Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

RESPONSE GRADIENT ESTIMATION USING MATHEMATICAL PROGRAMMING MODELS OF DISCRETE-EVENT SYSTEM SAMPLE PATHS

Wai Kin (Victor) Chan

Department of Decision Sciences and Engineering Systems
Rensselaer Polytechnic Institute
CII 5015, 110 Eighth Street
Troy, NY 12180 U.S.A.

Lee W. Schruben

Department of Industrial Engineering and Operations Research
University of California, Berkeley
4135 Etcheverry Hall
Berkeley, CA 94720 U.S.A

ABSTRACT

This paper illustrates the use of mathematical programming in computing gradient estimators. Consistency property of these estimators is established under the usual assumptions for IPA gradient estimator consistency. A finite difference tolerance limit is introduced. For complex discrete-event systems, more concise linear programming representations are developed. These new representations provide a direct way of calculating gradient estimates.

1 INTRODUCTION

Infinitesimal perturbation analysis (IPA) is a technique for estimating the gradient of a system performance measure by observing the sample path from a single simulation run (Ho et al. 1979, Glasserman 1991, Ho and Cao 1991, Fu and Hu 1997, Suri and Zazanis 1988, Freimer and Schruben 2001). Compared with the method of finite differences, which requires k additional simulation runs to calculate the gradients for k parameters, IPA not only saves a great amount of computation in simulation, but also avoids some difficulties in doing finite differences, for example, choosing a big value of the difference may result in a poor approximation to the gradient, while a too small of the difference could introduce a large variance in the estimation (Suri 1989). Another major benefit of IPA lies in the fact that estimation can be carried out online while a simulation is running. This is crucial to real-time optimization of systems in which replications with different values of parameters are difficult, if not infeasible, to perform (Cao and Chen 1997).

IPA algorithms can be implemented very efficiently in most discrete-event simulations. Suri (1987) provides an algorithm that incorporates IPA estimation in a general discrete-event simulation with only four additional lines of code. Graphical implementation of IPA has also been developed in Freimer and Schruben (2001), yielding a general and visual method of computing IPA estimates.

Despite all the aforementioned benefits, IPA has a limitation: It does not provide consistent or unbiased estimators for some systems; for example, the IPA estimator for non-identical multiple servers queues is neither consistent nor unbiased (Fu and Hu 1991). Cao (1985) and Glasserman (1991b) have provided conditions and treatments under which IPA estimators will have desirable statistical properties. Other types of perturbation analysis estimation techniques have also been proposed (see, for example, Gong and Ho 1987).

In this paper, a different approach to calculating gradient estimators is taken. This approach is based on mathematical programming representations of discrete-event system dynamics. Modeling sample paths of simulations as mathematical programs is a technique that allows the use of the algorithms and techniques of mathematical programming in the design and analysis of discrete-event systems (Schruben 2000, Chan and Schruben 2003, and Chan 2005).

Specifically, this paper illustrates the use of mathematical programming representations for discrete-event systems in computing gradient estimators. Since the mathematical programming representations are developed from event relationship graphs (ERGs)—a general graphical simulation modeling technique, a very brief review to ERGs is given in Section 2.1. Mathematical programming representations for discrete-event systems are presented in Section 2.2. Gradient estimators obtained from these mathematical programming representations are derived in Section 3. The consistency of these gradient estimators for systems in which IPA works is also established. For systems in which the IPA fails, the mathematical programming representations will be used to obtain information regarding the finite-difference sample path—a topic discussed in Section 4. Another type of gradient estimators—called linear programming perturbation analysis (LPA) estimators—for general discrete-event simulations is introduced in Section 5.

2 BACKGROUND

In this section, we first give a brief review to event relationship graphs. Then, the technique of mapping discrete-event system dynamics as mathematical programming formulations is illustrated by a simple example—a single-server queue.

2.1 Event Relationship Graph Models

Event relationship graphs (ERGs) are a general, minimalist means of explicitly expressing all the relationships between events in a discrete-event dynamic system model (Schruben 1983, Pegden 1986, Som and Sargent 1990, Wu and Chung 1991, Askin and Standridge 1993, Law and Kelton 2000, Seila, Ceric and Tadikamalla 2003).

The vertices of an ERG represent state changes that take place when a particular type of event occurs. The directed arcs of the graph represent the relationships between pairs of events. The state changes associated with each event vertex appear in braces. Labels on directed arcs—representing all the dynamic and logical relationships between events—specify the conditions and time delays between the occurrences of events. Following the ERG notation in Askin and Standridge (1993), a generic arc in an ERG is shown in Figure 1 and defined as follows: *after event A occurs, if condition i_{AB} is then true, event B will immediately be scheduled to occur t_{AB} time units into the future.*

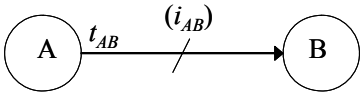

Figure 1: Basic Element of an Event Relationship Graph

A single-server, single-class of jobs with batch size 1, first-in-first-out, non-preemptive queueing system ($G/G/1$) will be used in this paper for illustration in subsequent sections. One of several possible ERGs for this system is shown in Figure 2 where the state is described by two integers: R = the number of currently idle resources (at most 1 in this example) and Q = the number of jobs currently waiting in line for service. The input data for simulating the ERG are the (random) customer interarrival times, a, and the (random) service times, s.

The queue in Figure 2 is assumed to be initially empty ($Q = 0$) with the server idle ($R = 1$). Initially scheduled events are indicated by a broken arrow as in Law and Kelton (2000). The only event initially scheduled here is the first job to arrive at time zero. The "&" is shorthand for the Boolean "AND" operator.

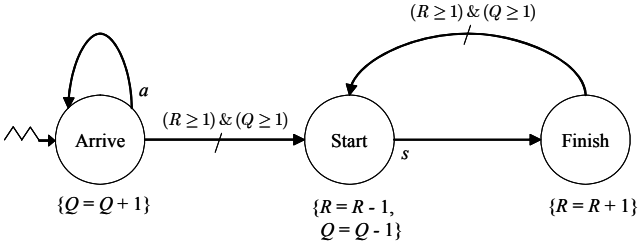

Figure 2: ERG for a $G/G/1$ Queue

2.2 *G*/*G*/1 queue Formulation

In this section, we illustrate the mapping an ERG into an LP using a $G/G/1$ queue example as done in Schruben, 2000. A general treatment of the translation of an ERG into an optimization model is given in Chan (2005).

The following notation for queueing networks will be used in this paper. For $i = 1,\ldots,n$, let a_i be the time interval between the i^{th} external arrival and the i-1^{th} external arrival (i.e., the i^{th} realization of random variable a in Figure 2), s_{ki} be the time interval needed for the i^{th} service at stage k (i.e., the i^{th} realization of random variable s in Figure 2), A_{ki} be the time of the i^{th} external arrival event occurrence at stage k, S_{ki} be the time of the i^{th} service start event occurrence at stage k, and F_{ki} be the time of the i^{th} service finish event occurrence at stage k.

Translating this ERG to an optimization program that generates the sample path for given interarrival times $\{a_i\}$ and service times $\{s_i\}$ for n jobs gives the following linear program—GG1-LP1. There are four constraints in GG1-LP1: the first two equalities are derived from the two non-zero-delay unconditional arcs, and the last two inequalities are generated from the two zero-delay conditional arcs by using the conditions specified there. The objective function, as in a simulation, is simply to execute all the events as early as feasible. The optimal solution of this LP is identical to the state trajectories (sample paths) generated by simulating an ERG model of the system (see Chan 2005). We assume all delay times are positive. Therefore, we do not include the nonnegative constraints on the variables.

GG1-LP1:

$$
\begin{array}{lll}
\min & \sum_{i=1}^{n}(A_i + S_i + F_i) & \\
st. & & \\
A_{i+1} & = A_i + a_{i+1} & ,i = 1,\ldots,n-1 \\
F_i & = S_i + s_i & ,i = 1,\ldots,n \\
S_i & \geq A_i & ,i = 1,\ldots,n \\
S_i & \geq F_{i-1} & ,i = 2,\ldots,n \\
A_1 = 0; A_i, S_i, F_i & \text{free}, \forall i\,. &
\end{array}
$$

For this simple system, there is one constraint for every arc in the ERG. The first constraint is the definition of the in-

terarrival time: the i+1th job will arrive (at time A_{i+1}) in a_{i+1} time units after the i^{th} job arrives at the system. The second constraint is the definition of the service time and states that the i^{th} job (started at time S_i) will finish its service s_i time units later (at time F_i). The third constraint ensures that the i^{th} service cannot start before the i^{th} job actually arrives. The last constraint enforces the constraint that the server cannot start work on the next job before it finishes the previous job.

In fact, we do not need to include the job arrival times in our objective function since they are not scheduled by conditional edges. Different ERGs, that are equivalent in the sense that all their sample paths are identical given the same input, can be created that model a particular discrete-event system. Some of these have distinct mathematical programming representations. These mathematical programs also may have distinct but equivalent formulations that correspond to different simulation models of the system. Heuristics and algorithms for eliminating redundant events in an ERG have been proposed (Schruben 1983 and Som and Sargent 1990, Seila, Ceric and Tadikamalla 2003).

For example, in GG1-LP1, the first constraint is only for the Arrival event times, which are input data for the simulation model and therefore can be removed from the formulation. The second constraint, being an equality, can also be eliminated from the formulation by incorporating it into the last constraint, thus replacing F_i with $S_i + s_i$, i = 1,...,n. Dropping the constant terms, i.e., $\sum_{i=1}^{n}(a_i + s_i)$, from the objective function yields a linear program for the $G/G/1$ queue that has only Start events as variables,

GG1-LP1(S):

$$\begin{array}{lll} \min \sum_{i=1}^{n} S_i & & \\ st. & & \\ S_i & \geq A_i \quad ,i=1,...,n & (U_i) \\ S_i - S_{i-1} & \geq s_{i-1} \quad ,i=2,...,n & (V_i) \\ S_i \text{ free } \forall i & & \end{array}$$

This is an example where developing ERG models from their equivalent optimization formulations can sometimes give different, and potentially more efficient, simulations.

GG1-LP1(S) can also be modified to a new formulation with an objective function of minimizing the average job waiting time ($W = \frac{1}{n}\sum_{i=1}^{n}(S_i - A_i)$) by replacing S_i with $W_i + A_i$, i = 1,...,n, in the formulation, dropping the constant term of the sum of the A_i's from the objective function, and dividing it by n. This new formulation and its dual are

GG1-LP1(W):

$$\begin{array}{lll} \min \frac{1}{n}\sum_{i=1}^{n} W_i & & \\ st. & & \\ W_i & \geq 0 \quad ,i=1,...,n & (U_i) \\ W_i - W_{i-1} & \geq s_{i-1} - a_i \quad ,i=2,...,n & (V_i) \\ W_i \text{ free } \forall i & & \end{array}$$

GG1-LP1-Dual(W):

$$\begin{array}{llll} \max \frac{1}{n}\sum_{i=1}^{n}(s_i - a_{i+1})V_i & & & \\ st. & & & \\ U_1 & -V_2 & = 1 & (W_1) \\ U_i + V_i & -V_{i+1} & = 1 \quad ,i=2,...,n-1 & (W_i) \\ U_n + V_n & & = 1 & (W_n) \\ U_i, V_i \geq 0, \forall i & & & \end{array}$$

Here U_i, and V_i, are the corresponding dual variables for each constraint. GG1-LP(W) is equivalent to the well-known Lindley recursion (Lindley 1952), $W_i = \max\{W_{i-1} + s_{i-1} - a_i, 0\}$. In fact, many of the max-plus representations of stochastic system sample paths have direct linear programming representations and hence, duals.

3 SENSITIVITY ANALYSIS

Consistency property of IPA estimators has been proved for certain queueing systems, see for example, Suri and Zazanis (1988), Fu and Hu (1991). When IPA fails, various generalizations or alternatives of perturbation analysis techniques have been developed. One of these is the smoothed perturbation analysis (SPA), which uses conditional probability to derive gradient estimators (Gong and Ho 1987). See Fu and Hu (1997) for a detailed discussion and comparison of various extensions of perturbation analysis techniques. Homem-de-Mello et al. (1999) also makes use of max-plus algebra to obtain sample path gradient for production scheduling problems under continuous distributions.

In this section, we illustrate the use of the duals of mathematical programming representations of ERGs to develop finite-difference gradient estimators.

Consider an LP representation for a discrete-event system. Let us call this LP, given in the following, as DES-LP1. To make our discussion more general, we will use the standard notation for linear programming. We have converted the inequality constraints to equality constraints by adding excess variables. This causes us to include nonnegative constraints in the model. However, in the original formulation without excess variables, the variables will automatically satisfy the nonnegative condition because the right-hand-side is assumed to be positive. Notation with subscript "B" (or "N") are related to basic variables (or

non-basic variables). Let $\boldsymbol{c} = (\boldsymbol{c}_B; \boldsymbol{c}_N)$ be an $(n+m)\times 1$ vector of objective coefficients, $\boldsymbol{A} = (\boldsymbol{B}, \boldsymbol{N})$ be an $m\times(n+m)$ coefficient constraints matrix, $\boldsymbol{d}(\theta) = (d_1(\theta); d_2(\theta);\ldots;d_m(\theta))$ be an $m\times 1$ vector of the right-hand side which is a function of scalar parameter θ, $\boldsymbol{x} = (\boldsymbol{x}_B; \boldsymbol{x}_N)$ be an $(n+m)\times 1$ vector of variables, and $\boldsymbol{u} = (u_1; u_2;\ldots;u_m)$ be an $m\times 1$ vector of dual variables. Here, m is the number of constraints, usually greater than n—the number of variables. It can be shown, assuming a non-degenerate formulation, that the number of possible sample paths is equal to the number of extreme points, $\binom{m+n}{m} = \frac{(m+n)!}{m!\,n!}$, bounded by the number of constraints, in the feasible region of the LP.

DES-LP1:

$$\begin{aligned} & z(\theta) = \min \boldsymbol{c}_B^T \boldsymbol{x}_B + \boldsymbol{c}_N^T \boldsymbol{x}_N \\ & s.t. \quad \boldsymbol{B}\boldsymbol{x}_B + \boldsymbol{N}\boldsymbol{x}_N = \boldsymbol{d}(\theta) \qquad (\boldsymbol{u}) \\ & \qquad \boldsymbol{x} \geq 0 \end{aligned}$$

Although the decision variables in this LP representation are the event times, other performance functions can still be evaluated by using the fact that the state of a DES is piecewise constant. For example, if $L_n(\theta)$ is a sample performance measure evaluated over a sample path containing n events, $C(\boldsymbol{Z}_i)$ is a bounded cost when the system is at state $\boldsymbol{Z}_i$, and x_i is event time of the i^{th} event, then we have (see e.g., Glasserman 1991a):

$$L_n(\theta) = \sum_{i=0}^{n-1} C(\boldsymbol{Z}_i)[\,x_{i+1} - x_i\,].$$

In this LP, the dual variables (shadow prices) represent how sensitive the objective function (system performance) is to changes in the right-hand-side random variables (input data) and therefore, provide information necessary for computing gradient estimators using the chain-rule as done in IPA gradient estimation. In fact, perturbations are propagated through all the binding constraints (constraints with zero excess) and the value of each dual variable represents the marginal effect of the corresponding right-hand-side random variable to the objective function. Therefore, all the binding constraints constitute an event-tree (the solution of the dual LP) similar to the one defined in Suri (1987). As a matter of fact, the LP formulation is a generic event-tree: given a sequence of input random variables, a realization of the event-tree can be constructed by solving the LP and connecting all binding constraints to form the branches.

However, the LP solution obtained from running the simulation might provides more information for a single simulation run because other perturbed sample paths can be reached from the current sample path by some additional computation (pivots), which might be easier than running a new simulation. From the computational point of view, the LP representations would be a potentially effective tool for other types of sensitivity analysis (in particularly, finite difference gradient analysis) when IPA fails, for example, using the dual-simplex method to get new sample paths. Performance of such sensitivity analysis is under investigation.

Let us now examine the consistency property of IPA gradient estimators computed using the dual variables. Dividing the objective function by n (this will not alter the optimal basis) and taking the limit $n \to \infty$ gives a consistent estimator of the mean of event times. $\overline{x}(\theta) = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} x_i^*$ a.s., where x_i^*'s are the optimal primal solutions, or equivalently working with the dual, $\overline{x}(\theta) = \lim_{n\to\infty} \frac{1}{n} z(\theta) = \lim_{n\to\infty} \frac{1}{n} \boldsymbol{d}(\theta)\boldsymbol{u}^*$ a.s., where $\boldsymbol{u}^*$ is the optimal dual vector. For a small perturbation $\Delta\theta$ provided that the order of events remains unchanged—an usual assumption of IPA—the current dual variables remain optimal and therefore, the objective function is perturbed by an amount of $\Delta z = \Delta \boldsymbol{d}\boldsymbol{u}^*$, where $\Delta\boldsymbol{d}$ is the amount of perturbation of the right-hand side due to the change in θ. The change to the mean event time is then

$$\begin{aligned} \overline{x}(\theta + \triangle\theta) - \overline{x}(\theta) &= \lim_{n\to\infty} \frac{1}{n} [\,\boldsymbol{d}(\theta + \triangle\theta) - \boldsymbol{d}(\theta)\,]\boldsymbol{u}^* \\ &= \lim_{n\to\infty} \frac{1}{n} \triangle \boldsymbol{d}\boldsymbol{u}^* \qquad a.s. \end{aligned}$$

Divided by $\Delta\theta$ and letting $\Delta\theta \to 0$ yields the derivative of the mean event time. Now, using the same assumptions typically made in IPA, i.e., the random variables $d_i(\theta)$'s are uniformly differentiable—a condition such that the random variables are smooth enough or well-behavior so that IPA works (see Cao 1985 or Ho and Cao 1991 for more details), we have

$$\begin{aligned} \frac{d\overline{x}(\theta)}{d\theta} &= \lim_{\triangle\theta\to 0} \lim_{n\to\infty} \frac{1}{n} \frac{\triangle \boldsymbol{d}(\theta)}{\triangle\theta} \boldsymbol{u}^* \qquad a.s. \\ &= \lim_{\triangle\theta\to 0} \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{m} \frac{\triangle d_i(\theta)}{\triangle\theta} u_i^* \\ &= \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{m} d_i'(\theta) u_i^* \end{aligned}$$

where $d_i'(\theta)$ is the derivative of $d_i(\theta)$ w.r.t. θ (assume exists) and the last equation uses the uniform differentiability condition. Observe that as n goes to infinity, so does m. Therefore, the dual variables provide a consistent estimator for the derivative of the mean event time under the usual IPA assumptions. If θ is a suitable (e.g., scale or shift) parameter of the arrival or service distributions, then the chain rule can be used in the usual IPA manner to compute the gradient estimates.

Linear programming formulations for closed and open tandem queueing networks are also given in Chan (2005).

There it is shown that the dual variables for different constraints have distinct physical meanings; for example, some of them represent the number of jobs in a busy period while the others equal the number of jobs in a local busy period (for definition of local busy period, see Fu and Hu 1997). Therefore, similar gradient estimators for queueing networks can also be computed using the dual variables.

4 TOLERANCE LIMIT FOR FINITE DIFFERENCE GRADIENT ESTIMATORS

Given a sample path, there exists a limit on the changes of the input parameters beyond which the perturbed sample path could look different from the original sample path either temporary or permanently. In this section, we derive an explicit formula for computing this tolerance limit.

When the right-hand side changes from $\boldsymbol{d}(\theta)$ to $\boldsymbol{d}(\theta+\triangle\theta)$, the value of the primal variables (event times) will change, so does the value of the objective function. The dual variables, however, will not change because the optimality condition, $\boldsymbol{c}_N^T - \boldsymbol{c}_B^T\boldsymbol{B}^{-1}\boldsymbol{N} \geq 0$, is independent of the right-hand side. The current primal variables will remain feasible until the feasibility condition is violated, or equivalently,

$$\boldsymbol{B}^{-1}\boldsymbol{d}(\theta) + \boldsymbol{B}^{-1}\triangle\boldsymbol{d} \geq \boldsymbol{0}.$$

Since $\boldsymbol{x}_B = \boldsymbol{B}^{-1}\boldsymbol{d}(\theta)$, this condition can be expressed in terms of each component:

$$\left(\boldsymbol{x}_{Bj} + \boldsymbol{B}^{-1}\triangle\boldsymbol{d}\right)_j \geq 0 \quad , j = 1,..,m$$

Therefore, the finite difference tolerance limit is given by

$$\min\{(x_B + \boldsymbol{B}^{-1}\triangle\boldsymbol{d})_j \geq 0, j = 1,...,m\}.$$

We note that this limit is strict. In some cases, although a small input parameter difference could cause the original sample path—nominal sample path—to change, Glasserman (1991a) has shown that under the so-called "commuting condition," the perturbed sample path will only deviate from the original sample path temporary.

The next natural question is what happens if this limit is exceeded; Can we get a finite difference gradient estimator using the information from the original sample path without running another simulation again? It turns out that this can be done by using parametric programming (Gal 1979). For example, if some primal variables are infeasible, then one can apply the dual simplex method to pivot other non-basic variables into the basis. After pivoting out all infeasible primal variables and obtaining a new optimal solution, the total change in the objective function would be

$$\triangle z = -\sum_{i\in\{i':(\boldsymbol{x}_B+\boldsymbol{B}^{-1}\triangle\boldsymbol{d})_{i'}<0\}} \frac{c_B^T\boldsymbol{B}^{-1}\boldsymbol{N}_j}{\left(\boldsymbol{B}^{-1}\boldsymbol{N}_j\right)_i}\left(\boldsymbol{x}_B + \boldsymbol{B}^{-1}\triangle\boldsymbol{d}\right)_i$$

where j is the index for an entering non-basic variable (for each infeasible primal variable indexed by i) such that it has the smallest ratio, that is,

$$\min\left\{\left|\frac{c_B^T\boldsymbol{B}^{-1}\boldsymbol{N}_j}{\left(\boldsymbol{B}^{-1}\boldsymbol{N}_j\right)_i}\right| : \left(\boldsymbol{B}^{-1}\boldsymbol{N}_j\right)_i < 0\right\}.$$

The computational requirement is in general less than that in performing a new experiment because the matrix $\boldsymbol{B}^{-1}\boldsymbol{N}$ is available from the original sample path.

5 SAMPLE-PATH LINEAR PROGRAMMING REPRESENTATIONS FOR GENERAL DISCRETE-EVENT SYSTEMS

For a complex discrete-event system, the mathematical programming representation derived using the technique given in Chan (2005) could be complicated and might also involve binary variables. One reason is that the mathematical programming representation contains all possible sample paths—recall that that the number of possible sample paths is equal to the number of extreme points. Calculating the shadow prices would then become time-consuming. To cope with this situation, we propose here another type of linear programming representations for general discrete-event systems that requires further investigation.

We observe that an IPA estimator is calculated from only one sample path and that other sample paths are not needed if order of events remains unchanged. These facts lead us to define the Sample-Path Linear Programming Representation (SPLP), which is the LP representation for a particular sample path.

Let $x_{\alpha i}$ be the time of i^{th} occurrence of the α event, i=1,…,n and $d_{\alpha i}$ be the duration from the time at which the i^{th} α event is scheduled (active) until it occurs. A SPLP is generated while the simulation is running. Specifically, all the constraints in a SPLP are generated using the following simple algorithm:

Algorithm LPA:

During the execution of a simulation run, whenever an event (say α) is scheduled to occur after a delay $d_{\alpha i}$, then

Step 1: Add the constraint $x_{\alpha i} - x_{\beta j} \geq d_{\alpha i}$ to the constraint matrix.
Step 2: Add $x_{\alpha i}$ to the objective function to be minimized.
Repeat Steps 1 and 2 until the simulation ends.

Therefore, the sample-path LP is created constraint by constraint while the simulation is running. Moreover, solving the SPLP is not required because the simulation results already give the values of all variables (including the optimal values of any binary variables if present). The IPA estimator can then be computed from the shadow prices in real-time. This will facilitate the implementation of online optimization algorithms, which use the gradient estimates from the SPLPs as the search direction.

6 CONCLUSION AND FUTURE WORK

One main goal of having the mathematical programming representations for discrete-event systems is to allow the application of the rich mathematical theory and algorithms of optimization to the study of discrete-event stochastic systems (Schruben 2000, Chan and Schruben 2003, Chan 2005). Getting IPA estimators from the shadow prices is just one application, opening an approach to sensitivity analysis. More study is needed in justifying statistical properties of these estimators when IPA gradient estimation fails. One fact is that the amount of changes due to a small perturbation on parameters is equal to the distance (difference) between two extreme points in the feasible region of the linear program. However it is not known now that this distance is consistent or unbiased.

Some on-going research along this line includes finite perturbation analysis. For example, without running the simulation again, can we answer questions like: what happen if maintenance is required for resources or what happen if a due date is added to some events? Using the LP representation, these questions might be answered by adding appropriate constraints and applying sensitivity analysis techniques.

ACKNOWLEDGMENTS

The authors wish to thank the National Science Foundation through grant DMI–0323765 for partial support of the research reported here.

REFERENCES

Cao, X. R. (1985). Convergence of parameter sensitivity estimates in a stochastic experiment. *IEEE Transactions on Automatic Control* 30(9): 845-853.

Cao, X. R. and H.-F. Chen (1997). Perturbation realization, potentials, and sensitivity analysis of Markov processes. *Automatic Control, IEEE Transactions on* 42(10): 1382-1393.

Chan, W. K. V. (2005). *Mathematical Representations of Discrete-Event System Dynamics*. Ph.D. dissertation, University of California, Berkeley.

Chan, W. K. V. and L. W. Schruben (2003). Properties of discrete event systems from their mathematical programming representations. *Proceedings of the 2003 Winter Simulation Conference (IEEE Cat. No. 03CH7499),* ed. Chick, S. E., Sanchez, P. J., Ferrin, D., and Morrice, D. J., *IEEE. Part Vol.1, 2003, pp.496-502 Vol.1. Piscataway, NJ, USA.*

Freimer, M. and L. Schruben (2001). Graphical representation of IPA estimation. *Proceeding of the 2001 Winter Simulation Conference (Cat. No.01CH37304), ed. Peters, B. A., Smith, J. S., Medeiros, D. J., Rohrer, M. W., IEEE. Part Vol.1, 2001, pp.422-7 vol.1. Piscataway, NJ, USA.*

Fu, M. and J.-Q. Hu (1997). *Conditional Monte Carlo : gradient estimation and optimization applications*, Kluwer Academic Publishers, Boston.

Fu, M. C. and J. Q. Hu (1991). Consistency of infinitesimal perturbation analysis for the *GI*/*G*/*m* queue. *European Journal of Operational Research* 54(1): 121-39.

Gal, T. (1979). *Postoptimal Analyses, Parametric Programming and Related Topics*, McGraw-Hill, London.

Glasserman, P. (1991a). *Gradient Estimation via Perturbation Analysis*, Kluwer Academic Publishers, Boston.

Glasserman, P. (1991b). Structural conditions for perturbation analysis of queuing-systems. *Journal of the Association for Computing Machinery* 38(4): 1005-1025.

Gong, W. B. and Y. C. Ho (1987). Smoothed (conditional) perturbation analysis of discrete event dynamic-systems. *Ieee Transactions on Automatic Control* 32(10): 858-866.

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

Ho, Y. C., M. A. Eyler and T. T. Chien (1979). A gradient technique for general buffer storage design in a production line. *International Journal of Production Research* 17(6): 557-580.

Homem-de-Mello, T., A. Shapiro and M. L. Spearman (1999). Finding optimal material release times using simulation-based optimization. *Management Science* 45(1): 86-102.

Lindley, D. V. (1952). The theory of queues with a single server. *Proceedings of the Cambridge Philosophical Society* 48(2): 277-289.

Schruben, L. W. (2000). Mathematical programming models of discrete event system dynamics. *2000 Winter Simulation Conference Proceedings (Cat. No. 00CH37165),* ed. Joines, J. A., Barton, R. R., Kang, K., and Fishwick, P. A., *IEEE. Part vol.1, 2000, pp.381-5 vol.1. Piscataway, NJ, USA.*

Suri, R. (1987). Infinitesimal perturbation analysis for general discrete event systems. *Journal of the ACM* 34(3): 686-717.

Suri, R. (1989). Perturbation analysis - the state of the art and research issues explained Via the *GI*/*G*/1 Queue. *Proceedings of the IEEE* 77(1): 114-137.

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

AUTHOR BIOGRAPHIES

WAI KIN (VICTOR) CHAN is an Assistant Professor of the Department of Decision Sciences and Engineering Systems at Rensselaer Polytechnic Institute. He received a bachelor's degree and a master's degree in electrical engineering from, respectively, Shanghai Jiao Tong University, China in 1997, and Tsinghua University, China in 2000, and a M.S. and Ph.D. degrees in industrial engineering and operations research from University of California, Berkeley in 2001 and 2005, respectively. His research interests include stochastic modeling, simulation optimization, mathematical programming, sensitivity analysis, applied statistical analysis, and semi-conductor manufacturing. His e-mail address is <chanw@rpi.edu>.

LEE W. SCHRUBEN is the Chancellor's Professor of the Department of Industrial Engineering and Operations Research at the University of California, Berkeley. His research is on discrete-event simulation modeling and analysis methodologies and a variety of applications – most recently in the area of bio-production. His e-mail address is <schruben@ieor.berkeley.edu>.