Proceedings of the 1995 Winter Simulation Conference
ed. C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman

STOCHASTIC OPTIMIZATION APPLIED TO A MANUFACTURING SYSTEM OPERATION PROBLEM

Robert W. Brennan
Paul Rogers

Department of Mechanical Engineering
University of Calgary
2500 University Drive NW
Calgary, Alberta T2N 1N4, CANADA

ABSTRACT

This paper deals with stochastic optimization of a discrete-event simulation model for the solution of a manufacturing system operation problem. Gradient estimates are obtained by the application of the infinitesimal perturbation analysis (IPA) technique. We begin with background material on stochastic approximation (SA) and the IPA technique, their potential value in finding optimal solutions to manufacturing system operation problems, and limitations concerning their applicability. Next we present our attempt to solve a real problem (the design of a partially-automated assembly line in an electronics manufacturing facility) using this approach. A sequence of models is described moving from one which embodies some restrictive assumptions through to models which more closely approximate the real system. All of the models are implemented in the SIMAN IV simulation language incorporating user-written code (written in C++) implementing the SA and IPA algorithms. We report and interpret the results obtained with the different models and close with concluding remarks on the current value of this technique in solving this kind of system design problem.

1 INTRODUCTION

This study concerns optimizing the performance of an asynchronous line used for component assembly in electronics manufacturing. As is typically the case with serial systems, the objective here is the minimization of station idle time and the maximization of throughput (Askin and Standridge 1993; Buzacott and Shanthikumar 1993). This is equivalent to minimizing the proportion of time that the bottleneck station spends in either the "blocked" or "starved" states.

The fixed layout of this assembly line as well as a mixture of both manual and automatic assembly stations combine to make line balancing impractical in this situation. An additional factor also makes this situation difficult to analyze: on any given shift, a different product type can be introduced to the line. This results in the need for a model that can respond to a changing system.

The model available to the authors at the time of the study was a discrete-event simulation model of the assembly line that was used during the detailed analysis stage of the manufacturing life-cycle (Singhal *et al.* 1987; Suri 1988). Re-use of this tool would be beneficial during the current operation stage in order to gain insight into the effect that individual stations have on line throughput performance. This type of insight would show where to look in the line in order to make changes that would most effectively improve overall line performance.

More specifically, the objective of this study is to determine an operational policy for a limited number of service personnel. These service personnel are utilized on this line (and others in the facility) to bring stations back on line when failures occur. As a result, we are concerned with the line's throughput sensitivity to changes in the mean-time-to-repair (MTTR) for the automatic stations.

One technique that can be used in combination with a discrete event-simulation, or in some cases, with the real system for sensitivity analysis is perturbation analysis (Ho and Cao 1991; Glasserman 1991). The advantage of this technique is that it can be used to estimate gradients of a performance measure with respect to multiple performance parameters of interest in a single simulation run. As well, these gradient estimates can then be used to determine optimum performance parameters when used in combination with a stochastic approximation (SA) algorithm (Kushner and Clark 1978). In this case, experiments will be performed to determine optimum MTTR values for the automatic stations in the line that can be then used as guidelines for the operational policies of the service personnel.

In addition to the objective of improved assembly line performance, the authors wish to investigate the applicability of perturbation analysis to real-world problems such as this one. This includes both the usefulness of the experimental results as well as the ease of use of this technique in the manufacturing system life-cycle. An attempt is made to implement this technique using a relatively generic approach that could be applied to the actual line as well to other simulations that meet the conditions of the perturbation analysis algorithm. To achieve this objective, the algorithm is written in an object-oriented programming language (C++) and can be interfaced with a simulation of the assembly line or the actual line.

In the following section a brief description of the SA algorithm and its application to this problem will be given; next, a more detailed description of the simulation experiments and results will follow; finally, a discussion of the applicability of this technique to problems of this nature will be given.

2 SINGLE RUN STOCHASTIC OPTIMIZATION

Recent advances in the area of single run gradient estimation techniques have provided the potential to move the powerful evaluative tool of discrete-event simulation into the same category as generative tools such as mathematical programming. This added insight, in combination with the reduction of experimental effort offered by these new techniques promises to widen the scope of modeling and simulation by providing efficient performance optimization techniques (L'Ecuyer *et al.* 1994), and sensitivity analysis (Strickland 1993).

Gradient estimation is an important requirement for the optimization of analytically intractable systems through stochastic approximation (Robbins and Monro 1951; Kiefer and Wolfowitz 1952). The gradient estimate's role in determining an optimum solution can be seen in the basic stochastic approximation algorithm:

$$\theta_{n+1} = \theta_n + \gamma_n Y_n \qquad (1)$$

where θ_n is a parameter value, γ_n is a deterministic sequence of gains, and Y_n is the gradient estimate at iteration n of the algorithm. In order to achieve convergence to an optimum parameter value, θ^*, the sequence of gains, γ_n, are decreased in steps to zero as the simulation experiment proceeds. The choice of a function for γ_n in this case should have the following properties:

$$\lim_{n\to\infty} \gamma_n = 0 \qquad (2)$$

$$\sum_{n=1}^{\infty} \gamma_n = \infty \qquad (3)$$

One example of a gain reduction equation that meets the requirements of equations (2) and (3) and has also been successfully applied to SA algorithms for the M/M/1 queue (Suri and Leung 1989; L'Eculyer *et al.* 1994) and for closed loop flexible assembly systems (Suri and Leung 1987) follows:

$$\gamma_n = \gamma_0 \, n^{-1} \qquad (4)$$

In the experiments that follow, this equation is used to determine the gain reduction after each iteration of the SA algorithm.

One difficulty in applying equation (4) to an optimization problem is the choice of the initial gain, γ_0: a value that is too large will result in θ_n "bouncing" between the upper and lower allowable limits of the parameter value; a value that is too small will converge very slowly. The method used here to choose the initial gain is based on the method used by (Suri and Leung 1987): γ_0 is chosen such that the first few steps of the SA algorithm are of the same order of magnitude as the actual θ_i values (i=1,2,...,M). The order of magnitude referred to here can be determined by observing the gradient estimates for each of the parameter values in a preliminary simulation experiment. In this case, perturbation analysis was used to determine the gradient estimates for each of the stations in the assembly line in a series of preliminary experiments (Rogers and Brennan 1995).

2.1 Single Run Gradient Estimation

As can be seen in equation (1), at the end of each iteration of the SA algorithm, a gradient estimate, Y_n, must be obtained from the simulation experiment. Since Y_n is required as the simulation experiment is run, single run gradient estimation techniques such as perturbation analysis (Ho and Cao 1991; Glasserman 1991) and likelihood ratio (Glynn 1987) are particularly suitable to this type of application. In this paper, we are primarily interested in the first technique; for an overview of other techniques that can be used with the SA algorithm see L'Ecuyer *et al.* (1994).

The infinitesimal perturbation analysis (IPA) technique takes advantage of the structure of a discrete-event dynamic system (DEDS) to obtain an estimate of the gradient of a performance measure, $\nabla_\theta L(\theta)$, using one experimental observation, where $L(\theta)$ is the performance measure of interest, and θ is the performance parameter. In order to develop this estimate, the technique is concerned with how the value $\nabla_\theta E[L(\theta)]$ will be calculated as well as what interactions exist between service events in the sample path when θ is perturbed by $\Delta\theta$. The basic requirements for IPA to obtain an estimate of the gradient are summarized by Suri and Zazanis (1988) into the following categories:

(a) *Perturbation Generation*: the effect of $\Delta\theta$ on a single service completion event.

(b) *Perturbation Propagation*: the effect of perturbations on current events and future events.

(c) *Effect on Performance*: the combined effect of the perturbation generation and propagation effects on the overall system performance.

The IPA algorithm is a combination of (a) and (b), with $\nabla_\theta E[L(\theta)]$, the gradient of an expectation, the combined result. The combination of the perturbation propagation rule and the perturbation generation rule allow perturbation analysis to be expressed in terms of an algorithm. There are several examples (Ho and Cao 1991; Suri and Leung 1987; Suri 1989) of IPA algorithms that can be applied to tandem networks with the addition of only a few lines of code to the simulation or monitoring program. In very general terms, the structure of an IPA algorithm is as follows:

1) *Initialize the accumulators*

2) *Update the accumulators*

   (a) *Perturbation Generation*: Add $dX(\theta)/d\theta$ to the accumulator, A_i, at the end of a service completion.

   (b) *Perturbation Propagation*: Apply the perturbation propagation rule to determine if the perturbation propagates (if a customer leaving station i terminates an idle or a blocked period at station k then copy A_i to A_k).

3) *Branch*: If the last service completion has been reached END, else go to 2.

4) *END*: Calculate the IPA estimate of the gradient from the accumulator values.

At this point, there are a few interesting observations that can be made about the IPA algorithm. Probably the most obvious observation is the simplicity of the algorithm; the few lines of code and minimal storage space that would be required to implement the algorithm would have very little impact on the processing time of the simulation. In addition to the simplicity of the algorithm, it is non-intrusive, working merely by observing the simulation program (or real system).

Finally, *k* gradient values can be estimated simultaneously from a single sample path (simulation experiment). If one considers the amount of experimental effort that this would require if a finite differences technique is used, it can be seen that the savings offered by IPA is very encouraging.

The IPA algorithm described above can now be combined with equation (1) to form an optimization algorithm for the assembly line. The form of the algorithm that is used is based on the algorithm of Suri and Leung (1987) for an M station assembly line:

1) *Initialize*: Initialize the parameter values, θ_i, $i=1,2,...,M$, and set $n=1$.

2) *Estimate Gradients*: Run the simulation until t_n parts are completed, calculating the gradients (*i.e.*, $dL(\theta_i)/d\theta_i$) using the IPA algorithm.

3) *Estimate Optimum Parameter Values*:

   (a) Apply equations (1) and (4) where,

$$Y_n = \frac{dL(\theta_i)}{d\theta_i} - \frac{1}{M}\sum_{j=1}^{M}\frac{dL(\theta_j)}{d\theta_j} \tag{5}$$

   (b) If θ_{n+1} is less (greater) than the interval $[\theta_{min}, \theta_{max}]$, set to θ_{min} (θ_{max}).

4) *Check Stopping Criterion*: End the algorithm if the stopping criterion is satisfied, *i.e.*,

$$\gamma_n \max_{i=1,2,...,M}(sg_i) < \varepsilon \tag{6}$$

where,

$$sg_i = \frac{dL(\theta_i)}{d\theta_i} - \frac{1}{M}\sum_{j=1}^{M}\frac{dL(\theta_j)}{d\theta_j}, \text{ if } n=1$$

or

$$sg_i = 0.8sg_i + 0.2\left\{\frac{dL(\theta_i)}{d\theta_i} - \frac{1}{M}\sum_{j=1}^{M}\frac{dL(\theta_j)}{d\theta_j}\right\}$$

if n>1.

5) *Return*: Set n = n + 1 and go to step 2.

At the beginning of each experiment, the initial θ_i values are chosen randomly from values uniformly distributed across the interval $[\theta_{min}, \theta_{max}] = [0.5, 5.0]$ minutes. The limits, θ_{min} and θ_{max} are related to the physical constraints of the system. The minimum value, θ_{min}, represents the shortest length of time that a repair can be completed. Similarly, the maximum value, θ_{max}, is the longest period of time that is acceptable for a repair. Each of these constraints are related to factors such as the number of service personnel available in the facility, the time it takes a service person to respond, and typical MTTR values for the stations in the assembly line.

The SA algorithm is then run until the stopping criterion is satisfied. As can be seen in equation (6), the stopping criterion is based on a compromise between the actual error and the number of iterations performed (*i.e.*, the term, γ_n will decrease as the simulation experiment progresses). The value, ε, is typically chosen by comparing experimental results with known optimal results (Suri and Leung 1987). Since optimal parameter values are not known for this practical application, ε is based on empirical results (*i.e.*, initial experiments are conducted to determine the value of sg_i when convergence appears to occur).

2.2 Applying IPA to the Assembly Line

As shown in Figure 1, the assembly line studied here consists of a combination of both manual and automatic stations connected by a conveyor system. The service times are deterministic for the automatic stations, and are modeled by a triangular distribution for the manual stations. As well, each station is prone to failure with exponentially distributed mean-time-between-failures (MTBF) and MTTR.

For any given shift, a single class of product types is introduced to the assembly line from what is, effectively, an infinite supply of parts. Each product follows a fixed routing, visiting each station in the line then waiting in the next station's buffer until that station is available.

As well as having random service times, the manual assembly stations also follow a break schedule illustrated in Figure 2. While the assembly workers are taking breaks, the line continues to run until each of the automatic stations are blocked or starved.

It has been shown that, for closed, tandem, single-class networks, the IPA algorithm produces consistent estimates of the gradient. The system studied here meets all of these requirements except the "closed network" requirement.

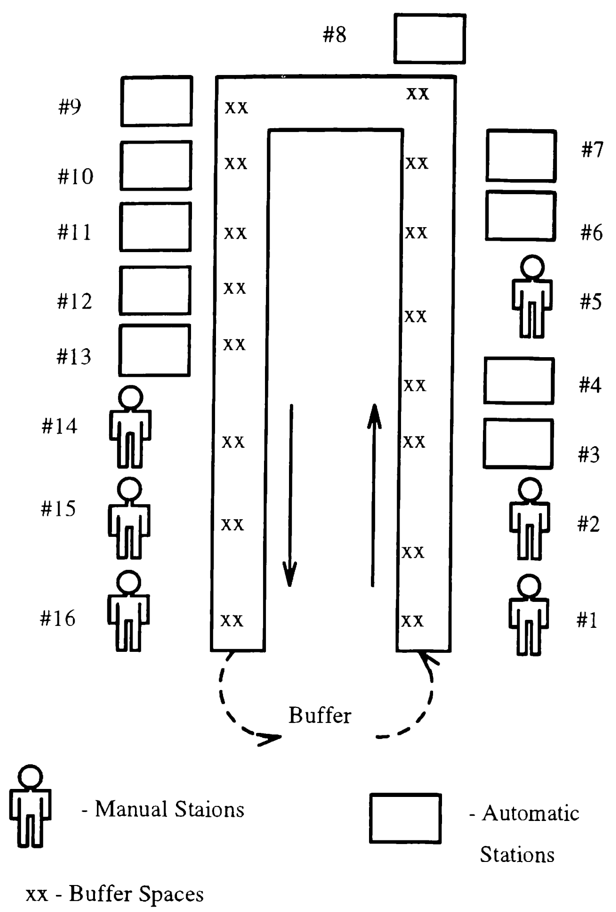

Figure 1: The Assembly Line

Time	Status	Time	Status	Note
7:00	down			meeting
		13:15	down	lunch
7:20	up	13:45	up	
8:00	down	14:00	down	stretch
8:07	up	14:07	up	
9:00	down	15:00	down	rotation
9:02	up	15:02	up	
9:15	down	15:15	down	break
9:30	up	15:30	up	
11:00	down	17:00	down	rotation
11:02	up	17:02	up	
11:15	down	17:15	down	break
11:30	up	17:30	up	
13:00	down			rotation
13:02	up			
		18:55	down	cleanup
		19:00	down	meeting

Figure 2: Manual Station Break Schedule

In order to use the algorithm here, the assembly line is approximated in our simulation by a closed line. As can be seen in Figure 1, an additional station has been added that is effectively an "infinite" buffer between the input and output stations. We assume that the input station is never starved and that the output station is never blocked. This assumption is reasonable in this situation.

In order to test the IPA algorithm for this application, a number of different scenarios are tried with each scenario more closely approximating actual operation of the assembly line. The main difference between each of the test cases is the representation of the manual stations' service time by the simulation model. These scenarios can be described as follows:

(a) Deterministic service times for both manual and automatic stations with no scheduled breaks.

(b) Random service times for manual stations with no scheduled breaks.

(c) Random service times for manual stations with scheduled breaks.

For each of these test scenarios, stochastic approximation will be used to maximize a cost function, $OF(\theta)$:

$$OF(\theta_i) = TH(\theta_i) - \frac{C}{\theta_i} \tag{7}$$

where C is a constant (a value of 0.1 is chosen for the experiments). The first term of equation (7) represents assembly line throughput, $TH(\theta)$; the second term represents the "cost" associated with assigning service personnel to station i. Differentiating equation (7) results in the gradient:

$$OF'(\theta_i) = TH'(\theta_i) + \frac{C}{\theta_i^2} \tag{8}$$

where $TH'(\theta)$ is represented by a location parameter (Glasserman 1991) in the IPA algorithm. In the following experiments the SA algorithm is used to maximize the cost function represented in equation (7).

3 EXPERIMENTAL RESULTS

3.1 The Experimental Testbed

The three test scenarios described in the previous section are implemented using the existing SIMAN discrete event simulation model with minor modifications to support an interface with a C++ program that performs the SA algorithm calculations. To allow SIMAN and the C++ program to communicate, the standard SIMAN user code (ANSI C) is used with data transfers taking place over a UNIX socket. A schematic of the implementation is illustrated in Figure 3. The modular implementation used here is intended to encourage re-use of the SA/IPA module with discrete-event simulation models of other lines in the manufacturing facility.

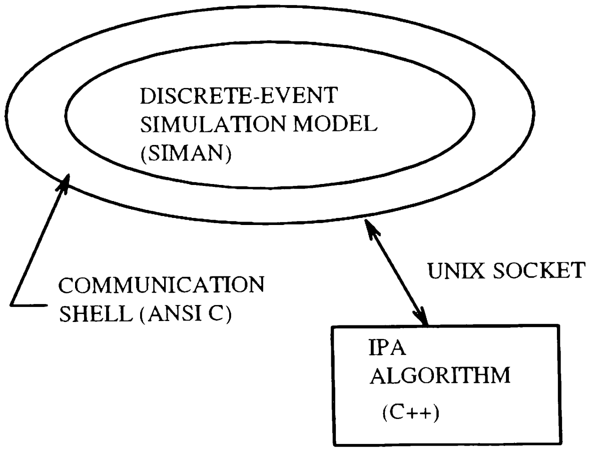

Figure 3: The Experimental Testbed

3.2 The Simulation Experiments

When a specific product order is sent to the shop floor in the actual system, the assembly line is run continuously until the order is completed. Manual assemblers operate in 12 hour shifts with the break schedule illustrated in Figure 2.

For the simulation experiments of the three scenarios described previously, a non-terminating analysis is used to allow the system to approach steady-state. Statistics on the SA algorithm are collected at the beginning and the end of each SA run (each SA "run" lasts the length of time required to reduce sg_i to below ε).

In order to provide a "warm-up" period for the system, the first iteration is started after 500 parts are completed, then subsequent iterations are performed after each 100 parts are completed. The length of time between SA iterations will have an effect on the accuracy of the IPA gradient estimates and, as a result, on the SA algorithm: short iterations may not provide the system with enough time to stablize after a parameter change, long iterations will adversely effect the rate of convergence. The value chosen here (*i.e.*, t_n = 100 parts) gives the system approximately one hour (real system time) to stablize after each iteration.

As noted previously, the initial system gains are based on the magnitude of the gradients investigated in previous experiments with this system (Rogers and Brennan 1995). These initial gains are shown in Table 1.

Table 1: Initial SA Algorithm Gains

Station	γ_0	Station	γ_0
1	250	9	110
2	90	10	120
3	110	11	110
4	120	12	120
5	10	13	130
6	110	14	90
7	110	15	90
8	30	16	80

The SA algorithm stopping criterion for the experiments is based on $sg_i < 0.2$. Once this criterion is met, a new set of initial parameters are randomly selected from a uniform distribution.

3.3 Results

Each of the three test scenarios described previously were simulated using the experimental testbed shown in Figure 3 in order to determine the convergence rate of the SA algorithm as well as the optimum MTTR values. In each of the three cases, the SA algorithm converged to consistent parameter values after each SA run. The convergence rate for the algorithm also appeared to be very similar for each test scenario:

(a) Deterministic service times: I = (76.9, 77.4)

(b) Triangularly distributed service times: I = (78.7, 79.0)

(c) Triangularly distributed service times with the break schedule of Figure 2: I = (67.4, 69.8)

where, I is the 95% confidence interval (C.I.) for the number of iterations. Figure 4 shows an example of the error reduction for a single SA run. Typical results for a single station (station 14) are shown in Figure 5. In each of these figures, of the three test scenarios shown, the third scenario approximates the actual assembly line most closely.

Tables 2, 3, and 4 show the 95% C.I. results for the MTTR values (by station, Stn.) for scenarios (a), (b), and (c) respectively. These results were generated at the end of each SA run by the C++ program described in the previous section.

The stations with θ_i values of (0.5,0.5) in Tables 2, 3, and 4 indicate those stations with optimal values that are less than θ_{min} (*e.g.*, stations 5 and 8 of Table 2). As a result of step 3(b) of the SA algorithm, θ_i is set to θ_{min} in this case.

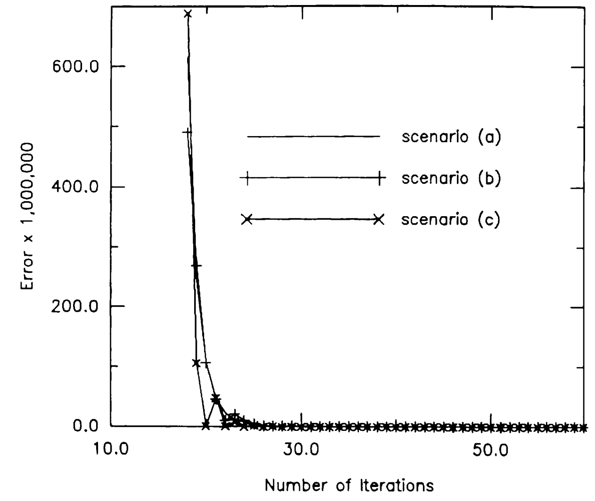

Figure 4: SA Algorithm Convergence (Error)

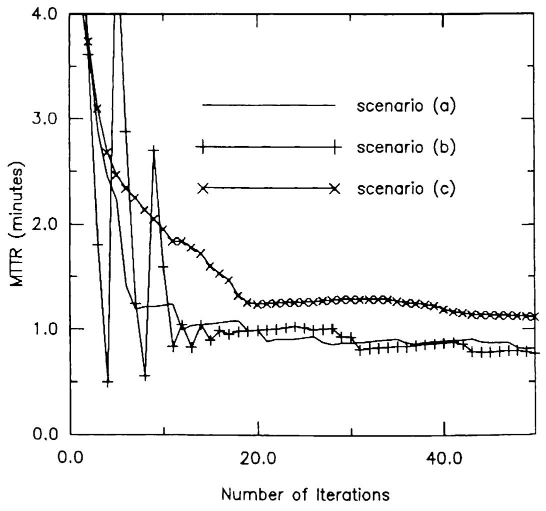

Figure 5: SA Algorithm Convergence for Station 14 (MTTR)

Table 2: SA Results for Scenario (a)

Stn.	θ_i (min)	Stn.	θ_i (min)
1	(1.416,1.463)	9	(1.793,1.813)
2	(1.402,1.461)	10	(1.970,1.990)
3	(1.716,1.735)	11	(1.440,1.465)
4	(1.442,1.462)	12	(1.873,1.917)
5	(0.5,0.5)	13	(1.054,1.080)
6	(1.587,1.600)	14	(0.643,0.665)
7	(0.651,0.659)	15	(2.743,2.988)
8	(0.5,0.5)	16	(0.923,0.947)

Table 3: SA Results for Scenario (b)

Stn.	θ_i (min)	Stn.	θ_i (min)
1	(1.244,1.271)	9	(1.841,1.868)
2	(1.678,1.726)	10	(1.025,2.059)
3	(1.757,1.777)	11	(1.489,1.531)
4	(1.439,1.458)	12	(1.840,1.892)
5	(0.5,0.5)	13	(1.188,1.213)
6	(1.383,1.403)	14	(0.756,0.768)
7	(1.222,1.232)	15	(2.794,2.965)
8	(1.017,1.034)	16	(0.930,0.966)

Table 4: SA Results for Scenario (c)

Stn.	θ_i (min)	Stn.	θ_i (min)
1	(1.925,1.999)	9	(2.145,2.314)
2	(2.443,2.670)	10	(2.219,2.409)
3	(2.532,2.745)	11	(1.824,1.965)
4	(2.133,2.265)	12	(2.052,2.177)
5	(0.5,0.5)	13	(1.514,1.699)
6	(1.867,1.976)	14	(0.899,0.940)
7	(1.689,1.784)	15	(3.533,3.820)
8	(1.468,1.532)	16	(1.309,1.393)

4 INTERPRETING THE RESULTS

As noted previously, the primary objective of the current study is to provide operational guidelines for the assembly line service personnel in order to determine which machines can be considered to be of high priority when failures occur. The results in Tables 2-4 do provide this type of information by indicating where service personnel should focus their attention on the assembly line. For example, Table 4 (random service times with scheduled breaks) indicates that stations 5 and 14 must be repaired as quickly as possible when a failure occurs. Station 15 in this case, can be considered to be of lower priority.

In order to validate these results, a comparison was made in a previous study (Rogers and Brennan 1995) between the IPA gradient estimates and gradient estimates obtained by finite differences. The two techniques showed a close correspondence for each of the three test scenarios described in this paper.

In addition to investigating the performance of the SA/IPA algorithms, the authors are also interested in a secondary objective of evaluating the applicability of this type of analysis to problems of the type described in this paper. The line studied here is part of a larger electronics manufacturing facility consisting of a number of similar lines used to manufacture a variety of components. Since a limited number of service personnel are are assigned to all of the equipment in the manufacturing facility, information similar to that given in Tables 2-4 can be of great benefit in determining service priorities.

As well, the modular approach used here is modifiable and extendible: the SA/IPA algorithms can be easily applied to discrete-event simulation models of other lines in the manufacturing facility. This extendibility will be dependent, of course, upon the applicability of the SA/IPA algorithms to the simulation model they are to be applied to.

Finally, the difficulty (or impracticality) of performing this type of analysis on the actual line makes this technique particularly beneficial as an off-line analysis tool. In the dynamic environment of an electronic manufacturing facility (*e.g.*, where changes in product type or, additions and removals of machines from the line can occur), a single-run analysis tool such as this can be used to provide insight into the effect of changes.

ACKNOWLEDGEMENTS

The authors wish to thank the Natural Sciences and Engineering Research Council of Canada for their generous support of this research under grant OGP-012-1522 and scholarship PGS-B #178095.

REFERENCES

Askin, R. and C. Standridge. 1993. *Manufacturing Systems Modeling*. John Wiley & Sons.

Buzacott, J.A. and J.G. Shanthikumar. 1993. *Stochastic Models of Manufacturing Systems*. Prentice Hall.

Glasserman, P. 1991. *Gradient Estimation via Perturbation Analysis*. Kluwer Academic Publishers.

Glynn, P.W. 1987. Likelihood ratio gradient estimation: an overview. In *Proceedings of the 1987 Winter Simulation Conference*, ed. A. Thesen, H. Grant, and W. David Kelton, 366-375.

Ho, Y.C. and X.R. Cao. 1991. *Perturbation Analysis of Discrete Event Dynamic Systems*. Kluwer Academic Publishers.

Kiefer, J. and J. Wolfowitz. 1952. Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics* 23:462-466.

Kushner, H.J. and D.S. Clark. 1978. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag.

L'Ecuyer, P., N. Giroux, and P.W. Glynn. 1994. Stochastic optimization by simulation: numerical experiments with the M/M/1 queue in steady-state. *Management Science* 40:1245-1261.

Robbins, H. and S. Monro. 1951. A stochastic approximation method. *Annals of Mathematical Statistics* 22:400-407.

Rogers, P. and R.W. Brennan. 1995. Applying infinitesimal perturbation analysis to a manufacturing system design problem. In *Proceedings of the Summer Computer Simulation Conference*, ed. T.I. Oren and L.G. Birta, 422-427.

Singhal, K., C.H. Fine, J.R. Meredith, and R. Suri. 1987. Research models for automated manufacturing. *Interfaces* 17:5-14.

Strickland, S.G. 1993. Gradient/sensitivity estimation in discrete-event simulation. In *Proceedings of the 1993 Winter Simulation Conference*, ed. G.W. Evans, M. Mollaghasemi, E.C. Russell, and W.E. Biles, 97-105.

Suri, R. 1988. A new perspective on manufacturing systems analysis. In *Design and Analysis of Integrated Manufacturing Systems*, ed. W. Dale Compton, 118-133. National Academy Press.

Suri, R. 1989. Perturbation analysis: the state of the art research issues explained via the GI/G/1 queue. *Proceedings of the IEEE* 77:114-137.

Suri, R. and Y.T. Leung. 1987. Single run optimisation of a SIMAN model for closed loop flexible assembly systems. In *Proceedings of the 1987 Winter Simulation Conference*, ed. A. Thesen, H. Grant, and W. David Kelton, 738-748.

Suri, R. and Y.T. Leung. 1989. Single run optimization of discrete event simulations-an empirical study using the M/M/1 queue. *IIE Transactions* 21:35-49.

Suri, R. and M. Zazanis. 1988. Perturbation analysis gives strongly consistent sensitivity estimates for the M/G/1 queue. *Management Science* 34:39-64.

AUTHOR BIOGRAPHIES

ROBERT W. BRENNAN is a graduate student in the Department of Mechanical Engineering (Division of Manufacturing Engineering) at the University of Calgary, working towards his Ph.D. degree. His research interests include control architectures for manufacturing systems, optimization of discrete-event simulation, and models for the analysis of manufacturing systems. He has over seven years industrial experience in project management and control systems and is a Professional Engineer and a member of IIE. He holds a B.Sc. degree in Mechanical Engineering from the University of Calgary.

PAUL ROGERS is an Associate Professor in the Department of Mechanical Engineering (Division of Manufacturing Engineering) at the University of Calgary. His research interests include discrete-event simulation, production planning and control systems, object-oriented modeling for intelligent manufacturing, and models for the analysis of manufacturing systems. He is a Professional Engineer and a member of IIE, INFORMS, SME, and SCS and serves on the Editorial Board of the *International Journal of Computer Integrated Manufacturing*. He holds Ph.D. and M.Eng. degrees from Cambridge University in England.