Perturbation Analysis: The State of the Art and Research Issues Explained via the GI/G/1 Queue

RAJAN SURI, MEMBER, IEEE

Invited Paper

Perturbation Analysis (PA) of Discrete Event Dynamic Systems (DEDS) enables parameter sensitivities of DEDS to be obtained by observing a single sample path of the system. Since the original paper in 1979 by Ho et al. [40], the technique has evolved from a body of experimental results to a new theoretical area with a rigorous basis. In particular, consistency and efficiency of PA have been proved for certain systems, and several developments have greatly extended the domain of applicability of PA. The aim of this paper is to use a simple GI/G/1 system to give an introduction to PA and illustrate the basic theoretical issues involved in this technique. After this we briefly cover the application of PA to networks of queues, and then discuss some of the recent extensions to PA. It is shown that many interesting open questions remain for PA, and areas for research are indicated. The paper is written for a broad audience and no prior knowledge of discrete event systems or queueing theory is assumed. It is also hoped that this paper will be suitable for introducing students and reseachers to the concepts of PA in a comprehensive and unified way.

I. Introduction

> To find out what happens to a system when you interfere with it you have to interfere with it (not just passively observe it)—G. E. P. Box [3].

This quote by a world-renowned statistician is sound advice for most experimenters, and based on many "abuses" of regression analysis [3]. However, the above view is often extrapolated to *all* experimental work. For example, in the simulation community, until recently, it was believed that if you observed a simulation of a communication network with link speed θ, then to predict what would happen with link speed of $\theta + \Delta\theta$, you would *have to* do another simulation. In this paper we will discuss a body of techniques that, for a certain class of systems, can deduce the outcome for the second experiment by simply "passively observing" the system during the first experiment.[1]

The importance of studying Discrete Event Dynamic Systems (DEDS) has already been stressed in several companion papers in this issue [35], [29], [21], [44], [70]. In the design, analysis, and operation of any system, whose performance (say Y) depends on the value of certain decision parameters (say θ, a vector), any information about the *gradient*, $dY/d\theta$ can be very useful to both engineers and managers. Such information tells us the sensitivity of the performance to any decision variable, and hence gives directions for improving these decisions. It provides the marginal benefits of resources and so lets us trade off between resources. Sensitivity information also alerts us to critical decisions—those that significantly impact performance, or noncritical ones—those that do not matter very much. Combined with mathematical programming methods, gradient information creates powerful tools for system optimization.

When studying a DEDS, analytical methods (e.g., flow models, Markov chains, and queueing theory) are typically the tools of first choice, since they usually provide insight and efficient analysis. However, as has been pointed out in several accompanying papers [35], [29], [52], [70], when it comes to detailed study of many of the DEDS in the modern world, we go beyond the limits of today's analytical methods. In such cases, one must resort to experimental approaches. Broadly speaking, we can classify these experimental approaches in two categories, which we shall call *computer simulation* and *physical simulation*. Since we are dealing with DEDS, the former is the domain of *discrete event simulation*, which involves Monte Carlo experimentation on a computer model. The latter includes doing experiments on a scale model of the system, or on a pro-

Manuscript received August 17, 1988; revised October 12, 1988. This work was supported in part by a Research Award from Ford Motor Company.

The author is with the Department of Industrial Engineering, University of Wisconsin, Madison, WI 53706, USA.

IEEE Log Number 8825477.

[1]We use the term "passively observing" in the same sense as in the above quote. That is to say, we may not disturb or affect the operation of the system; however, we *are* permitted to analyze any data that is available from observing the system.

0018-9219/89/0100-0114$01.00 © 1989 IEEE

totype of the system (*prototyping*), or on the actual system itself (*benchmarking*).

In this paper, we shall describe a recent methodology, called *Perturbation Analysis* (PA) of DEDS, which has the potential for being particularly efficient in obtaining gradient information whenever experimental studies are conducted on a DEDS. (All of the experimental methods above qualify for the application of the method.) Precise statements will be given later, but loosely speaking, we can say that for a class of systems:

i) gradient estimates have been obtained with significantly fewer experiments;
ii) "noise levels" in each estimate have been significantly lower; and
iii) parameter optimization algorithms have been implemented with significantly faster convergence rates.

Here "significant" does not mean just an effect that is experimentally "noticeable"—but rather, we will show examples where "significant" means *an order of magnitude or more* improvement over conventional methods. In the scientific parlance, it is reasonable to consider such an advance as a "breakthrough" rather than just a marginal improvement in the state of the art.

On the other hand, the technique of PA is very young by scientific standards, and many open problems remain. The class of DEDS for which the behavior of PA is understood is still small; much work needs to be done, both in theory and in implementation, to extend the technique to every day "real world" DEDS. Yet it is the potential of orders of magnitude increases in efficiency, as witnessed in the systems described below, that prompts us to describe the PA approach to a broader audience, in the hopes of stimulating more work in this area. Eventually, one might envision that incorporating PA into experiments on a DEDS would be as routine as differentiation is for analytical studies.

A. Overview of Paper

The aims of this paper are i) to introduce the PA ideas to a broad audience, and so it assumes no prior knowledge of PA; ii) to give an overview of the body of theoretical and experimental results available today, along with appropriate references; iii) to provide insight into the basics of PA as well as some of the resulting mathematical issues, all by means of a single-server queue example; iv) to provide an introductory, yet fairly comprehensive, paper to be used for introducing students and researchers to PA; and hopefully, if the above aims are achieved, v) to stimulate a wider set of people to develop the theory and algorithms for this still emerging area.

In order to accomplish these aims and also serve a broad audience, we have chosen to introduce PA in a particular way. The entire development of PA in Section III is done in practical terms (i.e., using "physical" arguments) with no mention of probability. The probabilistic assumptions and implications are brought in later, in Section IV and beyond. The purpose of this is to show the inherently simple and intuitive concepts behind PA. Also, concepts such as "perturbation generation" and "perturbation propagation," which are developed under more technical conditions in some papers, arise very naturally in our framework.

In the next section, we will define our example system. Section III will present the perturbation analysis of an experiment on this system, and lead to the infinitesimal PA (IPA) algorithm. Section IV will bring in specific probability structures and develop IPA for a simple queue. Section V then extends IPA to queuing networks and more general DEDS. In Section VI we show the failure of IPA in certain systems and then introduce more advanced PA methods. Section VII compares PA with other gradient estimation techniques. In Section VIII we discuss PA for discrete parameters. Finally, Section IX concludes with some general remarks and areas for research.

While this paper is not intended to be an all-encompassing survey of PA work, we do give a fairly extensive bibliography. Readers interested in pursuing the topic further should also see other recent papers (Ho [34], Suri [58], Suri and Zazanis [65], Cao [7]) which have quite up-to-date bibliographies.

II. A Simple Queueing System

Now we will describe a simple system which will serve as the main example to illustrate ideas throughout this paper. Consider a link in a communication network. The function of this link is to transmit any messages that arrive at its origin node A, to the destination node B. Messages arrive from external sources into a buffer at A. They are transmitted over the link and reach a buffer at B (see Fig. 1). For the purpose of this simple example, we will assume

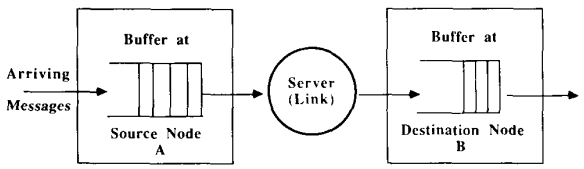

Fig. 1. Link in a communication network.

that the buffers are large enough and the link reliable enough that no messages are lost at either buffer or over the link. Also, there are no priorities among messages, so that they are transmitted using a first come, first serve (FCFS) order.

Suppose now that we are interested in the "system time" for a message, and that one of our design parameters is the link speed. (System time is defined here as the time from arrival of a message at A to the time when it is completely received at B. This includes the time spent waiting in queue at A, plus the time taken for the message to go over the link.) For clarity of analysis, we prefer to deal with the inverse of link speed as our design parameter. Thus, while link speed might be specified in number of bits transmitted per second, we will use the link service time, or amount of time to transmit a bit. We will also assume that each message has a fixed length header appended to it before transmission. Our basic design parameters are then

$$\theta = \text{link service time (s/bit)}$$

$$H = \text{header length (bits).} \tag{2.1}$$

Again for clarity of later analysis we find it convenient to define

$$\gamma = H\theta = \text{time to transmit header (s)} \tag{2.2}$$

and we can equally well use θ and γ as decision parameters instead of θ and H. Our design performance measure can now be defined precisely: it is the mean system time for a message arriving at the link when the system is in steady state.[2] Denote this mean value by $T(\theta, \gamma)$.

Now we will observe an experiment on the system. The experiment begins at a time when the system is empty (i.e., there are no messages in the buffer or in transmission). It lasts until a (predefined) number N messages complete transmission. During this experiment, let

$$L_i = \text{length of message } i \text{ (bits)}. \tag{2.3}$$

We also find it useful to define the transmission time (or "service time") for message i, which from (2.1) and (2.3) will be

$$X_i = (H + L_i)\theta \tag{2.4a}$$

$$= \gamma + L_i\theta. \tag{2.4b}$$

Note that X_i is *not* the system time—specifically, it does not account for any waiting time in the buffer. Indeed, let the system time for message i be t_i (as observed from the experiment). Then a simple estimate of the mean system time of a message, for this design, is

$$\hat{T}(\theta, \gamma, N) = \left(\frac{1}{N}\right) \sum_{i=1}^{N} t_i. \tag{2.5}$$

While more sophisticated estimates could be derived (e.g., by eliminating "transient" data), even this simple estimate has the property that (under fairly general conditions) $\hat{T}(\theta, \gamma, N)$ converges to $T(\theta, \gamma)$ as $N \to \infty$, and so it will suffice for this presentation.

Our next task is to derive an estimate of the sensitivity of the system to the two parameters, namely to estimate $dT/d\theta$ and $dT/d\gamma$. One conventional approach to estimating $dT/d\theta$ would be to do a second experiment with a link service time of $\theta + \Delta\theta$, to get a value $\hat{T}(\theta + \Delta\theta, \gamma, N)$. Then

$$\hat{F} = [\hat{T}(\theta + \Delta\theta, \gamma, N) - \hat{T}(\theta, \gamma, N)]/\Delta\theta \tag{2.6}$$

would be an estimate of this sensitivity. (Again, many refinements are possible, e.g., using a symmetric difference estimate or common random numbers in a simulation, and so on. These concepts will be defined and discussed in later sections. The main point we have to make will be apparent just from the simple form $\hat{F}$.)

We now make two basic observations about $\hat{F}$ (which extend to most of the refined estimates too). i) $\hat{F}$ involves doing an additional experiment at $\theta + \Delta\theta$. (If we prefer to get a symmetric difference estimate, *two* additional experiments would be required, one at $\theta - \Delta\theta$ and one at $\theta + \Delta\theta$.) Similarly, to estimate $dT/d\gamma$, we would require at least one additional experiment at $\gamma + \Delta\gamma$. In other words, conventional sensitivity analysis with respect to K parameters requires at least K additional experiments. ii) The choice of $\Delta\theta$ in (2.6) is difficult. Clearly if we choose $\Delta\theta$ too large, then we may not get a good approximation to the gradient. Unfortunately, if we choose $\Delta\theta$ too small, then $\hat{F}$, being the difference of two "noisy" quantities divided by a very small number, will have a very large variance (i.e., be extremely "noisy"). Thus obtaining a reasonable estimate for $dT/d\theta$ may require making N very large (details in Section VII).

While many refinements can be made to improve (somewhat) the accuracy of the above estimates, the basic problem is clear: sensitivity analysis of experiments can be tricky and require a large amount of experimental effort. In the remainder of this paper we will demonstrate how PA offers the possibility of overcoming both the above problems (i.e., repeated experiments, and noisy derivatives) for the case of DEDS.

[2]In order to avoid overly technical probabilistic statements at this stage, we will simply assume that the system does have a unique steady state and that this mean system time exists.

III. Infinitesimal Perturbation Analysis (IPA)

PA derives its properties by exploiting the structure of DEDS. In order to understand how it works, we need to study a typical experiment in some detail. Then we will introduce the IPA algorithm.

A. Detailed Evolution of an Experiment

Fig. 2 depicts a short time period starting from the beginning of the experiment. Along the top of the figure, the downward arrows depict the instants of arrival of messages (M_i denotes message i), and A_i denotes the time between arrival of M_{i-1} and M_i (A_1 is from the beginning of the experiment). Along the bottom of the figure, we see the instants of departure similarly shown. The Y axis of the graph is the number of messages present (including the one being transmitted) at the source node.

To understand the diagram in detail, let us trace through the time history shown (we will need this description to understand what follows). At time zero, the system is empty and remains so until A_1 when M_1 arrives. Since there are no other messages present, A_1 immediately starts to be transmitted with the total transmission time, as in (2.4), being X_1. At time $A_1 + A_2$, M_2 arrives and since M_1 is in transmission, it must wait. Similarly for M_3. At this point, three messages are present at the source node (see also the height of the graph). At time $A_1 + X_1$, M_1 completes transmission and departs, M_2 begins transmission, M_3 is still waiting, and the graph has a height of two. The reader should now follow the graph and interpret it for the remainder of the diagram. In particular, note that the departure of M_4 empties the system for an idle period of I_1 time units, and the arrival of M_5 terminates this idle period. The time from arrival of M_1 to departure of M_4 is called a busy period (BP) of the system.

For later use, we derive a simple relation for any BP. (For the reasons mentioned in the introduction, we will carry out the analysis in Section III with no reference to any probability distributions or assumptions. These will be brought into the analysis in Section IV.) Consider the first BP shown in Fig. 2. Let t_0 denote the starting time of this BP (here $t_0 = A_1$). The arrival time of M_1 must be t_0 (by definition) and the departure time must be $t_0 + X_1$ (since M_1 does not wait). For M_2, the arrival time is $t_0 + A_2$ and departure time is $t_0 + X_1 + X_2$. Continuing in a similar way, we find that the system time for $M_i (1 \le i \le 4)$ is

$$\left(t_0 + \sum_{j=1}^{i} X_j\right) - \left(t_0 + \sum_{j=2}^{i} A_j\right) = \sum_{j=1}^{i} X_j - \sum_{j=2}^{i} A_j \tag{3.1}$$

where the last sum is defined to be zero if $i < 2$. Finally, the total of system times for all messages in the BP is

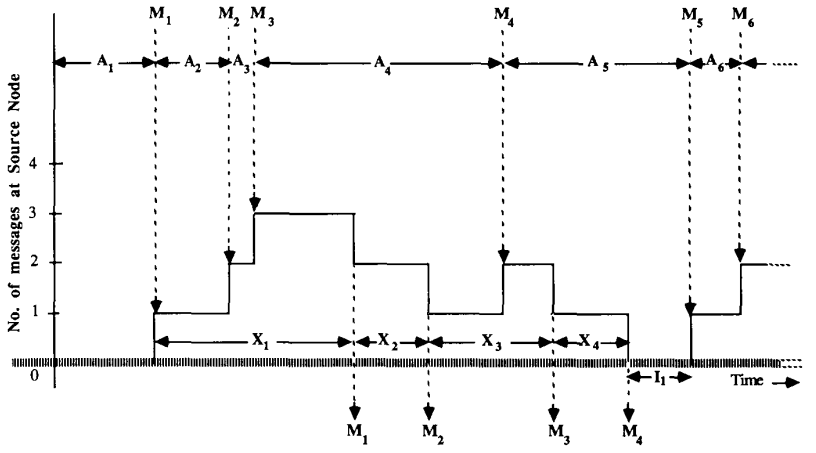

Fig. 2. Time evolution of the experiment (nominal path).

$$\sum_{i=1}^{4}\sum_{j=1}^{i} X_j - \sum_{i=1}^{4}\sum_{j=2}^{i} A_j. \tag{3.2}$$

This argument can be generalized to the mth BP (denoted BP_m) which begins with message $k_m + 1$ and lasts for n_m messages. The total of system times will then be

$$\sum_{i=1}^{n_m}\sum_{j=1}^{i} X_{k_m+j} - \sum_{i=1}^{n_m}\sum_{j=2}^{i} A_{k_m+j}. \tag{3.3}$$

Finally, suppose that the Nth message completes transmission during the Mth BP. The experiment then terminates and we use n_M to denote the number of messages completed during this last BP (which is unlike the preceding BPs since in general it will be truncated at the experiment termination time). From this experiment, the average system time of a message will be

$$\hat{T}(\theta, \gamma, N) = (1/N)\sum_{m=1}^{M}\sum_{i=1}^{n_m}\left(\sum_{j=1}^{i} X_{k_m+j} - \sum_{j=2}^{i} A_{k_m+j}\right). \tag{3.4}$$

B. Perturbation Analysis

Consider now the following question: "If the previously observed experiment had been conducted with the link service time set at $\theta + \Delta\theta$, what *would have been* the average system time of a message?" To answer this, without having to repeat the whole experiment, let us examine the sample path in detail. First note that M_i would have a transmission time of $(H + L_i)(\theta + \Delta\theta)$ or an increase in transmission time

$$\Delta X_i = (H + L_i)\Delta\theta \tag{3.5}$$

over its previous value of $X_i = (H + L_i)\theta$. We can rewrite (3.5) as

$$\Delta X_i = (\Delta\theta/\theta)X_i. \tag{3.6}$$

Consider again the sample path shown in Fig. 2. (We will call this observed sample path the *nominal* path and the one for the case of $\theta + \Delta\theta$ the *perturbed* path.) By hypothesis (i.e., by the way the above question was posed) the arrival pattern and message lengths (i.e., the A_i and L_i values) are defined to be the same for the case of $\theta + \Delta\theta$. However, M_1 would take ΔX_1 longer to leave the source node. Thus M_2 would wait ΔX_1 longer (than in the nominal) to start its transmission, and then take ΔX_2 longer in transmission, thus leaving the system a total of $\Delta X_1 + \Delta X_2$ later than in the nominal (see Fig. 3). In general, for any M_i in the first BP, the increase in system time (compared to nominal) would be

$$\Delta t_i = \sum_{j=1}^{i} \Delta X_j \tag{3.7}$$

and using (3.6) we can rewrite this as

$$\Delta t_i = (\Delta\theta/\theta)\sum_{j=1}^{i} X_j. \tag{3.8}$$

Coming to M_5, we need to proceed more carefully. At the departure of M_4 we distinguish between two possibilities. i) M_4 still departs before M_5 arrives and the system goes idle as in the nominal path—this case is shown in Fig. 3. ii) The accumulated delays are such that M_5 arrives before M_4 leaves and so the BP in the perturbed path does not terminate with M_4 but continues to M_5 and possibly beyond (see Fig. 4). We will now analyze these two cases.

In case i) even in the perturbed path M_5 arrives at an idle system and so it will experience an increase in system time of ΔX_5. Arguing as before, M_6 will experience an increase of $\Delta X_5 + \Delta X_6$. In general then, message $k_m + 1$ found the system idle in the nominal path, and if it finds the system idle in the perturbed path, then for message $k_m + 1$ through $k_m + n_m$ (i.e., those that formed BP_m in the nominal) the increase in system time would be

$$\Delta t_{k_m+i} = (\Delta\theta/\theta)\sum_{j=1}^{i} X_{k_m+j}, \quad i \le n_m. \tag{3.9}$$

Next, in case ii), let ΔS_1 be the amount of time by which BP_1 extends beyond the arrival of M_5 (see Fig. 4). Then M_5 will wait ΔS_1 before starting service, and an additional ΔX_5 in transmission, so that

$$\Delta t_5 = \Delta S_1 + \Delta X_5 \tag{3.10}$$

and arguing as before,

$$\Delta t_6 = \Delta S_1 + \Delta X_5 + \Delta X_6 \tag{3.11}$$

and indeed,

$$\Delta t_i = \Delta S_1 + (\Delta\theta/\theta)\sum_{j=1}^{i} X_j \tag{3.12}$$

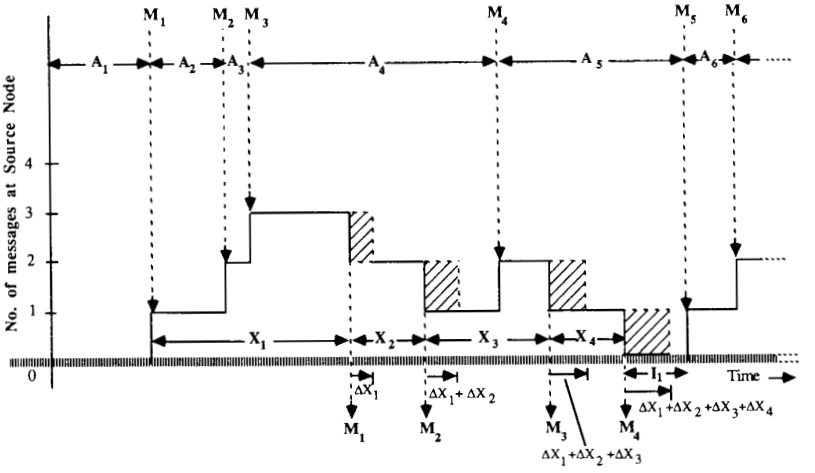

Fig. 3. Perturbations in the sample path for case i).

for any M_i that belonged to BP_2 in the nominal path. To generalize this, let $\Delta S_{m-1}(\Delta\theta)$ be the amount by which BP_{m-1} extends beyond the arrival of message $k_m + 1$. Then

$$\Delta t_{k_m+i} = \Delta S_{m-1}(\Delta\theta) + (\Delta\theta/\theta) \sum_{j=1}^{i} X_{k_m+j}, \quad i \le n_m. \tag{3.13}$$

Here we explicitly indicate the *dependence* of ΔS_{m-1} on the size of the change $\Delta\theta$, as we will need to discuss this below. Also note that the formula above is intended to *include* the case where BP_{m-2} may have "pushed" BP_m. The value of $\Delta S_{m-1}(\Delta\theta)$ included all effects of prior busy periods up through BP_{m-1}. In fact, we shall now let (3.13) include case i) as well, by simply defining $\Delta S_{m-1}(\Delta\theta)$ to be zero in that case.

Putting these arguments together, we can write an expression for the *change* in average system time of a message, due to the change $\Delta\theta$,

$$\Delta\hat{T}(\theta, \gamma, N; \Delta\theta) = (1/N) \sum_{m=1}^{M} \sum_{i=1}^{n_m} \cdot \left| \Delta S_{m-1}(\Delta\theta) + (\Delta\theta/\theta) \sum_{j=1}^{i} X_{k_m+j} \right|. \tag{3.14}$$

Note that in the perturbed path the pattern of BPs may be quite different. However, for this set of N messages, (3.14) is *exactly* the change in the average system time—no approximations have been made as yet.

Define the following set of BPs

$$B(\Delta\theta) = \{m | \Delta S_{m-1}(\Delta\theta) > 0\} \tag{3.15}$$

that is, the set of BPs that "get pushed" by the perturbation. Again, this set depends on the size of $\Delta\theta$. Then (3.14) can be written

$$\Delta\hat{T}(N; \Delta\theta) = (1/N) \left| (\Delta\theta/\theta) \sum_{m=1}^{M} \sum_{i=1}^{n_m} \sum_{j=1}^{i} X_{k_m+j} + \sum_{m \in B(\Delta\theta)} n_m \Delta S_{m-1}(\Delta\theta) \right| \tag{3.16}$$

where, for simplicity of notation, we have removed the arguments (θ, γ) from $\Delta\hat{T}$ on the left-hand side (LHS).

Consider the following quantity, which occurs within the first summation in (3.16):

$$H_{ni} = \sum_{i=1}^{n_m} \sum_{j=1}^{i} X_{k_m \cdot j}. \tag{3.17}$$

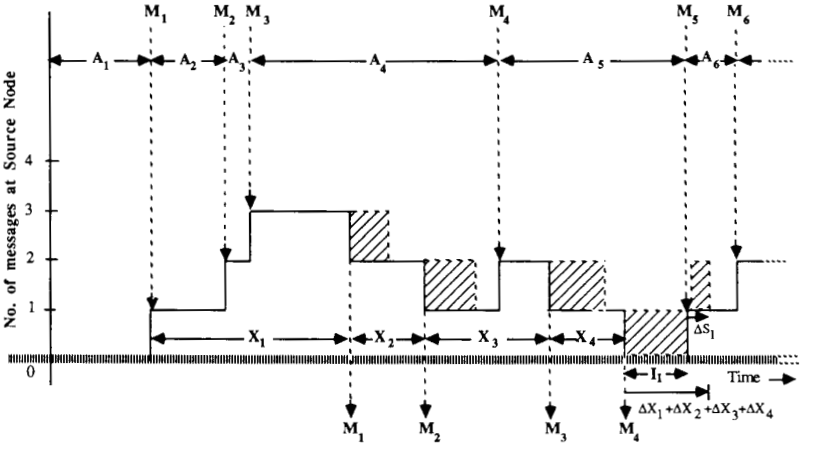

Fig. 4. Perturbations in the sample path for case ii).

This can be seen to be a well-defined function of a given BP. We will use this below.

Let us return to some basics for a moment. In the following, assume that we are given a sufficiently long sequence of interarrival times $\boldsymbol{A} = \{A_1, A_2, \cdots\}$ and message lengths $\boldsymbol{L} = \{L_1, L_2, \cdots\}$, and that all experiments (for whatever values of θ, γ, N) are performed with *this* set of inputs $\boldsymbol{A}$, $\boldsymbol{L}$. There is nothing *statistically* wrong with this procedure if it is physically possible to repeat the experiments this way, provided that the property $\lim_{N\to\infty} \hat{T}(\theta, \gamma, N) = T(\theta, \gamma)$ holds over the sequence $\boldsymbol{A}, \boldsymbol{L}$ for any of the chosen θ, γ values. Indeed, in simulation where inputs can be controlled, this experimental design is called common random numbers (CRN) and often used with effect to reduce computation time [24], [54]. With this experimental framework, the quantity $[\hat{T}(\theta + \Delta\theta, \gamma, N) - \hat{T}(\theta, \gamma, N)]/\Delta\theta$ is the CRN finite difference estimate of the gradient. The way that PA poses its hypothetical question ("what *would* have happened . . .", see beginning of Section III-B) is clearly similar to the CRN approach. (However, the limiting properties are different as will be seen below.)

Now in terms of our objective we are interested in estimating the sensitivity of T to θ, or formally, in the value

$$dT/d\theta = \lim_{\Delta\theta\to 0} [T(\theta + \Delta\theta, \gamma) - T(\theta, \gamma)]/\Delta\theta. \tag{3.18}$$

To relate the right-hand side (RHS) to experimental values $\hat{T}$, we write it as

$$dT/d\theta = \lim_{\Delta\theta\to 0} \lim_{N\to\infty} [\hat{T}(\theta + \Delta\theta, \gamma, N) - \hat{T}(\theta, \gamma, N)]/\Delta\theta \tag{3.19}$$

which can also (by our definition of $\Delta\hat{T}$) be written

$$dT/d\theta = \lim_{\Delta\theta\to 0} \lim_{N\to\infty} \Delta\hat{T}(N; \Delta\theta)/\Delta\theta. \tag{3.20}$$

Now we will introduce the following assumption which will be discussed in detail later.

Assumption (A1): The system being studied has the property that

$$\lim_{\Delta\theta\to 0} \lim_{N\to\infty} (1/N) \sum_{m\in B(\Delta\theta)} n_m \Delta S_{m-1}(\Delta\theta) = 0.$$

Under (A1), and from (3.20), (3.17), and (3.16) we get

$$\lim_{\Delta\theta\to 0} \lim_{N\to\infty} (1/N) \sum_{m=1}^{M} H_m/\theta = dT/d\theta. \tag{3.21}$$

However, $\Delta\theta$ has "canceled out" of the LHS, so we can remove the first limit! Thus

$$\lim_{N\to\infty} (1/N) \sum_{m=1}^{M} H_m/\theta = dT/d\theta \tag{3.22}$$

for systems which satisfy (A1). On the LHS we have a quantity which i) is defined completely by observing the nominal sample path alone—the H_m values are easily computed (see below) while the nominal experiment is evolving; ii) converges to the true gradient $dT/d\theta$ as N goes to ∞. This suggests that the following might be an experimental estimate of the gradient with respect to θ from the nominal experiment alone:

$$\hat{g}_\theta(N) = \left(\sum_{m=1}^{M} H_m\right)\Big/(N\theta). \tag{3.23}$$

Computation of $\hat{g}_\theta(N)$ is extremely easy as seen in Algorithm 1.

Algorithm 1: Estimation of $dT/d\theta$ during one experiment on a communication link.

0) Initialize: J ← 0; XSUM ← 0; HSUM ← 0; THETA ← θ
1) Update: At departure of next message (with service time observed to be XJ),
   1.1) J ← J+1
   1.2) XSUM ← XSUM + XJ
   1.3) HSUM ← HSUM + XSUM
   1.4) *If* link is now idle *then* XSUM ← 0
2) Test: *If* J = N *then* go to OUTPUT *else* go to UPDATE
3) Output: The IPA estimate of the gradient is HSUM/(N*THETA)

Lastly, we consider the estimation of $dT/d\gamma$. Note that

$$X_i = \gamma + L_i\theta \tag{3.24}$$

so that with $\gamma + \Delta\gamma$ we would have an increase in each X_i value of

$$\Delta X_i = \Delta\gamma. \tag{3.25}$$

In this case, the analog of (3.7) and (3.8) would be

$$\Delta t_i = \sum_{j=1}^{i} \Delta\gamma = \Delta\gamma \sum_{j=1}^{i} 1. \tag{3.26}$$

While we could write this as $\Delta\gamma i$, we leave it in the above form to show that for this case the estimation of the gradient just involves accumulating 1's. The corresponding gradient estimate is

$$\hat{g}_\gamma(N) = \left(\sum_{m=1}^{M} \sum_{j=1}^{n_m} \sum_{j=1}^{i} 1\right)\Big/N. \tag{3.27}$$

This estimate can be computed at the same time as the previous one: Algorithm 2 shows simultaneous estimation of $dT/d\theta$ and $dT/d\gamma$ while observing a single experiment. (Note that for (3.27) to be a good estimate a property similar to (A1) would have to be satisfied with respect to the γ parameter.)

Algorithm 2: Simultaneous estimation of $dT/d\theta$ and $dT/d\gamma$ during one experiment.

0) Initialize: J ← 0; XSUM ← 0; JSUM ← 0; HSUM ← 0; GSUM ← 0; THETA ← θ
1) Update: At departure of next message (with service time observed to be XJ);
   1.1) J + 1
   1.2a) XSUM ← XSUM + XJ
   1.2b) JSUM ← JSUM + 1
   1.3a) HSUM ← HSUM + XSUM
   1.3b) GSUM ← GSUM + JSUM
   1.4) *If* link is now idle *then* XSUM ← 0 and JSUM ← 0
2) Test: *If* J = N *then* go to OUTPUT *else* go to UPDATE
3) Output: The IPA estimate of $dT/d\theta$ is HSUM/(N*THETA)
   The IPA estimate of $dT/d\gamma$ is GSUM/N

The two estimates above, $\hat{g}_\theta(N)$ and $\hat{g}_\gamma(N)$ are infinitesimal PA (IPA) estimates and as can be seen from Algorithm 2 their computation is quite simple. We conclude by motivating the terminology "infinitesimal." Consider again the sample path in Fig. 2. We will argue informally, but a precise statement of assumptions and a formal derivation (for a general DEDS) can be found in Suri [58]. For a given experiment (which has been observed) there must exist an $\epsilon > 0$ such that $I_m > \epsilon$ for all m (I_m is the "idle time" between BPs, and by definition BPs must be separated). Now let

$$Y_m = \sum_{j=1}^{n_m} X_{k_m+j}, \quad \text{and } K = \max_m Y_m. \tag{3.28}$$

For the case where θ is changed, the amount a BP is lengthened in the perturbed path is $Y_m \Delta\theta/\theta$ (provided that no BPs coalesce). Now choose $\delta < \epsilon\theta/K$. Then for all $\Delta\theta < \delta$ we have

$$Y_m \Delta\theta/\theta \leq K\Delta\theta/\theta < K\delta/\theta < \epsilon < I_m. \tag{3.29}$$

So that indeed *no* BPs will coalesce for such $\Delta\theta$. Hence $\Delta S_m(\Delta\theta)$ will be zero for all m and from (3.16) and (3.23) we see that

$$\hat{g}_\theta(N) = \Delta\hat{T}(N; \Delta\theta)/\Delta\theta, \quad \text{for } \Delta\theta < \delta. \tag{3.30}$$

In other words, the IPA estimate equals the finite difference estimator for all $\Delta\theta$ "sufficiently small."

At this point one might be tempted to make the following (erroneous) argument: "Since the gradient can be obtained as the limit (as $\Delta\theta \to 0$) of CRN finite difference estimates, and since (after $\Delta\theta$ drops below δ) the IPA estimate equals the CRN finite difference estimate, so the IPA estimate must behave like the CRN finite difference estimate. In particular, as we use larger values of N, the IPA estimate will converge to $dT/d\theta$." The fallacy in this argument is to do with the subtleties of changing the order of two limits. The correct definition of the gradient, as seen in (3.19), involves letting N go to infinity first, and then letting $\Delta\theta$ go to zero. In the "erroneous argument," we have chosen a finite N, and then let $\Delta\theta \to 0$ for this fixed N. *After that*, we have let $N \to \infty$. So we have reversed the order of the limits required in the definition (3.19). If we try to fix this argument by letting $N \to \infty$ first, we encounter a different problem. As more and more BPs are observed, it is likely that there will be *some* observations of smaller and smaller values of I_m (this argument can be rigorously stated, see Section IV). Thus as $N \to \infty$ the value of ϵ (above) may become arbitrarily small. The corresponding value of δ above must then also become arbitrarily small, and this is the origin of the "infinitesimal" terminology. In the limit, there may be *no* $\delta > 0$ that satisfies the above requirements, so that we cannot generate a limiting sequence in $\Delta\theta$ for which (3.30) holds.

On the other hand, changing the order of limits is not necessarily wrong: often it may be admissible, and indeed, in many other analysis situations it is used with effect. So the key question here is: when is the change of limits admissible? *Whenever it is, we have the possibility of simple and efficient estimation of certain types of parameter sensitivities.*

We consolidate the ideas in this section by the following statement, made with respect to the θ parameter (an analogous one could be made for γ). For clarity, we add the argument θ_0 to ΔS_{m-1} below, to indicate the nominal parameter at which ΔS_{m-1} is calculated.

Proposition 3.1: For a single server queueing system, let θ^{-1} be the speed of the server. For an experiment on the system, let $\hat{T}(\theta, N)$ be the average system time observed over N customers, and define $T(\theta) = \lim_{N\to\infty} \hat{T}(\theta, N)$. If

i) $T(\theta)$ and $dT/d\theta$ exist in a neighborhood of θ_0
ii) $\lim_{\Delta\theta\to 0} \lim_{N\to\infty} (1/N) \sum_{m=1}^{M} n_m \Delta S_{m-1}(\theta_0, \Delta\theta) = 0$

then the IPA estimate $\hat{g}_\theta(N)$ generated by Algorithm 1 has the property that at $\theta = \theta_0$

$$\lim_{N\to\infty} \hat{g}_\theta(N) = dT/d\theta$$

(in other words Algorithm 1 provides a consistent estimate of the gradient while observing only one experiment).

The above proposition is stated so as to show the dependence of IPA's behavior on the limiting behavior of the ΔS_{m-1} quantity. However, we can also write the condition for consistency of IPA in a "raw" form as the exchange of two limits. From the discussion surrounding (3.30), we see that the LHS of (3.22) can be written as

$$\lim_{N\to\infty} \lim_{\Delta\theta\to 0} \frac{\Delta\hat{T}(N; \Delta\theta)}{\Delta\theta} \tag{3.31}$$

which is the value to which the IPA estimate will converge. The value $dT/d\theta$ on the other hand, is defined by (3.20). Thus we have this alternative statement.

Corollary 3.1: For the system in Proposition 3.1, IPA will provide a consistent estimate of the gradient if the system satisfies

$$\lim_{N\to\infty} \lim_{\Delta\theta\to 0} \frac{\Delta\hat{T}(N; \Delta\theta)}{\Delta\theta} = \lim_{\Delta\theta\to 0} \lim_{N\to\infty} \frac{\Delta\hat{T}(N; \Delta\theta)}{\Delta\theta}. \tag{3.32}$$

As already mentioned, the description in this section has proceeded with no reference to any assumptions about the probabilistic structure of the system. This is entirely intentional since PA is based on sample-path arguments and the intuitive ideas about it do not require specific probabilistic structures. However, in order to see whether a particular PA algorithm will work for a given system, we need to bring in specific assumptions for that system. Thus the natural place for the probability to be brought in is after the basic sample path arguments. As an example, for IPA applied to a single server queue, the critical question is: for what classes of queues do the conditions i) and ii) in Proposition 3.1 hold? This and related questions will be the subject of the next section.

IV. IPA for a GI/G/1 Queue

We now study the IPA algorithm of the preceding section applied to a GI/G/1 queue.[3] The development in this section is based on Suri and Zazanis [65] and Zazanis and Suri [68]. Although no knowledge of queueing theory is presumed, this section requires more familiarity with probability than the earlier part of the paper.

A. Development of Algorithm

A sample path for this queue is obtained by specifying two sequences of i.i.d. (independent and identically dis-

[3]This notation means the interarrival times have a general distribution but are independent of each other (GI), the service times have a general distribution (G) and there is one server (1)—see for example Kleinrock [45] for further explanation of such notation.

tributed) random variables: $\{A_1, A_2, A_3, \cdots\}$ is the sequence of interarrival times and $\{X_1, X_2, X_3, \cdots\}$ the sequence of service times (see Fig. 2). We assume that the r.v. (random variable) X_i depends on a parameter θ, the nature of this dependence being specified later. We are interested in the mean system time of a customer, in steady state, which we denote $T(\theta)$, and we are also interested in the sensitivity of this to θ, i.e., $dT(\theta)/d\theta$. Let $t_i(\theta)$ be the system time of the ith customer (as explained in Sections II and III). We will also assume that the system is stable, i.e., $EX_i < EA_i$. These assumptions imply that the system is ergodic and

$$\lim_{N\to\infty} (1/N) \sum_{i=1}^{N} t_i = T(\theta) \quad \text{a.s.} \tag{4.1}$$

(a.s. denotes "almost surely" or with probability 1). This equation means that, in practical terms, we can estimate the value of $T(\theta)$ by a sufficiently long experiment. To see this, let

$$\hat{T}(\theta, N) = (1/N) \sum_{i=1}^{N} t_i(\theta). \tag{4.2}$$

Then $\hat{T}(\theta, N)$ satisfies

$$\lim_{N\to\infty} \hat{T}(\theta, N) = T(\theta) \quad \text{a.s.} \tag{4.3}$$

or in experimental terms, the longer the experiment, the more accurately $\hat{T}$ estimates T, so that $\hat{T}$ can be considered a reasonable experimental estimator for T. Equation (3.4) shows the explicit dependence of $\hat{T}$ on the A_i, X_i, and N values.

Now we develop the IPA algorithm for this system. Let

$$\Delta X_i(\theta, \Delta\theta) = X_i(\theta + \Delta\theta) - X_i(\theta) \tag{4.4}$$

be the change in service time due to a change $\Delta\theta$ in the parameter. Similarly let $\Delta t_i(\theta, \Delta\theta)$ denote the corresponding change in system time. Then from the arguments in Section III, specifically those leading up to (3.13) we have

$$\Delta t_{k_m+i} = \Delta S_{m-1} + \sum_{j=1}^{i} \Delta X_{k_m+j} \quad i \le n_m. \tag{4.5}$$

(We have dropped the arguments $(\theta, \Delta\theta)$ in Δt_i and ΔS_{m-1} to simplify notation.) Analogous to (3.14) then we also have

$$\Delta\hat{T}(\theta, N; \Delta\theta) = (1/N) \sum_{m=1}^{M} \sum_{i=1}^{n_m} \left[\Delta S_{m-1} + \sum_{j=1}^{i} \Delta X_{k_m+j}\right]. \tag{4.6}$$

Now define, in the usual way

$$dX_i/d\theta = \lim_{\Delta\theta\to 0} [X_i(\theta + \Delta\theta) - X_i(\theta)]/\Delta\theta = \lim_{\Delta\theta\to 0} \Delta X_i/\Delta\theta \tag{4.7}$$

which limit we assume, for the moment, exists. We will also need a minor re-statement of assumption (A1) in probabilistic terms as follows.

Assumption (A1'): The system being studied has the property that (A1) holds a.s.

In addition, we also need some technical conditions on the service time r.v.'s. One possible approach is to use the concept of "uniformly differentiable a.s." as defined by Cao [4].

Assumption (A2): The r.v.'s $X_i(\theta)$ are uniformly differentiable a.s. at θ.

Readers not familiar with this concept may regard it simply as a condition that ensures that the $dX_i/d\theta$ values are sufficiently well-behaved for almost all outcomes of X_i. With these assumptions,

$$\begin{aligned} dT/d\theta &= \lim_{\Delta\theta\to 0} \lim_{N\to\infty} \Delta\hat{T}(\theta, N; \Delta\theta)/\Delta\theta \quad \text{a.s.} \\ &= \lim_{\Delta\theta\to 0} \lim_{N\to\infty} (1/N) \sum_{m=1}^{M} \sum_{i=1}^{n_m} \sum_{j=1}^{i} \Delta X_{k_m+j}/\Delta\theta \\ &= \lim_{N\to\infty} (1/N) \sum_{m=1}^{M} \sum_{i=1}^{n_m} \sum_{j=1}^{i} dX_{k_m+j}/d\theta. \end{aligned} \tag{4.8}$$

Here, in the second step, the ΔS_{m-1} term dropped out because of (A1). In the last step, it is easily verified that the uniform differentiability condition (A2) enables the limit with respect to $\Delta\theta$ to be performed first. (This was pointed out to the author by Chen [19].) Now if we define

$$h_m = \sum_{i=1}^{n_m} \sum_{j=1}^{i} dX_{k_m+j}/d\theta \tag{4.9}$$

we have

$$\lim_{N\to\infty} (1/N) \sum_{m=1}^{M} h_m = dT/d\theta \quad \text{a.s.} \tag{4.10}$$

so that the value

$$\hat{g}(N) = (1/N) \sum_{m=1}^{M} h_m \tag{4.11}$$

is a reasonable estimator of the gradient of T with repect to θ. All we need to do, in order to make $\hat{g}$ computable from a single sample path, is to discuss how $dX_i/d\theta$ may be estimated from the observation X_i; we do this next.

B. Sample Gradient of a Random Variable

A concept that is central to the PA approach is determining the effect on the value of a r.v. when one of its parameters is changed. In order for PA to be performed correctly, it is necessary to formalize some dependencies that are not explicitly considered in other treatments of DEDS. Specifically, we shall see how $dX_i/d\theta$ can be expressed in terms of X_i and θ. This section is based on the Appendix of Suri [58]. However the remarks here will be brief and informal; the reader should see [58] for details.

In engineering systems the parameterization of an r.v. usually models some physical relationship in a system. In such cases, formalizing the dependency is clear. For instance, in the communication network model of Section II, we had, from the physics of the system, that $X_i = \theta Y_i$ where $Y_i = H + L_i$, see (2.4). In such cases, θ is said to be a *scale* parameter of the distribution of X_i, and by straightforward differentiation we get $dX_i/d\theta = Y_i$ which can be written

$$dX_i/d\theta = X_i/\theta \tag{4.12}$$

a relationship that is true whenever θ is a scale parameter.

As a second common situation, let $X_i = \theta + Y_i$ in which case θ is called a *location* parameter. (In the communication network example, γ was a location parameter of X_i, see (3.24).) Again by differentiation we get

$$dX_i/d\theta = 1 \tag{4.13}$$

if θ is a location parameter.

What (4.12) and (4.13) mean is that the value of $dX_i/d\theta$ is "observable" from the values of X_i and θ (trivially so in the

location parameter case), so that a second experiment at $\theta + \Delta\theta$ is not required to estimate $dX_i/d\theta$. (In fact, one can see the heart of the idea of IPA right here! For these cases, the gradient of an r.v. can be estimated just from an observation on the r.v. However this is just IPA for a r.v.; IPA for a system requires determining how all these $dX_i/d\theta$ values interact through the system and affect the final performance.)

While location and scale parameters can be used to parameterize quite general distributions, several commonly used distributions are described by such parameters. A few examples are: θ is a scale parameter of the exponential distribution with mean θ; it is a location parameter of any normal distribution with mean θ and any uniform distribution with mean θ.

For many of the common distributions, even if a parameter is not immediately a location or scale parameter, it can be made so by an appropriate transformation. A rather simple example is where an exponential distribution is characterized by its rate λ. In this case the parameterization $\theta = 1/\lambda$ makes θ a scale parameter. Another common case is the normal distribution. Let the r.v. X have the distribution $N(\mu, \sigma^2)$. Then the r.v. $Y = X - \mu$ will have σ as a scale parameter. So $dY/d\sigma = Y/\sigma$. Also, $X = Y + \mu$ so that $dX/d\sigma = dY/d\sigma = Y/\sigma = (X - \mu)/\sigma$. The final expression gives the derivative in terms of X and its parameters, as desired. Similarly, if X is uniformly distributed in $[\mu - \delta, \mu + \delta]$ we can derive $dX/d\delta = (X - \mu)/\delta$.

In more general cases, it is possible to derive an expression for $dX/d\theta$ using results from [58]. As examples, if $F(x; \theta)$ is the cumulative distribution function of $X(\theta)$ then under certain conditions elaborated in [58]

$$\frac{dX}{d\theta} = \lim_{\Delta\theta \to 0} \frac{F^{-1}(F(X; \theta); \theta + \Delta\theta) - X}{\Delta\theta}. \tag{4.14}$$

Under those conditions the limit exists and the RHS can be seen to depend only on X and θ. Under additional conditions on the smoothness of F, we can write

$$dX/d\theta = -(\partial F/\partial\theta)/(\partial F/\partial x) \tag{4.15}$$

where the RHS is evaluated at X, θ. Again this gives a straightforward way of expressing $dX/d\theta$ in terms of X and θ. The purpose of this section is to justify the following assumption.

Assumption (A3): The r.v.'s $X_i(\theta)$ have the property that $dX_i/d\theta$ can be expressed as $\psi(X_i, \theta)$.

In terms of implementing the IPA algorithm, this assumption assures us that $dX_i/d\theta$ can be observed on the nominal sample path. (In the case of simulation, there are some other implications too, see [57], [58].) There is another property here that may be useful in practical implementations. In many instances, the actual form of the distribution function $F(x; \theta)$ for the r.v. $X_i(\theta)$ *need not be known*. For instance, if θ is a scale parameter (which we might know from the physical properties, e.g., for the communication network), then $dX_i/d\theta = X_i/\theta$, and by implementing IPA we can do a sensitivity analysis *without having to assume any distribution* for X_i. (In fact, even if the assumption that θ is a scale parameter is invalid, PA may still provide reasonable gradient estimates [13].) In the case of a location parameter, $dX_i/d\theta = 1$ and we do not even need to know the value of θ in order to do the sensitivity analysis! Readers interested further in this subject of differentiability of a r.v. should see the ideas in [58], [65], [57] for the PA view, and also a recent report by Glynn [28].

C. Statement of Algorithm

Assumption (A3), along with (4.9) and (4.11) leads to Algorithm 3 which is an IPA algorithm for estimating $dT/d\theta$ for a GI/G/1 queue from a single experiment. To understand Algorithm 3, simply replace Step 1.2) in Algorithm 1 with the more general Step 1.2) seen in Algorithm 3, where PSI($\cdot$, θ) is a function that returns the value $\psi(\cdot, \theta)$.

Algorithm 3: Estimation of $dT/d\theta$ from single sample path of GI/G/1 queue.

0) Initialize: J $\leftarrow$ 0; XSUM $\leftarrow$ 0; HSUM $\leftarrow$ 0
1) Update: At departure of next customer (with service time observed to be XJ):
   1.1) J $\leftarrow$ J + 1
   1.2) XSUM $\leftarrow$ XSUM + PSI(XJ, THETA)
   1.3) HSUM $\leftarrow$ HSUM + XSUM
   1.4) *If* server is now idle *then* XSUM $\leftarrow$ 0
2) Test: *If* J = N *then* go to OUTPUT *else* go to UPDATE
3) Output: The IPA estimate of the gradient is HSUM/N

Readers familiar with experimental approaches such as discrete event simulation will immediately see possibilities for reduction of bias in the above estimate (e.g., by eliminating the initial transient). Since the estimates obtained by PA are just some functions of experimental observations, the usual statistical techniques can be applied to get confidence intervals for the gradient or to reduce initial bias. These include independent replications, batch means, and regenerative methods [24], [54], [22]. Use of a regenerative technique is particularly appropriate with any PA algorithm since regenerative methods develop interval estimates using only one sample path. Indeed Suri and Zazanis [65] present several numerical examples showing gradient estimates, along with 95-percent confidence intervals, obtained from a single sample path by using an IPA algorithm along with regenerative techniques.

D. Consistency of IPA for the GI/G/1 Queue

We will consider now, more precisely than before, the conditions under which the IPA algorithm provides "good" estimates, in a sense to be defined here. Following the terminology of statistics we say the estimate $\hat{g}(N)$ in (4.11) is *strongly consistent* if

$$\lim_{N \to \infty} \hat{g}(N) = dT/d\theta \quad \text{a.s.} \tag{4.16}$$

and we say it is *asymptotically unbiased* if

$$\lim_{N \to \infty} E\hat{g}(N) = dT/d\theta. \tag{4.17}$$

These are the minimal properties that we should expect for a good estimator since (speaking in informal terms here) the former ensures that the variance of the estimator decreases with the length of the observation and the latter ensures a similar decrease in bias.

One way to study the properties of the IPA estimate would be to investigate the conditions under which (A1') holds. An alternative, direct way is to look at $\hat{g}(N)$ itself. After all,

$\hat{g}(N)$ is a function of the X_i values, albeit a complicated function. As a first step we might look at analytically tractable systems and see if for such systems the properties of $\hat{g}(N)$ can also be deduced. This approach was taken by Suri and Zazanis [65] for the M/G/1 queue. The interarrival times for this system are exponentially distributed. In [65] it was proved that, for a fairly general class of service time distributions, the IPA estimate $\hat{g}(N)$ is indeed strongly consistent and asymptotically unbiased. An IPA estimate of the gradient with respect to the rate parameter of the arrival distribution was also derived in [65] and this too was shown to be strongly consistent and asymptotically unbiased.

While the proof in [65] is too involved to discuss here, we will discuss a simpler case. A corollary given in [65] is that, for a customer entering an M/G/1 queue in steady state, the IPA value of the gradient of *this customer's system time* is an unbiased estimate of $dT/d\theta$. We will prove this for a simpler system, the M/M/1 queue. The proof here is based on a more general approach developed by Zazanis [69]. However, we use a simplified version of his proof as we need only a subset of his results.

An M/M/1 queue is described by exponentially distributed interarrival times with rate λ (i.e., mean $1/\lambda$), and by exponentially distributed service times with mean θ. The *traffic intensity* ρ for this system is given by $\rho = \lambda\theta$. We assume $\rho < 1$. Let B be the r.v. for the duration of an arbitrary BP. Some standard results from queueing theory are (see e.g., [45]): $T(\theta) = \theta/(1 - \rho)$, $EB = \theta/(1 - \rho)$, and $EB^2 = 2\theta^2/(1 - \rho)^3$. Differentiation gives $dT/d\theta = 1/(1 - \rho)^2$, the "true value" of the gradient. Also, since θ is a scale parameter of X_i, $dX_i/d\theta = X_i/\theta$.

Now consider a customer arriving at this queue in steady state, and let this customer be the jth customer in this BP. Then the IPA value of the gradient for this customer is (see arguments in Section III)

$$g = \sum_{i=1}^{j} dX_i/d\theta = (1/\theta) \sum_{i=1}^{j} X_i. \tag{4.18}$$

Now consider the term $\Sigma_{i=1}^{j} X_i$. From Fig. 2, we see that this is the total time since the beginning of the BP until the departure of customer j (e.g., try $j = 3$ and look at M_3). From the point of view of customer j we can decompose this total time into z_j, the age of the BP when customer j arrived, plus t_j, the system time of customer j. Thus

$$g = (z_j + t_j)/\theta. \tag{4.19}$$

We wish to evaluate Eg and see if it equals $dT/d\theta$. We can write

$$Eg = (Ez_j + Et_j)/\theta. \tag{4.20}$$

Now Et_j is simply the expected system time of an arbitrary customer, so it just equals $T(\theta)$. Next consider Ez_j. When this customer arrived at the queue, it must have found the server either idle or busy: denote these two possible events by I and b (respectively), and their probabilities by p_I and p_b. Then

$$Ez_j = E[z_j|I]p_I + E[z_j|b]p_b. \tag{4.21}$$

If the server was idle, z_j is identically zero, so we only need to consider the final term above. Since exponential arrivals take a "random" look at the system (see [45]), p_b is just the utilization of the server, which is ρ, and $E[z_j|b]$ is the average age of a BP seen by a random arrival into the BP. From a standard result in renewal theory [45] this average age is $EB^2/2EB$. Thus

$$Ez_j = \rho EB^2/2EB. \tag{4.22}$$

From these arguments and (4.20) we have

$$Eg = [\rho EB^2/2EB + T(\theta)]/\theta. \tag{4.23}$$

Substituting values for the M/M/1 queue in the RHS we get

$$\begin{aligned} Eg &= [\rho\theta/(1 - \rho)^2 + \theta/(1 - \rho)]/\theta \\ &= 1/(1 - \rho)^2 \end{aligned} \tag{4.24}$$

which indeed is the value of $dT/d\theta$ given above, and so the unbiasedness is proved. A more sophisticated extension of this argument is used by Zazanis [69] to consider IPA for higher moments of the system time.

Although the "direct" proof above, and the more general one for the M/G/1 queue in [65], are reassuring that IPA gives good estimates, these proofs do not give any insight as to "why" IPA works. Also, the whole purpose of PA is to tackle systems that are *not* analytically tractable and which require experimental methods. Nevertheless the results for simple systems have their place. First, if we are to prove consistency of PA algorithms for general DEDS we should *at least* be able to show it for some simple systems, so such proofs can be considered a first step in this direction. Second, from a historical perspective, the early PA papers encountered criticism that the algorithms might not converge to the true gradient values. At that time only experimental results were available and these were not considered to be sufficient "proof." Hence the first analytical results on simple queues and simple queueing networks (e.g., [36], [65], [7]) did much to convince the research community that IPA *could* work at all. The attention has now shifted to understanding *why it works when it does, and for what class of systems IPA will* work. We will now cover these questions.

We will illustrate why IPA works for the M/M/1 queue. The argument here will be informal, with the aim being mainly to provide insight. Suppose a BP has n customers in it. Consider the same BP with parameter $\theta + \Delta\theta$. From the arguments in Section III-B, the nth customer will leave the system an amount of time $(\Delta\theta/\theta)\ \Sigma_{j=1}^{n} X_j$ later than in the nominal. Call this value ΔY. The probability that this BP now coalesces with another BP is the probability of an arrival occurring within this ΔY period, which is $\lambda\Delta Y + o(\Delta Y)$. If such an arrival occurs, then each customer in this second BP (or rather, what was a second BP in the nominal) will be delayed by an amount ΔS_1 as explained in Section III-B, in addition to the amount calculated by the IPA algorithm. If I_1 is the idle time between the two BPs (e.g., see Fig. 2), then $\Delta S_1 = \Delta Y - I_1$, so that $\Delta S_1 < \Delta Y$. Thus the expected effect *per customer* in the second BP due to the coalition of the BPs is

$$\begin{aligned} &[\lambda\Delta Y + o(\Delta Y)]\Delta S_1 \\ &< [\lambda\Delta Y + o(\Delta Y)]\Delta Y \\ &= o(\Delta Y). \end{aligned} \tag{4.25}$$

Since $\Delta Y = (\Sigma_{j=1}^{n} X_j)\Delta\theta/\theta$, we see that the expected effect is $o(\Delta\theta)$. For gradient calculations we only need to account for effects of the same order as $\Delta\theta$. In informal terms then, the effect that IPA ignores in its calculations is indeed "ignorable" for the purposes of gradient estimation. Alter-

natively one can see this as an explanation of why Assumption (A1') holds for this system.

We should immediately caution the reader that while the above argument is intuitive, it is not rigorous. Specifically it ignores the question of the boundedness of the sum $\Sigma_{j=1}^{n} X_j$ and its square (ΔY^2) and on a more subtle level it ignores the fact that the effect ΔS_1 on the second BP may cause that to coalesce with a third, and so on. Nevertheless this approach can be made rigorous: see the Appendix of Heidelberger *et al.* [33] for the M/G/1 case, and Zazanis and Suri [68] for the GI/G/1 case.

The preceding paragraphs contain the fundamental insight about IPA in *any* system. Essentially, the IPA algorithm neglects certain occurrences (such as the coalescing of BPs) during its calculations. The net impact of these on the performance measure is given by the product of two quantities: the probability of such an occurrence, and the amount it affects the performance. If this product turns out to be of $o(\Delta\theta)$, then IPA will give a consistent estimate of the gradient. Understanding this elementary principle leads to a "core" understanding of IPA. More precise treatments of this point can be found in Cao [4] and Heidelberger *et al.* [33].

We will now re-state an analog of Corollary 3.1 in probabilistic terms. Our approach in Section III-A was motivated by an argument developed on one sample path, and this led to limits with respect to $N \to \infty$. We can, however, state the basic conditions for consistency of IPA with respect to Expectation instead. Let $t(\theta)$ be the system time for an arbitrary customer arriving at a system in steady state. For ergodic systems, the limiting average over customers in one sample path (i.e., $N \to \infty$ in our case) equals the expected steady state value, so that for such systems $T(\theta)$, the limiting sample path value in Proposition 3.1, equals $Et(\theta)$. Thus we can write $dT/d\theta$ as

$$\frac{d}{d\theta} E[t(\theta)]. \tag{4.26}$$

On the other hand, IPA calculates the values $dt_i/d\theta$, where the t_i are customer values as in (2.5), and in steady state for the above customer it would calculate the value $dt/d\theta$. As before, ergodicity implies that the limiting average of $dt_i/d\theta$ will be $Edt/d\theta$. So IPA will be consistent if

$$E\left[\frac{d}{d\theta} t(\theta)\right] = \frac{d}{d\theta} E[t(\theta)]. \tag{4.27}$$

Again, we see that consistency of IPA is related to changing the order of limits (Expectation is basically an integration operation, which is defined as a limit). Often in the literature one sees (4.27) as a condition for consistency of IPA rather than the statement in Corollary 3.1. Since most of the systems we deal with are ergodic, (4.27) can be seen as just a re-statement of Corollary 3.1 where the limit with respect to N has been replaced by the Expectation.

V. IPA for Queueing Networks and General Systems

So far we have illustrated the main ideas of IPA via a single server queue model of a communication link. Now we will look at IPA for more general systems. First a simple production line, with just two machines, will be analyzed. Then we will consider a general network of servers. Finally, we will consider a general discrete event system.

A. IPA for a Simple Production Line

We will now give an example from manufacturing. The development here is based on the early work of Ho, Eyler, and Chien [41]. Consider the production line in Fig. 5. Server

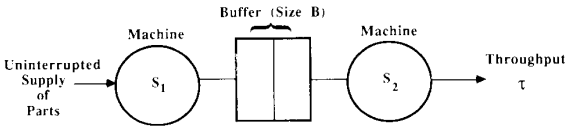

Fig. 5. A simple production line.

1 (S_1) is a machine whose cycle time depends on a parameter θ_1. We assume (for simplicity of the example) that S_1 has an uninterrupted supply of parts to work on. After S_1 finishes its work cycle on a part, it places the part in the buffer. The second machine S_2 picks one part from the buffer, works on it for a cycle time (which depends on a parameter θ_2) and then releases it to a finished goods area. The size of the buffer is B. If the buffer is full when S_1 completes a part then the part stays at S_1, which is then unable to work on another part and is said to be *blocked*. (This is the *transfer-blocking* definition. There is an alternative, *service-blocking*, more common in communication networks, see Suri and Diehl [61] for a discussion.) S_1 remains blocked until S_2 finishes its current cycle, releases its part, and takes the next part from the buffer, thereby releasing a buffer space. We will assume here that all transfers take place in a negligible amount of time, and that the finished goods area always has room for parts (i.e., S_2 is never blocked). The performance measure of interest for this system is its steady state throughput (number of parts produced per unit time), denoted $\tau(\theta_1, \theta_2)$.

We will now define an experiment on this system. The experiment begins with no parts in S_1, S_2, or the buffer. It ends when the Nth part is completed by S_2. Let T be the length of time of such an experiment. Then the experimental estimate of the throughput is

$$\hat{\tau}(\theta_1, \theta_2, N) = N/T. \tag{5.1}$$

While improved experimental estimates could be defined, e.g., to eliminate the initial transient, this will suffice for our current purposes. Under mild conditions that are usually satisfied in practice, this estimate will satisfy

$$\lim_{N \to \infty} \hat{\tau}(\theta_1, \theta_2, N) = \tau(\theta_1, \theta_2) \tag{5.2}$$

as desired for a good experimental estimate (see discussion in Section II).

As before, in order to understand IPA for this system, we need to look at a typical sample path and understand the basic "dynamics" in this system. Such a path is shown for $N = 10$ in Fig. 6. ($N = 10$ is chosen for illustration. In practice one might need N of the order of 10^4 or even 10^5 to make (5.1) reasonable.) The values X_i and Y_i denote the cycle times of S_1 and S_2 for the ith part. The vertical axis represents the total number of parts at S_2 and in the buffer. (The reader can think of the first unit in the graph's height representing a part at S_2 and the remaining units indicating parts present in the buffer.) The buffer size is $B = 2$ for this example. We will denote the ith part by P_i. At the start of the experiment S_1 begins a cycle on P_1 and S_2 is idle (dashed lines in figure).

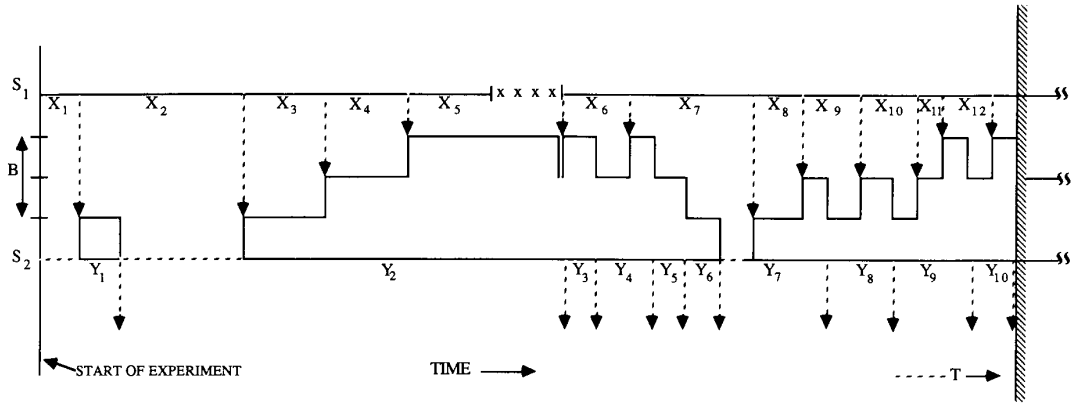

Fig. 6. Nominal sample path for the production line.

After X_1 time units P_1 goes to S_2 which works on it for Y_1 time units and P_1 leaves. S_1 is still busy with P_2 so S_2 goes idle again. Eventually S_1 completes its cycle on P_2 which goes to S_2. S_1 then also completes P_3 but since S_2 is still working on P_2, P_3 is placed in the buffer. Similarly P_4 is also put in the buffer. The total height of the graph is now 3 (one part at S_2 and two in the buffer) and the buffer is full. When S_1 completes P_5 it cannot move the part out and is now blocked (see crosses after X_5 in the figure). Finally S_2 completes its cycle on P_2 (endpoint of Y_2). Now a number of events occur in negligible time (in our model) so we need to follow them carefully: P_2 leaves S_2; P_3 immediately goes into S_2; P_4 moves down one position in the buffer; P_5 moves out of S_1 and into the buffer (see downward arrow at the end of the crosses); and lastly, S_1 is now "unblocked" and starts working on P_6. The vertical double line in the graph at this time instant indicates that the buffer contents dropped to one momentarily and then rose back to two. The reader should now follow and understand the sample path through to the end of the experiment at time T.

Let us now develop an IPA algorithm to estimate $dT/d\theta_1$ for this system. Let $\Delta X_i = X_i(\theta_1 + \Delta\theta_1) - X_i(\theta_1)$ denote the change in cycle times at S_1 due to a change $\Delta\theta_1$ in the parameter θ_1. The perturbed sample path is shown in Fig. 7. P_1 leaves S_1 at time ΔX_1 later, S_2 starts on P_1 at time ΔX_1 later works for a time Y_1 (which is unchanged), and thus P_1 leaves S_2 at time ΔX_1 later compared with the nominal. (In the figure we have not drawn the perturbed path but, rather, indicated the *changes* from the nominal path.) S_1 starts on P_2 at time ΔX_1 later and takes an extra amount of time ΔX_2, so P_2 leaves S_1 at time $\Delta X_1 + \Delta X_2$ later. S_2 thus starts on P_2 at time $\Delta X_1 + \Delta X_2$ later. In a similar way, we follow the perturbed path through to P_5 which finishes its cycle at S_1 $\Delta X_1 + \cdots + \Delta X_5$ later. *Now an interesting interaction occurs.* The buffer is still full so S_1 gets blocked (as in the nominal). S_2 finishes P_2 at time $\Delta X_1 + \Delta X_2$ later than the nominal, so S_1 gets unblocked at time $\Delta X_1 + \Delta X_2$ later than the nominal. So we see that while P_5 started its cycle at S_1 at time $\Delta X_1 + \cdots + \Delta X_4$ later, P_6 starts its cycle just $\Delta X_1 + \Delta X_2$ later. The dynamics of the system are becoming noticeable through the tracking of these perturbations. The reader should now track the perturbations through to the end of the experiment and verify that the perturbation in total experiment time is $\Delta T = \Delta X_1 + \Delta X_2 + \Delta X_6 + \Delta X_7$.

This is a good time to clarify an assumption implicit in the above paragraph. We are assuming (as is standard in IPA) that the perturbations are small enough so that the order of events does not change. In the above, we assumed that since P_5 got blocked in the nominal, it also got blocked in the perturbed path. (For an actual *finite* perturbation that is large enough, P_2 could complete at S_2 before P_5 completes at S_1 and so P_5 would no longer be blocked and the dynam-

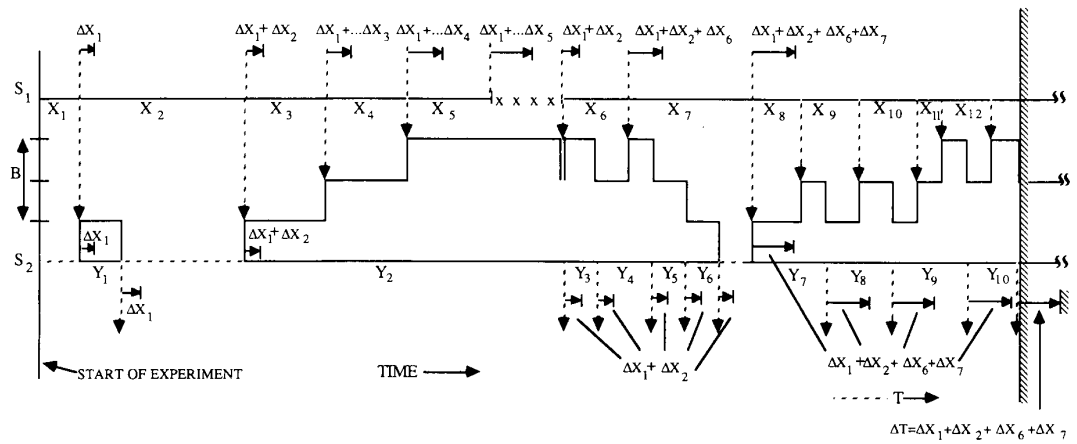

Fig. 7. Perturbations in the sample path for the production line.

ics above would not be correct.) For the IPA algorithm below we assume specifically that if P_i finds S_2 idle (or is blocked by S_2, respectively), then that remains the case in the perturbed path too.

With the above assumption, stating the IPA algorithm becomes particularly simple. Let AC_1 and AC_2 be accumulators associated with S_1 and S_2. At a given time instant, the value of AC_j represents the perturbation at S_j for the last part that left S_j. (In Fig. 7, the arrows ($\rightarrow|$) show the values of AC_1 and AC_2.) Since we are perturbing θ_1, the first rule is that at the completion of service of P_i at S_1, AC_1 is incremented by ΔX_i. The second rule is that if P_j finds S_2 idle, then AC_2 gets the value of AC_1 (see P_1, P_2, and P_7 in Fig. 7). The final rule is that if, by departing from S_2, P_k unblocks S_1, then AC_1 gets the value of AC_2 (as when P_2 leaves S_2 in Fig. 7). These three rules are all that is necessary for gradient calculation with respect to θ_1!

At the end of the experiment (when the Nth customer is served), let $\Delta T = AC_2$. AC_2 is the sum of a selected subset of ΔX_i values, say for $i \in I$. Assume also that (A3) holds (see Section IV-B). In that case, let

$$\frac{dT}{d\theta_1} = \lim_{\Delta\theta_1 \to 0} \frac{AC_2}{\Delta\theta_1} = \sum_{i \in I} \frac{dX_i}{d\theta_1} = \sum_{i \in I} \psi(X_i, \theta_1). \tag{5.3}$$

From (5.1), since N is fixed by definition of the experiment, we have

$$\frac{d\hat{\tau}}{d\theta_1} = -(N/T^2)\frac{dT}{d\theta_1} = -(N/T^2)\sum_{i \in I} \psi(X_i, \theta_1). \tag{5.4}$$

This shows that, if we directly accumulate $\psi(X_i, \theta)$, instead of ΔX_i in the first rule above, and call these new accumulators A_1 and A_2, then at the end of the experiment the value $-(N/T^2)A_2$ will be the IPA estimate of $d\hat{\tau}/d\theta_1$. The above development is summarized in Algorithm 4.

Algorithm 4: Gradient calculation for a simple production line.

0) Initialize: $A_1 \leftarrow 0$; $A_2 \leftarrow 0$; THETA1 $\leftarrow \theta_1$

1) Update: Whenever a part (say P_i) completes service, check these conditions:
   - i) If P_i completed service at S_1 then $A_1 \leftarrow A_1 + PSI(X_i, THETA1)$ (X_i is service time of P_i)
   - ii) if P_i leaves S_1 and terminates an idle period of S_2 then $A_2 \leftarrow A_1$
   - iii) If P_i leaves S_2 and terminates a blocked period of S_1 then $A_1 \leftarrow A_2$

2) Test: If S_2 has completed N parts then go to OUTPUT, else go to UPDATE

3) Output: Let T be the total time since the start of the experiment. The IPA estimate of $d\tau/d\theta$ is $-(N/T^2)A_2$

B. IPA for A General Network with Finite Buffers

Consider a general network of service stations with a single server at each station. Customers may go from any station to any other station, and each station may have a finite buffer size. IPA was extended to such networks by Ho, Cassandras, and Cao in various early papers [38], [12], [36]. The interesting observation we will now make is that the previous section has given us all the analysis needed to develop IPA for this general network. To see this, note that the only times when perturbations "propagate" from one server to another are when idle or blocked intervals are terminated by a customer moving from one server to another. The fact that there are only two servers in tandem is not necessary to the argument; the customer could have moved from any server to any other server. Thus we can generalize and still succinctly state the propagation rules ii) and iii) in Algorithm 4 as follows (below S_i and S_k are any two servers and A_i, A_k are the accumulators associated with them): If a customer leaving S_i terminates an idle or blocked period of S_k, then

$$A_k \leftarrow A_i. \tag{5.5}$$

(The reader should check that the above correctly translates to ii) and iii) in Algorithm 4.)

In a general network it is possible to arrive at a condition called a "chain" of blocking, where, say, S_k is blocked by S_i, and then in turn the buffer at S_k gets full and it ends up blocking S_j. In this case when the customer (say C) leaves S_i, we will have $A_k \leftarrow A_i$ (by the preceding rule). Immediately after this a customer (say D) will be "unblocked" at S_k and will leave for S_i. This customer D leaving S_k will unblock S_j, so we will have $A_j \leftarrow A_k$ (again by the preceding rule). Thus when there is a chain of blocking, no change is required in the rule but we just need to implement the propagation for each unblocked server in turn. (In fact as we see from this example all blocked servers in the chain will end up getting the value of A_i, where S_i is the server at the beginning of the chain.)

We can also generalize condition i) of Algorithm 4 to estimate $d\tau/d\theta_j$ (instead of $d\tau/d\theta_1$), for a stated j, by changing the statement of condition i) to: If a customer completed a service of duration X at S_j then

$$A_j \leftarrow A_j + \psi_j(X, \theta_j) \tag{5.6}$$

where the subscript j on ψ denotes the fact that this function may be different for the service time distribution at each server (this is shown as an additional argument to the "PSI" function in the algorithm below).

As a final, and rather powerful generalization we note that for a network with K servers it is just as easy to state the algorithm to compute all K gradients ($d\tau/d\theta_1, \cdots, d\tau/d\theta_K$) at the same time. First, we generalize the accumulators to be a two-dimensional array: A_{ij} will be the accumulator at server i for the gradient calculations with respect to θ_j. Second, for a general network we need to know where to measure the throughput. Let e be the index of the "exit" server, that is, the experiment will end when S_e completes N jobs. Algorithm 5 is then the requisite generalization of Algorithm 4.

Algorithm 5: Simultaneous estimation of K gradients for a queueing network.

(Comment: A_{ij} is the accumulator at S_i for gradient with respect to θ_j)

0) Initialize: $A_{ij} \leftarrow 0$, i = 1, $\cdots$, K; j = 1, $\cdots$, K
   THETA$_i \leftarrow \theta_i$, i = 1, $\cdots$, K

1) Update: Whenever a customer (say C) completes service, check these conditions:
   - i) If C completed service of length X at S_i then $A_{ii} \leftarrow A_{ii} + PSI(i, X, THETA_i)$

ii) if C leaves S_i and terminates an idle or blocked period of S_m then
$A_{mj} \leftarrow A_{ij}$, for $j = 1, \cdots, K$
(If there is a chain of blocking then continue this procedure through the chain as explained in the text)

2) Test: If S_e has completed N parts then go to OUTPUT, else go to UPDATE

3) Output: Let T be the total time since the start of the experiment. The IPA estimates of the K gradients $d\tau/d\theta_j$ $(j = 1, \cdots, K)$ are: $-(N/T^2)A_{ej}$ $(j = 1, \cdots, K)$

There are several points to be noted regarding Algorithm 5. First, no statement has been made regarding the routing mechanism of customers or the service time distributions: the IPA algorithm as such is derived in a very general way. Second, K gradient values are being computed in parallel from a single sample path. Third, note the simplicity of this general algorithm. It is easily implemented, for example, in any of the common simulation languages—see Suri and Leung [64] for an implementation in the SIMAN language.

We now return to the fundamental question regarding IPA, that is, under what conditions do the estimates in Algorithm 5 converge to the true gradient values? Since the class of queueing networks is very large with many subclassifications, we will not go into details here but rather give an overview of the known results. Our statement will be somewhat technical and require a rudimentary knowledge of queueing network terminology. Currently all the analysis has been for single-server stations. For a closed tandem single-class network with finite buffers (such as an automatic assembly system [64]) Algorithm 5 produces consistent estimates in the case of exponential service times (Cao and Ho [10]). For open and closed tandem single-class networks with finite buffers and more general service time distributions, a consistency proof is in Chen and Suri [20]. For open and closed single class networks with unlimited buffer sizes, IPA has been proven to produce consistent estimates of gradients of sojourn times and throughput[4] for the case of exponential service times (Cao [7], Cao and Ho [9]). In general networks with blocking, however, IPA will usually produce biased estimates (Cao [4]). Such is the case also for networks with multiple classes of customers (Cao [5], Heidelberger *et al.* [33]). In these cases, more powerful PA algorithms are needed, as discussed later.

[4]In the case of unlimited buffer sizes, throughput gradients with respect to service time are meaningful only for closed networks. Algorithm 5 is the correct IPA algorithm for throughput gradients in closed networks too.

C. *IPA for General DEDS*

As a final generalization we consider the application of IPA to any DEDS. Here we will just summarize the results. The basic idea is straightforward. A fairly general class of DEDS is defined formally by Suri [58]. For such DEDS it is shown in [58] that the events in the system are related to each other via a "tree" structure (in the computer science meaning of the term "tree"). We will illustrate this for our simple production line from Section V-A. In [58] all changes in the system that take place instantaneously are considered as part of one event. Thus if a part completes service at S_1, goes to S_2, and starts service at S_2, all in "zero time" (as assumed in Section V-A), then this is all considered to be one event. Fig. 8 shows the events for the nominal sample path from Fig. 6. E_{ij} is the event "P_j completes service at S_i." The first event in the path is E_{11}. This causes P_1 to go to S_2 and start service there (instantaneously) and hence (given the value of Y_1) E_{11} determines the time of E_{21}. A precise definition of this causal relationship from E_{11} to E_{21} is given in [58] and denoted by *schedules*, i.e., E_{11} schedules E_{21}. Similarly, E_{11} schedules E_{12}. The tree structure of events in Fig. 8 is based on this relation: a branch of the tree goes from E_{ij} to E_{mn} whenever E_{ij} schedules E_{mn}. The reader should follow through this tree in Fig. 8 to verify this structure. In particular, note the branch going *upwards* from E_{22} to E_{16}, which has to do with the unblocking of S_1 when E_{22} occurs. Because of the way the example was set up, all events in Fig. 8 are on one tree. In general, there may be several (disjoint) trees.

The significance of this tree structure is twofold. First, under the IPA assumptions it is proved in [58] that perturbations in event times affect other events *only* by propagating "downstream" along the trees (i.e., towards the right, in Fig. 8). Thus a perturbation in E_{13} will propagate to E_{14} and E_{15} but not to E_{11} or E_{12} (they are upstream) nor to E_{22} or E_{16} (they are downstream in time, but not reachable by going

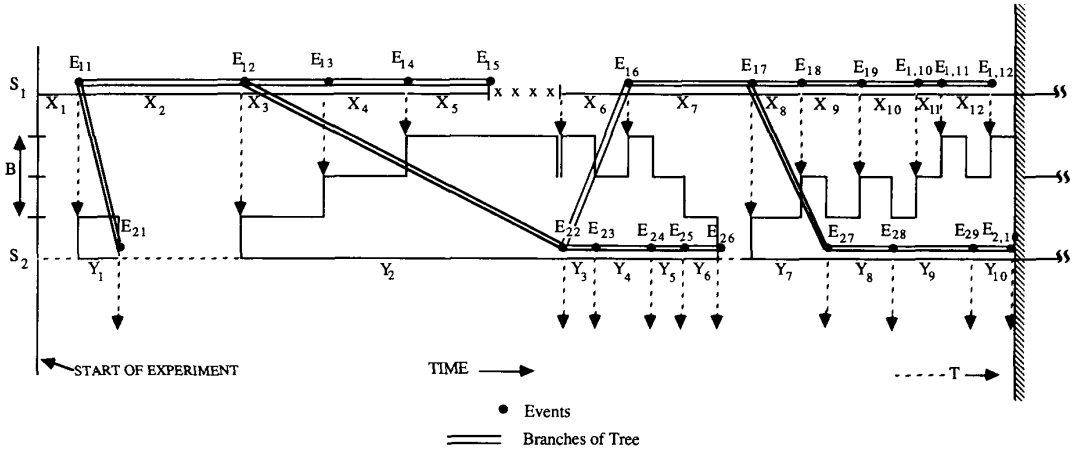

Fig. 8. Tree structure illustrated for events in the production line.

downstream along the tree from E_{13}). Also, events on different trees would not be affected by E_{13}. (These points should be intuitively clear from the meaning of "schedules" and the IPA logic in Section V-A.) Second, for a class of performance measures it is shown that, by using the tree structure, an IPA algorithm can be implemented very efficiently. In fact [58] shows that using the structure of a typical simulation program as a starting point, implementation of IPA requires just four lines of additional code.

To further illustrate the relation between the tree structure and the IPA calculation, consider the perturbation in the experiment completion time, due to a perturbation $\Delta\theta_1$, which was calculated in Section V-A to be $\Delta T = \Delta X_1 + \Delta X_2 + \Delta X_6 + \Delta X_7$. We know that the effect of $\Delta\theta_1$ is to perturb each X_i value. However, as we see from this, only some of these perturbations affect the final performance measure. If we trace back upstream from the final event $E_{2,10}$ we find only the following X_i's are upstream: X_1, X_2, X_6, and X_7! The reader may find it interesting to think about this relationship further, or else read the formal development of these concepts in [58].

In the next section, we will give examples of systems for which IPA is not consistent. Nevertheless, for the cases where IPA is known to work, one has to remark on the simplicity of the procedure (such as Algorithm 5 for networks, or the general algorithm in [58]), which implies that it could be implemented with little effort on a simulation or on actual data being observed from a DEDS. This simplicity, combined with the benefits obtainable by knowing the vector of gradients, plus the "low noise" properties of IPA estimates (discussed in Section VII), make it attractive for researchers to seek out and define classes of systems where IPA will be consistent.

VI. Extensions of IPA

In this section, we first give some examples where IPA clearly will not work. This will motivate the extensions that are described next, which enable gradient estimation for a wider class of systems.

A. Example of Failure of IPA

Consider again the communication link studied in Section III. Suppose, however, that the performance measure of interest is the average number of messages sent between idle periods of the link. If we model the link as a single server queue, this performance measure is the average number of customers served in a BP. Denote this average by $\beta(\theta)$, where the argument reminds us that the value of β depends on the parameter θ (we will omit the dependence on γ for this section). A simple experimental estimate for $\beta(\theta)$ would be to observe M BPs and then to let

$$\hat{\beta}(\theta, M) = \left(\frac{1}{M}\right) \sum_{m=1}^{M} n_m \tag{6.1}$$

where n_m is (as in Sections III and IV) the number of customers served in BP_m.

Now let us consider again the arguments presented in Section III and see how they apply to this new performance measure. The IPA estimate is based entirely on the assumption that no BPs will coalesce, the situation we called "case i)" and showed in Fig. 3. However, if we make our parameter change small enough so that no BPs coalesce, then each n_m value in (6.1) will remain the same, so that there will be no change in the estimate of the performance measure. Thus the IPA estimate of sensitivity will be identically zero! Intuitively it is clear that this estimate must be wrong: if we increase the link service time we should expect (on average) more messages between idle times. So what has happened here?

The key to understanding the failure of IPA is to first understand the nature of the performance measure. Suppose for illustration, that we conduct our experiment for one BP only, and let us think through what the perturbed path would be for various increments $\Delta\theta$. Refer to Figs. 3 and 4. As we increase the size of $\Delta\theta$, the number of messages served in the first BP does not change for a while (it stays at n_1), then as $\Delta\theta$ passes through a critical value, the first BP "collides with" the second BP and all of a sudden the number of messages served in the first BP (of the perturbed path) equals $n_1 + n_2$ (values from the nominal path). Thus, with respect to the variable $\Delta\theta$, the sample performance measure is quite discontinuous: it is in fact stepwise constant. Now let us relate this to some of the statements made at the end of Section IV, about when IPA works. We mentioned there that IPA ignores the possibility of some events. If the probability of occurrence of such events, multiplied by the effect of these events on the performance, is insignificant (in the sense of $o(\Delta\theta)$), then IPA will be consistent. In this case however, the *only* events that lead to an effect on the performance measure (i.e., a coalition of BPs) are *precisely* the events that IPA ignores! Therefore, in this case IPA is doomed to failure by its very assumption that BPs will not coalesce. Another salient observation here is that while the probability of coalition of BPs is the same in this example and for the case of average system time in Sections III and IV, the effect on the performance measure, when BPs coalesce, is very small for the case of average system time, but quite significant for the case of average number of customers in a BP.

This example presents a clear instance of the failure of IPA—the very nature and assumptions of IPA are such that they cannot measure the basic effects that impact the performance measure in this case. While far more subtle examples exist and are discussed in the references, this example gets to the heart of why IPA may fail.

B. Smoothed Perturbation Analysis

Motivated by the failure of IPA to work for the simple case above, Gong and Ho [32] used the powerful tool of conditional probability to develop an extension to IPA, which they termed smoothed PA (or SPA). Before describing SPA in its generality, let us develop an estimator for the gradient of the above performance measure, and use this to give us the insight about SPA. We will assume that the link can be represented by a GI/G/1 system—this is not necessary in general, but we do so to keep the analysis here straightforward.

Let the interarrival times have a cumulative distribution function $G(\cdot)$ and a corresponding density function $g(\cdot)$. In order to make the following analysis rigorous, several technical assumptions need to be made regarding the interarrival and service time distributions (see Zazanis and Suri [68]). These will be omitted here and the analysis will be informal, aiming to give the main insight. Consider again

the experiment on one BP as described above, and refer to Fig. 3. This experiment would terminate with the departure of M_4, and we would have $\hat{\beta} = n_1 = 4$. Now we ask the question, for a given $\Delta\theta$, what is the *expected* change in the value of n_1, based on the observed BP_1? Looking at the shaded areas in Fig. 3, which denote the changes induced by $\Delta\theta$, we see that the value of n_1 will only change if an arrival occurs during the shaded interval following M_4, because in this case for the perturbed path the BP will not end after M_4. Let this last shaded interval have length ΔY. Also, at the time that the nominal BP_1 ended, let z_1 be the time since the last arrival (in Fig. 3 this is the time from the arrival of M_4 to the departure of M_4). Then the probability of an arrival during ΔY is

$$P(\text{arrival in } \Delta Y \mid BP_1) = \Delta Y g(z_1)/[1 - G(z_1)] + o(\Delta Y). \quad (6.2)$$

Here the conditioning $(\cdot \mid BP_1)$ means we have observed all the events in BP_1. Now the value of ΔY can be written as

$$\Delta Y = \sum_{i=1}^{n_1} \Delta X_i = \sum_{i=1}^{n_1} \psi(X_i, \theta)\Delta\theta + o(\Delta\theta). \quad (6.3)$$

Here we have used Assumption (A3) in Section IV-B which states that $dX_i/d\theta$ can be expressed as a function $\psi(X_i, \theta)$. The purpose of this is to get the RHS of (6.3) explicitly in terms of $\Delta\theta$ and the observations on BP_1.

Next we calculate the effect on n_1 of such an arrival. We can think of this arrival as one which would have started a new BP in the nominal, but which gets combined with the first BP in the perturbed path (as in Fig. 4). Because the BPs in a GI/GI/1 system are probabilistically independent [45], the expected number of customers in this second BP is just the expected number of customers in a BP, or $\beta(\theta + \Delta\theta)$—note that the argument indicates that this second BP should be a representative BP for the new parameter value. Thus the expected change in n_1 will be the probability of an arrival multiplied by the effect of the arrival, or

$$E[n_1(\theta + \Delta\theta) - n_1(\theta) \mid BP_1] = \frac{g(z_1)}{1 - G(z_1)}\left[\sum_{i=1}^{n_1} \psi(X_i, \theta)\right] \cdot \beta(\theta + \Delta\theta)\Delta\theta + R(\Delta\theta). \quad (6.4)$$

Here the term $R(\Delta\theta)$ denotes a remainder which accounts for two effects: i) the fact that the second BP that coalesces with the first does not remain unchanged, in fact, it is pushed back by the amount ΔS_1 (see Fig. 4) and this may lead to this second BP coalescing with a third, and so on; and ii) all the $o(\Delta\theta)$ terms from (6.2) and (6.3) which need to be explicitly considered through a further step below.

It is important to note here a fundamental difference between the above analysis and that done for IPA in Section III. The entire IPA algorithm there could be developed from the physical dynamics of the system and the observed sample values, with no need to bring in any probability. On the other hand, above we have taken an *expectation* to get an estimate not of the *actual* change that would occur but rather the *average* change that would occur with the perturbation. This is an essential difference between the SPA and IPA approaches and we shall amplify on it later.

The analysis above holds for any BP in the nominal path, so we can generalize (6.4) to get, for the mth BP

$$E[n_m(\theta + \Delta\theta) - n_m(\theta) \mid BP_m] = \frac{g(z_m)}{1 - G(z_m)}\left[\sum_{i=1}^{n_m} \psi(X_{k_m+i}, \theta)\right] \cdot \beta(\theta + \Delta\theta)\Delta\theta + R(\Delta\theta) \quad (6.5)$$

where k_m is as defined in Section III. Next we will take the expectation of this quantity over all possible occurrences of BP_m, and use the relation $E\{E[X|Y]\} = E[X]$ to get

$$\beta(\theta + \Delta\theta) - \beta(\theta) = E\left\{\frac{g(z_m)}{1 - G(z_m)}\left[\sum_{i=1}^{n_m} \psi(X_{k_m+i}, \theta)\right]\right\} \cdot \beta(\theta + \Delta\theta)\Delta\theta + ER(\Delta\theta). \quad (6.6)$$

Zazanis and Suri [68] show that under certain assumptions on the GI/G/1 system, $ER(\Delta\theta) = o(\Delta\theta)$. Hence, dividing (6.6) by $\Delta\theta$ and taking limits as $\Delta\theta$ goes to 0, we get

$$\frac{d\beta}{d\theta} = E\left\{\frac{g(z_m)}{1 - G(z_m)}\left[\sum_{i=1}^{n_m} \psi(X_{k_m+i}, \theta)\right]\right\}\beta(\theta). \quad (6.7)$$

The expectation on the RHS is in general intractable for a GI/G/1 queue, and also, the value of $\beta(\theta)$ is not known in closed form. On the other hand, both of these are very easy to estimate given an observed sample path. Looking at the form of (6.7) we see that a possible estimator for $d\beta/d\theta$ based on a sample path of M BPs, is

$$\hat{g}_\beta(M) = \left[\frac{1}{M}\sum_{m=1}^{M}\left\{\frac{g(z_m)}{1 - G(z_m)}\sum_{i=1}^{n_m}\psi(X_{k_m+i}, \theta)\right\}\right]\left[\frac{1}{M}\sum_{m=1}^{M} n_m\right]. \quad (6.8)$$

Here the first bracket estimates the expected value in (6.7), and the second, $\beta(\theta)$. This translates easily into an algorithm which is only slightly different from our Algorithm 3 for estimating the gradient of mean system time—Algorithm 6 estimates the gradient $d\beta/d\theta$ from a single sample path. (The functions $G1(\cdot)$ and $G2(\cdot)$ in the algorithm denote $g(\cdot)$ and $G(\cdot)$ respectively.)

Algorithm 6: Estimation of $d\beta/d\theta$ from single sample path of GI/G/1 queue.

0) Initialize: MCOUNT ←0; NMSUM ←0; PSISUM ←0; BIGSUM ←0; THETA ← θ.
1) Update: At departure of next customer (with service time observed to be XJ):
   1.1) NMSUM ← NMSUM+1
   1.2) PSISUM ← PSISUM+PSI(XJ,THETA)
   1.3) *If* server is now idle *then*
      1.3.1) BIGSUM ← BIGSUM+ PSISUM*G1(ZM)/[1−G2(ZM)] (where ZM is time since last arrival)
      1.3.2) MCOUNT ← MCOUNT+1
      1.3.3) PSISUM ← 0
2) Test: *If* MCOUNT = M *then* go to OUTPUT *else* go to UPDATE
3) Output: The IPA estimate of the gradient is BIGSUM*NMSUM/(M*M)

Thus we have once again a relatively simple algorithm that enables estimation of the gradient of the mean number of customers served in a BP, from a single sample path. As we mentioned above, the key to this algorithm is bringing a conditional expectation into the analysis to develop the appropriate estimator. Zazanis and Suri [68] used a similar conditioning to get an estimator for the *second derivative* of mean system time with respect to a service time parameter, for a GI/G/1 queue from observations on a single sample path. Gong and Ho [32] develop this conditioning idea

into a general approach which they termed SPA and which we now describe briefly.

Let $J(\theta)$ be a performance measure of a DEDS and let $L(\theta)$ be an unbiased sample estimator for J, that is

$$E[L(\theta)] = J(\theta). \tag{6.9}$$

For many situations, such as the performance measure $\beta(\theta)$ above, the sample performance estimate can be discontinuous with respect to θ (as is $n_1(\theta)$ above), but the expected performance measure, $J(\theta)$, may still be continuous. (For instance, in the M/M/1 queue, which is a special case of the above example, $n_1(\theta)$ is discontinuous in θ exactly as described above, yet $\beta(\theta)$ is analytic in θ.) Intuitively speaking, the expectation operator in (6.9) "smooths out" the discontinuities by averaging over a large number of sample paths. This smoothing property is well known in probability and filtering theory.

Now let $z(\theta)$ be any subset of the data observable from the experimental sample path, and let $\Delta L(\theta) = L(\theta + \Delta\theta) - L(\theta)$. Gong and Ho [32] define the SPA estimator as

$$\hat{g}_{\mathrm{SPA}} = \lim_{\Delta\theta \to 0} \frac{E[\Delta L(\theta) \mid z(\theta)]}{\Delta\theta}. \tag{6.10}$$

As an illustration of this, for our previous example, let us examine (6.8). There z consists of the values: $M, n_1, \cdots, n_M$, $z_1, \cdots, z_M$, and all the X_i. Also, the RHS of (6.8) is precisely the limiting conditional expectation on the RHS of (6.10).

In order for the general SPA estimator in (6.10) to be unbiased we need

$$E \lim_{\Delta\theta \to 0} \frac{E[\Delta L(\theta) \mid z(\theta)]}{\Delta\theta} = \frac{d}{d\theta} E[L(\theta)] \tag{6.11}$$

whereas for the IPA estimator to be unbiased we must have (see end of Section IV)

$$E \lim_{\Delta\theta \to 0} \frac{\Delta L(\theta)}{\Delta\theta} = \frac{d}{d\theta} E[L(\theta)]. \tag{6.12}$$

Comparing the LHSs of (6.11) and (6.12) we see that (6.11) has one more expectation in it. It is observed in [32] that (6.11) is more likely to hold since the inner expectation in it "smooths out" the function we are differentiating. Thus SPA can be expected to work for a broader class of systems and performance measures than IPA.

In the form above, SPA is a rather general estimator. As shown in [32] by different choices of z one can make the SPA estimator range from IPA to closed form solutions from classical queueing theory. This means that the method is very powerful and can be "tuned" to the amount of closed form solutions that can be developed prior to doing the experiment. However, it also means that in applying SPA to a practical system one needs to examine closely two coupled aspects: i) what is a good choice of z, and ii) can we obtain an analytic expression for the requisite conditional expectation? Thus applying SPA to any system will not be routine, as it is for IPA (e.g., Suri [58] states an IPA algorithm for a general DEDS). On the other hand, among all the extensions to IPA, at the present time SPA is the preferred one where it can be developed, since it is likely to have the best variance properties as we shall see in Section VII.

We will not consider more detailed analysis of examples related to SPA here. Gong and Ho [32] apply to several simple systems to illustrate its use, and Gong [31] applies it to a routing problem in communication networks. An interesting point is that in [31] it is proved by the SPA arguments that an algorithm proposed in 1976 by Bello and Segal [2], [56] for networks is in fact unbiased (the algorithm was believed to be biased in [2]).

C. Extended Perturbation Analysis

Another approach to overcome the potential inconsistency of IPA was introduced by Ho and Li [42] who termed it extended PA (EPA). The basic assumption of IPA for the single server queue is that the BPs remain the same in the nominal and perturbed paths. The generalization of this assumption for arbitrary DEDS is that the *sequence* of events in the two paths remains the same, although the time of events may change [58]. Now we know from the above discussions that it is not necessary for the actual perturbed path to have the same sequence of events, but only that the change in the perturbed path be such that its effect is statistically small. However, if the structure of the system being analyzed is such that this change is statistically significant, then IPA will not be consistent.

For systems that can be represented by a continuous time markov chain, the EPA method can be applied. (An example of such a system would be a multiple class queueing network with class-dependent exponential service times. Any practically sized network would be analytically intractable, so simulation would be the usual analysis tool.) Here we will just give an outline of EPA. This method works by choosing a *finite* $\Delta\theta$ and then predicting, from the nominal path, where the perturbed path would have branched to a different state, say B, while the nominal path continues in, say, state A. Up to this point, an IPA-like estimator is used to compute the effects of the perturbation, but at this point, the computation is "frozen." The algorithm then waits until the system eventually enters state B during the nominal path (clearly this means that EPA can only be applied to systems where all states will recur in finite time). At this point the EPA computation restarts. (Certain "end-effects" to do with patching together these two pieces need to be considered too.) The justification of this approach is that when the system enters state B in the nominal path, from the Markov property its future evolution is statistically the same as it would have been when it entered B earlier on in the perturbed path (provided we can continue to predict the future differences between these paths as well). Suppose the nominal experiment is designed such that its output $L(\theta)$ is an unbiased estimate of $J(\theta)$, that is, $E[L(\theta)] = J(\theta)$. Then by continuing with this algorithm to the end of the experiment, we will get, in addition, an estimate, say $H(\theta, \Delta\theta)$, such that $E[H(\theta, \Delta\theta)] = J(\theta + \Delta\theta)$. From these two estimates (computed from *one* experiment) we can therefore get an estimate $D(\theta, \Delta\theta) = H(\theta, \Delta\theta) - L(\theta)$ which satisfies $E[D(\theta, \Delta\theta)] = J(\theta + \Delta\theta) - J(\theta)$. Hence $D(\theta, \Delta\theta)/\Delta\theta$ can be used as an estimate of the gradient of $J(\theta)$.

We can see right away that the EPA algorithm cannot be as efficient as IPA, since it may remain "inactive" for significant sections of the nominal experiment. However, there are two factors that make its performance better than one might expect. The first is that in most applications one is doing gradient estimation with respect to a number of parameters simultaneously—in this case it can turn out that

several of the gradient computations are "active," on average, during the observations and the savings is still better compared to multiple experimentation. The second is that from a practical point of view, one can often aggregate the states of a system to fewer subsets, and use the aggregate state to decide whether to activate or deactivate the EPA calculation. Not only does this keep the computations active for longer segments of the experiment, but it also enables EPA to be applied to non-Markovian systems. All these points are elaborated by Ho and Li [42]. As an example, we mention that in [42] an EPA algorithm is applied to gradient estimation in a multiple class queueing network and experimental results show that there still exists the possibility of an order of magnitude or even more reduction in experimental time compared with the use of repeated experiments.

An important point to note from the preceding discussion is that its properties of EPA will be similar to the CRN finite difference estimator. This is because the EPA algorithm is just a clever way of estimating the value of $J(\theta + \Delta\theta)$ without a second experiment. On the other hand, both the IPA and SPA estimators directly estimate a derivative, not a finite difference. This point is actually quite critical, because it impacts the variance properties of these estimators, discussed in Section VII.

D. Finite Perturbation Analysis

This extension, which we denote FPA, historically predates EPA and SPA. In fact, the original paper on PA by Ho *et al.* [40] used a version of FPA, but the formal distinction between IPA and FPA methods was made later [41], [38], [39]. The basic idea of FPA is to tackle IPA's weakest assumption, namely that events to not change order. However, in order to avoid doing a complete new experiment to track changes in order of events, FPA only considers changes in order of events up to a predefined limit. For example, "first order" FPA only considers the possibility that adjacent events might change order, and ignores any effects of changes in order beyond adjacent events. The way it works then is to introduce perturbations and propagate them while observing the nominal path, but limiting its calculations by only extrapolating to predict the effects of such changes in order [39].

While the original FPA algorithms were all heuristic in nature and the indications of their consistency were purely experimental (e.g., [39], [40]), Cao [5] placed FPA on a theoretical footing by proving consistency of an FPA algorithm for a simple multiclass queueing network. In retrospect, however, we can see the work by Cao [5] as being the finite difference step in the SPA estimator (e.g., the step in (6.6) above), and it appears better to take such an analysis one step further, where possible, by letting $\Delta\theta \to 0$ to get the expression in (6.7) with corresponding estimator (6.8). Indeed, Gong and Ho [32] develop such an SPA estimator for the system considered in [5]. The early papers on FPA also make for very difficult reading for researchers starting on PA. In addition, in the light of recent developments on SPA and EPA, it is likely that these methods will cover the domain for which FPA was initially developed. Thus it would be this author's recommendation for interested readers to focus on these new approaches and save reading some of the earlier papers on FPA until they are thoroughly familiar with the recent thoughts on PA and definitely feel the need to understand FPA (or wish to explore the origins of PA!).

E. Making IPA Work by Change of Parameter or Representation

Although IPA may not work for a given problem, there may be a transformation of the problem for which IPA will work. An age-old trick in mathematics, when one encounters a seemingly new problem, is to see whether it can be transformed to a problem that has already been solved. Ho and Cao [37] use this idea to convert the problem of sensitivity analysis with respect to a routing parameter (for which IPA is not consistent) to an equivalent problem with respect to a service time parameter (for which it is). Additional examples of the use of such a transformation remain a topic for investigation.

Some very recent research on the applicability of IPA has produced the interesting observation that whether IPA will work or not can be a consequence of the *observer's* internal model of the system being observed. It is possible to construct two alternative such models, both of them stochastically correct representations of the actual system, and yet the resulting IPA algorithm may produce consistent results in one representation and not in the other! For example, Heidelberger *et al.* [33] provide an example of a birth-death process and show that (for their model of the process) an IPA algorithm provides a gradient estimate which is clearly biased since it has the wrong sign.

In analyzing this example, Glasserman [25] discovered that by using an alternative representation of the birth-death process the resulting IPA algorithm produced consistent estimates. In other words, some representations may produce sample paths that are "smoother" with respect to a given parameter, than other representations. This is consistent with the recent work of Glynn [28] who makes the same observation about random variables (Glasserman's statement was for DEDS). The result by Glasserman raises the intriguing question: for the cases where IPA is currently considered to be inconsistent, might there be alternative representations for the systems which would make IPA consistent? Glasserman and Gong [26] provide such an alternate representation for an M/G/1/K queue. Because of the simplicity of IPA, as well as its excellent variance properties (Section VII), finding such representations would be very worthwhile.

VII. Comparison of PA with Other Gradient Estimation Methods

Although PA offers the ability to estimate a gradient from a single experiment, for stochastic systems one also needs to understand the "noisiness" of the PA estimates, in order to fully convince ourselves that PA is worthwhile. In fact, in signal processing, it is common knowledge that differentiating a signal tends to amplify the noise in it. Since IPA does just that, one's intuition might lead one to expect highly "noisy" derivatives from IPA. However, we will find surprising results below! Also, other "single-experiment" methods have recently been suggested for gradient estimation and it is useful to understand the properties of these in comparison with PA. We begin by comparing PA to the natural alternative, namely finite difference estimates using multiple experiments.

A. Comparison of IPA with Finite Difference Methods

The simplest conventional approach to gradient estimation for a DEDS would be the finite difference method which would conduct two experiments, one at the nominal value of the parameter (θ), and one at the value $\theta + \Delta\theta$, and then use the forward difference (FD) estimate in (2.6) to get an approximation to the gradient. Compared with this, PA has the apparent advantage in that it produces estimates from a single sample path. However, this advantage may not be realized if the PA estimates are so noisy that we require a longer sample path to get the same accuracy as the FD method. Thus in order to convincingly demonstrate the computational savings offered by PA, one has to also investigate the issue of "noisiness" in both estimates. Here we will consider first the IPA estimator and then discuss other PA methods. The discussion below is based on the treatment in Zazanis and Suri [67] (also see [4] for a related analysis).

Clearly, the accuracy of the gradient estimate obtained by the FD method will depend on the choice of $\Delta\theta$. The FD method however, suffers from a basic incompatibility of two considerations: i) in order to estimate the gradient as accurately as possible one would like to make $\Delta\theta$ *as small as possible*, yet ii) if we look at the form of the FD estimate in (2.6) we see it is the difference of two noisy quantities divided by a small quantity ($\Delta\theta$) so that this estimate will have a variance proportional to $1/\Delta\theta^2$, and to minimize this variance we should choose $\Delta\theta$ *as large as possible*.

To understand this tradeoff better we need to quantify the total error in the FD estimate and for this we can use the mean squared error (MSE): if g is the actual gradient value and $\hat{F}$ is the estimate then the MSE is $E[g - \hat{F}]^2$. This MSE can be expressed as the sum of the variance in $\hat{F}$ and the square of the bias in $\hat{F}$. (The bias arises from using a finite difference as an approximation to the gradient). Since the bias increases with $\Delta\theta$ while the variance decreases with $\Delta\theta$, as can be expected then, there is a "best" choice of $\Delta\theta$ which minimizes the MSE. The performance of the FD estimate then will be, at best, that obtained if one chooses this value of $\Delta\theta$. Zazanis and Suri [67] analyze this problem for the case where the experiment at θ consists of M independent observations, and that at $\theta + \Delta\theta$ consists of another M independent observations (all $2M$ observations are mutually independent). Under these circumstances, they show that the MSE of the above "best" FD estimate is proportional to $M^{-1/2}$. For systems where IPA is consistent, this MSE turns out to be proportional to M^{-1} for the IPA estimate. These two rates are drastically different. To see this, note that for statistical estimates, confidence intervals are roughly proportional to the square root of the MSE. Thus if we wish to increase the accuracy of a gradient estimate by one significant digit, using IPA we will need to obtain a hundred times as many observations. (This might sound high but it is true of any statistical estimator, not just IPA—here IPA is just behaving as well as the best statistical estimator.) On the other hand, to get this increase in accuracy using the FD method will require 10 000 times as many observations! *This example points out the fundamental numerical problems associated with gradient estimation in stochastic systems.* It also shows, however, that IPA offers not only the opportunity to reduce the number of experiments, but also the opportunity to get the same accuracy for far fewer observations. (Practical examples in [67] indicate that the constant of proportionality is about the same for both methods, so the M terms dominate.) These statements, of course, apply only in the cases where IPA is itself unbiased. In such cases though, by combining the fact that we can estimate multiple gradients with IPA, with this additional efficiency in reduced noise, we can conclusively state that IPA provides the possibility of orders of magnitude reduction in computational effort.

The FD estimator above can be improved by using symmetric difference (SD) estimates, namely, one experiment at $\theta - \Delta\theta$ and one at $\theta + \Delta\theta$. This does improve the MSE properties somewhat [67], however, the price we pay for this is that we now need *two* additional experiments after the nominal one at θ. In fact, for N parameters, the FD method requires N additional experiments, the SD method $2N$ additional, while IPA requires *no* additional experiments. Also, as shown in [67] this SD estimator still does not achieve the accuracy of IPA. Another possibility is, if experimental design permits, to use common random number (CRN) methods along with the FD estimate, as discussed in Section II-B. An analysis in Cao [4], while not based on MSE properties, suggests that whenever IPA is unbiased its convergence properties will be better than CRN FD estimates. A rigorous comparison of the MSE of IPA with CRN FD methods remains to be done. (This author's intuition suggests that while use of CRN methods will improve the constant of proportionality in the MSE, the fundamental dependence on observation length will remain at $M^{-1/2}$ and so the FD estimate will remain quite inefficient even with CRN methods.)

B. Other PA Estimates

We come next to the question of accuracy (in the above sense) of the SPA, EPA, and FPA estimates. While rigorous analyses remain to be done on these, we can make some general statements based on the knowledge of these estimates and the basic methodology used by Zazanis and Suri [67]. Any estimator that directly provides an unbiased estimate of the gradient will definitely have an MSE proportional to M^{-1}. On the other hand, any estimator that first estimates the performance at $\theta + \Delta\theta$ and then uses this to estimate the gradient by a finite difference, is likely to have an MSE proportional to $M^{-1/2}$. Therefore, SPA falls into the most efficient category along with IPA, while EPA and FPA are likely to behave as the FD estimator. Note that these statements are in regard to the MSE property only; in terms of number of experiments, all these PA methods provide estimates from only one sample path.

At any rate, from the above observations as well as the discussion in the previous section, we can now justify our earlier statements that IPA is the simplest and also most efficient estimator and should be used where possible, and after this, SPA is to be preferred over EPA or FPA. Also, the issue of MSE for EPA is somewhat involved—see the discussion in Section IX—so the statements in this section should be regarded as somewhat speculative until rigorous results are available.

C. Likelihood Ratio Methods

Gradient estimation in stochastic systems has recently been receiving a lot of attention from the research community. A number of methods, all based on likelihood

ratios, have been proposed which, like PA, also obtain gradient estimates while observing only one sample path. Fairly general classes of likelihood ratio (LR) methods have been proposed and analyzed by Glynn and Sanders [30] and Reiman and Weiss [51], and a particular case of these methods, called the Score Function method, has been proposed by Rubinstein [55]. Interestingly enough, it seems that an LR method for gradient estimation in simulations was actually introduced in the Russian literature in 1968 [1], but apparently never attracted any attention! While we do not cover the technical details of these approaches here, for readers knowledgeable in statistics we mention that these methods all use a technique that is related to "importance sampling" or "change of measure." Without covering the analytical aspects, it may still be useful for the reader if we make some remarks that compare PA with LR methods. The simulation community has been attempting to make broad statements comparing the two methods, but as we hope to show, the comparison is quite complex and neither method dominates. In making our comparison, we first talk about IPA, then PA in general.

The LR algorithm can be stated for general systems, general performance measures, and certain types of parameters. In this respect it is similar to IPA which can also be stated in general terms. Like IPA, LR also involves a (different) change of limits. It appears however, that LR is applicable to a wider class of stochastic systems (not just DEDS) and performance measures, and that the characterization of systems where it is consistent has been more thoroughly completed than for IPA.

On the other hand, LR is restrictive in the parameters it can analyze. Essentially it cannot be applied to a parameter that changes the support of a probability distribution for an r.v. (i.e., the domain over which the r.v. takes values). This excludes whole classes of parameters, such as the mean of a uniform distribution. By its very nature, LR simply cannot be applied to deterministic parameters. In constrast, as shown by our development in Section III, IPA does not require any probability statements for its basic arguments. Thus, as shown by Suri and Zazanis [65] IPA can be applied to, and will be consistent for, sensitivity analysis with respect to the deterministic service time in an M/D/1 queue. For discrete valued r.v.'s the comparison is rather interesting. Consider a discrete r.v. that takes the values α_i with probability p_i ($i = 1, \cdots, n$). LR algorithms can be developed with respect to the p_i parameters, but not the α_i parameters, while quite the opposite, IPA algorithms can be developed with respect to the α_i parameters, but not the p_i parameters! (We say only "can be developed" to remind us that the question of consistency of these algorithms would still need to be clarified in either case.)

Now we turn to the wider class of PA methods. The development of SPA, EPA, and FPA algorithms has (theoretically or experimentally) extended the domain of applicability of PA, in terms of the classes of systems, parameters, and performance measures that can be analyzed, so some of the above disadvantages of IPA can be overcome. However, as already mentioned, these extensions of PA may require considerable analytical work on the part of the algorithm developer, with some "customization" for each problem, while LR has the advantage of remaining a generally definable algorithm wherever it can be applied. One other area where PA can be applied, but not LR, is that of discrete-valued parameters, which can be important in practical applications such as buffer-sizing decisions (see Section VIII).

Finally, there is the question of "noise" in the estimates. For problems involving steady-state performance measures, LR methods need to use regenerative techniques, and the variance of their estimates increases with the length of the regeneration cycle. This can lead to quite "noisy" estimates. For example, the LR estimates in [51] seem to have an order of magnitude or more variance than corresponding IPA ones in [65]. In steady-state estimation then, if IPA is known to produce consistent estimates, then it should be the method of choice [6]. Both methods can be used for transient performance estimation too (e.g., for IPA estimation of transient throughput see [7]). The comparison of variance in this case remains an open question.

Thus, as we hope we have shown, the two methods (PA and LR) have overlapping domains of applicability, but none contains the other. Also, the decision to use one method or the other is compounded by issues of algorithm coding, estimate consistency, and variance. In this author's opinion, both methods represent very promising advances in the state of the art, and both methods, as well as their comparison, are worthy of further research efforts.

D. Optimization Implications Including Single-Run Optimization

As we mentioned in the introduction to this paper, an obvious application of the gradient information supplied by PA would be for use with parameter optimization algorithms. The problem of optimizing stochastic systems, using experimental observations, is inherently difficult. Because of the randomness in the observation, a few sample observations may lead us to change the parameter in the wrong direction. Thus, any good stochastic optimization scheme must be able to converge to the optimum despite the confusion of noise. This problem has received a fair amount of attention in the statistics literature, and a number of traditional approaches exist to tackle it, including response surface methods, various search methods, and stochastic approximation techniques (e.g., see the references in [49], [54], [63]). Empirical investigations of these methods shows that, while they may work on practical problems, they require enormous amounts of experimental effort. These investigations are supported by analytical results too. For example, without the availability of gradient information, the convergence rate of the Kiefer–Wolfowitz method (a classical stochastic approximation method [46]) is typically $O(M^{-1/3})$, where M is the experimental effort. This means that to improve the estimate of the optimum by one significant digit requires increasing the experimental effort by a factor of 1000!

As we shall see, the knowledge of gradient information can improve this situation considerably. With regard to stochastic approximation methods, for example, if we can directly estimate the gradient from each experiment, then the optimum seeking problem for the objective function is replaced by a root finding problem for the gradient. Of course, the gradient is still a noisy observation, and we need to apply root-finding methods for noisy observations. The classic stochastic approximation method for this is the Robbins–Monro (RM) procedure [53]. Let us say we are interested in optimizing some steady state performance mea-

sure of a production line, with respect to a service time parameter. An obvious optimization approach would then be to conduct an experiment, get an estimate of the steady-state gradient, and then use the RM procedure to update the parameter value. Such an approach was used successfully by Ho and Cao [36] for several queueing network examples. The RM procedure has a convergence rate of $O(M^{-1/2})$, so we now need (only!) 100 times as many experiments for each significant digit. We see at once the considerable effect of being able to obtain the gradient with each experiment.

It seems, however, that we can do even better than this approach. Since the PA estimate of the gradient is available as the experiment is being observed, why not use this estimate to improve the parameter value *while the system is operating* and, by continuously doing so, optimize the system *during a single experiment!* Of course, this raises several interesting issues. First, each time we change the parameter value, we introduce transient effects into the system, which may create bias in our updating scheme if we are really interested in optimizing a steady state performance measure. Second, because of such transient and bias effects, will this scheme really end up taking a longer experiment than the sum of the individual experiments required with the previous method?

Only preliminary answers are known to these questions. Still, the evidence that is available suggests that such *single-run optimization* methods may be very promising in overcoming the computational demands of stochastic optimization problems. Preliminary experiments with such methods were reported by Meketon [48], [49] and Suri and Zazanis [65], and a comprehensive empirical study was done for a simple system (M/M/1 queue) by Suri and Leung [63]. Single-run optimization of a manufacturing system with multiple parameters is in [64]. The results for the PA-RM single-run (PARMSR) algorithm in [63] are quite exciting. For example, in one case where the parameter was updated after every 5 customers were served, the algorithm converged to its estimate of the optimum after about 500 customers (average over a number of sample runs). The same number, for a single-run optimization algorithm operating without gradient information, was around 25 000 customers. Not only that, but the average absolute deviation from the (theoretical) optimum cost, after convergence, was 2.5 percent for the PARMSR algorithm but 7.4 percent for the other algorithm. In other words, the reduction in run time was achieved along with an increase in accuracy. Recently, Fu and Ho [71] obtained a further reduction in run time by utilizing second derivative information obtained simultaneously with the gradient information from a single sample path. No doubt, all these results are preliminary and many open issues remain. However, one cannot but be excited by the convergence in 500 customers, because, for the M/M/1 queue, it typically takes 100 000 observations just to get reasonable estimates of performance at *one* parameter value! (e.g., see [65], [50]). This seeming contradiction (ability to optimize the parameter faster than we can estimate the system performance) is an intriguing empirical result that invites further research (see the discussion in Section IX).

While the convergence of these single-run optimization algorithms remains an open issue in general, a few preliminary results have recently been obtained. Using results from adaptive control theory, Leung and Suri [47] recently presented a convergence proof for a simple single-run optimization scheme. This point may be relevant to the audience of this paper, because it shows that people with knowledge of conventional (i.e., continuous) system theory may be able to contribute much needed new ideas in the area of DEDS as well. Glynn [27] presents a convergence proof for an algorithm which is also single-run. By using a clever trick where parameter updates are done only at the end of every two regenerative cycles, he eliminates bias from the estimates, but a concern with this method may be the length of a regenerative cycle in practical systems, which could be hundreds or even thousands of customers in a queueing network problem. (With the PARMSR algorithm in [63] each update step was applied every 5 customers.) Fu [23] extends Glynn's method to prove convergence of a single-run optimization algorithm for a GI/G/1 queue. However, his algorithm still needs to wait till the end of one regenerative cycle before it updates the parameter. Recently Wardi [66] has presented a convergence proof for a single-run optimization algorithm operating on a GI/G/1 queue, using however, a nonstandard measure of convergence, and an alogrithm that requires increasingly longer run lengths between parameter updates. Proof of "convergence with probability 1" for the PARMSR algorithm (i.e., updating every 5 customers), even for a simple queueing system such as M/M/1, remains an open question.

PA also provides a tool for distributed optimization algorithms. This might be particularly useful in communication networks where nodes could update their parameters asynchronously based on local information; applications of this idea can be found in [14], [16], [18].

VIII. PA for Discrete Parameters

In practical systems, many parameters (such as buffer sizes, or number of servers at a station) are discrete in nature. Although this paper has focused on IPA, which by its nature can be applied only to continuous parameters, from a historical perspective the subject of discrete parameters has special significance for PA. The origins of PA are in a paper by Ho, Eyler, and Chien [40] on buffer size optimization for a production line. The approach in that paper seemed to be a special heuristic designed to solve a particular problem. However, subsequent work by the same authors [41] plus the many other authors cited throughout this paper, took the ideas in [40] and developed them into a general body of theory and algorithms. Here we will cover the subject of discrete parameters only briefly.

A. FPA for Discrete Parameters

The application of PA to sensitivity analysis with respect to a discrete parameter θ is concerned with estimating the performance of the system at the values $\theta + 1$ and/or $\theta - 1$, while observing just the nominal sample path (at θ). Clearly, in this case, by the very nature of the parameter, one is concerned not with gradient estimation, but finite differences. Also, the perturbations introduced in the sample path cannot be made arbitrarily small, they are of size 1, which can often be relatively large compared with the nominal parameter value. These perturbations can introduce significant changes in event order in the sample path, and thus, in order to propagate these perturbations, one

has to invoke FPA rules. PA for discrete parameters can thus be quite involved (as can be seen from the original paper [40]). We do not recommend beginning researchers on PA to look into this aspect until they are thoroughly familiar with the basics of IPA and some of the other recent extensions. Also, the theory of PA for discrete parameters, being dependent on FPA, is not rigorously developed, so it can properly be considered a set of heuristic algorithms at the present time. Nevertheless, the empirical results in [40], [59] show that reasonable estimates of perturbed system performance can be obtained with discrete parameters from a single sample path. There is plenty of scope for research here (see Section IX).

B. The Augmented Chain Approach

A somewhat different approach for discrete parameters has been developed recently by Cassandras and Strickland [15], [17] and applied to scheduling problems [60]. Although it arises from the authors' earlier work on PA, it might well be classified as a separate approach because of some structural differences in its basis. Applicable to continuous time Markov chains, it is termed the "augmented chain" approach. The idea is that for such systems, the sample path for the system with parameter $\theta + 1$ or $\theta - 1$ differs only occasionally from that with parameter θ. Thus while simulating the nominal system, with a little extra computational effort, one can simultaneously be simulating the system at $\theta + 1$ and/or $\theta - 1$. This idea can also be extended to the case where one is observing an actual system. Unlike FPA, this approach is exact (in that it can be rigorously proven to give consistent estimates). We mention this method here as it appears to be more promising than FPA for design analysis of systems that can be modeled as Markov chains, and also because we wish to refer to it in our discussion of research topics later.

IX. Research Issues

As we mentioned in the introduction, PA is a very young discipline by scientific standards and many interesting open problems need to be addressed. We mention here a few that we consider to be significant. Others can be found in the body of this paper as well as many of the recent works on PA.

A. Understanding and Expanding the Domain of IPA

Although IPA may not be consistent for many classes of systems and/or performance measures, when it is consistent it combines several desirable features: it is the simplest to understand and implement, can be written as a "universal" algorithm for any DEDS [58] and has the best variance properties [65]. It is of great interest then, to understand and expand the domain of applicability of IPA as much as possible. First, continuing to prove the consistency of IPA for common classes of systems provides a lot of insight and understanding about the nature of IPA. Thus, more results along the lines of [65], [7], [10], [68] would be useful. What can be said for more general systems, e.g., classes of GSMPs [29]? Second, and possibly more significant, we have seen that by suitable transformations, consistent IPA algorithms can be developed for systems in which IPA appears inconsistent at first glance [37], [25], [26]. Can we find "preferred" representations for classes of stochastic processes so that IPA will be consistent in these representations?

B. FPA and PA for Discrete Parameters

At the present time, implementing FPA for general systems remains in the realm of heuristics. EPA works for continuous time Markov processes (albeit, inefficiently if the state space is large), and one can view FPA as an approximate implementation of EPA where various states have been aggregated. However, a better understanding of FPA is needed, along with rigorous statements about its accuracy. The use of PA for discrete parameters also has much room for work. The difficulty in dealing with such parameters is that a change in the parameter (e.g., increase in buffer size) can generate a sample path that is impossible in the nominal path, so predictions based on the nominal path are difficult. The augmented chain method is one way of getting around this. Are there other PA approaches (e.g., variations of SPA or EPA) that could be used with effect?

C. Efficiency of the Methods

More extensive work can be done, both empirical and theoretical, on the efficiency of various PA and other gradient estimation methods. Here "efficiency" should include measures of MSE as well as total computer time required for each approach. An open question (see Section VII-A) is, how well do CRN methods compare with IPA? For EPA, the question of efficiency is complicated: the perturbed path it develops is of shorter length than the nominal, yet correlated, with it on segments. What can we say about the theoretical efficiency of EPA? In regard to LR methods, can we derive analytical expressions for the efficiency of IPA compared with LR in the steady state estimation case? How does IPA compare with LR methods for transient estimation?

D. Using PA for Single-Run Optimization

The experimental results obtained with such algorithms are exciting [49], [63]–[65]. What can we say about the convergence of such methods? Why do the experimental results indicate extremely fast convergence, even faster than might be expected? Is it because current statistical theory on convergence of stochastic approximation algorithms is based on long-sample characteristics, while single-run algorithms appear to converge so quick that they need short-sample analysis of their rates? Understanding these issues and developing well-tested algorithms for multiple parameter problems would be practically very useful.

E. Getting More Information from a Sample Path

Quite a bit of the recent development in DEDS analysis is related to the question: how can we get more information out of this set of observations? When PA was first introduced, many researchers were skeptical, even unbelieving, that it could possibly work. (How can you get something for nothing, or for almost nothing?) In retrospect, it seems less surprising. A DEDS has a lot of structure to it, which is reflected in its sample path (e.g., Fig. 8). By exploiting this structure, a great deal of information may be gleaned from the observations emanating from the system. Most of the traditional "output analysis" approaches had looked at a

DEDS as a "black box" and only performed statistical analysis on the *outputs* of the "box." The PA methods are all means of extracting additional information from a sample path. The same is true of the LR methods and the augmented chain method. This points out a general research direction: techniques for getting more information out of DEDS sample paths (see [43]).

X. Conclusion: Impact of "Dynamic Systems" Viewpoint

As a final, somewhat philosophical remark relevant to the audience of this paper, it is worth noting that many of the recent developments in DEDS have come from people with a "systems" background (e.g., dynamic systems, control theory, signal processing, etc.) The development of PA is firmly rooted in system dynamics. A preliminary convergence result for single-run optimization is based on adaptive control theory [47]. In the past the "continuous variable dynamic systems" (CVDS) people and the DEDS people have tended to have separate communities, conferences, and journals. It seems that both communities can benefit by serious exchange of ideas and by researchers crossing the traditional boundaries to work on problems of the other side. In particular, given the growing importance of DEDS in today's world, people with CVDS background should view this as an opportunity to explore and contribute in the DEDS area. The development and maturing of PA is but one instance of the potential of such a migration over to DEDS from people schooled in CVDS. Other instances can be found in some of the accompanying papers in this special issue. It is hoped that this paper will stimulate many more persons into investigating PA in particular, and the area of DEDS in general.

Acknowledgment

This paper builds on the work of many researchers who have developed the field of PA, and without all their contributions this paper would never have come to pass. Most notably, acknowledgment is due to Prof. Y. C. Ho whose early insights and perseverance persuaded all of us to devote our energies to exploring this topic. Much of the development here on the single server queue is based on joint work with Prof. M. A. Zazanis. I am indebted to him for many discussions and insights on that system as well as on PA in general.

References

[1] V. M. Aleksandrov, V. I. Sysoyev, and V. V. Shemeneva, "Stochastic optimization," *Eng. Cybern.*, vol. 5, pp. 11–16, 1968.
[2] M. Bello, "The estimation of delay gradient for purpose of routing in data communication networks," S. M. Thesis, Dept. Elec. Eng. and Comp. Sci., M.I.T., Cambridge, MA, Sept. 1976.
[3] G. E. P. Box, "Use and abuse of regression," *Technometrics*, vol. 8, pp. 625–629, 1966.
[4] X. R. Cao, "Convergence of parameter sensitivity estimates in a stochastic experiment," *IEEE Trans. Automat. Contr.*, vol. AC-30, pp. 834–843, 1985.
[5] ——, "First order perturbation analysis of a simple multiclass finite source queue," *Performance Evaluation*, vol. 7, pp. 31–41, 1987.
[6] ——, "Sensitivity estimates based on one realization of stochastic system," *J. Statistical Computation and Simulation*, vol. 27, pp. 211–232, 1987.
[7] ——, "On a sample performance function of Jackson queueing networks," *Oper. Res.*, vol. 36, pp. 128–136, 1988.
[8] X. R. Cao and Y. C. Ho, "Perturbation analysis of Sojourn time in queueing networks," in *Proceedings of the 22nd IEEE Conf. on Decision and Control*, (San Antonio, TX), pp. 1025–1029, Dec. 1983.
[9] ——, "Estimating Sojourn time sensitivity in queueing networks using perturbation analysis," *J. Optimization Theory and Applications*, vol. 53, pp. 353–375, 1987.
[10] ——, "Sensitivity analysis and optimization of throughput in a production line with blocking," *IEEE Trans. Automat. Contr.*, vol. 32, no. 11, pp. 959–967, 1987.
[11] C. G. Cassandras, "On-line optimization for a flow control strategy," *IEEE Trans. Automat. Contr.*, vol. AC-32, no. 11, pp. 1014–1017, 1987.
[12] C. G. Cassandras and Y. C. Ho, "An event domain formalism for sample path perturbation analysis of discrete event dynamic systems," *IEEE Trans. Automat. Contr.*, vol. AC-30, no. 12, pp. 1217–1221, 1985.
[13] C. G. Cassandras and W. B. Gong, "Robustness of perturbation analysis algorithm for queueing systems," manuscript, Univ. of Mass.-Amherst, 1988.
[14] C. G. Cassandras and J. I. Lee, "Applications of perturbation techniques to optimal resource sharing in discrete event systems," in *Proc. 1988 American Control Conf.*, pp. 450–455, June 1988.
[15] C. G. Cassandras and S. G. Strickland, "An 'augmented chain' approach for on-line sensitivity analysis of Markov processes," in *Proc. of the 26th IEEE Conf. Decision and Control*, pp. 1873–1880, 1987.
[16] ——, "Perturbation analytic methodologies for design and optimization of communication networks," *IEEE J. Selected Areas Commun.*, pp. 158–171, 1988.
[17] ——, "On-line sensitivity analysis of Markov chains," to appear in *IEEE Trans. Automat. Contr.*, 1989.
[18] C. G. Cassandras, M. V. Adibi, and D. Towsley, "Distributed routing with on-line marginal delay estimation," in *Proc. IEEE INFOCOM '88*, pp. 603–612, Mar. 1988.
[19] L. Chen, personal communication, 1988.
[20] L. Chen and R. Suri, "Consistency of IPA for tandem queueing networks," manuscript, Dept. Ind. Eng., Univ. of Wisconsin-Madison, 1988.
[21] G. Cohen, P. Moller, J. P. Quadrat, and M. Viot, "Algebraic tools for the performance evaluation of discrete event systems," this issue.
[22] M. A. Crane and A. J. Lemoine, *An Introduction to the Regenerative Method for Simulation Analysis*. New York, NY: Springer, 1977.
[23] M. C. Fu, "Convergence of a stochastic approximation algorithm for the GI/G/1 queue using infinitesimal perturbation analysis," manuscript, Harvard University, Div. of Applied Sci., 1988.
[24] G. S. Fishman, *Principles of Discrete Event Simulation*. New York, NY: Wiley, 1978.
[25] P. Glasserman, "IPA analysis of a birth-death process," *Oper. Res. Lett.*, vol. 7, no. 1, 1988.
[26] W. B. Gong and P. Glasserman, "Infinitesimal perturbation analysis for the M/G/1/K queue," in *Proc. 27th IEEE Conf. Dec. and Control*, (Austin, TX), pp. 1114–1118, Dec. 1988.
[27] P. W. Glynn, "Optimization of stochastic systems," in *Proc. 1986 Winter Simulation Conf.*, 1986.
[28] ——, "Construction of process-differentiable representations for parametric families of distributions," Tech. Rep., Mathematics Research Center, University of Wisconsin-Madison, 1987.
[29] ——, "A GSMP formalism for discrete-event systems," this issue.
[30] P. Glynn and J. L. Sanders, "Monte Carlo optimization in manufacturing systems: two new approaches," in *Proc. of ASME Computer in Engineering Conference*, (Chicago, IL), 1986.
[31] W. B. Gong, "Smoothed perturbation analysis for a routing problem in communication networks," manuscript, 1988.
[32] W. B. Gong and Y. C. Ho, "Smoothed (conditional) perturbation analysis of discrete event dynamic systems," *IEEE Trans. Automat. Contr.*, vol. AC-32, no. 10, pp. 858–866, 1987.
[33] P. Heidelberger, X. R. Cao, M. Zazanis, and R. Suri, "Convergence properties of infinitesimal perturbation analysis estimates," *Management Sci.*, vol. 34, no. 11, pp. 1281–1302, Nov. 1988.

[34] Y. C. Ho, "Performance evaluation and perturbation analysis of discrete event dynamic systems," *IEEE Trans. Automat. Contr.*, vol. AC-32, no. 7, pp. 563-572, 1987.
[35] ——, "Scanning the issue," this issue.
[36] Y. C. Ho and X. R. Cao, "Perturbation analysis and optimization of queueing networks," *J. Optimiz. Theory Appli.*, vol. 40, no. 4, pp. 559-582, 1983.
[37] ——, "Performance sensitivity to routing changes in queueing networks and flexible manufacturing systems using perturbation analysis," *IEEE J. Robotics Automat.*, vol. 1, pp. 165-172, Dec. 1985.
[38] Y. C. Ho and C. Cassandras, "A new approach to the analysis of discrete event dynamic systems," *Automatica*, vol. 19, no. 2, pp. 149-167, 1983.
[39] Y. C. Ho, X. R. Cao, and C. Cassandras, "Infinitesimal and finite perturbation analysis for queueing networks," *Automatica*, vol. 19, pp. 439-445, 1983.
[40] Y. C. Ho, M. A. Eyler, and T. T. Chien, "A gradient technique for general buffer storage design in a serial production line," *Int. J. Prod. Res.*, vol. 17, no. 6, pp. 557-580, 1979.
[41] ——, "A new approach to determine parameter sensitivities of transfer lines," *Management Sci.*, vol. 29, no. 6, pp. 700-714, 1983.
[42] Y. C. Ho and S. Li, "Extensions of infinitesimal perturbation analysis," *IEEE Trans. Automat. Contr.*, vol. AC-33, no. 5, pp. 427-438, 1988.
[43] Y. C. Ho, S. Li, and P. Vakili, "On the efficient generation of discrete event sample paths under different parameter values," *Mathematic and Computation in Simulation*, to appear in 1988.
[44] K. Inan and P. Varaiya, "Algebras of discrete event models," this issue.
[45] L. Kleinrock, *Queueing Systems, Vol. I: Theory.* New York, NY: Wiley, 1975.
[46] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *Ann. Math. Stat.*, vol. 23, pp. 462-466, 1952.
[47] Y. T. Leung and R. Suri, "Convergence of a single run simulation optimization algorithm," in *Proc. American Control Conf.*, 1988.
[48] M. S. Meketon, "A tutorial on optimization on simulations" (unpublished), presented at Winter Simulation Conf., 1983.
[49] ——, "Optimization in simulation: A survey of recent results," in *Proc. Winter Simulation Conf.*, pp. 58-67, 1987.
[50] M. S. Meketon and P. Heidelberger, "A renewal theoretic approach to bias reduction in regenerative simulations," *Management Sci.*, vol. 28, no. 2, pp. 173-181, 1982.
[51] M. I. Reiman and A. Weiss, "Sensitivity analysis via likelihood ratios," in *Proceedings of the 1986 Winter Simulation Conf.*, pp. 285-289, 1986.
[52] R. Righter and J. Walrand, "Distributed simulation of discrete-event systems," this issue.
[53] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Stat.*, vol. 22, pp. 400-407, 1951.
[54] R. Y. Rubinstein, *Monte Carlo Optimization, Simulation, and Sensitivity Analysis of Queueing Networks.* New York, NY: Wiley, 1986.
[55] ——, "The score function approach of sensitivity analysis of computer simulation models," *Math. and Computation in Simulation*, vol. 28, pp. 351-379, 1986.
[56] A. Segall, "The modeling of adaptive routing in data communication networks," *IEEE Trans. Commun.*, vol. COM-25, pp. 85-95, Jan. 1977.
[57] R. Suri, "Implementation of sensitivity calculations on a Monte-Carlo experiment," *J. Optimiz. Theory Appl.*, vol. 40, no. 4, pp. 625-630, Aug. 1983.
[58] ——, "Infinitesimal perturbation analysis for general discrete event systems," *J. ACM*, vol. 34, no. 3, pp. 686-717, July 1987.
[59] R. Suri and X. Cao, "The phantom customer and marked customer methods for optimization of closed queueing networks with blocking and general service times," *ACM Performance Evaluation Review*, vol. 12, no. 3, pp. 243-256, Aug. 1983.
[60] J. B. Suk and C. G. Cassandras, "Optimal scheduling of two competing queues with blocking," in *Proc. 27th IEEE Conf. Dec. and Control*, to appear in Dec. 1988.
[61] R. Suri and G. W. Diehl, "A variable buffer size model and its use in analyzing closed queueing networks with blocking," *Management Sci.*, vol. 32, no. 2, pp. 206-224, 1986.
[62] R. Suri and J. W. Dille, "A technique for on-line sensitivity analysis of flexible manufacturing systems," *Ann. Oper. Res.*, vol. 3, pp. 381-391, 1985.
[63] R. Suri and Y. T. Leung, "Single run optimization of discrete event simulations: An empirical study using the M/M/1 queue," to appear in *IIE Transactions.*
[64] ——, "Single run optimization of a SIMAN model for automatic assembly systems," in *Proc. Winter Simulation Conf.*, pp. 738-748, 1987.
[65] R. Suri and M. A. Zazanis, "Perturbation analysis gives strongly consistent sensitivity estimates for the M/G/1 queue," *Management Sci.*, vol. 34, no. 1, pp. 39-64, Jan. 1988.
[66] Y. Wardi, "Simulation based stochastic algorithm for optimizing GI/G/1 queues," manuscript, Ben Gurion University of the Negev, 1988.
[67] M. A. Zazanis and R. Suri, "Comparison of perturbation analysis with conventional sensitivity estimates for stochastic systems," 1985, submitted to *Oper. Res.*
[68] ——, "Estimating first and second derivatives of response time for G/G/1 queues from a single sample path," 1986, submitted to *Queueing Systems.*
[69] M. A. Zazanis, "Unbiasedness of infinitesimal perturbation analysis estimates for higher moments of the response time of an M/M/1 queue," to appear in *Oper. Res.*
[70] B. P. Ziegler, "DEVS representation of dynamical systems: event-based intelligent control," this issue.
[71] M. C. Fu and Y. C. Ho, "Using perturbation analysis for gradient estimation, averaging and updating in a stochastic approximation algorithm," in *Proc. 1988 Winter Simulation Conf.*, pp. 509-517.

Rajan Suri (Member, IEEE) received the B.S. degree from Cambridge University, England, and the M.S. and Ph.D. degrees from Harvard University, Cambridge, MA.

He is Professor of Industrial Engineering at the University of Wisconsin-Madison, where he is also one of the faculty responsible for the Manufacturing Systems Engineering Program. He is also a principal of Network Dynamics Inc., a firm specializing in software for manufacturing systems. He has been instrumental in extending the theories of queueing networks and perturbation analysis for manufacturing applications, and is the author of over 50 technical publications, several books, and edited volumes.

Dr. Suri is Associate Editor of the *Journal of Manufacturing Systems* and of the *International Journal of Flexible Manufacturing Systems.* In 1981, he received the Eckman Award from the American Automatic Control Council for outstanding contributions in his field.

Queueing Models for Systems with Synchronization Constraints

FRANÇOIS BACCELLI AND ARMAND M. MAKOWSKI, MEMBER, IEEE

Invited Paper

In this paper, we consider queueing models which occur naturally in the study of a class of resource sharing problems under synchronization constraints such as resequencing and Fork-Join primitives. These queueing models are amenable to a representation in terms of a state recursion. The proposed methods of analysis are complementary and draw on classical ideas of queueing theory as well as on mathematical tools from the theory of stochastic ordering and ergodic theory. The state recursion is at the center of all aspects of the analysis, be it for developing the exact solutions, obtaining bounds on system performance or establishing the stability conditions. The ideas are illustrated on simple models of resequencing and Fork-Join synchronization, with emphasis put on deriving computable bounds on the performance measures.

I. Introduction

Although *synchronization* constraints are inherent to the operation of many computer, communication, and production systems, their impact on system performance is far from being well understood. This may be partially attributed to the penury of models which meaningfully incorporate the synchronization constraints of interest, and which are nevertheless analytically tractable. This difficulty is of course not specific to this class of applications.

Numerous attempts have been made at modeling and analyzing the effects of (hard) synchronization constraints. We shall not report on this activity here in any detail, but in the interest of developing some perspective on the approach taken in this paper, we briefly mention one of the most common approaches, namely the one based on *Petri nets* [1], [15]. We recall that the theory of Petri nets was initiated in response to modeling needs in protocol validation and that its ideas were especially adapted to handle process synchronization and provide a very general framework for accurately describing the system logical functions. Although Petri net models could in principle be used to represent most notions of synchronization encountered in applications, quantitative models based on Petri nets typically exhibit high dimensionality and their analysis yields few structural properties to assist in the performance evaluation.

On the other hand, many of the interesting applications involving synchronization constraints are concerned with problems of *resource sharing*. It is widely recognized that such problems are adequately described in terms of queueing network models which have a long tradition of providing quantitative insights into system performance [31], [33]. With this in mind, we propose to investigate the performance issues associated with synchronization within the confines of queueing network modeling. However, the explicit incorporation of the synchronization constraints in these queueing models often destroys important properties, such as the *product form* [12], [29] and *insensitivity* [4] properties, which have fueled the development of various methodologies for the performance evaluation of data networks. Consequently, product-form networks do not provide a natural framework for attacking the performance evaluation of systems with synchronization constraints. This state of affairs points to the need of expanding the theory of queueing networks into new directions if a quantitative theory of such systems is to be developed.

In this paper, we take a step along these lines by restricting attention to a class of queueing networks which arises in modeling interesting synchronization mechanisms present in many applications. Loosely speaking, we can characterize the queueing systems of interest as the ones where the time behavior of a natural *state variable* is given through a *recursion* endowed with certain *monotonicity* and *convexity* properties. Typical state variables include customer waiting time or response time, and appropriate generalization thereof. This approach is very much in the spirit of

Manuscript received April 21, 1988; revised September 20, 1988. The work by F. Baccelli was initially supported by a grant from the Minta Martin Aeronautical Research Fund of the College of Engineering at the University of Maryland. The work by A. M. Makowski was supported in part through ONR Grant N00014-84-K-0614 and NSF Grant ECS-83-51836. Research grants were contributed by AT&T Bell Laboratories and GM Laboratories. The research environment provided by the Systems Research Center was made possible through the National Science Foundation's Engineering Research Centers Program NSFD CDR 88-03012. This paper evolved from a set of lecture notes prepared by F. Baccelli for a tutorial on queueing methods presented at the Cours INRIA-C3 d'Algorithmique Distribuée, La Colle-sur-Loup, France, April 1987.

F. Baccelli is with INRIA-Centre de Sophia Antipolis, 06565 Valbonne Cédex, France.

A. M. Makowski is with the Electrical Engineering Department and Systems Research Center, University of Maryland, College Park, MD 20742, USA.

IEEE Log Number 8825171.

0018-9219/89/0100-0138$01.00