1978

Homogeneous Approximations of General Queueing Networks

Gianfranco Balbo

Peter J. Dennins

Report Number:
78-290

HOMOGENEOUS APPROXIMATIONS OF GENERAL QUEUEING NETWORKS

Gianfranco BALBO

Peter J. DENNING

Department of Computer Sciences
Purdue University
West Lafayette, IND 47906
USA

July 1978
(Revised November 1978)
CSD TR 290

Preprint: To appear in Proc. 4th International Symposium on Computer System Modelling and Performance Evaluation, Vienna, 6-8 February 1979.

To appear in Proc. 4th International Symposium on Computer System Modelling and Performance Evaluation, Vienna, 6-8 February 1979.

HOMOGENEOUS APPROXIMATIONS OF GENERAL QUEUEING NETWORKS[1]

Gianfranco Balbo[2] and Peter J. Denning

Department of Computer Sciences
Purdue University
West Lafayette, IND. 47907
USA

Abstract: Product form queueing networks are of special interest because the algorithms for computing their performance statistics are fast. The concept of homogeneity, which arises in the operational analysis of queueing network models, asserts that the product form is exact if each device's on-line service function is the same as would be observed off-line under constant loads. (A device's service function gives the mean time between departures conditioned on the queue length.) We will show that, corresponding to any given general queueing network, there exists a product-form queueing network of the same topology whose queue length distributions are identical to those of the given network; this suggests that errors in approximations based on product-form models can be confined to parameter estimation. A numerical study compares several practical methods of approximating the service functions from on-line measurements.

---

[1] Work reported herein was supported in part by NSF Grants GJ-41289 and MCS78-01729 at Purdue University.

[2] Presently on leave from Istituto di Scienze dell'Informazione, Universita' di Torino, ITALY.

1. INTRODUCTION

Queueing network models in which the solution for state occupancies p(n) [the proportion of time the system spends in state n] are of the product form lead to very fast algorithms for computing performance metrics [BALB77, BUZE73, DENN78, REIS75, REIS78]. Stochastic queueing network models satisfying the BCMP theorem [BASK75] are of this kind. So are homogeneous operational queueing networks, in which the on-line behavior of every device is the same as its off-line behavior [DENN77, DENN78]. The fast algorithms make product-form queueing network models very useful even when all the assumptions are not exact.

Exact solutions for p(n) in nonhomogeneous systems are computationally slow. In principle, one can use Cox's method to approximate each device, with arbitrary precision, as a group of "stages". One can then write a set of balance equations among quantities like p(n,m), where n is an apportionment of the jobs among the devices and m is a vector specifying the stage of service currently in progress at each device. Gauss-Seidel elimination, one of the most efficient solution techniques known for such equations, often runs for a considerable time before converging on a solution, especially if the coefficient of variation between departures for some device is high.

To avail themselves of fast solution techniques, analysts prefer to work with a product-form model that approximates the real system. A number of successful approximations have been discovered, but little is known in general about their errors.

The first result of this paper is that there exists a product-form queueing network model in which the marginal queueing distribution of each device is identical to that of a given arbitrary general queueing network. We call this the homogeneous equivalent model (HEM) of the given general queueing network. This result suggests that the product-form model is not the fundamental limitation of approximations; parameter estimation is the limit. The paper also describes a numerical study comparing three methods of approximating the equivalent model: load independent devices, load dependent devices whose service functions are the ones measured on-line in the real system, and the extended product form (EPF).

The experimental part of this investigation complements and extends prior work by other authors -- e.g., CHAN75, CHAN78, GELE75, GELE78, SAUE75, SEVC77, and SHUM78. We omitted the diffusion approximation [GELE75, GELE78, KOBA74] because it is outside the context of product-form queueing network models. We omitted approximations based on the Chandy-Herzog-Woo theorem, which already have been well studied elsewhere [CHAN75, CHAN78, SAUE75]. We also omitted Kuhn's method of accounting for nonexponential devices because it too is not based on product-form queueing networks [KUHN78]. The contributions of this experimental study are a) a comparison of the effects of the coefficient of variation (CV) on various approximations for utilizations and mean queue lengths; b) an evaluation of the relative importance of backlogs caused by bottlenecks or by high CVs; and c) a study of on-line and off-line behavior of devices.

2. FORMAL PROBLEM STATEMENT

A product-form closed queueing network with N jobs and M devices has states $\underline{n} = (n_1, n_2, \ldots, n_M)$ where $N = n_1+n_2+\cdots+n_M$. The proportion of time state $\underline{n}$ is occupied is given by the product form

$$p(\underline{n}) = \frac{1}{G} \prod_{i=1}^{M} F_i(n_i)$$

where G is a normalization constant and F_i is a "device factor":

$$F_i(n) = V_i^n \; S_i(n)S_i(n-1) \cdots S_i(1)$$

where V_i, the visit ratio, is the mean number of visits to device i per job, and $S_i(n)$, the service function of device i, is the mean time between completions conditioned on the queue length being n.[1] If device i is load independent with mean time between completions S_i, its device factor is $F_i(n) = (V_iS_i)^n$. The queueing distribution at device i is defined as

$$p_i(n) = \sum_{\underline{n},\ n_i=n} p(\underline{n}) \ .$$

The question of primary interest here is: Does there exist a choice of service functions $\{S_i(\cdot)\}$ for which the $p_i(n)$ calculated in the model are identical to the $p_i(n)$ observed in the real system? Such service functions define the equivalent homogeneous devices corresponding to the real devices. We will exhibit a fast, iterative method for computing these equivalent functions. We will then compare approximations in terms of the $\{S_i(\cdot)\}$ they generate.

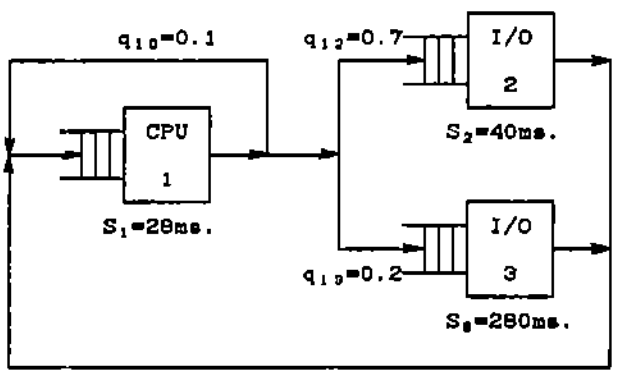

Figure 1. Example network

[1] The service function of device i is defined operationally as the ratio $S_i(n) = T_i(n)/C_i(n)$, where $T_i(n)$ is the total time during which device i is observed to have a queue of length n and $C_i(n)$ is the total number of service completions observed when the queue length is n. Note that $1/S_i(n)$ is the device's output rate observed relative to time intervals in which queue length is n. Note also that $S_i(n)$ need not be constant even if the device contains a single, load independent server [DENN78].

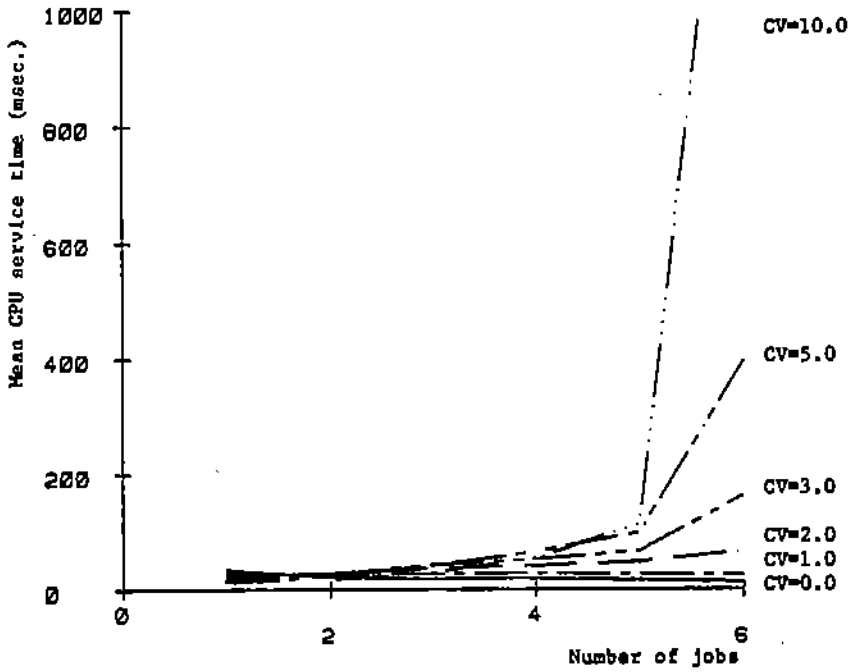

Figure 2. On-line service function for CPU in example network.

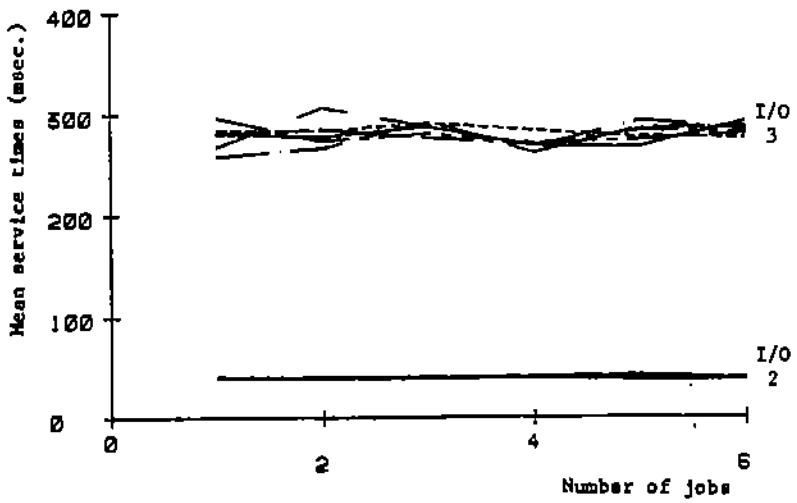

Figure 3. On-line service functions of I/O devices in the example network for various CVs of CPU.

3. METHODOLOGY OF THE INVESTIGATION

To illustrate the homogeneous equivalent service functions and their approximations, we used a simulator of a central server network (Figure 1, with one CPU and two I/O devices) to generate behavior sequences. These sequences were interpreted as observations of a real system. The network is closed with N jobs in it for N=8. (No significant changes in the results were observed for N>8.) Except when we note explicitly otherwise, the simulations used exponential service distributions (CV=1) at the two I/O devices and either hyperexponential service (CV>1) or Erlangian service (CV<1) at the CPU. All three devices were single servers.

Busy times and completion counts for each device at each queue length ($0 \leq n_i \leq N$) were measured from the observed behavior sequence and used to calculate the visit ratios $\{V_i\}$, the queue length distributions $\{p_i(n)\}$, the on-line service functions $\{S_i(n)\}$, the utilizations $\{U_i\}$, and the mean queue lengths $\{\bar{n}_i\}$. These measurements were repeated for a variety of combinations of the CVs of the three devices. This allowed us to study the effect of several high CVs on the errors arising from various approximations to the homogeneous equivalent service functions.

The box below summarizes the three product form approximations which will be compared later in the paper.

CODE	DESCRIPTION
HEN	Homogeneous model equivalent to real system.
HLI	Homogeneous Load Independent Model; the mean service times of devices match those measured on the real system.
HLD	Homogeneous Load Dependent Model; the service functions match the on-line service functions observed in the real system.
EPF	Extended Product Form model for a network of load independent servers; the mean and CV of the service distributions match those measured on the real system.

4. CAUSES OF NONHOMOGENEOUS BEHAVIOR

The coefficient of variation (CV) of a random variable is the ratio of its standard deviation to its mean. A large amount of variation in the times between service completions at a device will cause the device's on-line service function to differ from its off-line service function.

Consider a single-server device (i) with high CV in its distribution of service requests. When observed off-line under any fixed, nonzero queue length (n), the mean time observed between

completions ($S_i(n)$) will always be the mean (S_i) of the request distribution. But when device i is on-line, there will be long requests that block large numbers of short ones, causing long departure times to be correlated with long queue lengths — i.e., $S_i(n)$ will tend to increase in n.[2]

Figure 2 depicts the on-line service function $S_i(n)$ of the CPU in the network of Figure 1, for various values of the CV of the CPU burst distribution.

This figure confirms the above intuition: the higher the CV the stronger the tendency for $S_i(n)$ to increase with the queue size n. The high CV can, like a bottleneck, generate a backlog at a device. The line corresponding to CV = 1 is close to the off-line service function, which is constant at 28 msec.

For small CV (requests nearly all of the same size) there is only a weak tendency for a long queue to build and, hence, the service function will decrease as queue length increases. Figure 2 shows that this decrease is nowhere near as severe as the increase wrought by high CV. Indeed, our experiments (described later) show that very small CVs do not cause much error in the approximations made by product-form models.

Figure 3 shows the on-line service functions of the two (exponential) I/O devices of Figure 1. The presence of the high CV at the CPU has a marginal influence of no consistent pattern on the service functions of these other devices. This is not surprising since exponential servers are homogeneous.

The high CV in a device's output process causes the on-line service function to differ from the off-line because long request will block large numbers of short requests. It follows that a device with a high degree of internal parallelism can mitigate this effect by providing alternate paths on which short requests can bypass a long one. Three examples of such devices are the infinite server, the processor-sharing server, and the last-come-first-served server of the BCMP Theorem [BASK75]; these kinds of devices exhibit the same service functions on-line as they do off line and, hence, they are homogeneous.

An arbitrary subsystem can be approximately homogeneous relative to its environment if it allows a high level of concurrency among jobs inside it. In this case, long jobs cannot effectively block groups of subsequently arriving short ones and, consequently, the on-line and the off-line behaviors will be similar. A study of the Purdue MACE System confirms this; it revealed that the CV of time between completions of jobs in the same time sharing workload

---

[2] An extreme case will illustrate this intuition. Suppose that the N jobs of a closed system have CV=N-2 at device i. It can be shown that one job must require at least $(N-1)S_i$ sec. at device i and the others collectively require less than S_i sec. there. If, at time t, the long job leaves behind in the queue of device i the N-1 short jobs, then all N jobs will leave device i in the interval $[t,t+S_i]$; if the response time of the rest of the network is not too large, then all the short jobs will return to join the queue behind the long one. In this case the high CV at device i forces the frequent recurrence of the state in which all jobs are present at device i.

was 1.08, whereas the CV in the total CPU requirement was 2.15.

It is tempting to conjecture that using the actual on-line service functions as parameters to the product form solution would cause the product form to be exact. It will be evident from our numerical studies that this is not so. (Thus, the homogeneous equivalent service functions differ from the on-line service functions.)

5. EQUIVALENT HOMOGENEOUS NETWORK OF QUEUES

The objective is to calculate service functions $\{S_i(n)\}$ which, when used in the product form solution, lead to the same values of $\{p_i(n)\}$ as observed in the real system. Each resulting $S_i(n)$ is interpreted as the service function of a homogeneous device equivalent to device i in the real system.

The idea of replacing homogeneous subnetworks with equivalent devices is not new; it is the basis of equivalence and decomposition approaches used by Brandwajn [BRAN74, BRAN77] and by Chandy, Herzog and Woo [CHAN76]. The new aspect of our result is that a homogeneous equivalent for an arbitrary queueing network exists. This result demonstrates that homogeneity is not the inherent limitation of product-form queueing network models. The errors arise in parameter estimation, particularly the queue-dependent service functions of devices.

The definition of the normalizing constant G in the product form expression for $p(\underline{n})$ is

$$G = g(N,M) = \sum_{\underline{n}\in S(N,M)} \prod_{i=1}^{M} F_i(n_i)$$

where $S(N,M)$ is the set of all M-component vectors whose nonnegative elements sum to N, and $F_i(n_i)$ is the factor for device i. The proportion of time during which $n_M=n$ can be written as

$$\begin{aligned} p_M(n) &= \sum_{\substack{\underline{n}\in S(N,M) \\ n_M=n}} p(\underline{n}) \\ &= \frac{F_M(n)}{g(N,M)} \sum_{\underline{n}\in S(N-n,M-1)} \prod_{i=1}^{M-1} F_i(n_i) \\ &= F_M(n) \frac{g(N-n,M-1)}{g(N,M)} \end{aligned}$$

Because $F_M(n) = V_M S_M(n) F_M(n-1)$, this reduces to an expression for the homogeneous equivalent service function of device M:

$$S_M(n) = \frac{1}{V_M} \frac{p_M(n)}{p_M(n-1)} \frac{g(N-n+1,M-1)}{g(N-n,M-1)}, \quad n=1,\ldots,N,$$

This expression can be written in the form of a balance equation

$p_M(n)/S_M(n) = p_M(n-1)\lambda(n-1)$, where

$$\lambda_M(n) = V_M \frac{g(N-n,M-1)}{g(N-n+1,M-1)}$$

is the arrival rate to device M generated by the remainder of the network.

Figure 4 shows the algorithm for calculating the homogeneous equivalent service function. It iteratively improves trial functions until the error between trials functions becomes small. A component of the algorithm is a queueing network evaluation routine,

$$G := QN(\{V_i\},\{S_i(n)\},j),$$

for which $\{V_i\}$ are the visit ratios, $\{S_i(n)\}$ are the trial service functions, and j is a "distinguished device" ($1 \leq j \leq M$). QN reindexes the devices so that j corresponds to the last column of the matrix $g(\cdot,\cdot)$ and then executes the standard algorithm [BUZE73] until the M-1st column of the matrix $g(\cdot,\cdot)$ is filled. QN returns an array G[0,...,N] such that $G[r] = g(r,M-1)$, r=0,...,N.

Steps 3 and 4 in the algorithm iterate until there is no significant further change in the trial $\{S_i(n)\}$. (In the cases we studied, two iterations of Step 3 the model yielded estimates of the metrics of the real system to one significant digit of accuracy; ten iterations of step 3 gave three digits of accuracy.) Step 5 ensures that the throughputs of the equivalent network match those of the real system.

The algorithm works for the following reasons. First, there is a unique set of $\{S_i(n)\}$ whose throughputs match the real system and which generate the original $\{p_i(n)\}$. The existence of these $\{S_i(n)\}$ is guaranteed by the equation used in Step 3.2.1. An examination of the product form solution for $p(\underline{n})$ shows that changing any proper subset of the values $\{S_i(n)$: all i and n$\}$ would change the speed of some part of the system relative to others; this would cause relative changes in queueing and hence in the $\{p_i(n)\}$, contradicting the supposition that $\{S_i(n)\}$ are the equivalent functions. Second, the equivalent network has the same topology as the original system. Therefore, its throughputs $\{X_i\}$ obey to the throughput equations of the real system. This means that the throughputs of the model after Step 4 are correct to within a constant. Step 5 scales all the service functions to cause the model's $\{X_i\}$ to match those of the real system. (Scaling <u>all</u> the $\{S_i(n)\}$ by the same factor does not change the $\{p_i(n)\}$.) Third, convergence is assured because, in closed networks, the global effect of changing a parameter is less than the change itself [WILL76].

Figure 5 compares the homogeneous equivalent service functions with the actual on-line service functions of the CPU in the network of Figure 1. Notice that the homogeneous equivalents are relatively flat for $0 \leq n \leq N-1$ with a sharp rise for n=N; the service-time at n=N is higher for larger CVs. Figure 6 shows the homogeneous equivalents for the two I/O devices in Figure 1; even though they are exponential, their equivalent functions depend on the load.

INPUT: N, M, T, {C(i)}, {V(i)}, {$p_i(n)$} } i=1,...,M .
OUTPUT: {$S_i(n)$} } n=1,...,N .

ALGORITHM:

1. As an initial trial service function for each i, use the overall mean time between completions (T is the total observation time):

$$S_i(n) = \frac{T - T_i(0)}{C(i)} \qquad n=1,\ldots,N.$$

2. Initialize error measure: E := 0.

3. For j=1,...M do: (for each device)

   3.1. Calculate G := QN({V_i},{$S_i(n)$},j)

   3.2. For n=1,...,N do: (for each queue length)

      3.2.1. Calculate new trial service function value

$$Y = \frac{p_j(n)}{p_j(n-1) * V_j} \frac{G[N-n+1]}{G[N-n]} .$$

      3.2.2. Aggregate the squared error

$$E := E + \left[\frac{S_j(n)-Y}{S_j(n)}\right]^2 .$$

      3.2.3. Update the service function

$$S_j(n) := Y .$$

4. If E > desired ε, repeat from Step 3.

5. (Ensure throughput constraint.)
Let $a = \frac{1}{X(1)} \sum_{n=1}^{N} p_1(n)/S_1(n)$, where X(1) = C(1)/T.
Scale by replacing each $S_i(n)$ with $S_i(n)/a$.

Figure 4.

6. EXPERIMENTAL STUDIES

Because there is no known direct measurement of the real system that will yield the homogeneous equivalent service functions, we conducted a series of experiments to see how well models based on easily measured quantities approximate the equivalent service functions. The easiest approximation replaces each function $S_i(n)$ with the overall mean service time, $(T-T_i(0))/C_i$; we called this the homogeneous load independent (HLI) model. A second approximation puts each $S_i(n)$ equal to the actual on-line service function; we called this the homogeneous load dependent (HLD) model. We also studied the extended product form (EPF) of Shum and Buzen, which has given good results elsewhere [SHUM78, SHUM77]; the EPF incorporates explicitly the CV of each device's service distribution. The three approximations were evaluated by comparing calculated utilizations and mean queue lengths of devices with the corresponding values measured in the behavior sequences of the simulated system.

6.1. HLI Approximation

In this series of experiments we used the simulator to generate behavior sequences for values of the CPU's CV ranging from 0 to 10. For each sequence and for each device we measured the overall mean times between completions $\{S_i\}$ and the visit ratios $\{V_i\}$, which we then used as the parameters for the product-form load-independent queueing network evaluator.

Figure 7 compares the actual utilization of the CPU with the values obtained from the HLI model, and displays the relative error between the HLI estimate and the actual value. Since the the same $\{V_i\}$ and $\{S_i\}$ were used in all the behavior sequences, the utilization ratios are independent of the CV [DENN78]:

$$\frac{U_i}{U_j} = \frac{V_i S_i}{V_j S_j} .$$

Therefore the utilizations of the I/O devices are scaled versions of the CPU utilization: the relative error between the HLI approximation and actual utilization is the same for all devices.

The HLI model consistently overestimates the utilizations when the CPU's CV > 1. This is because, first, the utilizations are constrained to be in fixed ratios, and, second, the high CV causes jobs to backlog at the CPU which in turn lowers the utilization of the I/O devices.

For utilizations, the HLI model has very small error when $0 \leq CV \leq 2$, but its errors may exceed 10% as soon as $CV \geq 6$. A separate series of experiments with all the CV = 0 also showed that the model's estimates of utilizations are correct within 3%.

Figure 8 compares the means of the queue lengths estimated by the HLI model with the actual means. The errors are zero when CV = 1 because, in the simulations used to generate behavior sequences, the CPU's service distribution was exponential (and therefore homogeneous) when CV = 1. The HLI model is much less robust in estimating queue lengths than it is in estimating utilizations: in this case, for example, its estimates of n_i are accurate to within

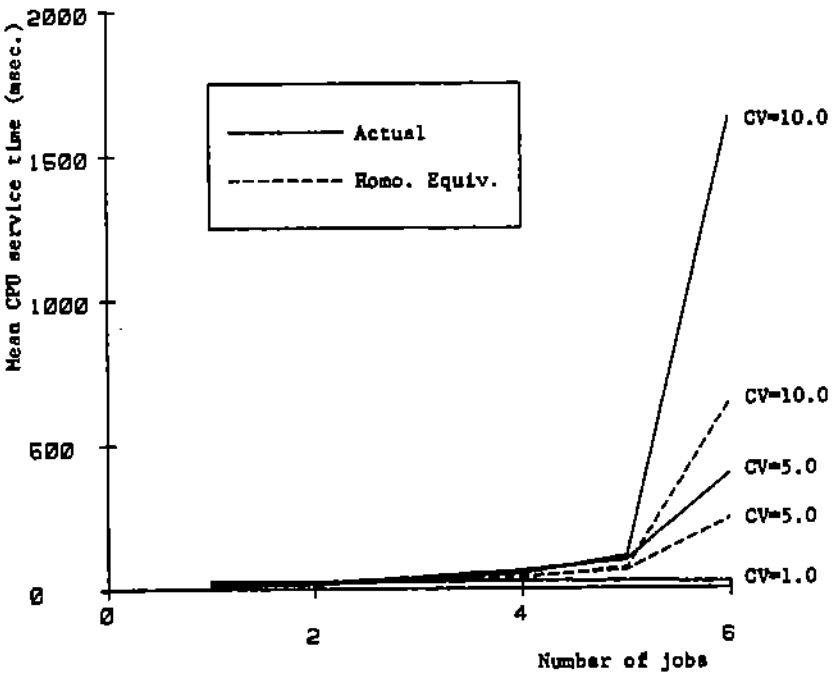

Figure 5. Comparison between actual on-line service function of the CPU and the homogeneous equivalent service function.

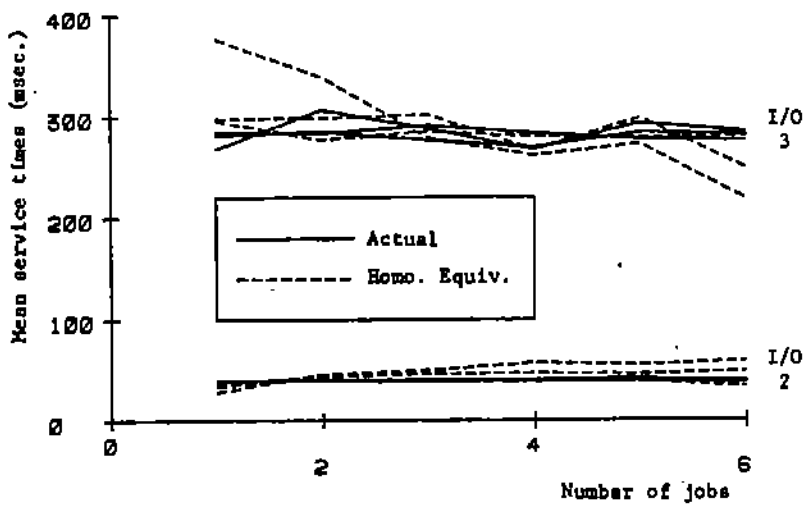

Figure 6. Homogeneous equivalent service functions for I/O devices for various CVs of CPU.

10% only for $0.6 \leq CV \leq 1.8$.

As noted in Section 4, the high CV can generate a backlog at a device. But the only kind of backlog representable in a load-independent model is the standard bottleneck, such as device 3 in our example. The high CV transfers some of the backlog from device 3 (I/O) to device 1 (CPU). This explains why the HLI model overestimates the queue at device 3 and underestimates it at device 1. (Since device 2 causes no backlogs, the HLI model estimated its queue length well.)

To study how CV-induced backlogs can compete with or reinforce bottleneck-induced backlogs, we ran a series of experiments using progressively higher values of mean CPU time (S_1) with the CPU's CV held fixed at 5.0. The results are shown in Figures 9 and 10.

Figure 9 shows, as in Figure 7, that the HLI model consistently overestimates utilizations no matter what device is the bottleneck. Figure 10 shows, as was noted above, that the HLI model model consistently overestimates the queue length at the bottleneck, even if the bottleneck is also a device of high CV. In our example, S_1 = 58 msec causes balance between the CPU and the I/O device; at this point $U_1 = U_3$ and the two devices are of equal importance as bottlenecks. For $S_1 < 58$ msec, device 3 (I/O) is the bottleneck while for $S_1 > 58$ msec device 1 is the bottleneck.

Of special interest in Figure 10 is that the HLI model is nearly exact at the point where the two dominant devices of the system are balanced ($U_1 = U_3$). This is consistent with a suggestion by Courtois [COUR77, pp 83 ff], who argued that balanced devices tend to be more decomposable than unbalanced ones — thus conforming more closely to the homogeneity assumption.[9] However, the HLI model gives about 15% overestimate in the CPU utilization at the balance point. The relations among bottlenecks, decomposability, and the accuracy of the HLI model are, evidently, interesting subjects of further research.

6.2. HLD model Approximation

Figure 5 showed large difference between the CPU's homogeneous equivalent service function and the actual service function measured on-line. Our experimental study revealed that these differences cause significant errors in estimating utilizations and mean queue lengths by the load-dependent model (LDM).

Because a high CV at the CPU can cause a backlog there, we expect that system states

$$(CPU, I/O, I/O) = (n_1, n_2, n_3) = (N, 0, 0)$$

account for progressively larger proportions of state occupancy when the CPU CV increases. We found that, indeed, $p(N,0,0)$ grows with the CV, but that the HLD model seriously overestimates $p(N,0,0)$. Figure 11 confirms the tendency of the HLD model to overestimate the

[9] It is also interesting that the M/M/1 formula for mean queue length, $\bar{n} = U/(1-U)$, is nearly exact at the balance point, where U is the actual utilization of the device.

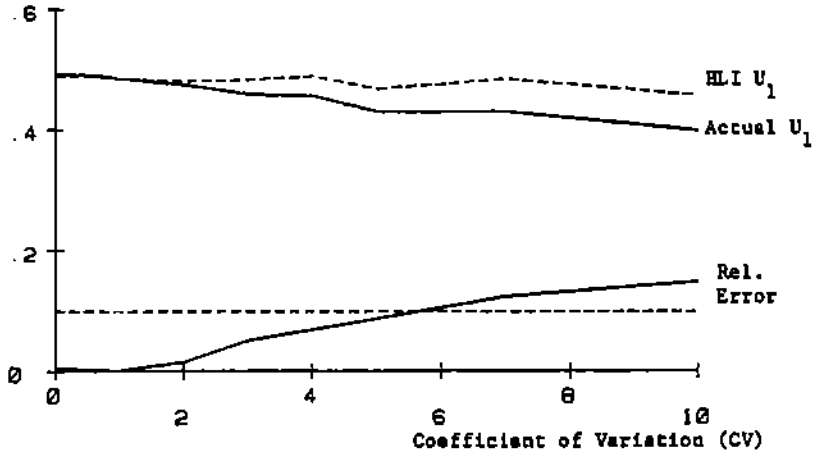

Figure 7. CPU utilization: Actual values, HLI estimates, and relative errors.

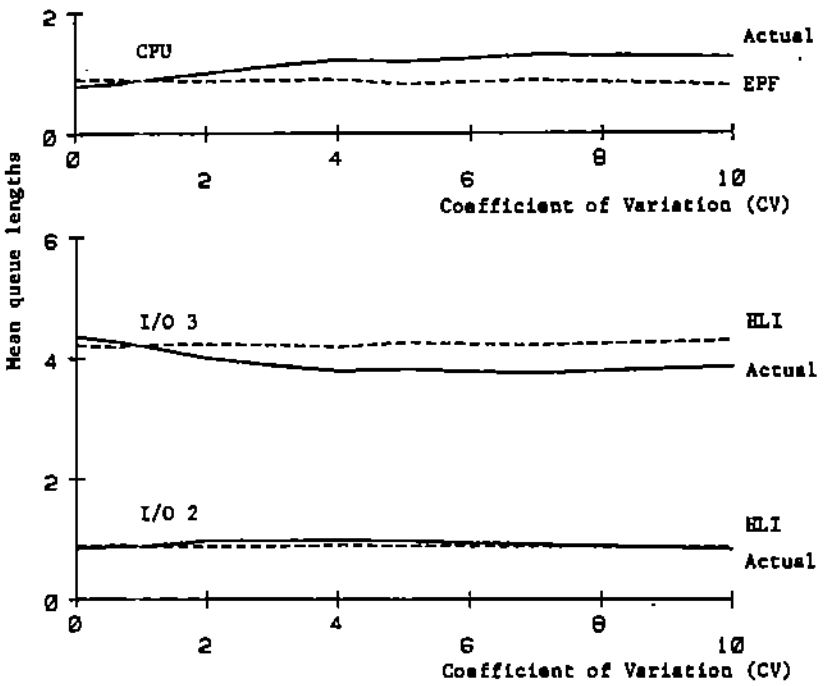

Figure 8. Mean queue lengths in HLI model.

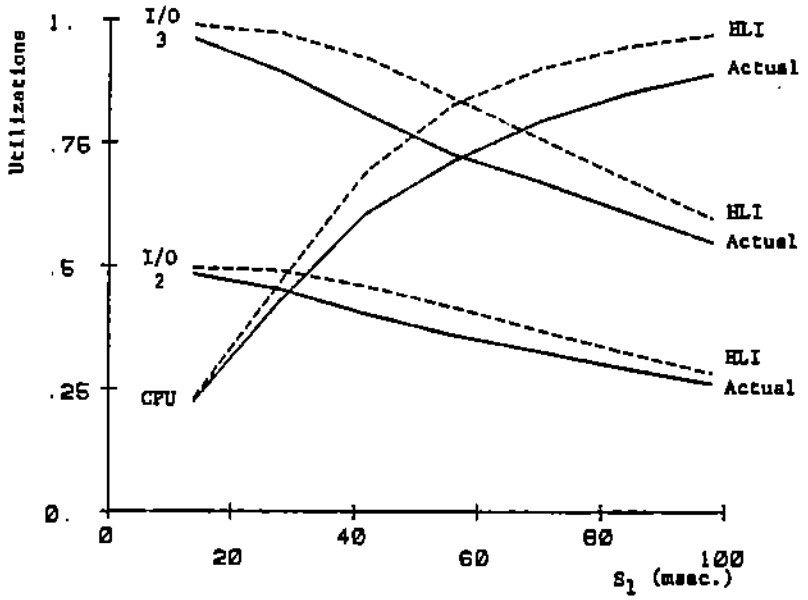

Figure 9. Effect of location of bottleneck on utilization estimates in HLI model.

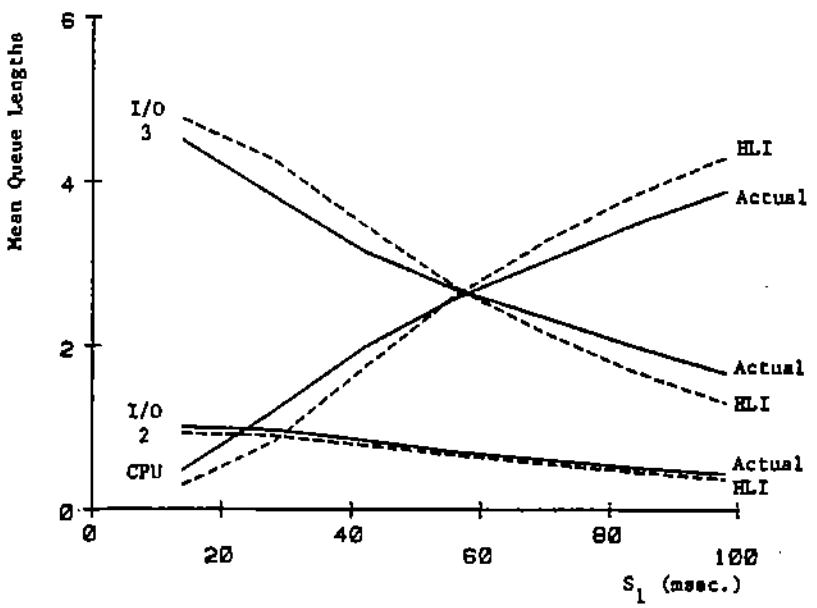

Figure 10. Effect of location of bottleneck on mean queue length estimates in HLI model.

backlog at the device of high CV. The states are ordered according to decreasing likelihood of being observed in the system with N = 6 and CV = 5. The state actually occupied the largest portion of time is (0,0,6) -- reflecting that device 3 is the bottleneck. The state (6,0,0) is actually occupied only about 1/3 as often as (0,0,6) -- showing that the bottleneck is more important than the backlog caused by high CV. In contrast, the HLD model also estimates that states (5,0,1) and (5,1,0), corresponding also to CV-induced backlogs, are more likely than in actuality.

The conclusion is that the HLD model, which uses the on-line service functions attaches far too much importance to backlogs induced by high CV. Backlogs caused by bottlenecks are more important. Because the on-line service function has a shape similar to the homogeneous equivalent service function (See Figure 5), it is possible that a scaling transformation could construct a good approximation to the homogeneous equivalent starting from the on-line function. However we have not investigated this possibility.

Other experiments showed that the HLD model estimated utilizations with almost no error as long as $0 \leq CV \leq 2$. However the utilization errors rapidly multiplied for larger CVs, reaching 10% at CV=3 and 40% at CV=10. The HLD model estimated CPU mean queue length to within 10% only for $0 \leq CV \leq 1.5$. Only when the CPU was the bottleneck did the HLD model give accurate results. The overall conclusion is that the HLD model is less robust than the simpler, HLI model.

6.3. EPF Approximation

In 1976 Shum and Buzen reported an approximation called the extended product form (EPF) [SHUM76, SHUM77]. They noted that the device-factors $\{F_i(n)\}$ in the product form expression for $p(n)$ are proportional to the queue length distributions $\{p_i(n)\}$ in an M/M/1/N queueing system. This insight suggested instead substituting for the $\{F_i(n)\}$ the solution of an M/G/1/N queueing system. By thus incorporating the CVs of the service distributions of the devices into the calculations, this would increase the accuracy of the approximations. A few trial cases suggested that the EPF approximation could estimate actual utilizations and mean queue lengths to within 5% even for CV = 10.

We constructed a version of the EPF algorithm. It is possible to derive service functions $\{S_i(n)\}$ as they would be in an M/G/1/N queueing system, thereby viewing the EPF approximation as another method of estimating the homogeneous equivalent service functions. However, it being more convenient to adopt Shum and Buzen's method intact, we did not explicitly calculate $\{S_i(n)\}$ for the EPF.

As specified by Shum and Buzen, the EPF approximation is computationally difficult to use. The reason is that the $\{S_i(n)\}$ corresponding to the M/G/1/N queueing system depend on the absolute values of the arrival rates $\{\lambda_i\}$ at the devices. Since, in a closed network, the $\{\lambda_i\}$ are not known initially, it is necessary to search the space of all $\{\lambda_i\}$ satisfying the throughput equations of the system until a set $\{\lambda_i\}$ is found for which the completion rate of each device i is also λ_i.

We found instances of the network of Figure 1 in which all the devices had high CVs, and the EPF algorithm could not find a

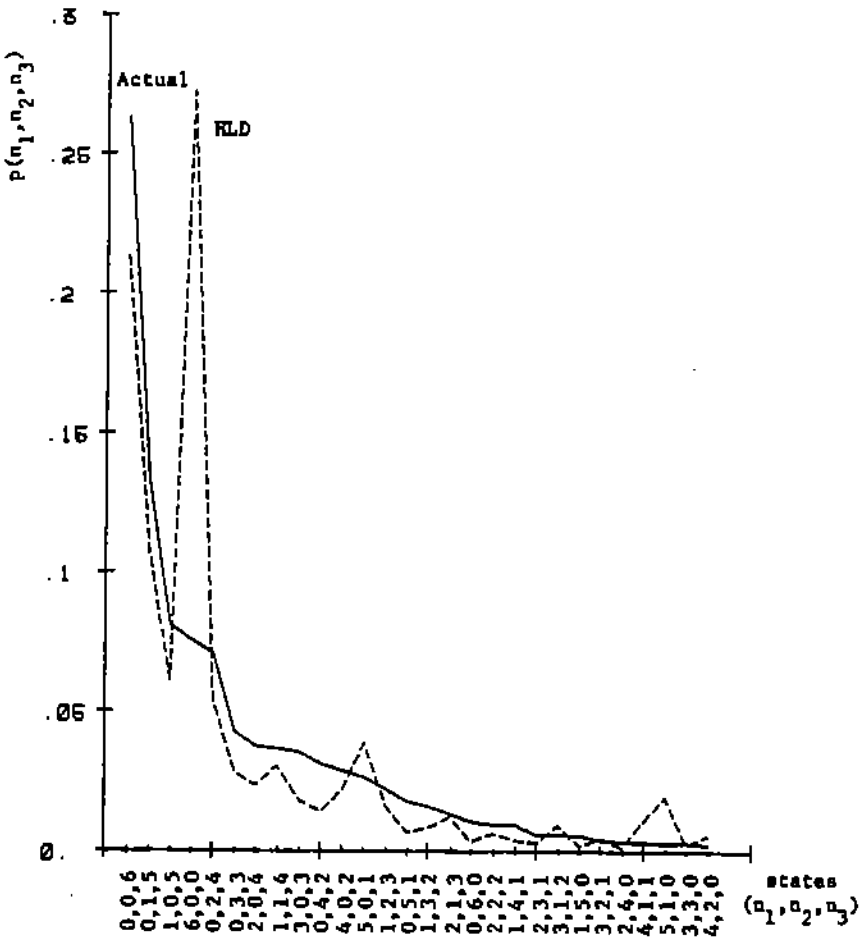

Figure 11. State occupancies.

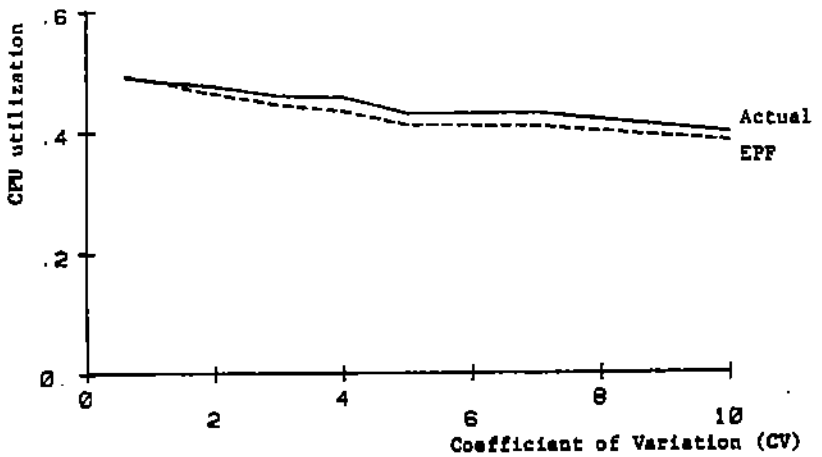

Figure 12. CPU utilization in EPF model.

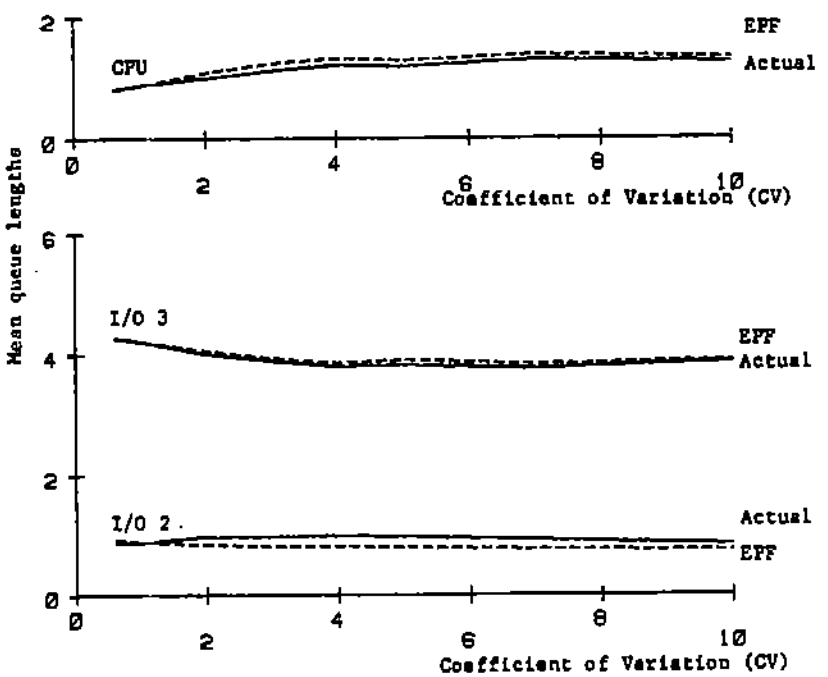

Figure 13. Mean queue lengths in EPF model.

solution. We have since discovered a modified EPF, based on a convex error function, which converges rapidly to a solution in all test cases. (It is the subject of another paper.) The EPF approximation estimates utilizations to within 5% and mean queue lengths to within 10% for all CVs.

Figure 12 illustrates the CPU's utilization (U_1), as estimated by EPF and in actuality, for $0.8 \leq CV \leq 10$. The maximum error is 5%. Similar behaviors were observed for U_2 and U_3, with EPF underestimating consistently and within 5%.

Figure 13 illustrates the mean queue lengths, as estimated by EPF and in actuality, for $0.8 \leq CV \leq 10$. The maximum error in any queue length is 10%, and the error lessens for very high CVs.

The conclusion is that EPF is indeed a robust approximation, but slightly less so than one might expect from reading SHUM76 or SHUM77. The major present limitation is that the published versions of the EPF algorithm may not converge if all devices have high CVs.

7. CONCLUSIONS

We have shown that there exists a product-form queueing network whose marginal queueing distributions are identical to those of a given arbitrary queueing network. A straightforward algorithm rapidly computes the service functions of the equivalent devices when the queueing distributions $\{p_i(n)\}$ are given. This result suggests that the fundamental limitation of queueing network models is not the homogeneity assumption, but rather the inability to estimate the service functions accurately.

The homogeneity assumption asserts that a device's service function measured on-line will be the same as when the service function is measured off-line under constant load. By causing a backlog at a device, a high coefficient of variation (CV) can destroy the homogeneity assumption for that device, producing an error between the product-form model's estimates of utilizations or mean queue lengths and the true values.

Possibly the simplest approximation results from the homogeneous load independent (HLI) model, which sets each $S_i(n)$ to the overall mean service time (S_i) for each value n of the queue length. Our experimental study showed that this model's estimates of utilizations were consistently high and accurate to within 15% for a range on the CPU's CV from 0 to 10. However, this model's estimates of the mean queue length can be significantly in error (e.g. 40%) when the CPU's CV is high. In our study the HLI model was a good estimator (< 10%) of utilization when $0 \leq CV \leq 8$ and for mean queue length when $0.5 \leq CV \leq 1.8$.

The location of the system bottleneck -- at device of high CV or elsewhere -- does not significantly affect the HLI model's (over)estimate of utilization. However, if the devices likely to generate backlogs -- either because of bottlenecks or high CVs -- are in balance (approximately equal utilizations), the HLI model seems to estimate mean queue lengths with very low error. This seems to be consistent with Courtois's argument that balanced systems are more decomposeble (and hence more homogeneous) than unbalanced ones. The relations among balance, backlogs, and HLI

model accuracy are worthy of further investigation.

An approximation suggested directly by the definition of homogeneity is the homogeneous load dependent (HLD) model, whose service functions are the ones observed on-line in the real system. This approximation gives much poorer results than the on-line = off-line intuition leads one to expect. The reason is that this model attaches far too much importance to backlogs caused by high CV, and too little importance to backlogs caused by bottlenecks. In reality, "bottleneck backlogs" appear more influential than "CV backlogs". The similarity of shape between the on-line and the homogeneous equivalent service functions suggests that there may exist simple scaling transformations that estimate the latter from the former.

The extended product form (EPF) approximation (in effect) constructs estimates of service functions by using the solution of the M/G/1/N queueing system. This permits both the mean and CV of a device's interdeparture times to be used in the calculation. The drawback of published implementations of the EPF approximation is their searching the space of solutions to the system's throughput equations; this search is slow and may not converge if too many devices have high CVs. When the EPF algorithm does locate a solution, it usually estimates utilizations to within 5% and mean queue lengths to within 10%. Its minimum error occurs for small CVs ($0.8 \leq CV \leq 2$) or very large CVs.

Figures 14 and 15 compare these approximations for the example studied in this paper.

There remains the question of how important the more sophisticated approximations are in practice. In the Purdue Time Sharing Subsystem, for example we observed the CVs shown in Table 1.

Table 1

RANDOM VARIABLE	CV
Times between job submission from all terminals	1.16
Times between job completions by central computing subsystem	1.06
Total CPU requirement per job	2.15

These CVs are sufficiently low that the HLI model can be used with sufficient accuracy. Many subsystems contain sufficient parallelism as to rule out the possibility of observing a high CV in the times between their job-completions.

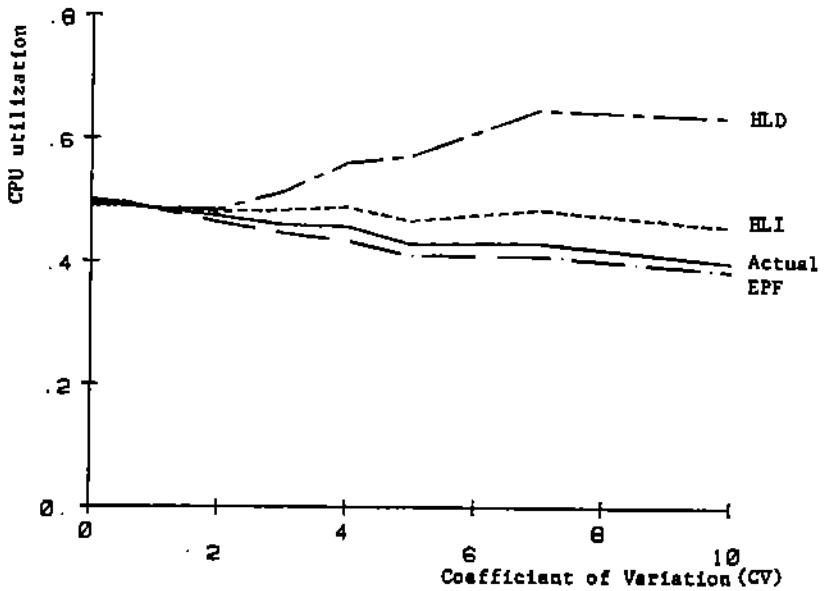

Figure 14. Comparison of CPU utilizations in all models.

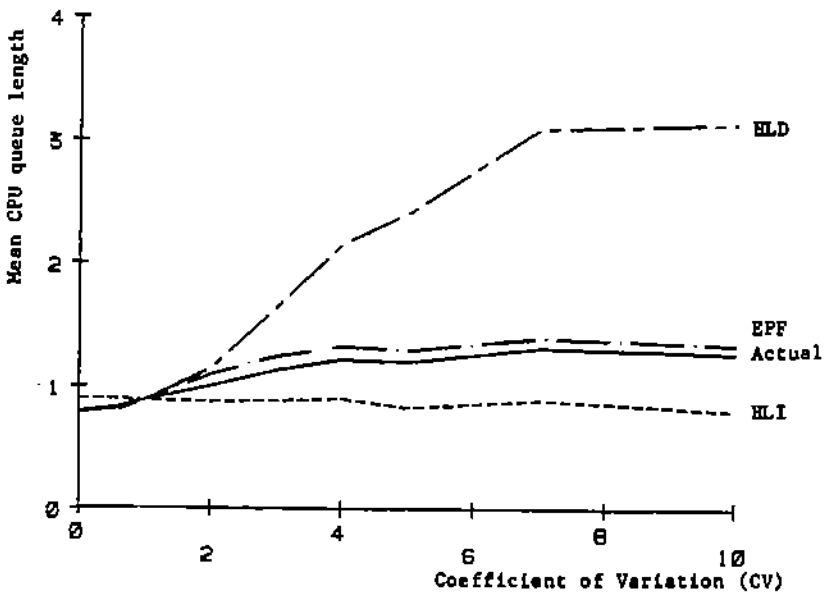

Figure 15. Comparison of mean CPU queue in all models.

ACKNOWLEDGEMENTS

We are grateful to Herbert D. Schwetman for assistance in getting our simulator working; to Steven C. Bruell for providing data on the Purdue System and for helping to implement the EPF algorithm; to Jeffrey Buzen and James Bouhana for insights on the reasons why the high CVs interfere with the predictions of product-form queueing network models.

REFERENCES

BALB77 G. Balbo, S.C. Bruell, and H.D. Schwetman, "Customer Classes and Closed Network Models - A Solution Technique," Proc. 1977 IFIP Congress , Toronto, CANADA, (August 1977), 559-564.

BRAN74 A. Brandwajn, "A Model of a Time Sharing System Solved Using Equivalence and Decomposition Methods," Acta Informatica 4, 1 (1974), 11-47.

BRAN77 A. Brandwajn, "An Approach to the Numerical Solution of Some Queueing Problems," Proc. of the International Symposium on Computer Performance Modeling, Measurement and Evaluation, North-Holland Publishing Co., (August 1977), 83-112.

BUZE73 J. P. Buzen, "Computational Algorithms for Closed Queueing Networks with Exponential Servers," Comm. ACM 16, 9 (September 1973), 527-531.

BUZE76 J. P. Buzen, "Operational Analysis: the Key to the New Generation of Performance Prediction Tools," Proc. IEEE COMPCON, 1976, IEEE, New York.

CHAN75 K.M. Chandy, U. Herzog, and L. Woo, "Approximate Analysis of General Queueing Networks," IBM J. R. & D. 19, (January 1975), 43-49.

CHAN78 K.M. Chandy, and C.H. Sauer, "Approximate Methods for Analyzing Queueing Network Models of Computer Systems," Computing Surveys 10, 3(September 1978).

COOP72 R.B. Cooper, Introduction to Queueing Theory, The MacMillan Company, New York, (1972).

COUR77 P.J. Courtois, Decomposability, ACM Monograph Series, Academic Press (1977).

DENN77 P.J. Denning, and J.P. Buzen, "Operational Analysis of Queueing Networks," Proc. 3rd Int'l Symposium on Modeling and Performance Evaluation of Computer Systems, North-Holland Publishing Co. (1977).

DENN78 P.J. Denning, and J.P. Buzen, "The Operational Analysis of Queueing Network Models," Computing Surveys 10, 3 (September 1978).

GELE75 E. Gelenbe, "On Approximate Computer System Models," J. ACM 22, 2 (April 1975), 261-269.

GELE76 E. Gelenbe, and G. Pujolle, "The Behaviour of a Single Queue in a General Queueing Network," Acta Informatica 7, (1976), 123-136.

KOBA74 H. Kobayashi, "Applications of the Diffusion Approximation to Queueing Networks,"J. ACM 21, 2 (April 1974), 316-338.

KUHN76 P. Kuhn, "Analysis of Complex Queueing Networks by Decomposition," Proc. 8-th International Teletraffic Congress, Melbourne, Australia, (November 1976).

REIS75 M. Reiser, and H. Kobayashi, "Queueing Networks with Multiple Closed Chains: Theory and Computation Algorithms," IBM J. R. & D. 19 (May 1975), 283-294.

REIS78 M. Reiser, and C.H. Sauer, "Queueing Networks Models: Methods of Solution and Their Program Implementations," in Current Trends in Programming Methodology III (K.M. Chandy, and R. Yeh, Eds) Prentice-hall (1978), 116-167.

SAUE75 C.H. Sauer, "Configuration of Computing Systems: an Approach Using Queueing Network Models," Ph.D. Thesis, The University of Texas at Austin, (May 1976).

SEVC77 K.C. Sevcik, A.I. Levy, S.K. Tripathi, and J.L. Zahorjan, "Improving Approximations of Aggregated Queueing Network Subsystems," Proc. of the International Symposium on Computer Performance Modeling, Measurement and Evaluation, North-Holland Publishing Co., (August 1977), 1-22.

SHUM76 A.W. Shum, "Queueing Models for Computer Systems with General Service Time Distributions," Ph.D. Thesis, Division Engrg. & AppliedPhysics, Harvard University, Cambridge, MA 02138 (December 1976).

SHUM77 A.W. Shum, and J.P. Buzen, "The EPF Technique: A Method for Obtaining Approximate Solutions to Closed Queueing Network with General Service Times," Proc. of the 3rd. International Symposium of Modelling and Performance Evaluation of Computer Systems, North-Holland Publishing Co., (October 1977), 201ff.

WILL76 A. C. Williams, andR. A. Bhandiwad, "A Generating Function Approach to Queueing Network Analysis of Multiprogrammed Computers," Networks 6, 1 (1976), 1-22.