Proceedings of the 1998 Winter Simulation Conference
D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, eds.

USE OF SIMULATION IN OPTIMIZATION OF MAINTENANCE POLICIES

Farhad Azadivar
Industrial and Manufacturing Systems Engineering
Kansas State University
Manhattan, KS 66506, U.S.A

J. Victor Shu
Talus Solutions Inc.,
650 Castro Street, Suite 300
Mountain View, CA 94041, U.S.A

ABSTRACT

Selecting an optimum maintenance policy independent of other parameters of the production system does not always yield the overall optimum operating conditions. For instance, high levels of in-process inventories affect the performance of a given maintenance policy by reducing the effects of machine breakdowns. In this study, parameters of the production system, in particular the allowable in-process buffers, and the design parameters of the maintenance plan are considered simultaneously as integral parts of the whole decision process for selection and implementation of a maintenance policy. The results from the simulation experiments show that the response surfaces for these systems are of the forms that yield themselves to an optimization search. However, the optimization problem itself is not trivial, as the performance of the system depends on a combination of qualitative and policy variables (the choice of the maintenance policy) as well as a set of quantitative variables (allowable buffer spaces). In this paper, a methodology is presented for solving this class of problems that is based on a combined computer simulation and optimization integrated with a genetic algorithm search.

1 INTRODUCTION

Allowing for build up of work-in-process (WIP) can often reduce the effects of machine breakdowns on a system's productivity. In most production systems, however, attempts are made to keep WIP at an absolute minimum. These conflicting effects give rise to a new way of thinking about selection and optimization of maintenance policies. The idea here is that in order to obtain an overall optimum operating condition the selection and optimization of maintenance policies should be considered simultaneously with deciding on the levels of allowable in-process inventories. A systematic optimization of such systems, however, is not simple. Decision variables for this optimization procedure consist of some qualitative factors (the type of maintenance policy) and some quantitative parameters (the size of allowable in-process inventories).

Earlier researches reported on maintenance planning usually studied push-type production systems. They are also more directed towards monitoring and control of the system under a given maintenance policy than specification and optimization of the policy to select. Examples of these systems appear in the work by Kobbacy (1992), and Ulusoy et al. (1992). There has also been work done on dynamic scheduling of maintenance jobs for optimal results, in particular policies by researchers such as Enscore and Burns (1983) and Wu et al. (1992). Bruggeman and Dierdonck (1985) suggested applying the Manufacturing Resource Planning (MRP II) concept to maintenance resource planning.

For JIT type systems, Abdulnour et al. (1995), using computer simulation and experimental design, developed some regression models to describe the effects of three preventive maintenance policies on performance of a production system. In an earlier work, Azadivar and Shu (1996) ranked maintenance policies in terms of their performance on JIT systems defined by certain characteristic factors.

In designing an overall optimum maintenance policy it is necessary to set optimum values for all decision parameters of the system. Unfortunately, most of the factors affecting the performance of maintenance policies are inherent to the system and cannot be used as decision variables. One of the few factors that plays a role in the performance and at the same time can be considered as a design parameter for all systems is the size of buffer spaces allowed. There are, however, other decision variables which are applicable only when a particular maintenance policy is selected. These are the frequency of maintenance when a preventive or predictive policy is selected.

In this work, a systematic method is proposed for the overall optimization of the system. The optimum determined consists of specifying the maintenance policy to employ along with its design parameters, if applicable, and the level of in-process inventories that will result in an overall optimum performance. A general algorithm has been developed based on the genetic algorithm approaches that yields a near-optimum solution for a combination of quantitative and qualitative variables.

2 OBJECTIVES

Five maintenance policies were investigated. These are: predictive maintenance policy, reactive policy, opportunistic policy, time-based preventive policy, and MTBF-based preventive policy.

There are many system performance measures that could be used as the objective of the optimization. In most production systems, where in-process inventory is kept at a minimum level, late delivery has a significant disruptive effect on the downstream processes. This makes on-time delivery one of the most important aspects of systems operations. In this study, a function of on-time delivery, the *service level*, has been selected as the measure of performance. The service level is defined as the percentage of jobs delivered on time.

3 RESPONSE SURFACE TOPOLOGY

Derivation of analytical forms for the response surface of performance as a function of the maintenance policy and other controllable variables is not possible; systems could get very complicated and some decision factors are qualitative in nature. Here, response surfaces were derived by evaluating the system at several points using computer simulation. The results were then used to depict them graphically.

Four examples, P1 through P4 were used in this study. These examples represent different problems with various levels of complexity and sizes containing between 12 to 60 controllable variables. The simplest and most complicated of these examples are represented in Figure 1 and 2. In these figures nodes represent the status of the product and

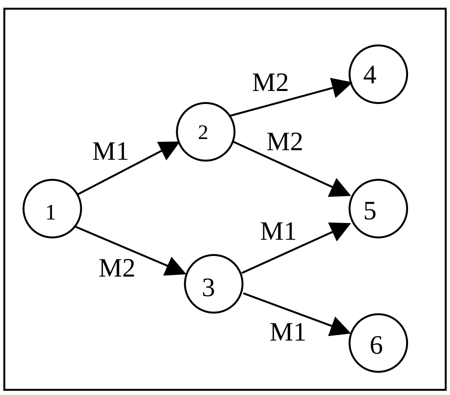

Figure 1: Graphic Representation of P1

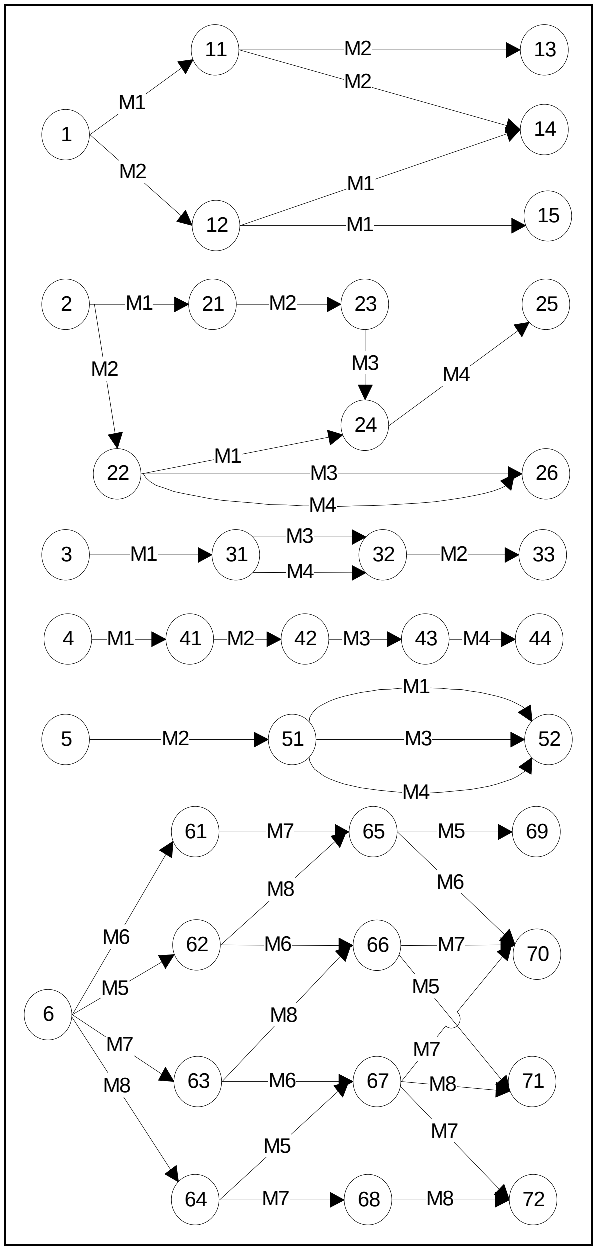

Figure 2: Graphic Representation of P4

arrows are the processes used to change the part from one status to the next. The letters on the arrows denote the resource used for each process represented by the corresponding arrow.

For each problem a set of points in the feasible region were chosen at which the performance measures of the system were evaluated. Each point represents a set of values for allowable buffer spaces in front of each station and a maintenance policy. The set of values for buffer spaces used in evaluating example P1 is given in Table 1. The results of performing the evaluations for all combinations of the sets of buffer spaces specified above and maintenance policies for this example is summarized in Table 2. A plot of these values in a 3-dimensional space is given in Figure 3. Similar results were obtained for the other three examples. The form of this graph, which indicates a local maximum at the predictive policy and buffer allocation No. 3, suggests that there is a potential for a systematic search to yield a desirable answer to this type of maintenance policy optimization problems.

Table 1: Buffer Placements Configurations for P1

	Buffer placement configurations					
Buffer capacity at nodes	1	2	3	4	5	6
2	1	2	2	2	1	0
3	1	1	0	2	2	2
4	2	2	3	1	1	0
5	1	1	2	1	1	2
6	2	1	0	1	2	3

4 FORMULATION OF THE PROBLEM

This problem has some special characteristics that distinguishes it from regular non-linear programming type optimization routines. It has both quantitative variables such as buffer allocations and qualitative variables such as the policy choices. It also has both explicit constraints such as the total available inventory storage spaces at all work stations and implicit constraints such as the lead time for delivering a product. A simulation optimization that can solve qualitative controllable variables must require only the responses from discrete points in the search space.

Existing methods that can solve that kind of problems are all direct search methods and often require complete enumeration. The existence of structural (qualitative) decision variables eliminates even these direct search methods. For these methods to be applicable, the variables must be quantitative so that the feasible space can be represented by a geometrical n-dimensional space. As a result, for these problems the

Table 2: Service Level as a Function of Buffer Allocations and Maintenance Policy

buffer allocation	policy				
	reactive	opportune	predictive	MTBF-PM	time-PM
No. 1	0.653	0.693	0.818	0.784	0.764
No. 2	0.724	0.754	0.876	0.837	0.827
No. 3	0.785	0.809	0.951	0.917 (0.003)	0.885
No. 4	0.654	0.680	0.801	0.768	0.759
No. 5	0.572	0.613	0.720	0.680	0.668
No. 6	0.422	0.425	0.479	0.462	0.462

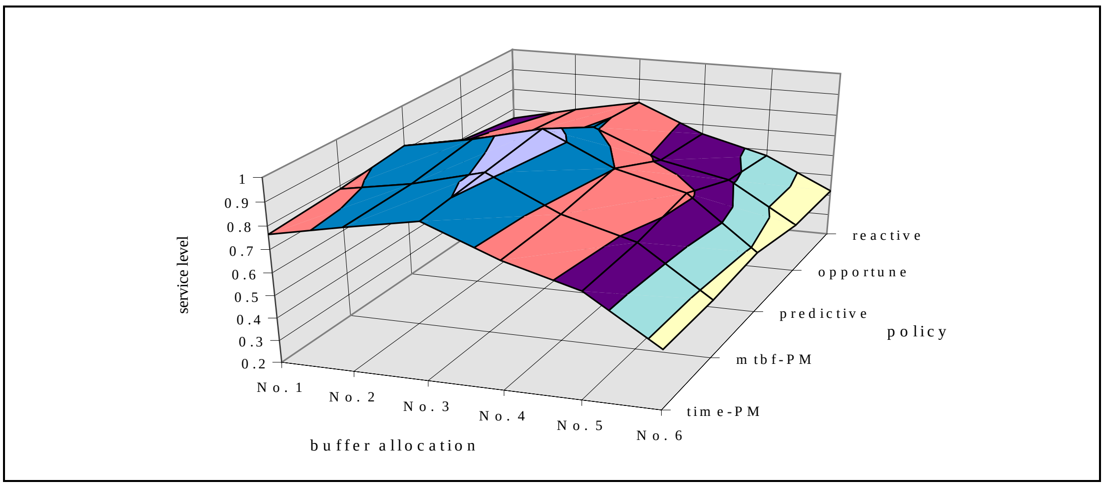

Figure 3: 3-D Representation of the Response Surface for Example P1

random search has often been used as the only feasible approach. In this work, a method based on a genetic algorithm has been developed and implemented and the results have been compared with those obtained from a random search.

The general form of the maintenance policy optimization formulation can be stated as follows,

Maximize:

$$E[f(X,Y)] = E[f(x_1, x_2, ..., x_n, y_1, y_2, ... y_m)]$$

Subject to:

$$h_j(x_1, x_2, ..., x_n) \leq a_j, \quad \text{for } j = 1, 2, .., p$$
$$g_k(X,Y) = \leq b_k \quad \text{for } k = 1, 2, ..., q$$
$$L_i \leq x_i \leq U_i, \quad \text{for } i = 1, 2, ..., n$$

where X: x_1, x_2, ..., x_n is the vector of quantitative controllable variables, which are deterministic in nature and are bounded by upper and lower limits U_i and L_i , Y: y_1, y_2, ..., y_m is the vector of the qualitative factors, f(X,Y) is the objective function, which is an implicit function of the controllable variables and can be evaluated only by simulation. $h_j(x_1, x_2, ..., x_n) \leq a_j$, for j = 1, 2, ..., p, denotes the explicit constraints, whose analytical forms are known and can be evaluated analytically. a_j is the limit for the j-th explicit constraint. $g_k(X,Y)$, for k = 1, 2, ..., q, denote implicit constraints. b_k is the pre-set goal for the k-th implicit constraint.

Note that the explicit constraints, like the objective functions, are also responses of the simulation model. For instance, one such constraint may require the average lead time not to exceed a certain value. These constraints are stochastic. Therefore, they may be obeyed in one replication of the simulation, yet violated in another. Thus they need to be treated stochastically. The way they are treated in this study is that it is assumed that they will always have a chance to be violated, but a probability limit is assigned to this violation according to the comfort level of the decision maker. Assuming this comfort level to be α_k, the k-th implicit constraint can be written as:

$$P[g_k(X,Y) < b_k] \geq 1 - \alpha_{Ik}.$$

In simulation experiments the form of these constraints can be transformed into

$$UCL_{1-\alpha k} g_k(X,Y) \leq b_k$$

where $UCL_{1-\alpha k}$ denotes the upper confidence limit for the response g_k at 1 - α_k level.

4.1 Definitions

X = Vector of quantitative variables consisting of vectors $\mathbf{X_1}$ and $\mathbf{X_2}$

Y = Vector of qualitative variables or policy choices consisting of y_1 and y_2

$\mathbf{X_1}$ = Vector of buffer sizes

$\mathbf{X_2}$ = Vector of preventive maintenance frequencies

y_1 = Variable representing the maintenance policy

y_2 = Variable representing maintenance task priority

SL = The random variable representing the response for the service level

LT_s = The random variable representing the response for lead time for part s

LTD_s = The desired value for LT_s

IL = The random variable representing the total inventory level in the system

ILD = The desired level for IL

$\mathbf{x}_{1i}$ = the buffer size at the i-th machine station,

s_i = the storage limit at the i-th machine station,

SS = Upper bound on the total number of storage spaces.

Then, the optimization problem can be formulated as

$$\text{Max} \quad E[SL(\mathbf{X_1}, \mathbf{X_2}, \mathbf{Y_1}, \mathbf{Y_2})]$$

$$\text{S.T.} \quad P[LT(\mathbf{X_1}, \mathbf{X_2}, \mathbf{Y_1}, \mathbf{Y_2}) < LTD_s] \geq 0.95$$
$$P[IL(\mathbf{X_1}, \mathbf{X_2}, \mathbf{Y_1}, \mathbf{Y_2}) < ILD] \geq 0.95$$
$$\sum_i x_{1i} \leq SS, \quad \text{for } k = 1, 2, ..., p$$

$$0 < x_{1I} < s_i$$

4.2 Applying Optimization Procedures

The search methods were implemented and compared on four examples. These examples denoted by P1 through P4 represent different problem sizes containing between 12 to 60 controllable variables. Response surfaces were generated as functions of maintenance policy, maintenance task priority, preventive maintenance frequency, and buffer placements.

During the search process, explicit constraints are checked first because they are in analytic forms, thus easy to check. Implicit constraints are checked after simulation runs are made. The result obtained from applying each optimization technique is accepted only if both explicit and implicit constraints are satisfied.

In the constrained random search, a finite number of randomly selected points in the feasible space are checked and evaluated. A point is generated by a pre-determined scheme and checked against all explicit constraints. If none is violated, the simulation runs are performed and then all implicit constraints are checked. If no implicit constraint is violated, the value of the response is compared with the best existing solution and is retained as the best if it is better, but is discarded otherwise. The process continues with the next point in the feasible region until all points are evaluated and compared. The best result is considered the optimal solution to the problem.

Genetic algorithms (GA) are a set of search methods that mimic the process of biological evolution. In genetic algorithms, controllable variables are usually encoded in fixed-length strings the same way as genetic information is encoded in chromosomes in the biological world. A population of these strings, which each represents a solution to the problem at hand, is created randomly and then transformed according to some probabilistic transition rules. Transition rules usually include selection, crossover and mutation. The selection process generates parents from the current population to reproduce offspring in the next generation. Crossover is used to combine a partial string from one parent string with a partial string at the same location from another parent string to form a whole new string. Mutation, which introduces diversity to the population, is used to change the value of the variable at one position in the string according to a random decision suggested by a probability distribution.

These transformation processes generate new string patterns that do not exist in their parents. These new patterns may represent better results for the system. A better result in the genetic algorithm terminology is referred to as a better fitness value. In an environment where the right to reproduce is determined by individual fitness and luck, better fitness provides better chances to survive and reproduce and over time, the average fitness of the population improves. For further information on genetic algorithms used in optimization, refer to textbooks such as those of Davis(1991), Goldberg(1989), and Chambers(1995).

The results of performance of different search methods were compared in terms of the value of the objective function and the number of simulation runs to obtain the solution. In performing the comparison the attempt was made to compare the results obtained for the same number of simulation runs spent. This was possible by using as many number of runs that took for GA to find the solution be spent on random search.

Table 3 through 6 show these comparisons for service levels obtained by random search and GA for a given number of simulation runs for problems P1 through P4. The general trends in these figures indicate that GA performs relatively better than the random search and its superiority increases as the problem size increases.

In these experiments a simple GA without many available enhancements was employed. Additional experimentation with various values for GA parameters was conducted. The results indicated a significant improvement over the simple GA. For instance, when a binary representation, rank scaling, generational stability stopping rule, elitist roulette wheel selection, 1-point crossover, and bit swap mutation were used to achieve

Table 3: Comparison for P1

number of evaluations	random search	genetic algorithm
111	0.713	0.713
130	0.713	0.720
292	0.720	0.738
1802	0.720	0.751

Table 4: Comparison for P2

number of evaluations	random search	genetic algorithm
593	0.484	0.574
598	0.484	0.523
608	0.511	0.579
1901	0.511	0.588
2183	0.531	0.657
2504	0.531	0.634
2973	0.603	0.697
4846	0.603	0.682

Table 5: Comparison for constrained P3

number of evaluations	random search	genetic algorithm
218	0.321	0.492
332	0.321	0.512
568	0.357	0.491
739	0.357	0.510
1049	0.363	0.528
1070	0.403	0.571
1432	0.403	0.535
1569	0.405	0.535
1756	0.405	0.519
1831	0.418	0.577
2508	0.418	0.533

the same value or even better values for the service level, the number of simulation runs required decreased drastically. These results for P2 are shown in Table 7. The numbers in parenthesis show the number of simulation runs used to obtain the indicated service level.

Table 6: Comparison for P4

number of evaluations	random search	genetic algorithm
117	0.861	0.883
473	0.861	0.897
660	0.875	0.893
2634	0.875	0.919
3241	0.876	0.913
4381	0.876	0.925

Table 7: Comparison of Basic and Enhanced GA for P2

	crossover = 0.4		crossover = 0.5	
Pop. size	**Basic GA**	**Enhanced GA**	**Basic GA**	**Enhanced GA**
60	0.907 (397)	0.918 (271)	0.918 (607)	0.911 (334)
90	0.915 (726)	0.924 (382)	0.918 (1120)	0.917 (435)
120	0.925 (1218)	0.925 (616)	0.919 (1381)	0.920 (507)
150	0.923 (1558)	0.915 (647)	0.918 (2748)	0.919 (916)
180	0.933 (2306)	0.931 (798)	0.938 (2816)	0.938 (956)

5 CONCLUSIONS

Optimum maintenance policy for a manufacturing system has to consider all of the factors that have influence on the consequences of machine breakdowns. A major factor in these systems that has such an effect is the level of WIP at each workstation, as low WIP leaves little tolerance for machine failures. In order to make an overall best decision on the type of maintenance policy and the other characteristics of the system, in this paper the problem was formulated as an optimization problem consisting of quantitative decision variables as well as qualitative and policy variables. A simulation-optimization procedure based on genetic algorithms was developed and applied to four problems ranging from a very simple to a very complex system. A procedure was developed to automatically build and execute simulation models for configurations suggested by the optimization algorithm. The results obtained were then compared with those obtained from a random search.

The results obtained demonstrated that the proposed formulation could indeed provide acceptable solutions for this complex problem. In particular, the genetic algorithm based optimization routine demonstrated a great flexibility in solving problems that are defined by a set of combined quantitative and qualitative variables.

ACKNOWLEDGEMENT

This project was sponsored by a grant from the Advanced Manufacturing Institute of Kansas State University.

REFERENCES

Abdulnour, G., Dudek, R.A., and Smith, M.L. 1995. Effect of maintenance policies on the just-in-time production system. *International Journal of Production Research,* 33: 565-583.

Azadivar, F. and Shu, J. 1996. Maintenance policies for JIT manufacturing environments. In *Proceedings of 1996 International Manufacturing Engineering Conference*, ed. Ioan D Marinescu, 123-125.

Bruggeman, W. and Van Dierdonck, R. 1985. Maintenance resource planning - an integrative approach. *Engineering Costs and Production Economics.* 9:147-154.

Chambers, L. 1995. *Practical Handbook of Genetic Algorithms*. CRC Press.

Davis, L. 1991. *Handbook of Genetic Algorithms*, Van Nostrand Reinhold.

Enscore, E.E. and Burns, D.L. 1983. Dynamic scheduling of a preventive maintenance programme. *International Journal of Production Research.* 21:357-368

Goldberg, D.E. 1989. *Genetic Algorithms In Search, Optimization and Machine Learning*. Addison Wesley.

Kobbacy, K.A.H. 1992. The use of knowledge-based systems in evaluation and enhancement of maintenance routines. *International Journal of Production Economics.* 24: 243-248

Ulusoy, G., Or, I., and Soydan, N. 1992. Design and implementation of a maintenance planning and control system. *International Journal of Production Economics*. 24:263-272

Wu, B., Seddon, J.J.M. and Currie, W.L. 1992. Computer-aided dynamic preventive maintenance within the manufacturing environment. *International Journal of Production Research*. 30:2683-2696

AUTHOR BIOGRAPHIES

FARHAD AZADIVAR is professor of Industrial and Manufacturing Systems Engineering at Kansas State University. He is also the director of the Advanced Manufacturing Institute, a research and technology transfer Center of Excellence in Manufacturing for the State of Kansas. His areas of expertise are in modeling and optimization of manufacturing systems, simulation. optimization and management of technological innovation. He holds a Ph.D. in Industrial Engineering from Purdue University.

J. VICTOR SHU is a project manager at Talus Solutions Incorporated. His research interests include optimization, computer simulation and maintenance management. He holds a Ph.D. in Industrial Engineering from Kansas State University.