Comput Syst Sci & Eng (2000) 6: 399–404


International Journal of
Computer Systems
Science & Engineering

# A technique for determining queuing network simulation length based on desired accuracy

**P Katsaros and C Lazos**

*Department of Informatics, Aristotle University of Thessaloniki, 54006 Thessaloniki, Greece. Email: katsaros@csd.auth.gr*

Although simulation is a commonly used approach by the computer systems performance engineers and the communication systems engineers, not enough consideration is usually given to the quality of simulation results. However, the queuing network simulation is not only a representation of the dynamic behavior of a system, but it is basically a stochastic process whose output has to be statistically analyzed. Moreover, the random selection of the simulation run length often leads to multiple replications of the same experiment in order to achieve the desired level of estimation accuracy. This paper shows an algorithm for dynamically determining the simulation run length in regard to the required accuracy. The last, is defined as an acceptable confidence interval length of the estimated values (confidence intervals are produced by the statistical analysis of the estimation results with the use of the regenerative approach).

**Keywords**: queuing networks, simulation, regenerative method

## 1. INTRODUCTION

Every discrete event simulation on a finite or countably infinite state space is a *Generalized Semi-Markov Process* (GSMP). The GSMP is defined as a stochastic process that makes a state transition when an event associated with the occupied state occurs. Several possible events, associated with a state, compete with each other to trigger the next transition. At each transition of the GSMP, new events may be scheduled. For each of these new events, a clock indicating the time when the event is scheduled to occur, is set, according to an independent (stochastic) mechanism. If a scheduled event does not trigger a transition but is associated with the next state, its clock continues to run; if such an event is not associated with the next state, it ceases to be scheduled and its clock reading is abandoned.

However, from the system analyst's point of view, the discrete event simulation is a random experiment and the estimation of the required characteristic(s) of a simulation model is a statistical process. Since most statistical theories assume that observations are independent, we can have reliable results more easily from independent, rather non-independent observations. The regenerative simulation is generally adopted as the easiest method for generating independent observations. The reason is that when using independent replications we have to estimate the extent of the transient phase and then to discard the biased (by the selection of the initial state of the system) observations of this period, whereas in the regenerative case the system is initially assumed to be in an equilibrium condition. On the other hand, the applicability of the regenerative property is not always clear and a study of the stochastic nature of the process is certainly necessary.

The formal definition of the regenerative property as given by G.Shedler [1] is in terms of *stopping times* for a stochastic process:

**Definition 1**
A stopping time for a stochastic process $\{X(t) ; t \geq 0\}$ is a random variable T (taking values in $[0,+\infty)$) such that for every finite $t \geq 0$, the occurrence or non-occurrence of the event $\{T \leq t\}$ can be determined from the history $\{X(u) ; u \leq t\}$ of the process up to time t.

**Definition 2**
The stochastic process $\{X(t); t \geq 0\}$ is a regenerative process in continuous time provided that:

(i) there exists a sequence $\{T_k; k \geq 0\}$ of stopping times such that $\{T_{k+1} - T_k; k \geq 0\}$ are independent and identically distributed
(ii) for every sequence of times $0 < t_1 < t_2 < ... < t_m$ ($m \geq 1$) and $k \geq 0$, the random vectors $\{X(t_1), ..., X(t_m)\}$ and $\{X(T_k+t_1), ..., X(T_k+t_m)\}$ have the same distribution and the processes $\{X(t); t < T_k\}$ and $\{X(T_k+t); t \geq 0\}$ are independent.

From the definition of the regenerative process, it is clear that we can obtain independent and identically distributed observations of a random variable from a single simulation run of the process by dividing it into *regenerative cycles*. Every time a regenerative cycle is completed, the process probabilistically restarts. These time instants are called *regeneration points* and they occur when the process returns to some fixed state.

The applicability of the regenerative method relies on the presence of the regenerative property in the underlying stochastic process of a simulation model. An *irreducible* and *recurrent non-null Markov process* with a finite state space is a typical example of a regenerative process. The successive entrances to any fixed state, s, form a sequence of regenerative points.

However, the problems arising in systems performance modeling are usually quite complex. Open (for communication systems) or closed (for computer systems) queuing networks with multiple job types and different service priorities among job types are usually used to represent contention for the multiple resources that comprise a system.

In the case of a closed network with probabilistic job routing and exponential service times, the underlying stochastic process is a Markov process with finite state space, but not necessarily irreducible. There may be one or more transient states and more than one irreducible, closed sets of recurrent states. However, it can be proved that

**Theorem 1**
*If there is a service center that sees only one job class or it is such that jobs of the lowest priority are subject to pre-emption, then the underlying Markov process has a single irreducible closed set of recurrent states.*

Here it is important to make clear the meaning of the job class in the queuing networks bibliography and to distinguish it from the job type. At every epoch of continuous time, each job is of exactly one class and one type, but jobs may change class as they traverse the network. On the contrary, jobs do not change type. The type of a job may influence its routing path through the network as well as its service requirements at each service center. Service priorities can also be associated with job types. It can be proved that the aforementioned theorem is also valid for queuing networks with multiple job types.

Theorem 1 yields that the Markov process, which is restricted to the set of recurrent states, is irreducible and recurrent non-null and so, it is a regenerative process. Since we are only interested in the steady-state behavior of the system, it is sufficient to be restricted to the set of the recurrent states. To achieve this, we have to initialize our simulation model in a recurrent state, say, s. The successive entrances to the state, s, will then form a sequence of regeneration points.

Finally, it is easy to prove that any state with all the jobs of a closed network being of the same class and queued in the same service center is such a recurrent state.

In cases where service times can not be assumed as being exponentially distributed it is not possible to formulate the queuing network as a Markov process. Let us assume the existence of a state, s, with all the jobs at a service center which sees only one class, or is such that, jobs of the lowest priority are subject to pre-emption and the set D of all states accessible from s. The GSMP which is restricted to the set D is irreducible and it is easy to prove that this process is characterized by the regenerative property. The selection of the regenerative state is similar to the case of exponential service times.

To conclude, in cases of open queuing networks where a steady-state distribution does not always exist (the condition under which a steady-state distribution exists is mentioned in §4) and so the regenerative method can not be applied, it is important to use an upper limit for the length of the 'regenerative' cycle. This allows the simulation to successfully terminate, producing point estimates of the required characteristic(s).

## 2. ESTIMATION RESULTS

Let us denote by $\tau_k$ the length of the kth cycle of X, where $X = \{X(t) ; t \geq 0\}$ a regenerative process in continuous time. If the expected time between regeneration points is finite, then X has a limiting distribution. Consider a real-valued function f (a queuing network characteristic to be estimated) whose domain is the state space of X, and

$$r(f) = E[f(X)]$$

$$Y_k(f) = \int_{T_{k-1}}^{T_k} f(X(u)) \cdot du$$

The definition 2 of a regenerative process yields that

**Theorem 2**
*The sequence* $\{(Y_k(f), \tau_k); k \geq 1\}$ *consists of independent and identically distributed random vectors.*

Also, the following can be proved [1]:

**Theorem 3**
*If* $\tau_1$ *is aperiodic, the expected time between regeneration points is finite and* $E[|f(X)|] < \infty$, *then*

$$E[f(x)] = \frac{E[Y_1(f)]}{E[\tau_1]} = r(f)$$

Theorem 3 asserts that the behavior of a regenerative process within a cycle determines the limiting distribution of the process as a ratio of expected values.

Suppose, that the goal of the simulation experiment is the estimation of the quantity $r(f)$. Thus, we wish to estimate the ratio of the expected values of two dependent random variables from a sequence of independent observations of the pair of random variables. To obtain an estimator for $r(f)$,

along with a confidence interval, set

$$Z_k(f) = Y_k(f) - r(f)\,\tau_k$$

and observe that the kth cycle of the process $\{X(t);\ t \geq 0\}$ determines the quantity $Z_k(f)$. The sequence $\{Z_k(f);\ k \geq 1\}$ of random variables are independent and identically distributed and the two aforementioned theorems imply that $E[Z_k(f)] = 0$. Then

$$\begin{aligned}\sigma^2 &= \text{var}\,(Z_1(f)) \\ &= E[(Z_1(f))^2] - (E[Z_1(f)])^2 \\ &= E[(Y_1(f) - r(f)\,\tau_1)^2] \\ &= E[((Y_1(f) - E[Y_1(f)]) - r(f)\,(\tau_1 - E[\tau_1]))^2]\end{aligned}$$

It follows that

$$\sigma^2 = \text{var}\,Y_1(f) - 2\,r(f)\,\text{cov}\,(Y_1(f), \tau_1) + (r(f))^2\,\text{var}\,\tau_1$$

Let N be the number of regenerative cycles and let $\bar{Y}(N)$, $\bar{\tau}(N)$, $s_{11}^2$ (N), $s_{22}^2$ (N), $s_{12}^2$ (N) be point estimates of $E[Y_1(f)]$, $E[\tau_1]$, var $(Y_1(f))$, var$(\tau_1)$ and cov $(Y_1(f),\tau_1)$ respectively, we have:

$$\bar{Y}(N) = \frac{1}{N}\sum_{k=1}^{N} Y_k(f)$$

$$\bar{\tau}(N) = \frac{1}{N}\sum_{(k=1)}^{N} \tau_k$$

$$S_{11}^2(N) = \frac{1}{N-1}\sum_{k=1}^{N} (Y_k(f) - \bar{Y}(N))^2$$

$$S_{22}^2(N) = \frac{1}{N-1}\sum_{k=1}^{N} (\tau_k - \bar{\tau}(N))^2$$

$$S_{12}^2(N) = \frac{1}{N-1}\sum_{k=1}^{N} (Y_k(f) - \bar{Y}(N))\,(\tau_k - \bar{\tau}(N))^2$$

Thus, the following point estimates for r(f) and $\sigma^2$ can be obtained

$$\hat{r}(N) = \frac{\bar{Y}(N)}{\bar{\tau}(N)}$$

$$s^2(N) = s_{11}^2(N) - 2\hat{r}(N)\,s_{12}^2(N) + (\hat{r}(N))^2\,s_{22}^2(N)$$

As a consequence of the central limit theorem it can be proved that

$$\frac{\sqrt{N}\,(\hat{r}(N) - r(f))}{\sigma / E[\tau_1]}$$

follows the standard normal distribution and so an 100 a % confidence interval for r(f) is

$$\left[\hat{r}(N) - \frac{s(N)\cdot F^{-1}\left(\frac{1+a}{2}\right)}{\sqrt{N}\,.\,\bar{\tau}(N)},\ \hat{r}(N) + \frac{s(N)\cdot F^{-1}\left(\frac{1+a}{2}\right)}{\sqrt{N}\,.\,\bar{\tau}(N)}\right]$$

## 3. HOW TO DETERMINE THE SIMULATION LENGTH

It is obvious that as N increases, the length of the confidence interval decreases and the midpoint converges to the true value. If the desired accuracy will be determined by the number δ, so that the half length of the 100a% confidence interval to be obtained will be not more than 100δ% of r(f), then

$$\delta.\,\hat{r}(f) \geq \frac{s(N)\cdot F^{-1}\left(\frac{1+a}{2}\right)}{\sqrt{N}\cdot\bar{\tau}(N)}$$

and the number N of the required regenerative cycles can be dynamically determined as follows:

$$N \geq \left(\frac{F^{-1}\left(\frac{1+a}{2}\right)}{\delta}\right)^2 \cdot \left(\frac{s(k)}{\hat{r}(k)\cdot\bar{\tau}(k)}\right)^2 \qquad (A)$$

where s(k), $\hat{r}(k)$ $\bar{\tau}(k)$ are the sample estimates, after the kth cycle of the simulation experiment. Let, for example, our aim is to compute a 90% confidence interval for the mean throughput, of a network queue. Also, we want the confidence interval length, to be, not more than 4% of the estimated value. In this case we have:

$$\alpha = 0.9 \text{ and } \delta = 0.02$$

If k regenerative cycles of the simulation experiment have just been completed and the number k is still less than L,

$$L = \left(\frac{F^{-1}\,(0.95)}{0.02}\right)^2 \cdot \frac{S_{TR}}{\overline{TR}^2\cdot\overline{CL}^2}$$

where

$F^{-1}\,(0.95) = 1.645$

$S_{TR}^2$ is the variance

$\overline{TR}$ is the average throughput

$\overline{CL}$ is the average cycle length,

then the experiment has to go on. Otherwise, the simulation has already produced the desired confidence interval length and there is no reason to continue.

If more than one queues exist in the simulation model, then the number of the realised regenerative cycles has to be compared to the maximum L of all the network queues.

## 4. QUEUING NETWORK SIMULATION IN PRACTICE

In practice, the software for queuing network simulation, should allow the analysts to be able to define both the regenerative state of the specified model and the required level of accuracy as well as an upper limit for the length of the 'regenerative' cycle in cases where a steady-state distribu-

tion does not exist.

However, in most commonly used simulators, the regenerative method, has not been implemented (APLOMB and RESQ are the only known exceptions [2]). On the other hand, even in cases where the simulator makes use of a subroutine interface, for increased modeling flexibility, it is usually impossible to use the regenerative method, because this would mean interference in the core simulation mechanism.

The only alternative is the use of programming languages like SIMULA, SIMSCRIPT or other, designed for simulation modeling applications. In these cases, the analyst, models the system by defining the changes that occur at event times. He has to determine the events that can change the system state and then to develop the logic associated with each event type. A simulation of the system is produced by executing the logic associated with each event in a time-ordered sequence. Thus, the logic associated with the end of a regenerative cycle can be easily included in an event.

We have developed a queuing network simulator which is being used for obtaining reliable results from open or closed queuing network simulations with multiple job types. The development platform was the Sun Sparc workstation running the SunOS operating system and the software has been implemented in SIMSCRIPT II.5 [3].

The simulator produces estimates for the following queuing network characteristics: the mean throughputs, the mean utilizations, the mean response times and the mean queue lengths. The desired accuracy is being specified by the analyst by the number $\delta$, so that the half length of the 100a% confidence intervals to be obtained will be not more than 100$\delta$% of the estimation, for all the queuing network estimators and for every queue. The required number of regenerative cycles is dynamically determined as the maximum number N (where N is calculated according to formula A) for all the queuing network estimators and for every queue in the network.

The cyclic queue model (Figure 1) [4, 5] can be used to illustrate the regenerative closed queuing network simulation. Let us assume that there is one CPU and two identical I/O devices and the degree of multiprogramming is three.

We also assume that both queues have FCFS scheduling disciplines and that service times are independent and identically distributed with exponential distributions and rates 0.15 for the CPU and 0.1 for each service center of the I/O devices. All the jobs are of the same type and thus they all have the same priority.

The system has to be initialized in a regenerative state. As mentioned in §1, any state with all the jobs being of the same class and queued in the same service center is a recurrent state and so, the successive entrances to this state form a sequence of regenerative points. In our case the regenerative state was defined as the state with all the jobs placed at the CPU. Finally, the maximum half length of the 90% confidence intervals to be produced, is required to be no more than 2% of the estimated value. The results obtained are shown in Figure 2.

The required confidence interval lengths were achieved after the completion of 3426 regenerative cycles. The use of formula A for all the queuing network estimators and for every queue in the network, has helped us to stop the simulation on time, thus obtaining the required accuracy in only one simulation run. Also, the enhanced control over the simulation length, allows us to enclose the simulation execution in a repetitive loop where the model parameters can be easily varied according to a specified algorithm. Thus we can easily study the effect of changes in the values of various network input parameters, on the system performance.

It is also interesting to illustrate the regenerative open queuing network simulation by using the model shown in Figure 3 where the job arrival process follows the Poisson distribution with rate 1/8 ms.

The condition for existence of steady-state for open queuing networks is that the system is not 'saturated' or 'overloaded' i.e. the demand for service must be less than the service capacity. Practically, this means that for every node i in the system, the ratio $\lambda_i/\mu_i$ of the job arrival rate divided by the node service rate must be less than the number of servers available in the node.

Obviously, our model fulfils the aforementioned condition and we can proceed with the selection of the regenerative state. Taking into account that the probability of a network state is the product of the probabilities of the states of the individual queues [7] (except FCFS queues with non- exponential or job type dependent service time distributions), we infer that the most frequent network state is the one, where each queue is in its most frequent state. From the other hand, it is easy to prove [6] that for a queue with exponential inter-arrival times and exponential service times, the queue length distribution is given by

$$P(N) = (1-U)\,U^N, \qquad N = 0, 1, 2, \ldots\ldots$$

where U is the server utilization ($= \lambda/\mu$) and since U is less than 1, P(0) is the most probable queue length. Generally, for most open queuing networks, the state where no jobs are in the network, is certainly one of the most frequently occurring recurrent states.

Thus, the empty network was selected as the regenerative state, since it is also the easiest to test for. The maximum half length of the 90% confidence intervals to be produced, is required to be no more than 3.8% of the estimated value. The results obtained are shown in Figure 4.

It is important to note that these systems are highly variable, because of the variability in the total number of jobs in the system. As a consequence, it is essential to use observations of a large number of regenerative cycles in order to produce confidence intervals of an acceptable length (recall that the confidence interval length is proportional to the standard deviation s(N)).

The use of formula A for all the queuing network estimators and for every queue in the network, has helped us to determine the required number of regenerative cycles without having to repeat this time consuming simulation more

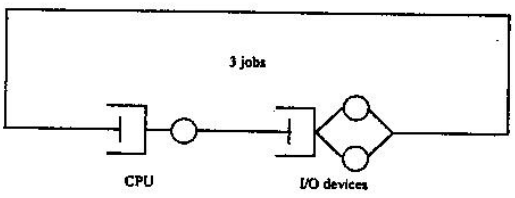


**Figure 1** The cyclic queue model

| | | | |
|---|---|---|---|
| SIMULATED TIME: | 101 008 ms | NUMBER OF EVENTS: | 24 686 |
| NUMBER OF CYCLES: | 3426 | AVERAGE NUMBER OF EVENTS: | 7.205 |
| AVERAGE CYCLE LENGTH: | 29.474 ms | 90% CONFIDENCE INTERVAL: | (28.667, 30.281) |

**RESULTS REGARDING THE SERVICE ACTIVITIES**

| | THROUGHPUT jobs/ms | | | UTILIZATION | | | RESPONSE TIME ms | | |
|---|---|---|---|---|---|---|---|---|---|
| | LB | MEAN | UB | LB | MEAN | UB | LB | MEAN | UB |
| CPU | 0.121 | 0.122 | 0.124 | 0.802 | 0.809 | 0.816 | 12.772 | 13.033 | 13.293 |
| I/O | 0.121 | 0.122 | 0.124 | 1.200 | 1.217 | 1.233 | 11.320 | 11.518 | 11.715 |

**RESULTS REGARDING THE QUEUES**

| | LENGTH jobs | | |
|---|---|---|---|
| | LB | MEAN | UB |
| CPU | 1.570 | 1.593 | 1.615 |
| I/O | 1.385 | 1.407 | 1.430 |

**Figure 2** The results obtained from initializing the system in a regenerative state

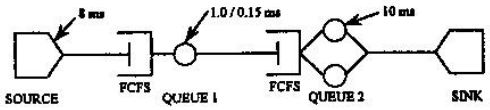


**Figure 3** A simple regenerative open queuing network simulation

than one time.

As mentioned in §1 the regenerative method is also applicable in cases with multiple job types and different service priorities associated with them. To illustrate this, a simple open queuing network has been simulated with two different job types (Figure 5).

Jobs released from the source 1 are assumed to have a greater service priority than the jobs released from the source 2. In this experiment, the maximum half length of the 90% confidence intervals to be produced was required to be no more than 3.95% of the estimated value. The results obtained are shown in Figure 6.

Comparing these results to the results of the former simulation (Figure 4) we observe noticeable differences in utilizations, queue lengths and response times.

More regenerative simulation experiments were also carried out. Different queuing disciplines (Last-Come-First-Served-Preemptive-Resume and processor sharing) were used in

| | | | |
|---|---|---|---|
| SIMULATED TIME: | 1 532 664 ms | NUMBER OF EVENTS: | 573 651 |
| NUMBER OF CYCLES: | 7526 | AVERAGE NUMBER OF EVENTS: | 76.223 |
| AVERAGE CYCLE LENGTH: | 203.623 ms | 90% CONFIDENCE INTERVAL: | (196.885,210.361) |

**RESULTS REGARDING THE SERVICE ACTIVITIES**

| | THROUGHPUT jobs/ms | | | UTILIZATION | | | RESPONSE TIME ms | | |
|---|---|---|---|---|---|---|---|---|---|
| | LB | MEAN | UB | LB | MEAN | UB | LB | MEAN | UB |
| QUEUE 1 | 0.124 | 0.125 | 0.125 | 0.827 | 0.831 | 0.835 | 37.403 | 38.861 | 40.320 |
| QUEUE 2 | 0.124 | 0.125 | 0.125 | 1.238 | 1.245 | 1.251 | 16.042 | 16.260 | 16.478 |

**RESULTS REGARDING THE QUEUES**

| | TOTAL LENGTH jobs | | |
|---|---|---|---|
| | LB | MEAN | UB |
| QUEUE 1 | 4.657 | 4.848 | 5.040 |
| QUEUE 2 | 1.998 | 2.029 | 2.059 |

**Figure 4** The results obtained for the empty network.

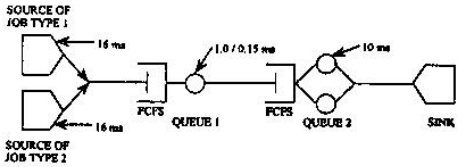


**Figure 5** An open queuing network simulation with multiple job types

SIMULATED TIME: 1 848 851 ms
NUMBER OF CYCLES: 8800
AVERAGE CYCLE LENGTH: 210.073 ms
NUMBER OF EVENTS: 695 661
AVERAGE NUMBER OF EVENTS: 79.052
90% CONFIDENCE INTERVAL: (203.502,216.643)

**RESULTS REGARDING THE SERVICE ACTIVITIES**

| | THROUGHPUT jobs/ms | | | UTILIZATION | | | RESPONSE TIME ms | | |
|---|---|---|---|---|---|---|---|---|---|
| | LB | MEAN | UB | LB | MEAN | UB | LB | MEAN | UB |
| QUEUE 1 | 0.125 | 0.125 | 0.126 | 0.833 | 0.837 | 0.841 | 39.404 | 40.956 | 42.508 |
| QUEUE 2 | 0.125 | 0.125 | 0.126 | 1.247 | 1.252 | 1.258 | 16.299 | 16.491 | 16.682 |

**RESULTS REGARDING THE QUEUES**

| | TOTAL LENGTH jobs | | |
|---|---|---|---|
| | LB | MEAN | UB |
| QUEUE 1 | 4.934 | 5.137 | 5.340 |
| QUEUE 2 | 2.041 | 2.068 | 2.096 |

**Figure 6** The results obtained for the open queuing network with multiple job types

simple models. More complex models with passive queues [2], job fission and fusion nodes [7] and non exponential service times have been also tested. In all cases, the implemented algorithm helped us achieve an acceptable confidence intervals length in only one simulation run.

## 5. CONCLUSION

In this paper, we suggest a useful algorithm for dynamically determining the queuing network simulation length. The use of this algorithm helps the analyst obtain the required accuracy of the results in only one simulation run. This is particularly useful for long simulations i.e., complex queuing networks with many jobs running, open queuing networks and queuing networks with many different job types.

Applications were also discussed and demonstrated. The suggested method can be applied without problems in all cases of queuing networks (open or closed), where a steady-state distribution and a regenerative structure exists.

## REFERENCES

1 Shedler, G. *Regenerative Stochastic Simulation*, California, Academic Press (1993)
2 Sauer, C H and Macnair, E A. 'Queueing Network Software for Systems Modelling, Software', *Practice and Experience* 9, (1979) pp. 369–80
3 Kiviat, P J and Villanueva, R and Markowitz, H M. *The SIMSCRIPT II.5 programming language*, CACI Products Company (1987)
4 Chiu, W W and Dumont, D and Wood, R. 'Performance Analysis of a Multiprogrammed Computer System', *IBM Journal of Research and Development* 19, (1975) pp. 263–71
5 Chiu, W W and Chow, W-M. 'A Performance Model of MVS', *IBM Systems Journal* 17, (1978) pp. 444–62
6 Mitrani I. *Modelling of Computer and Communication Systems*, Cambridge, Cambridge University Press (1987)
7 Sauer, C H and Chandy, K M. *Computer Systems Performance Modelling*, New Jersey, Prentice-Hall (1981)