$\mathrm{ATI}_i = [n_i + (1 - Pa_i)(N_i - n_i)]/[1 - (p_i^{\circ} + \mathrm{AOQ}_{i-1})(1 - \beta_i) - (1 - p_i^{\circ} - \mathrm{AOQ}_{i-1})\alpha_i].$$

Statement of the problem:

$$\min C = \sum_{i=1}^{m} \frac{(a_i + b_i + x_i)}{1 - p_i \cdot (1 - \beta_i) - (1 - p_i)\alpha_i} \cdot [n_i + (1 - Pa_i) \cdot (N_i - n_i)]$$

subject to

$$\mathrm{AOQ}_i \le \mathrm{AOQ}_i^*$$
$$Pa_i \ge Pa_i^*$$
$$n_i > c_i$$
$$n_i, c_i \text{ integer}$$

where AOQ_i^* and Pa_i^* are quality targets for each stage in the process.

4. Some solutions to the inspection problem

The problem of screening as a quality tool, in different modalities, has been dealt with by several authors. Tang and Tang (1994) review a number of screening procedures based on economic or quality objectives and amount of information or resources available. The techniques described to solve the problem include heuristic methods, dynamic programming approaches and all or nothing screening based on economic criteria. The problem they analyse is not the same as the one presented in this paper since they are not trying to monitor the quality level of a process but to achieve a target proportion of defectives by replacing non-conforming items with good product.

Barad (1990) describes a break-even approach to inspection in a multistage production process where screening is allocated only at some stages depending on economic criteria. When screening does take place, 100% of the product processed at that stage is inspected. One of the variables used to decide whether to inspect is the quality level at some point in the manufacturing process. Barad suggests allocating most of the inspecting resources to stages with a relatively high proportion of non-conforming product. Partial inspection could be used to identify these stages.

Lee and Unnikrishnan (1998) account for the possibility of inspection errors in a multistage manufacturing system and then tackle the problem of allocating a fixed number of inspections subject to time constraints and capability of screening. Again, this is a methodology that tries to allocate inspection in order to achieve specific quality goals and not as a monitoring tool to detect changes in production performance. In an interesting remark, the authors explain why the dynamic programming approach employed in other methodologies becomes quite impractical as the set of possible combinations grows exponentially and opt instead for three heuristic approaches: local search, dynamic programming subject to time constraints and an economic-probabilistic model.

Liou *et al.* (1994) solve the problem of optimally placing sequential partial inspections subject to errors when a pool of inspectors are available to screen the *same items* over and over and their rates of errors are unequal but known. Also known are the fractions of items to be inspected at every screening station. Under their assumptions, a closed solution to the problem of how to achieve a target outgoing quality level is developed.

Raz and Thomas (1983) developed a realistic cost function for the problem of partial inspection subject to errors. However, their definition of partial inspection is somewhat different from the one employed in this paper: in their case, an inspector selects a fraction f from the lot to screen and *accepts the rest of the lot*. Their objective is to determine the optimal number of inspectors (from a theoretically large pool) to repeatedly examine different fractions of the lot so as to meet a specific quality level at minimum cost. A numerical example is solved using branch and bound techniques.

These papers illuminate certain aspects of the multistage inspection problem the most important of which are perhaps the flexibility, variety and complexity of options available in screening procedures: the answers obtained depend entirely on how the problem is presented. In the following sections, we will develop a technique for the optimization of the multistage inspection problem subject to errors described in this paper using GA.

5. Genetic algorithms for function optimization

Describing more than a single technique, GA is a term used whenever the concept of progressive evolution by a large population of diverse solutions is applied to solve a problem. The main premise behind them is the idea that by *combining* different pieces of information relevant to the problem, new and better solutions can be made to appear. The algorithm accumulates information throughout a run and uses it to create new solutions. These are refined and used again until some convergence criterion is met. GA were first analysed by Holland (1975) and have been used, with varying degrees of success, in a number of problems from computer science (Rasmussen and Barrett 1995, Scott *et al.* 1999, Alander 1999) manufacturing engineering (Koza 1993, Ansari and Hon 1997) and scheduling (Falkenauer 1998, Vainio *et al.* 1995, Bierwith and Malfeld 1999) among other areas.

GA maintain a number of potential solutions (a *generation*) throughout the course of a run. Members of a generation (the *chromosomes* or *individuals*) are combined and altered through mechanisms resembling those of the classic natural selection theory first proposed by Darwin. GA start with a relatively large population that is representative of the space where an optimal solution may be found. This is generally done by randomly generating a fixed number of individuals. The algorithm then applies several procedures to the population in an attempt to improve their *fitness*. Fitness can be measured as the value of an appropriate objective function that represents, usually — but not necessarily — with a single real number, whatever combination of properties are required of a good solution. Although a number of different techniques are available, the evolving mechanisms are, in general, as follows.

- Recombination: information from two (or more) individuals is exchanged with the purpose of constructing a better solution.
- Mutation: small, random changes are applied to a few individuals in the population to overcome some of the limitations of a restricted grid search and to learn about potentially good (or bad) directions of expansion.
- Selection: promising individuals (or their offspring) are kept in the population at the expense of others that are perceived to be ill equipped for the search.

Using these steps, the initial population moves around its environment trying to find solutions with above average fitness. An initial period of rapid improvements is usually followed by gradual convergence to one or a small set of solutions. In the following sections, the procedure outlined above is explained in more detail as it is applied to the multistage partial inspection problem.

6. Solution of the multistage inspection problem through genetic algorithms

The GA used in this paper works with a multidimensional decision vector and seeks to optimize one objective function. Since the process contains multiple responses, a desirability function — involving all the required objectives — was created for each manufacturing scenario considered.

The desirability function approach allows the practitioner to include multiple responses in the optimization procedure. Each of the responses involved in the optimization is assigned a desirability function whose values vary between zero (for unacceptable response values) and one (response attains or exceeds target value). The objective function is then formed as the geometric mean of all individual desirability values. For a more detailed explanation regarding the construction and properties of the desirability function approach for multiple response optimization, see Derringer and Suich (1980) and Myers and Montgomery (1995). For an excellent and very complete survey of multiobjective optimization techniques using GA, see Coello Coello (2000).

An encoded individual (or *chromosome*) in our problem is a twelve dimensional, integer-valued vector whose first two entries are the number of inspected items and the acceptance number at stage one respectively. These are followed by the corresponding entries in stages 2–6:

2500	85	1780	103	1200	95	1900	98	2500	168	500	40
n_1	c_1	n_2	c_2	n_3	c_3	n_4	c_4	n_5	c_5	n_6	c_6

We will refer to any entry in this vector as a *gene*.

It should be mentioned that this encoding differs from the traditional approach commonly used in other GA (the original GA theory called for the encoding of a chromosome as a binary-valued string). In the encoding we use, all entries represent actual values for the decision variables and there is no need to spend computing effort encoding and decoding solutions throughout a run. Besides the benefits obtained by simplifying computations, this type of encoding allows the user to easily understand the solution to the problem and to identify the effect that changes in evolutionary mechanisms will have on it.

GA are based on the premise that most chromosomes contain at least some useful information and that, by sharing it, an improved solution can be created. For this reason, the initial population should be as large and diverse as reasonably possible to provide the recombination step with a large number of building blocks. In the literature, and for relatively simple theoretical objective functions, initial populations of sizes between 20 and 50 individuals are common (Wiley 1999, Falkenauer 1998). This number should be used with caution. Successful use of GA depends on a number of interconnected factors and the size of the initial population is key to equip the algorithm with enough working material. In this paper, initial populations of 20 or 50 randomly generated inspecting schedules were deemed appropriate.

6.1. *Recombination*

The recombination operator is the most important element in a successful GA. It enables the exchange of information obtained by the individuals and its transmission to the next generation.

The *crossover* mechanism is the most commonly used form of recombination. In its simplest form (and the one applied in this paper), two distinct chromosomes are selected at random from the parent population and broken at the same position (also chosen at random). The offspring individual (different from either parent) is obtained by combining the first portion of one parent with the last portion of the second. This operation is performed until a relatively large offspring population has been created. In our case, and with a pool of either 20 or 50 parent chromosomes, we create 100 or 250 offspring individuals respectively.

Parent 1.	2500	85	1780	103	1200	95	1900	98	2500	168	500	40
Parent 2.	1850	53	2200	99	785	67	1000	102	750	75	900	69
Offspring	2500	85	1780	103	1200	95	1900	98	750	75	900	69

This is called *one point discrete* crossover and was used to find a solution to our problem.

6.2. *Mutation*

Mutation is used to alter the genetic material of a relatively small number of individuals in a random fashion, enhancing the diversity of the population and expanding the volume of the current search space. As with recombination, several mutation procedures have been proposed in the literature. The one used in this paper is presented next.

Gaussian mutation ensures that the search space does not remain limited to that defined by the parent population. In this procedure, a number of parent individuals are selected and small changes are performed in all of their genes so that the resulting chromosomes are located somewhere within a small neighbourhood of their parents. The alterations performed to the genes are normally distributed with mean zero and standard deviation (SD) σ_i where the subscript indicates a particular gene in the chromosome. In this way, every gene could have an independent distribution and the user can define the appropriate magnitude of the change for every variable involved. This type of mutation places the offspring in the hyperellipsoids of constant probability defined by the multivariate normal distribution used, allowing the individuals to move and take steps in directions that may maximize improvement.

Bäck (1996) has suggested a log-normal approach for altering the standard deviations of this mutation mechanism to ensure that they remain positive during the course of a run. This method produces a mutation vector in one of the following ways:

$$\overleftrightarrow{\sum_i} = N_i(0,1) * \sigma_i \exp[\tau \cdot N(0,1) + T \cdot N_i(0,1)]$$

$$\nu_i = \nu_i + \sum_i$$

$$i = 1, 2, \ldots, n,$$

where the subscript in the normally distributed variables indicates that they must be sampled independently for each gene. In our case—and since all our decision variables have integer restrictions—the entries in the mutation vector are rounded to the nearest integer value before being added to the chosen chromosome.

Each gene of the parent individual, ν_i, selected for mutation becomes:

$$\overleftrightarrow{\sum_i} = \sigma_i \exp[\tau \cdot N(0,1) + T \cdot N_i(0,1)].$$

The values selected for τ and T (along with σ_i) will determine the average size of the mutation step taken by a parent individual. It has been suggested (Bäck 1996) that the mutation variables are included in the chromosome of decision variables so that the algorithm may evolve fixed parameters along the way. The idea behind this proposition is that there is a connection between the ‘correct’ mutation parameters and fitness of the individuals and that, by letting the algorithm arrive to its own values, the optimum can be found. In this paper, however, the mutation parameters are not evolved with the rest of the chromosome. In every generation, we subject to mutation some 10–12% of the parent chromosomes.

6.3. *Selection*

This mechanism is the operation that transforms the volume-oriented search performed by the steps described so far into a path-oriented exploitation of promising regions. It does this by imposing a minimum degree of fitness in every succeeding generation. After the offspring individuals have been created using the mechanisms described previously, a number of them are selected to form the next parent generation using a ranking procedure.

Ranking selection orders the current population according to their objective function value and selects from this list, in descending order, until the desired number of parent individuals is reached.

Two versions of this selection method are (μ, λ) and $(\mu + \lambda)$ where μ and λ represent the parent and offspring populations respectively. The mechanism used in this paper, (μ, λ)-selection, uses ranking selection excluding the parent population from the process so that every parent individual survives for exactly one generation. This minimizes the probability that the algorithm will become trapped in a local optimum. Convergence using this version of the selection procedure is very fast and a sufficiently large mutation rate is necessary to prevent a premature narrowing of the search space.

Repeating the procedures of recombination, selection and mutation result in convergence to a solution after a few generations. For the examples we have analysed here, little was achieved by running the algorithm beyond 10 generations. This means that, after the 10th generation, the improvement of the best solution from one generation to the next was $<1\%$. Each generation performs 100 evaluations of the objective function so we were able to find an answer using 1000 objective function evaluations. Although this number was used for the comparisons in our examples, it does not represent a fixed goal. The number of objective function evaluations necessary to obtain a good solution will vary depending on the choice of particular GA mechanisms employed and, of course, the difficulty of the problem.

The convergence criterion we have used may not necessarily hold for other types of problems and we employed it because we had evidence that the fitness value of the objective function would not be appreciably affected by letting the algorithm run for a very long time. The question of when a GA has converged is a problematic one to answer and the practitioner may choose from a variety of criteria which include: a predetermined number of generations without improvement, successive generations without appreciable improvement or, the most common, a fixed number of generations. Regardless of the criterion chosen, there is always a chance that a better solution lies just ahead but, as we will see, the GA is usually so reliable that this risk does not appear to be large.

The reader will have noticed the large number of parameters that make up the GA. Population sizes, mutation parameters and convergence criteria, among others. All must be selected before a run and, although the algorithm is fairly robust with respect to these choices, it is clear that optimal values for these parameters could be found. It is by now common to employ some combination of designed experiments and direct search to optimize the GA parameters and we have employed this approach for our algorithm. A full factorial experiment using parent population size, mutation rate and all three mutation parameters (σ, τ, T) was developed using final desirability value and speed of convergence for one of our examples as the responses of interest. Once the significant effects were identified, polynomial models of the responses as a function of the parameters were fitted and optimized. Although all the examples we present were solved using the optimized version of the algorithm, its speed and convergence properties are not greatly affected by suboptimal choices in the GA parameters.

7. Examples of the optimization of a multistage inspection problem

Consider the process with the parameters shown in table 1.

This process consists of six major consecutive manufacturing stages and its size, typical of many industrial operations, presents a challenging situation for the selection of a nearly optimal inspecting schedule that satisfies multiple requirements. The

Factors	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6
Lot size (N)	28 800	28 000	29 000	26 000	28 000	30 000
Sample size (n)	1500	922	1750	964	1921	1032
Prop. def. (p)	0.0100	0.0222	0.0320	0.0234	0.0331	0.0131
p^o	0.01	0.02	0.01	0.02	0.01	0.01
Type I error	0.01	0.01	0.02	0.01	0.01	0.01
Type II error	0.01	0.02	0.03	0.02	0.02	0.02
P(def)	0.0198	0.0315	0.0504	0.0327	0.0421	0.0227
c	25	94	75	184	68	168
Pa	0.2216	1.0000	0.0812	1.0000	0.0767	1.0000
AOQ	0.0022	0.0222	0.0035	0.0233	0.0231	0.0129
ATI	23 211.22	952.00	28 209.32	996.62	27 914.92	1055.96
Manufacturing cost	1.0	2.0	3.0	4.0	5.0	6.0
Inspection cost	0.25	0.3	0.4	0.5	0.6	0.7
Replacement cost	1.3	3.5	7.0	13.0	17.0	21.0
Stage cost	59 188.60	5521.60	293 376.93	17 440.77	630 877.11	29 250.12
Total cost	1 035 655.13					

c, Acceptable number of non-conforming items; Pa, Probability of accepting any particular lot. Obtained using the binomial approach; AOQ, average outgoing quality; level of quality obtained at every stage; ATI, average total inspection expected given the values of the variables; p^o, proportion of defective items particular to every stage; Prop. def. (p), actual proportion of defectives at every stage; P(def), apparent proportion of defectives given the actual amount and the inspection errors; stage cost is directly proportional to ATI; manufacturing, inspection and replacement costs are reported on a per item basis; total cost is the sum of the cost for all individual stages.

Table 1. Example of the relevant parameters for the inspection problem.

reported cost of every stage is directly proportional to the average number of items inspected in it.

The current schedule stresses inspection on Stages 1, 3 and 5 (with nearly all items being inspected at those stations) to keep the proportion of non-conforming items low. It is preferable to inspect more items at the selected stages than at the rest since costs are smaller but the error rates for the inspector in station 3 are higher than in other stages making this an unlikely choice for a large number of inspections. The AOQ remains very stable despite the addition of non-conforming items at every stage and the proportion of defective items reaching the consumer is fairly low. Moreover, since the effect of removing non-conforming items early in the process tends to have negligible impact on the final quality level, there seems to be no choice but to eliminate defectives in alternating stages. The GA was asked to find an alternative schedule that produced a proportion of defectives reaching the consumer no higher than 2.0% but with a lower cost. Results are presented in table 2.

The solution found — a good alternative to the original schedule — produces a final AOQ $< 1.7\%$ (compared with 1.3% in the original process) with savings of over $250,000. The inspection efforts have been shifted almost entirely to the fifth stage (since it is cheaper to inspect than the last stage) where most of the non-conforming items are removed.

It is important to know just how effective and efficient the GA is when trying to solve the example we have presented. The GA arrived at the final schedule after 1000 evaluations of the objective function and, for comparison purposes, a simulated algorithm (SA) was implemented to solve the same problem.

SA is another non-deterministic algorithm used to solve difficult optimization problems. In this technique, a single initial solution (typically generated at random)

Factors	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6
Lot size (N)	28 800	28 000	29 000	26 000	28 000	30 000
Sample size (n)	1041	997	1550	546	1686	377
Prop. def. (p)	0.0100	0.0296	0.0347	0.0540	0.0659	0.0165
p^{o}	0.01	0.02	0.01	0.02	0.01	0.01
Type I error	0.01	0.01	0.02	0.01	0.01	0.01
Type II error	0.01	0.02	0.03	0.02	0.02	0.02
P(def)	0.0198	0.0388	0.0529	0.0624	0.0739	0.0260
c	99	44	187	130	109	129
Pa	1.0000	0.8328	1.0000	1.0000	0.0785	1.0000
AOQ	0.0098	0.0249	0.0347	0.0564	0.0066	0.0167
ATI	1063.05	5874.15	1636.62	582.3362	27 799.96	387.07
Manufacturing cost	1.0	2.0	3.0	4.0	5.0	6.0
Inspection cost	0.25	0.3	0.4	0.5	0.6	0.7
Replacement cost	1.3	3.5	7.0	13.0	17.0	21.0
Stage cost	2710.77	34 070.09	17 020.88	10 190.77	650 879.21	10 721.76
Total cost	725 593.48					

Table 2. Results from the minimization of the proportion of defective items as a minimum cost obtained using a genetic algorithm.

is allowed to change in such a way that the resulting answer lies somewhere in the neighbourhood of the preceding one. The change is always accepted if it has produced an improvement in the value of the objective function and accepted with a probability that decreases with time, and the magnitude of the change, if it has not. The cornerstone of this procedure is the cooling schedule, the decreasing probability values that allow a solution to explore the feasible space and eventually settle near or at an optimum point. In the form implemented here, the changes to the current solution are made using the Gaussian mutation pattern we have described for the

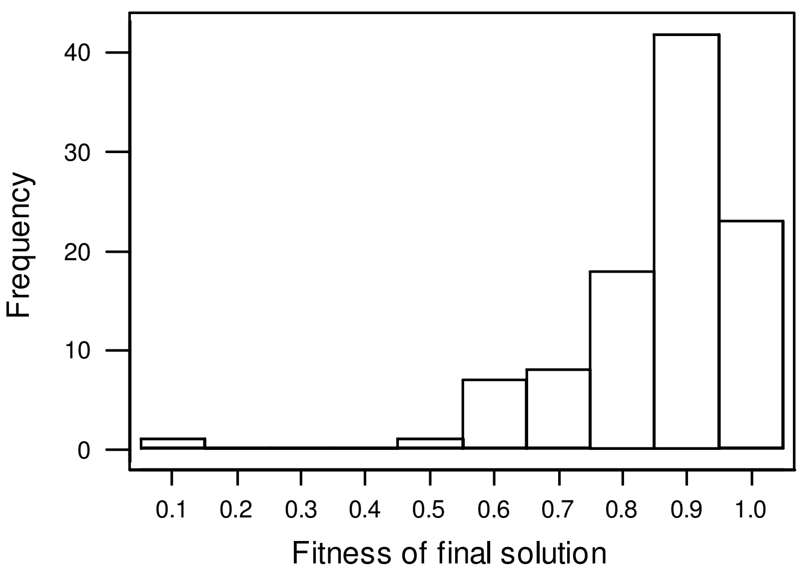

Figure 1. Frequency distribution of the fitness of 100 independent solutions to the first example presented using a SA.

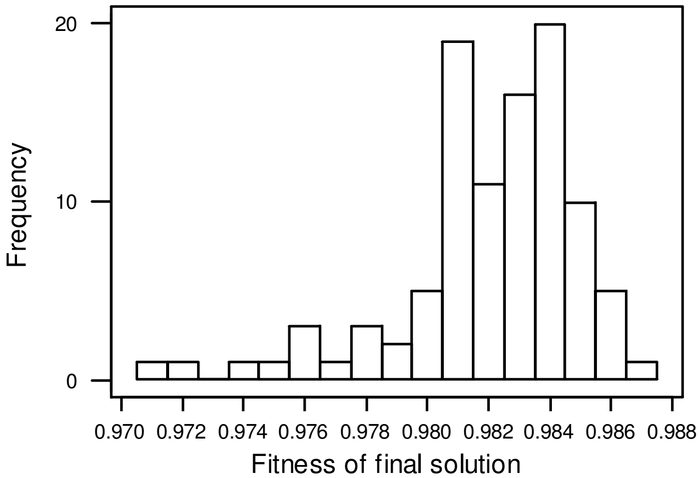

Figure 2. Distribution of the desirability of the final solution for 100 independently started runs of the first example using a GA. The distribution is much tighter than the one found using the SA. Since both techniques use the same number of function evaluations, this indicates a superior performance by the GA.

GA. We have run the SA multiple times always using a number of objective function evaluations comparable to that employed by the GA. The behaviour of the solutions found using 100 independently started runs of this methodology and the same number for the GA are presented in figures 1 and 2.

Although the SA is capable of producing answers of very good quality, their distribution is very far from the performance exhibited by the GA. Despite the presence of many random elements in the GA, from the generation of the initial population to the mutation of chromosomes, the fact that more than one solution is always under consideration allows this methodology to overcome the trap of local optimality. In practice, it is still a good habit to perform multiple runs whenever a solution technique cannot guarantee global optimality but with the GA, in this particular problem, this seems to be less pressing of an issue than with other algorithms.

Although the case just presented involves all 12 decision variables, there were only two responses involved in the optimization (final quality level and total cost). A more complicated scenario calls for a high probability of acceptance for the inspected lots ($Pa \geq 0.9$ in all stages), minimization of the proportion of non-conforming items reaching the final consumer at a minimum cost for the manufacturer. This problem has 8 responses to be optimized simultaneously. Results are shown in table 3. The high probability of acceptance translates into minimal inspection effort (and thus reduced costs) but it will also affect overall quality in a negative way so a compromise must be reached.

The solution found satisfies all the Pa requirements and calls for an increase in inspection efforts in the last three stages (where the removal of non-conforming items is more effective for final quality purposes). An almost constant outgoing quality level is maintained in all of them despite the fact that more non-conforming items are being introduced into the process. Notice how the tightened constraints imposed on the Pa values have dramatically affected the number of defective items out of the last

Factors	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6
Lot size (N)	28 800	28 000	29 000	26 000	28 000	30 000
Sample size (n)	998	1172	1974	2267	2007	2053
Prop. def. (p)	0.0100	0.0297	0.0368	0.0545	0.0576	0.0613
p^{o}	0.01	0.02	0.01	0.02	0.01	0.01
Type I error	0.01	0.01	0.02	0.01	0.01	0.01
Type II error	0.02	0.02	0.03	0.02	0.02	0.02
P(def)	0.0197	0.0388	0.0550	0.0629	0.0658	0.0695
c	86	54	127	157	146	157
Pa	1.0000	0.9121	0.9673	0.9007	0.9009	0.9002
AOQ	0.0099	0.0271	0.0352	0.0480	0.0518	0.0555
ATI	1018.06	3746.37	3022.96	4934.28	4991.94	5204.15
Manufacturing cost	1.0	2.0	3.0	4.0	5.0	6.0
Inspection cost	0.25	0.3	0.4	0.5	0.6	0.7
Replacement cost	1.3	3.5	7.0	13.0	17.0	21.0
Stage cost	2596.04	21 728.95	31 438.75	86 349.95	112 817.82	144 154.89
Total cost	399 086.40					

Table 3. Results from the simultaneous optimization of eight responses (all Pa, AOQ_6 and total cost) using a genetic algorithm.

stage indicating that it is not possible to simultaneously accept the majority of the inspected lots and maintain high quality levels. As we have seen in the previous example, the GA usually arrives at a solution with practically the same desirability value time after time. However, due to the way the objective function is constructed, the final solutions may actually involve an array of different values for each of the individual objectives included in the desirability function and different inspecting schedules. Far from being an obstacle, this fact may present an interesting set of choices for the practitioner.

Consider the solution in table 3. Five other inspection schedules with very similar desirability produce the values for the objectives shown in table 4.

Although the values for the objectives are fairly similar (particularly the total cost), these solutions have very different ways of allocating resources and the practitioner has now some flexibility in deciding how to distribute the inspection effort. Compare some of these results with those of table 3 and notice how a relatively small increase in total cost can bring the outgoing percentage of defective items in the process down by almost a full percentage point. This indicates that the quality levels are very sensitive to the amount of money available to carry out the task.

To complete the analysis of this particular example, the simulated annealing SA we implemented and the GA were run 50 times. Each of these independent runs was stopped after 1000 evaluations of the objective function with the best solution found throughout a run reported as the answer. The GA produces an average desirability value of 0.96 with a minimum of 0.95 and a maximum of 0.97. The average for the SA is 0.87 with a minimum of 0.03 and a maximum of 0.96.

Now suppose that we are interested in knowing if the $Pa \geq 0.9$ constraint is exerting too much influence in the final quality level and whether it would be possible to relax this restriction slightly in order to improve AOQ. The original Pa requirement can be modified so that *on average* the Pa at all stages is ≥ 0.9. Results from this new formulation of the problem are presented in table 5.

The GA has found a solution that maintains nearly all the Pa constraints and reduces the proportion of defectives reaching the final consumer by almost 1% at an extra cost of $100,000. Using it in this way, the algorithm can analyse alternative scenarios and assess sensitivity of current inspecting schedules. Notice how this last schedule compares to those of table 4. Since in table 5 we are forcing the algorithm to keep the *Pa* for every stage at a high level, the amount of money spent is much larger than that spent in all the cases in table 4 without making a large impact on final quality level. Notice that virtually the same final AOQ shown in table 5 can be obtained by allowing one fairly low Pa in one of the stages as shown in all instances in table 4.

In our comparison with the SA, we have verified that using the Gaussian mutation mechanism produces good solutions to the problem but without maintaining the

n	2177	896	1435	632	1464	2241
c	62	43	80	151	98	162
Pa	0.9979	0.9398	0.5404	1.0000	0.8892	0.9481
AOQ	0.0094	0.0277	0.0210	0.0416	0.0461	0.0523
Cost	5811.11	15 542.22	155 323.04	11 633.74	107 955.32	108 951.33
Total cost	405 216.77					
n	1469	1811	969	1332	1069	1878
c	55	74	108	78	134	142
Pa	1.0000	0.7159	1.0000	0.5043	1.0000	0.9999
AOQ	0.0097	0.0208	0.0311	0.0262	0.0363	0.0456
Cost	3821.52	57 186.69	10 597.74	252 188.744	25 294.84	55 066.91
Total cost	404 156.10					
n	1710	2156	1709	1942	2395	1162
c	40	90	101	118	139	102
Pa	0.8803	0.8644	0.9213	0.6814	0.9037	1.0000
AOQ	0.0085	0.0236	0.0307	0.0339	0.0383	0.0488
Cost	12 883.85	34 757.68	42 305.02	178 575.82	117 739.32	34 119.12
Total cost	403 379.80					
n	2294	2296	2257	2131	2096	1667
c	77	96	118	173	168	189
Pa	1.0000	0.8221	0.6108	1.0000	1.0000	1.0000
AOQ	0.0094	0.0231	0.0200	0.0383	0.0471	0.0573
Cost	5967.58	42 279.12	138 822.32	39 189.37	50 205.47	49 383.55
Total cost	325 847.41					
n	1921	1945	2170	1409	2008	1790
c	48	75	108	79	120	177
Pa	0.9557	0.5753	0.8715	0.8399	0.9001	1.0000
AOQ	0.0091	0.0164	0.0223	0.0352	0.0399	0.0494
Cost	8096.36	80 493.06	61 169.54	98 532.84	111 847.36	52 634.19
Total cost	412 773.34					

Table 4. Five different solutions to the problem of optimizing all *Pa*, the final AOQ and the total cost of the operation. The practitioner now has the option of selecting the one that best suits some particular arrangment of available resources.

Factors	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6
Lot size (N)	28 800	28 000	29 000	26 000	28 000	30 000
Sample size (n)	2077	2254	2409	2316	1155	2449
Prop. def. (p)	0.0100	0.0285	0.0344	0.0498	0.0531	0.0584
p^o	0.01	0.02	0.01	0.02	0.01	0.01
Type I error	0.01	0.01	0.02	0.01	0.01	0.01
Type II error	0.02	0.02	0.03	0.02	0.02	0.02
P(def)	0.0197	0.0376	0.0527	0.0583	0.0615	0.0667
c	49	96	141	149	81	174
Pa	0.9094	0.9010	0.9062	0.9001	0.8986	0.8176
AOQ	0.0086	0.0247	0.0304	0.0435	0.0489	0.0473
ATI	4589.02	5073.52	5176.90	4972.96	4218.95	8008.32
Manufacturing cost	1.0	2.0	3.0	4.0	5.0	6.0
Inspection cost	0.25	0.3	0.4	0.5	0.6	0.7
Replacement cost	1.3	3.5	7.0	13.0	17.0	21.0
Stage cost	11 702.00	29 426.41	53 839.80	87 026.85	95 348.36	221 830.55
Total cost	499 173.97					

Table 5. Results from the less constrained optimization scheme. Final AOQ has been reduced by almost 1% compared with that shown in table 3.

performance of the GA. As a final characterization, we have analysed the contribution of recombination and mutation toward the final solution of a problem. This analysis allows us to verify that it is indeed the right combination of factors that makes for a robust algorithm and that there is much to gain from the integration of the appropriate evolving mechanisms.

In figure 3, the random mutation curve shows the typical behaviour obtained when a uniformly random mutation mechanism is used (instead of the Gaussian technique employed in all of our examples). In this procedure, a selected number of

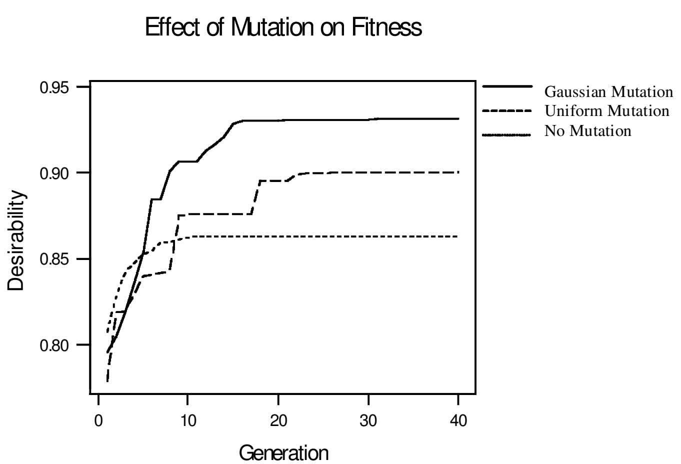

Figure 3. Typical effect of different mutation mechanisms on the progressive fitness of the population of solutions for an example of the inspection problem. In each case all other parameters remain equal. The curve marked 'No Mutation' shows the results that are commonly obtained using the recombination and selection mechanisms alone. For the mutation curves, up to 10% of the parent population (of 200 individuals in this particular case) was allowed to change in every generation.

genes in the chosen chromosome are replaced by uniformly random values obtained from the range where the decision variables are defined. It is apparent that Gaussian mutation, with its ability to look for a better solution in a close neighbourhood of the current population, is better suited to improve the fitness of the selected individuals.

It is interesting to notice that recombination and selection alone (represented by the curve marked 'No Mutation' in figure 3) tend to shrink the search space much too rapidly and convergence to a poor solution is quickly obtained. These two mechanisms, acting alone, have no resources to expand the space spanned by the initial parent population. In addition, as we have seen, mutation alone acting on an initial population would amount to not much more than a random search from which good results could be obtained only after a very long time.

8. Conclusions

The combined use of GA and the desirability function for the solution of the multistage partial inspection problem has allowed us to handle the peculiarities of the problem (integer-valued variables, exponential growth of the search space, multiple responses) very easily and efficiently. The GA searches the feasible space restricted only by the constraints imposed on the problem itself. Unlike some other solution methods, the GA does not need to impose artificial demands on the problem in order to decrease its size or complexity. This hands-off approach minimizes the number of decisions that should be made before a run.

The GA usually exhibits highly reliable behaviour, converging to a narrow range of good solutions starting with randomly generated initial populations, and this sets it apart from other non-deterministic algorithms. Although final solutions found in different runs may have nearly identical values for the objective function, the inspection schedules involved may be quite different. This could be the result of a highly multimodal space, the way in which the objectives are blended in the desirability function or both. Multiple optimal or nearly-optimal answers to a problem can present the practitioner with an opportunity to make a decision based on some other goodness criterion such as time of inspection, convenience of inspecting more heavily in a particular stage or some other measurement of quality.

Throughout this paper, we have analysed the effect that removal or modification of the procedures within a GA have on the solutions to our examples. Our results suggest that the right combination of search and expansion mechanisms — and not a single portion of the GA — is responsible for its good characteristics.

We have implemented our partial inspection examples and the GA using only an Excel spreadsheet coupled with Visual Basic procedures. This choice underscores the ease with which a GA may be used in actual industrial problems without the need to employ specialized software.

References

Alander, J. T., 1999, Population size, building blocks, fitness landscape and genetic algorithm search efficiency in combinatorial optimization: an empirical study. In L. D. Chambers (ed.), *Practical Handbook of Genetic Algorithms: Complex Coding Systems,* Vol. III (Boca Raton: CRC Press).

Ansari, N. and Hon, E., 1997, *Computational Intelligence for Optimization* (Norwell: Kluwer).

Bäck, T., 1996, *Evolutionary Algorithms in Theory and Practice* (New York: Oxford University Press).

BARAD, M., 1990, A break-even quality level approach to location of inspection stations in a multi-stage production process. *International Journal of Production Research,* **28,** 29–45.
BIERWITH, C. and MALFELD, D. C., 1999, Production scheduling and rescheduling with genetic algorithms. *Evolutionary Computation,* **7,** 1–17.
COELLO COELLO, C. A., 2000, An updated survey of GA-based multiobjective optimization techniques. *ACM Computing Surveys,* **32,** 109–143.
DERRINGER, G. and SUICH, R., 1980, Simultaneous optimization of several response variables. *Journal of Quality Technology,* **12,** 214–219.
FALKENAUER, E., 1998, *Genetic Algorithms and Grouping Problems* (Chichester: Wiley).
HOLLAND, J. H., 1975, *Adaptation in Natural and Artificial Systems* (Cambridge, MA: MIT Press).
KOZA, J. R., 1993, Hierarchical automatic function definition in genetic programming. In L. D. Whitley (ed.), *Foundations of Genetic Algorithms 2* (San Mateo: Morgan Kauffman)
LEE, J. and UNNIKRISHNAN, S., 1998, Planning quality inspection operations in multistage manufacturing systems with inspection errors. *International Journal of Production Research,* **36,** 141–155.
LIOU, M., TSENG, S. and LIN, T., 1994, Optimal sequence of partial inspection subject to errors. *International Journal of Production Economics,* **33,** 109–119.
MONTGOMERY, D. C., 2001, *Introduction to Statistical Quality Control,* 4th edn (New York: Wiley).
MYERS, R. H. and MONTGOMERY, D. C., 1995, *Response Surface Methodology* (New York: Wiley).
RASMUSSEN, S. and BARRETT, C. L., 1995, Elements of a theory of simulation. In *Proceedings of the European Conference on Artificial Life* (Berlin: Springer).
RAZ, T. and THOMAS, M. U., 1983, A method for sequencing inspection activities subject to errors. *IIE Transactions,* **15,** 12–18.
SCOTT, S. D., SETH, S. and SAMAL, A., 1999, A synthesizable VHDL coding of a genetic algorithm. In L. D. Chambers (ed.), *Practical Handbook of Genetic Algorithms: Complex Coding Systems,* Vol. III (Boca Raton: CRC Press).
TANG, K. and TANG, J., 1994, Design of screening procedures: a review. *Journal of Quality Technology,* **26,** 209–222.
VAINIO, M., SCHÖNBERG, T., HALME, A. and JAKUBIK, P., 1995, Optimizing the performance of a robot society in structured environments through genetic algorithms. In *Proceedings of the European Conference on Artificial Life* (Berlin: Springer).
WILEY, K., 1999, An evolutionary algorithm that solves the nonintuitive problem of black and white pixel distribution to produce tiled patterns that appear gray. In L. D. Chambers (ed.), *Practical Handbook of Genetic Algorithms: Complex Coding Systems,* Vol. III (Boca Raton: CRC Press).