IMPROVING SIMULATION ACCURACY THROUGH THE USE OF SYNTHETIC ALIGNMENT INTERVALS

Jeffrey P. Buzen
Independent Consultant
12-607 Mountain Laurels Dr.
Nashua, NH 03062, USA
E-mail: buzen@post.harvard.edu

KEYWORDS

Stochastic simulation, queueing systems and network models, stochastic models, Markov models.

ABSTRACT

Monte Carlo simulations are used in many disciplines to evaluate the steady state distributions of stochastic models. This paper introduces a new procedure for improving the accuracy of such simulations. The approach is based on constructing synthetic "alignment intervals" that are appended to the output of the original simulations, creating extended simulations whose output conforms to certain mathematical relationships. Satisfying these relationships is shown to be sufficient to guarantee that the underlying steady state distributions have been computed accurately.

INTRODUCTION

Variability is present in the behavior of many real world systems. It is often convenient to regard this variability as the physical manifestation of some underlying stochastic process. This assumption leads to the creation of stochastic models of system behavior. These models can then be evaluated through analytic techniques, numerical methods, or Monte Carlo simulation.

In many cases, analysts are interested in determining the steady state distribution of the underlying stochastic process. When Monte Carlo simulation is being used for this purpose, the Ergodic Theorem insures (with probability one) that - if the simulation runs "long enough", and if the random number generator is "good enough" - the output of the simulation will provide an accurate characterization of the underlying steady state distribution. The first few sections of this paper present a new procedure for testing the output of a simulation to determine if it has, in fact, provided an accurate characterization of the underlying steady state distribution.

The tests are based on the equivalence of two different methods for deriving the equations that characterize this distribution. The first method employs the classical approach of setting the time derivative of the transient distribution equal to zero. The second method is based on a new approach: a direct analysis of observable quantities that simulation programs actually generate. Both methods are shown to produce exactly the same equations for characterizing the steady state distribution. This provides the rationale for the new testing procedure. A simple example is used to illustrate the issues involved.

The discussion then turns to cases where the output of the simulation does not satisfy the conditions sufficient to guarantee accuracy. It is shown that accuracy can be improved in such cases by appending specially constructed "alignment intervals" to the end of the original output trajectories. Since system behavior during these specially constructed alignment intervals is driven by external calculations rather than calls on a random number generator, legitimate philosophical concerns arise regarding the validity of this approach. These issues are addressed in the final sections of this paper.

EXAMPLE – SIMPLE QUEUEING NETWORK

Begin by considering the simple queueing network shown in Figure 1. There are two queues, each served by a single server. A total of three customers circulate around the network, cycling between one server and the other.

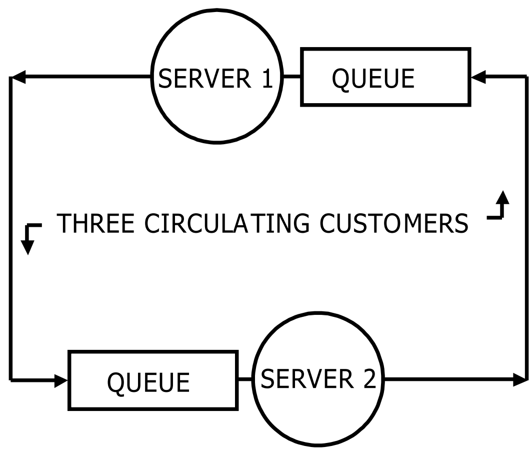


Figure 1: Simple Queueing Network

Proceedings 21st European Conference on Modelling and Simulation
Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007
ISBN 978-0-9553018-2-7 / ISBN 978-0-9553018-3-4 (CD)

Assume that the service times at servers 1 and 2 are determined by sampling from exponentially distributed random variables with means of 2 seconds and 4 seconds respectively. The queueing network in Figure 1 can thus be regarded as the realization of a stochastic process. The structure of this process is identical to that of an M/M/1/3 queue

Let P(n) be the steady state probability that the number of customers at server 1 is equal to n (for n = 0, 1, 2, 3). Analytic expressions for P(n) are well known to queueing theorists. However, suppose for purposes of this example that P(n) is not known but is instead being evaluated through a Monte Carlo simulation.

Figure 2 depicts two possible trajectories that such a simulation might generate. Both trajectories are exactly 30 seconds in duration. This is, of course, too short an interval to obtain reliable estimates of an underlying steady state distribution, but it is still sufficient for purposes of this discussion.

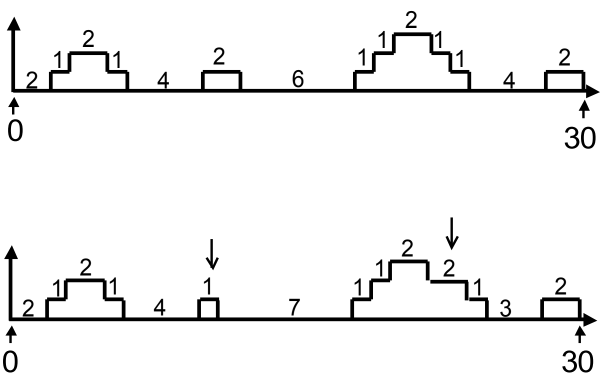


Figure 2: Two Possible Simulated Trajectories

Begin by considering the values of P(n) that are associated with the upper trajectory.

There are zero customers at server 1 for a total of 2 + 4 + 6 + 4 = 16 seconds. Thus $P(0) = {}^{16}/_{30}$.

Similarly, there is one customer at server 1 for a total of 1 + 1 + 2 + 1 + 1 + 2 = 8 seconds. Thus $P(1) = {}^{8}/_{30}$.

Likewise, there are two customers at server 1 for a total of 2 + 1 + 1 = 4 seconds, implying $P(2) = {}^{4}/_{30}$.

Finally, there are three customers at server 1 for a total of 2 seconds, which implies $P(3) = {}^{2}/_{30}$.

As already noted, the values of P(n) for the underlying stochastic process can also be expressed analytically in this simple case. Using the standard notational conventions of queueing theory, assume that service times at server 1 are exponentially distributed with mean $1/\mu$ and that service times at server 2 are exponentially distributed with mean $1/\lambda$. Then the values of P(n) can be expressed as well known functions of the ratio λ/μ.

$$P(0) = 1/[1 + \lambda/\mu + (\lambda/\mu)^2 + (\lambda/\mu)^3] \quad (1)$$

$$P(1) = (\lambda/\mu)\, P(0) \quad (2)$$

$$P(2) = (\lambda/\mu)^2\, P(0) \quad (3)$$

$$P(3) = (\lambda/\mu)^3\, P(0) \quad (4)$$

Note that Equations (1) – (4) pertain to the steady state distribution of the stationary stochastic process associated with Figure 1. The parameters of this process are μ and λ.

In the example being considered here, $1/\mu = 2$ seconds (the mean service time at server 1) and $1/\lambda = 4$ (the mean service time at server 2). The ratio λ/μ is thus equal to $(1/4) \div (1/2) = ½$. Replacing λ/μ by ½ in Equations (1) – (4) yields the following solution for the steady state distribution.

$$P(0) = 1/[1 + (½) + (½)^2 + (½)^3] = {}^{8}/_{15} = {}^{16}/_{30}$$

$$P(1) = {}^{1}/_{2} \text{ x } {}^{8}/_{15} = {}^{4}/_{15} = {}^{8}/_{30}$$

$$P(2) = (½)^2 \text{ x } {}^{8}/_{15} = {}^{2}/_{15} = {}^{4}/_{30}$$

$$P(3) = (½)^3 \text{ x } {}^{8}/_{15} = {}^{1}/_{15} = {}^{2}/_{30}$$

Note that the attained state distribution actually observed in the upper trajectory of Figure 2 is identical to the theoretical steady state distribution computed using the mathematical equations derived from the underlying stochastic process.

This perfect alignment between observed and theoretical results is, of course, the goal of Monte Carlo simulation. However, in this case the 30 second simulation interval seems much too short to expect this goal to be satisfied.

Is the observed alignment between the simulator's output and the analytic solution merely the consequence of a highly specialized and artfully constructed trajectory, or is it the result of fundamental principles that can be generalized to a broad class of possible trajectories?

As this paper will demonstrate, fundamental principles are indeed involved, and these principles extend well beyond the simple example that has just been presented.

CLASSICAL ANALYSIS APPROACH

It is helpful to begin by returning to first principles and reviewing the classical approach for deriving the steady state distribution of any continuous time Markov process.

The first step is to identify the states of the Markov process and the "permissible" transitions between these states. Essentially, a transition is "permissible" if the specification of the Markov process allows it to occur with a probability that is greater than zero.

In the example illustrated in Figure 1, the state of the process will simply be defined as an integer that represents the number of customers at server 1. Thus, state can be equal to 0, 1, 2 or 3.

Permissible transitions occur when a single customer completes service at one of the servers and proceeds to the queue at the other server. This results in a state change of plus or minus one. The four possible states and the six permissible transitions are illustrated in Figure 3.

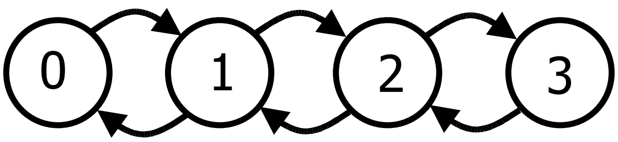


Figure 3: State Transition Diagram for Figure 1

The state of this Markov process at any time t can be represented by a random variable with distribution P(n,t). In other words, P(n,t) is the probability that the Markov process is in state n at time t. P(n,t) is a function of t that depends on the initial state of the process at time t = 0 and on the values of the parameters (in this case, μ and λ.).

Intuitively, it is reasonable to believe that, if the value of t becomes large enough, the dependence of P(n,t) on the initial state will become negligible and P(n,t) will become completely independent of t. Stochastic process that conform to these intuitive notions are said to be "ergodic".

From a mathematical perspective, the condition that P(n,t) is independent of t can be made precise by stating that the derivative of P(n,t) with respect to t is equal to zero for each value of n. Since this is a Markov process, system state at time $t + \Delta t$ depends only system state at time t and on the parameters μ and λ. These observations make it possible to write down the derivative of P(n,t) with respect to t for each value of n. Setting each derivative equal to zero yields the following set of four equations that the steady state distribution must satisfy:

$$\mu P(0) = \lambda P(1) \tag{5}$$

$$[\mu + \lambda]P(1) = \mu P(0) + \lambda P(2) \tag{6}$$

$$[\mu + \lambda]P(2) = \mu P(3) + \lambda P(3) \tag{7}$$

$$\mu P(3) = \lambda P(2) \tag{8}$$

Equations derived in this manner are commonly referred to global balance equations. When combined with a "normalization equation" expressing the fact that the sum of P(n) over all values of n must be equal to 1, it becomes possible to solve these equations for P(n). This yields the steady state distribution that has already been presented in Equations (1) – (4).

ALTERNATIVE ANALYSIS APPROACH

In the preceding section, the global balance equations (5) – (8) were obtained by setting certain derivatives equal to zero. These same equations can also be derived through an alternate set of highly intuitive arguments that refer solely to the observable properties of the output generated by Monte Carlo simulations.

To proceed, note that the state transition diagram shown in Figure 3 can be adapted to track the progress of any simulation as it executes over time. Simply add a token to the diagram and assume that it moves from circle to circle via one of the arrows whenever the length of the queue at server 1 changes. The position of the token at any instant represents the current state of the system.

It should be immediately apparent that, during the course of any finite simulation interval, the number of transitions that the token makes out of a given state will be equal to the number of transitions it makes into that state. The only exceptions are the initial state, which has one extra transition out (just after the start of the simulation) and the final state, which has one extra transition in (just before the end of the simulation). If the initial and final states are identical, these two extra transitions will balance one another, implying that the number of transitions out is equal to the number of transitions in for all possible states. This condition is often referred to as "flow balance".

IMPLICATIONS OF FLOW BALANCE

To examine the implications of flow balance from a mathematical perspective, let C(n) denote the number of times during the execution of a simulation that a customer completes service at server 1 while the system is in state n. Similarly, let A(n) denote the number of times that a customer completes service at server 2 while the system is in state n.

For state 0, transitions out occur only as a result of service completions at server 2 (causing a transition from state 0 to state 1). The number of times these transitions occur is equal to A(0). Similarly, the only transitions into state 0 occur while the system is in state 1 and the single customer at server 1 completes its service. The number of times these transitions occur is equal to C(1). The flow balance condition thus implies:

$$A(0) = C(1)$$

For state 1, there are two possible transitions in (a completion at server 2 while in state 0 or a completion at server 1 while in state 2). Similarly, there are two possible transitions out (a completion at server 1 while in state 1 or a completion at server 2 while in state 1). For state 2, flow balance implies:

$$A(1) + C(1) = A(0) + C(2)$$

Similar considerations regarding state 3 imply:

$$A(2) + C(2) = A(1) + C(3)$$

State 3 is similar to state 0:

$$C(3) = A(2)$$

The next step is to convert these four flow balance equations so they are expressed in terms of transition rates rather than raw counts. Suppose that T(n) is the amount of time the system spends in state n during the simulation interval. Then $T = T(0) + T(1) + T(2) + T(3)$ must equal the total length of the simulation interval. Also, T(n)/T must be equal to the proportion of time (during the simulation interval) that the system spends in state n. This quantity is denoted by P(n):

$$P(n) = T(n)/T$$

Simple algebra then implies that the four flow balance equations given above can be re-written as follows:

$$[A(0)/T(0)]\,P(0) = [C(1)/T(1)]\,P(1)$$

$$[A(1)/T(1) + C(1)/T(1)]\,P(1) = [A(0)/T(0)]\,P(0) + [C(2)/T(2)]\,P(2)$$

$$[A(2)/T(2) + C(2)/T(2)]\,P(2) = [A(1)/T(1)]\,P(1) + [C(3)/T(3)]\,P(3)$$

$$[C(3)/T(3)]\,P(3) = [A(2)/T(2)]\,P(2)$$

Note that these four equations are valid for any trajectory generated by any simulation that conforms to the state transition diagram illustrated in Figure 3. The only requirement is that the trajectory must satisfy flow balance: this is, the initial and final states must be the same. No distributional assumptions of any type are required.

EFFECT OF DISTRIBUTIONAL ASSUMPTIONS

In this particular example, service times at server 1 are assumed to be generated by sampling from an exponentially distributed random variable whose mean is $1/\mu$ seconds. In addition, service times at server 2 are assumed to be generated by sampling from an exponentially distributed random variable with mean $1/\lambda$. These distributional assumptions have important implications for the expected values of A(n)/T(n) and C(n)/T(n) that appear in the flow balance equations at the bottom of the preceding column.

Consider server 1 first. C(n) is the total number of requests completed by this server while the system is in state n (i.e., while there are n customers at server 1). Also, T(n) is the total time spent in state n. Thus, C(n)/T(n) is the conditional request completion rate at server 1 while the system is in state n.

Since service times at server 1 are assumed to be generated by sampling from a random variable with mean $1/\mu$, the unconditional completion rate at server 1, computed over all times that the server operates, is expected to equal μ. This is true regardless of the service time distribution. The additional fact that the service time distribution is exponential implies that the service completion process is memoryless (i.e., the probability that a completion will occur at any instant is independent of the amount of service that the customer currently being served has already consumed).

The memoryless property implies that conditional completion rates (conditioned on system state) will always be the same as the overall (unconditional) service completion rate. In other words, the quantity C(n)/T(n) is expected to equal μ for all values of n (for n = 1, 2 and 3). This condition, which can be verified by direct inspection of any trajectory, is referred to as completion rate homogeneity.

Applying the same argument to server 2, it also follows that the values of A(n)/T(n) are expected to equal λ for n = 0, 1 and 2. In other words, the completion rates at server 2 are also expected to be homogeneous

If the flow balance equations at the bottom of the preceding column are modified so that C(n)/T(n) is replaced by μ for n = 1, 2 and 3 and A(n)/T(n) is replaced by λ for n = 0, 1 and 2, these four equations become identical to Equations (5) – (8).

To summarize the main point, Equations (5) – (8) have just been derived under two very different sets of assumptions. First, it was shown that Equations (5) – (8) are satisfied by the steady state distribution of the Markov process associated with Figure 1. Then, it was shown that Equations (5) – (8) are also satisfied by the output of any simulation model based on Figure 1, provided the output satisfies the observable conditions of flow balance and homogeneous completion rates at servers 1 and 2.

Solving Equations (5) – (8) for P(n) will always yield Equations (1) – (4). It follows that both sets of assumptions are sufficient to generate exactly the same values of P(n).

APPLICATION TO SIMULATION OUTPUT

This observation leads to a simple test for the validity of simulation results. For example, suppose that a Monte Carlo simulation of the system illustrated in Figure 1 generates a trajectory that satisfies flow balance (initial state = final state). Suppose further that completion rates at server 1 and server 2 are homogeneous: that is, $C(n)/A(n) = \mu$ for n = 1, 2, 3 and $A(n)/T(n) = \lambda$ for n = 0, 1, 2. Then the attained distribution associated with this trajectory (i.e., the measured values of P(n)) will be identical to the values obtained by evaluating the analytic expression for the steady state distribution with parameters set equal to corresponding measured values.

The upper trajectory in Figure 2 provides a concrete example of such a trajectory. To verify that all the values of C(n)/T(n) are equal, note that the values of T(0), T(1), T(2) and T(3) are 16, 8, 4 and 2 respectively. In addition, the values of C(1), C(2), and C(3) are 4, 2 and 1 respectively. Thus, the value of C(n)/T(n) is equal to .5 for n = 1, 2 and 3.

Similarly, the values of A(0), A(1) and A(2) are equal to 4, 2 and 1 respectively. This implies that the values of A(n)/ T(n) are equal to .25 for n = 0, 1 and 2.

These simple relationships, combined with the observation that the initial and final state are the same, are sufficient to guarantee with 100% certainty that the attained distribution associated with this particular trajectory will be identical to the steady state distribution of underlying Markov process (with parameter values $\mu = .5$ and $\lambda = .25$).

Note that this conclusion does not require *a priori* knowledge of the steady state distribution associated with the underlying stochastic process. As long as flow balance is satisfied and the conditional completion rates for server 1 and server 2 are homogeneous, the simulation is guaranteed to produce an accurate characterization of the steady state distribution of the underlying stochastic process.

The Ergodic theorem and the Law of Large Numbers imply that, if the simulation runs long enough, this same conclusion can be asserted with probability one. However, as the upper trajectory in Figure 2 illustrates, it is not always necessary to run the simulation for a lengthy interval of time to insure that the conclusion is correct.

Of course, it is entirely possible for a Monte Carlo simulation to generate a trajectory that does not satisfy these goals. The lower trajectory in Figure 2 provides such an example. The two areas of difference between the upper and lower trajectories are marked with vertical arrows.

For the lower trajectory $\mu = (4+2+1)/(7+5+2) = .5$ and $\lambda = (4+2+1)/(16+7+3) = .25$. Since the parameters of the underlying Markov process are the same, the values of P(n) should be the same if the simulation has produced an accurate result. However, P(1) is 7/30 rather than 8//30, and P(2) is 5/30 rather than 4/30. The appearance of inaccurate results should not be surprising in this case since the completion rates at both server 1 and server 2 fail to satisfy homogeneity.

For server 1: C(1)/T(1) = 4/7
C(2)/T(2) = 2/5
C(3)/T(3) = 1/2

For server 2: A(0)/T(0) = 4/16
A(1)/T(1) = 2/7
A(3)/T(3) = 1/5

GENERALIZATIONS & EXTENSIONS

Thus far, this discussion has concentrated on the simple model shown in Figure 1 and on the associated Markov process characterized by the state transition diagram in Figure 3. It is a routine matter to apply exactly the same reasoning to any system being modeled by a continuous time Markov process. Simply draw the state transition diagram and use it to identify those conditional completion rates that will have to satisfy homogeneity conditions.

Once the homogeneity conditions are identified, simply run a simulation and test the generated trajectory to see if these conditions have been satisfied. If they have, and if flow balance is also satisfied, it can be stated with absolute certainty that the simulation has generated an accurate characterization of the steady state distribution of the underlying stochastic process.

Of course, the parameters of the underlying stochastic process must be determined by actually measuring the generated trajectory (rather than referring to the input parameters of the simulation program). Assuming that the simulation has run for a reasonably long interval of time, these measurements should yield values that are nearly identical to the parameter values specified as input to the simulation program.

Note that this entire process can be carried out in cases where the analytic expression for the steady state distribution is unknown.

SYNTHETIC ALIGNMENT INTERVALS

As this discussion has illustrated, analysts who use Monte Carlo simulations to evaluate steady state distributions should, under ideal circumstances, expect these simulations to generate trajectories that satisfy flow balance and various forms of homogeneity.

In cases where these conditions are not satisfied, the accuracy of the simulation can – in principle – be improved by extending the trajectory so that the conditions can be met.

There are two ways to extend the simulation. The first is to allow the simulation program to run for an additional interval of time. The Ergodic theorem and the Law of Large Numbers imply that, in the limit, the required assumptions will almost always be satisfied. The flow balance requirement can, in principle, be satisfied simply by modifying the simulation program so that it only terminates at points where the final state is equal to the initial state. However, the time needed to achieve these goals may be unacceptably long.

As an alternative, consider the possibility of appending specially constructed "alignment intervals" to the end of the original simulations. The objective of these alignment intervals is to bring the extended trajectory into compliance with the required assumptions. Since synthetic alignment intervals are not generated by calling upon the random number generator that drives the simulation program, it constitutes an artificially tailored appendage to the original simulation. Thus, care must be taken when interpreting the results of the extended simulation.

To understand some the issues involved, note that the information represented within a steady state distribution is only a subset of the information that can be obtained by analyzing the associated stochastic process. For example, in the case of a single server queue, the steady state distribution contains detailed information about queue length, but contains no information at all about the lengths of busy periods. In fact, two stationary stochastic processes with different internal structures (and different busy period distributions) can have exactly the same steady state distribution (Buzen 2006a).

The point is that appending a non-random alignment interval to the output of a Monte Carlo simulation is almost certain to interfere with various behavioral properties of the extended trajectory. Homogeneity and flow balance are not sufficient to insure that all properties of the stochastic process are preserved. Even though the extended trajectory is no longer "random", and even though it no longer exhibits all the properties of a faithful stochastic simulation, it is still possible to demonstrate that the extended trajectory does in fact provide a completely accurate characterization of the underlying steady state distribution. This observation provides the philosophical justification for appending a non-random alignment interval to the end of a stochastic trajectory.

In many cases, it is reasonable to assume that the original Monte Carlo simulation has generated a steady state distribution that is quite close to the exact solution. The approach outlined here can be used to inspect the output of the simulation and identify the most serious violations of homogeneity assumptions (while ignoring minor violations of homogeneity). Synthetic alignment intervals that correct only the most serious violations can then be appended to the original trajectory, resulting in an incremental improvement in accuracy.

Such a step-wise approach to improving simulation accuracy may ultimately prove to be the most effective procedure for applying these results in practice. It is important to note in this regard that simulation results can be surprisingly accurate even though the required homogeneity assumptions are not satisfied exactly (Suri 1983).

A specific example of an alignment interval that correctly adjusts the lower trajectory in Figure 2 is presented in the next section. General algorithms for the construction of synthetic alignment intervals have not yet been developed.

EXAMPLE

Figure 4 displays two trajectories, both having durations of 30 seconds. The upper trajectory in Figure 4 is identical to the lower trajectory in Figure 2.

Assume that the upper trajectory in Figure 4 has been generated by a Monte Carlo simulation of the stochastic process illustrated in Figure 1. As previously discussed, the completion rates at servers 1 and 2 are not homogeneous for this trajectory. In particular,

For server 1: $C(1)/T(1) = 4/7$
$C(2)/T(2) = 2/5$
$C(3)/T(3) = 1/2$

For server 2: $A(0)/T(0) = 4/16$
$A(1)/T(1) = 2/7$
$A(3)/T(3) = 1/5$

As already noted, the values of P(n) for this trajectory differ from the correct values obtained by setting λ/μ equal to ½ in Equations (1) – (4). In other words, the simulation results shown in the upper trajectory in Figure 4 do not accurately characterize the steady state distribution of the underlying stochastic process.

The lower trajectory in Figure 4 displays an alignment interval that can be used to correct this problem. If the lower trajectory is appended to the upper trajectory, the resulting 60 second trajectory will exhibit homogeneous service times for servers 1 and 2. This then implies that the values of P(n) associated with the 60 second trajectory are identical to the values of P(n) obtained from Equations (1) – (4) with $\lambda/\mu = ½$.

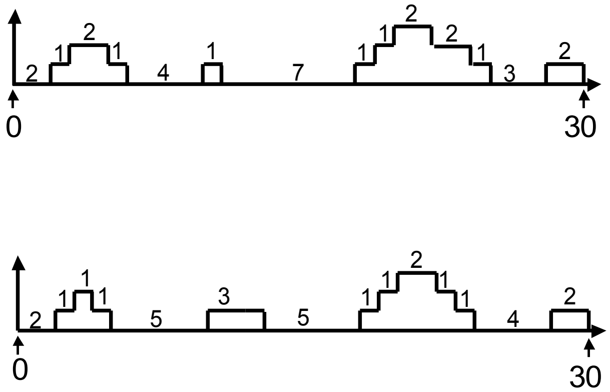


Figure 4: Simulated Trajectory & Alignment Interval

The following calculations, which are based on the complete 60 second trajectory, verify these remarks.

T(0) = 2 + 4 + 7 + 3 + 2 + 5 + 5 + 4
= 32

T(1) = 1 + 1 + 1 + 1 + 1 + 2 + 1 + 1 + 3 + 1 + 1 + 2
= 16

T(2) = 2 + 1 + 2 + 1 + 1 + 1 = 8

T(3) = 2 + 2 = 4

C(1) = 8

C(2) = 4

C(3) = 2

A(0) = 8

A(1) = 4

A(2) = 2

For server 1,

C(1)/T(1) = 8/16 = .5

C(2)/T(2) = 4/8 = .5

C(3)/T(3) = 2/4 = .5

For server 2,

A(0)/T(0) = 8/32 = .25

A(1)/T(1) = 4/16 = .25

A(2)/T(2) = 2/8 = .25

Since flow balance is satisfied and completion rates at both server 1 and server 2 are homogeneous, the values of P(n) must accurately characterize the steady state distribution of the underlying stochastic process.

To verify this conclusion, note that the values of P(n) from the 60 second trajectory are:

P(0) = T(0)/T = 32/60 = 8/15

P(1) = T(1)/T = 16/30 = 4/15

P(2) = T(2)/T = 8/30 = 2/15

P(3) = T(3)/T = 4/30 = 1/15

The parameters of the underlying stochastic process are

$\mu = (8 + 4 + 2)/(16 + 8 + 4) = .5$

$\lambda = (8 + 4 + 2)/(32 + 16 + 8) = .25$

If these values of λ and μ are substituted into the exact analytic solution given in Equations (1) – (4), a routine calculation demonstrates that the values of P(n) in the 60 second trajectory are correct.

In this example, the length of the alignment interval happens to be identical to the length of the original trajectory. This is not a requirement. For example, the time scale for all events in the alignment interval could be reduced by a factor of two so that the interval was only 15 seconds in length. Homogeneity would still be satisfied in the 45 second extended interval and the values of P(n) would still be computed correctly.

CONCLUSIONS

By testing the output of a Monte Carlo simulation to see if certain mathematical relationships are satisfied, it is possible to determine if the simulation has generated an accurate result (i.e., if the simulation has provided an accurate characterization of the steady state distribution of the underlying stochastic process).

If the output fails to pass the appropriate tests, the accuracy of the simulation can be improved by appending a synthetic (non-random) alignment interval to create an extended simulation interval that does in fact possess the desired characteristics. An example illustrating this procedure has been provided.

The development of algorithms for the construction of synthetic alignment intervals is – at present – an open research problem. If general algorithms can be developed, simulation times can be shortened and confidence in the accuracy of simulation results can be enhanced.

The approach presented here can, in principle, be extended to any Monte Carlo simulation of a continuous time Markov process.

BIBLIOGRAPHIC NOTES

The material presented in this paper represents a new application of Operational Analysis. Introduced thirty years ago (Buzen 1976a), operational analysis is concerned with the development of equations that characterize the observable behavior of systems as they operate over time. No assumptions are made regarding the existence of an underlying stochastic process. Instead, all assumptions are formulated in terms of relationships among quantities that can be observed and measured under normal operating conditions.

The concept of homogeneity as used in this paper – along with derivations based on homogeneity and flow balance – closely parallel material that was originally presented in (Buzen 1976b). These derivations were subsequently extended to a broad class of queueing network models (Denning and Buzen 1978). Suri's analysis of the robustness of queuing network formulas was based upon operational analysis and the concept of homogeneity, but his work did not consider implications for Monte Carlo simulation (Suri 1983).

The application of operational analysis to the output of Monte Carlo simulations is a very recent development that has been characterized as Operational Analysis 2.0 (Buzen 2006b). Since Monte Carlo simulations can be regarded as explicit realizations of underlying stochastic processes, new issues are raised by the introduction of this additional consideration (e.g., the generation of synthetic alignment intervals).

BIBLIOGRAPHY

Buzen, J.P. 1976a. "Fundamental Operational Laws of Computer System Performance," *Acta Informatica*, 7, No. 2 (June), 167-182.

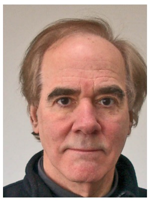
Buzen, J.P. 1976b. "Operational Analysis: The key to the new generation of performance prediction tools". In *Proceedings of IEEE COMPCON 76* (Washington, DC, Sept.) , 166-171

Buzen, J.P. 2006a. "Counter intuitive aspects of statistical independence in steady state distributions". In *Communication Networks and Computer Systems*, J.A. Barria (Ed.), Imperial College Press, London, 89 – 104.

Buzen, J.P. 2006b. "New Perspectives on Benchmarking, Modeling and Monte Carlo Simulation: Operational Analysis 2.0" In *CMG 2006 Conference Proceedings* (Reno, NV, Dec. 4 – 8), Computer Measurement Group, Turnersville, NJ.

Denning, P.J and Buzen, J.P. 1978. "The Operational Analysis of Queuing Network Models," *ACM Computing Surveys*, 10, No. 3, (Sept.), 225-261.

Suri, R. 1983. "Robustness of Queuing Network Formulas," *Journal ACM*, 30, No. 3 (July), 564-594.



JEFFREY P. BUZEN pioneered the use of queueing network models for the analysis of computer system performance in the early 1970s. After developing the central server model and the convolution algorithm for his Ph.D. dissertation at Harvard (1971), he co-founded BGS Systems, Inc. (1975) where, for 23 years, he led the team that commercialized these research results by creating the widely used BEST/1 family of modeling tools for computer performance management and capacity planning. Prior to founding BGS Systems, Dr. Buzen was a member of the Harvard's Computer Science faculty (where he taught a course on Operating Systems attended by Microsoft's Bill Gates and supervised the Ph.D. dissertation of Ethernet inventor Bob Metcalfe). He has returned recently to Harvard and is now serving as a consultant to the Initiative for Innovative Computing. Email: buzen@post.harvard.edu