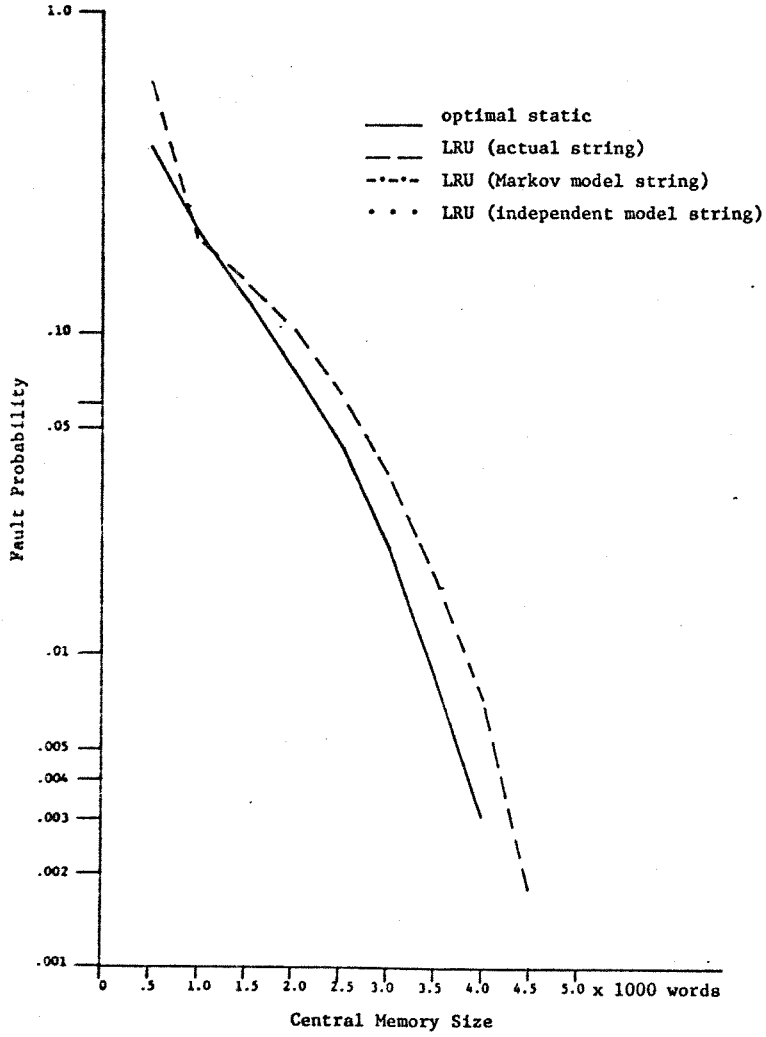


Figure 3.15: Reference Fault Probability vs. CM Size for Tape5

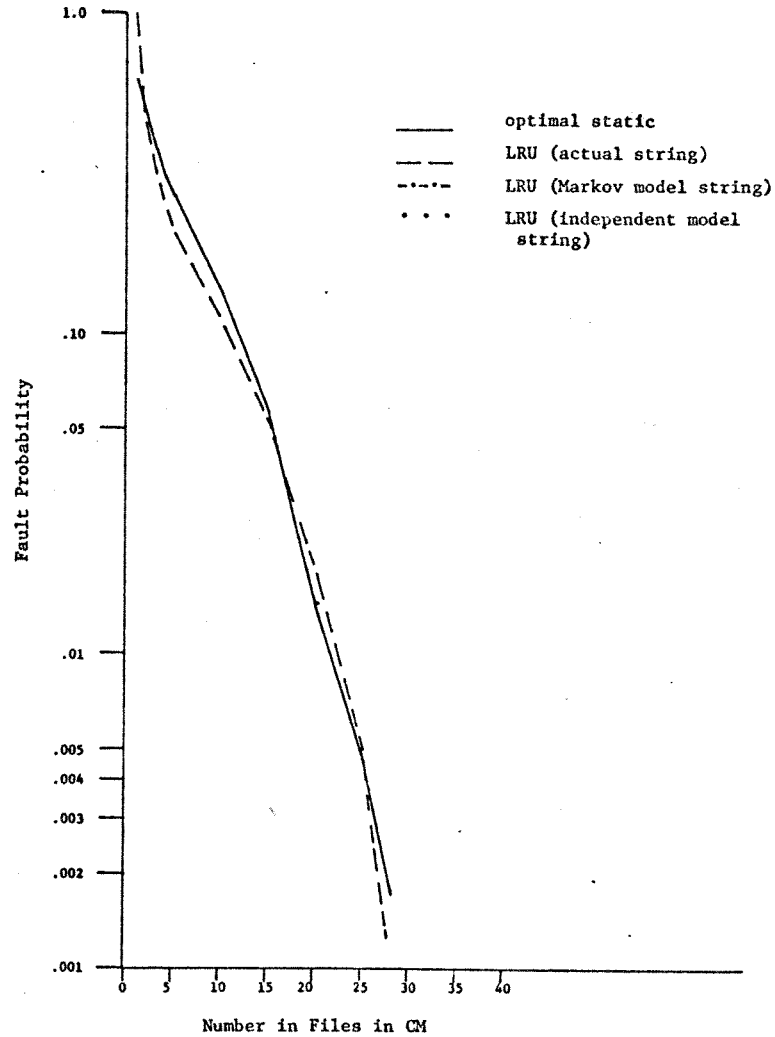


Figure 3.16: Reference Fault Probability vs. M for Tape5

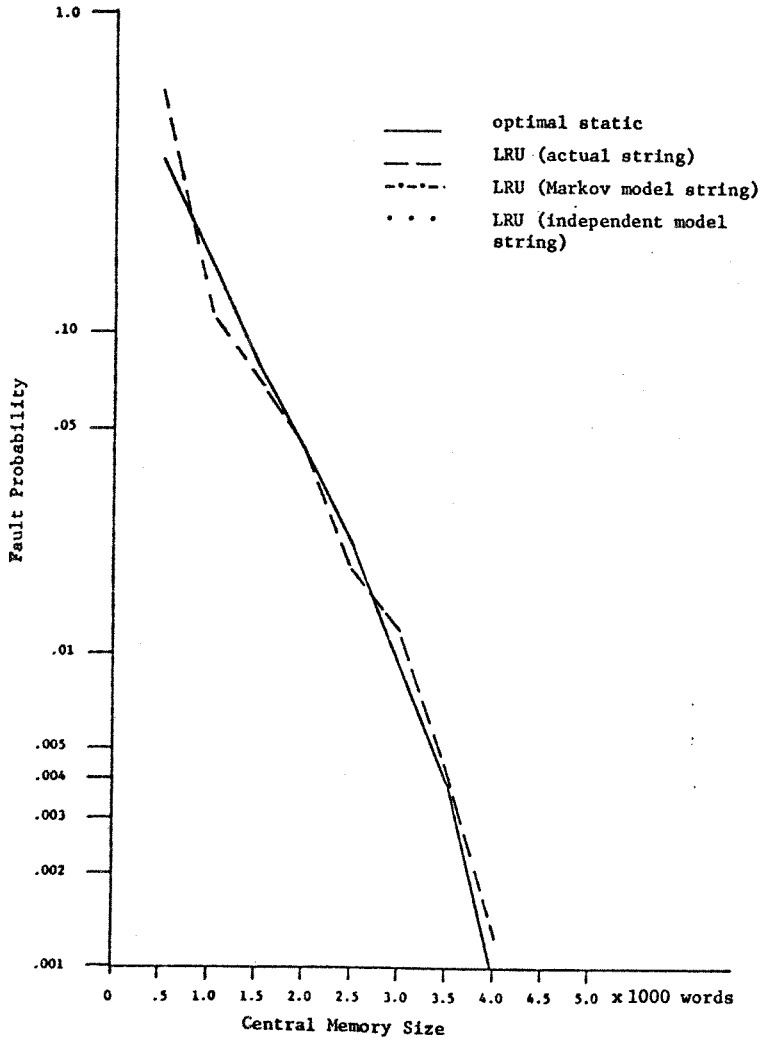


Figure 3.17: Reference Fault Probability vs. CM Size for Tape4a

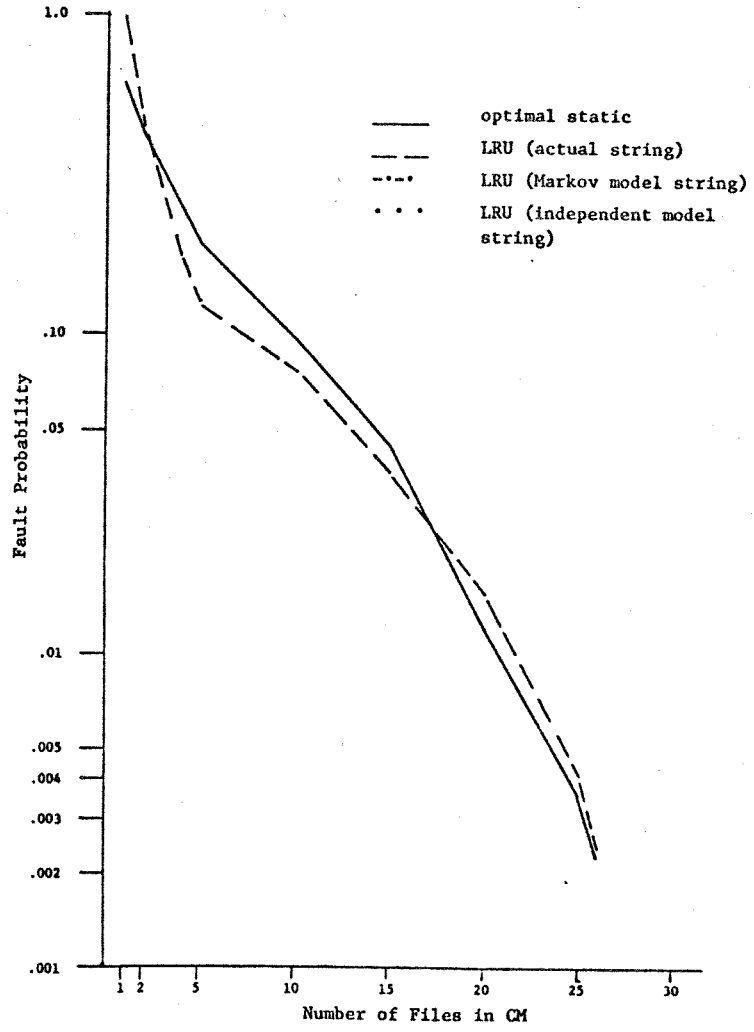


Figure 3.18: Reference Fault Probability vs. M for Tape4a

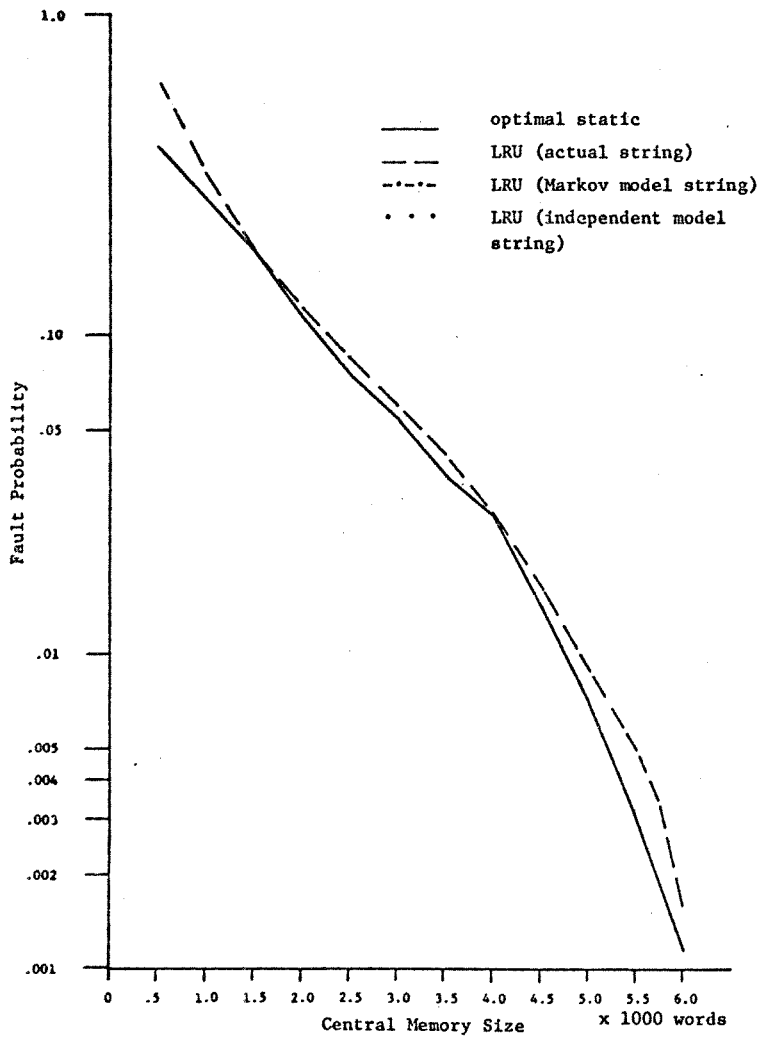


Figure 3.19: Reference Fault Probability vs. CM Size for Tape4

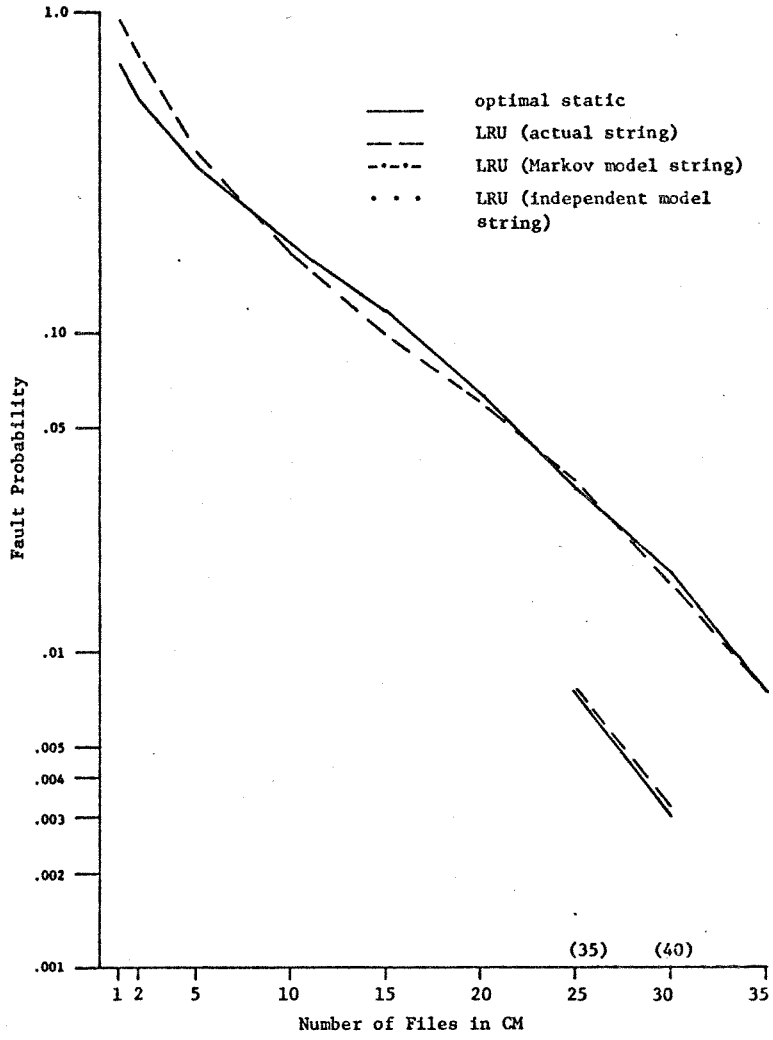


Figure 3.20: Reference Fault Probability vs. M for Tape4

better for larger CM and M. The same is true for the composite tape (figure 3.15) for CM. For different values of M, however, we note that the two curves interweave throughout the range studied. The same is true for the batch monostream and multistream strings (figures 3.18 and 3.20, respectively). We note that the batch monostream curves (figure 3.17) is also interleaved for CM. Curiously, for the batch multistream, CM case (figure 3.19) the optimal static is superior to LRU over the entire range, although the curves are very close. Thus the effect of file sizing makes a difference in this qualitative analysis; a difference so slight, however, that sizing could be ignored.

A comparison between the batch (tape4) and interactive traces reveals the impact of the workload environment upon the strategies. Note that the batch environment is more favorable to the dynamic strategy than the interactive environment. This is a reasonable result, as the total number of user programs active during the course of the trace is smaller for the batch environment. Thus the total behavior of programs over the time interval studied is less homogenized (more dynamic).

3.10 Conclusions

This chapter considered the effect of possible patterns in passive resource requests on a library file system under different memory management strategies. Two goals of our investigation were (1) to increase our knowledge of program behavior, specifically file reference patterning, and (2) to determine the effect of this patterning

on memory management strategies for the library file system. The investigation was broken into two principal phases. The conclusions we draw from each phase follow.

First, a performance comparison was made between allocation strategies via trace-driven simulation models of the file system. We conclude that a static allocation strategy will outperform a dynamic strategy in the actual system. This conclusion is based on the fact that an LFU (∞) allocation strategy outperformed an LRU strategy over a range of memory sizes. The optimal static strategy, in turn, will outperform LFU (∞), since in the long run LFU (∞) produces a static allocation with CM packed according to greatest-frequency-first. This intuitively obvious relationship was verified in initial comparisons which we do not present.

Second, an investigation was made of the user program reference generation process by testing the validity of reference generation models. We conclude that user program file references do display the property of locality but not to the degree to which a dynamic allocation strategy would better the performance of an optimal static strategy in actual implementation. The reason for this conclusion is the observation that the dynamic strategy is, at best, only slightly better than the static. This slight advantage is far outweighed by greater overhead for the dynamic strategy. We observe that a simple Markov process is a poor model for reference generation, but more accurate than a model assuming the independence of references. From this we conclude that the assumption of independence of references does not hold for this system.

More complex Markov models to adequately predict the reference string are intractable.

A positive result of this analysis is the recommendation that the existing static strategy in use by the system be changed to the optimal static strategy. The expense of implementing the strategy is minimal, consisting of the collection of accurate long-term reference frequencies. From the reference frequencies and program sizes algorithm 3.1 can be used to generate a partitioning of the existing library between CM and disk so as to minimize the reference fault probability. How often the library should be re-partitioned depends upon the rate at which the frequencies change. From our analysis we conclude that the reference frequencies do not change significantly within the approximately 30 minute interval recorded by the trace mechanism. This does not mean the reference frequencies do not change over longer periods.

IV. The Effects of Resource Request Patterns in Queueing Network Models

4.1 Introduction

We have considered the possibility of locality in system file references in the previous chapter. In this chapter we examine the effects of more generalized customer behavior patterns upon system performance. Again we base our examination upon queueing network models. The central question in this investigation is "How do customer resource request behavior patterns impact existing queueing network models of computer system performance?" The motivation for asking this question comes from the observation that jobs do exhibit patterns in their behavior. One behavior pattern is locality of references, discussed in the last chapter. Another pattern is CPU burst time "modes" investigated by Lo [L1]. The increasingly popular queueing network models for which product-form solutions hold are seemingly insensitive to many types of patterned job behavior, remaining accurate performance predictors of systems in which behavior patterns surely exist. Why? The answer to this question affects the robustness of existing queueing network models (and hence our confidence in their accuracy). If the impact of behavior patterns is important, then clearly the analyst must carefully characterize the workload behavior patterns, a formidable task since the practical definition of pattern is necessarily vague. If, however, the impact of patterning is negligible, then the analyst may ignore behavior

patterns and be satisfied that grosser workload characterizations will suffice.

In this chapter we define a means of formally modeling patterns of active resource requests for customers in queueing network models, designated customer task graphs (ctg's). It is shown that this formalism can represent a very large class of behavior patterns. We prove that for queueing network models obeying local balance [C1] customer behavior patterns cannot impact performance measures of the models, so long as the ctg's representing arbitrary behavior patterns affect the same server workloads upon the model. We next consider queueing network models with arbitrary ctg's which are not in local balance. Since closed form solutions to these models do not exist, we cannot make as definitive a statement as for local balance models. A series of experiments are conducted over what are considered reasonable computer job behavior patterns and computer system models. The results of the experiments for these models show that job behavior patterns have little impact on the common performance metrics of the systems. The implications of these results for the systems analyst are then considered.

4.2 Customer Task Graphs

We define the sequencing of customer service requests by a directed graph which we label a customer task graph (ctg). The ctg is composed of n nodes, N_i, i=1,...,n. Associated with each N_i is the server S_i requested by the customer and a customer type T_i. Let the set of servers be S, of size s, and the set of types T, of

size t, such that $S_i \epsilon S$ and $T_i \epsilon T$, i=1,...,n. Let an arc from N_i to N_j represent the possibility of the customer next specifying a request for server S_j with type T_j upon completion of service at server S_i with type T_i. Let $q_{i,j}$ represent the non-zero probability associated with this event, with $\sum_j q_{ij} = 1$.

Now consider a queueing network populated by M customers whose service requests are determined by a ctg, with all customers obeying the same ctg. The service time distribution at each server S_j, j=1,...,s, may vary by customer type. Chandy [C1] and Baskett, Chandy, Muntz, and Palacios [B1] have determined closed form solutions to a class of these queueing networks which have the property of local balance. The ctg is implicitly defined in this class of networks by the branching probabilities p(i,r:j,s), which is the probability of a customer of type r transiting to queue j with type s upon completion of service at queue i. We define a graph notation instead of utilizing the p(i,r:j,s) convention because the ctg notation utilizes fewer customer types, and because the graph convention appears more natural for the representation of customer behavior.

Consider the example ctg in figure 4.1. The p(i,r:j,s) notation would require n_1+n_2-1 types to insure the sequencing of n_1 accesses to queue 1 followed by n_2 accesses to queue 2, whereas the equivalent ctg requires but two types (at the expense of n_1+n_2 nodes). This example makes clear the equivalence of the two notations, as we need only to specify one type in the p(i;r:j,s)

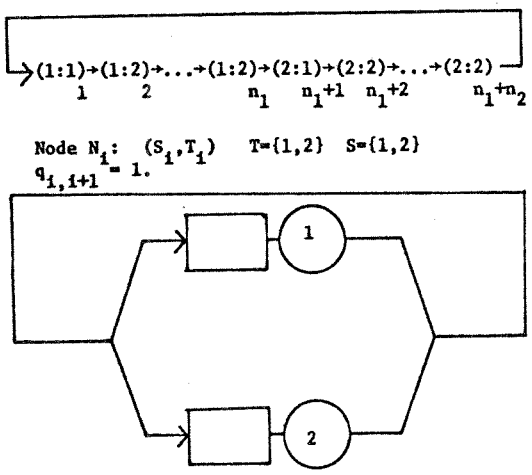


Figure 4.1: A Customer Task Graph and Its Associated Queueing Network

notation for each node in the ctg.

4.3 Proof of Equivalence of Customer Task Graphs

In this section we consider ctg queueing networks which meet the local balance condition. We prove that for every ctg queueing network with many customer types there exists an equivalent queueing network with a single type. Two queueing networks are equivalent if they have the same queue-length distributions by server, with queue length being defined as the total number of customers occupying the queue and its server. If two ctg queueing networks are equivalent to the same single-type queueing network, then they are obviously equivalent. The consequences of this property are discussed at the end of this section.

Consider the class of ctg queueing networks described in [B1]. In the preceding section we established that the ctg notation and that of [B1] are equivalent. We slightly modify the notation of [B1] for ease of exposition. We define a closed queueing network, QN, as composed of L queues, their associated servers, and M customers. There is an arbitrary but finite number of customer types R. Customers traverse the network and change type by branching probabilities p(i,r:j,s), with the transition being from queue i to queue j and from type r to type s. The service discipline at each queue is either processor-shared (PS) or first-come-first-served (FCFS). Service times for FCFS servers are assumed exponentially distributed and identical for all types. Service time distributions for PS servers must have rational Laplace transforms and may vary by customer

type.

Let μ_{ir}^{-1} be the mean service time for a customer of type r at server i with $\mu = \{\mu_{ir}, i=1,\ldots,L \text{ and } r=1,\ldots,R\}$. Let e_{ir} be defined by the set of linear equations

$$\sum_{L,R} e_{ir} \cdot p(i,r:j,s) = e_{js}.$$

The e_{ir} may be interpreted as the relative frequency of visits to queue i for customers of type r and are defined to within an arbitrary multiplicative constant. Closed form solutions for aggregate states of the system are derived in [B1], with the probability of the system in state S being

$$P[S=(y_1,y_2,\ldots,y_L)] = Gg_1(y_1)g_2(y_2)\ldots g_L(y_L) \qquad (4.1)$$

where $y_i=(n_{i1},n_{i2},\ldots,n_{iR})$ and n_{ir} is the number of customers in queue i of type r. G is a normalizing constant and is defined as $\Sigma P[S=(y_1,y_2,\ldots,y_L)]$, summed over $y_1+\ldots+y_L=M$. The g_i are defined as

$$g_i(y_i) = n_i! \prod_{r=1}^{R} (1/n_{ir}!)\beta_{ir}^{n_{ir}} \qquad (4.2)$$

where

$$\beta_{ir} = \begin{cases} e_{ir}/\mu_{ir} \text{ for PS queues} \\ e_{ir}/\mu_i \text{ for FCFS queues} \end{cases} \qquad (4.3)$$

Let us define another marginal distribution by introducing the aggregate state $A = (n_1,n_2,\ldots,n_L)$ so that

$$P[A=(n_1,n_2,\ldots,n_L)] = Gf_1(n_1)f_2(n_2)\ldots f_L(n_L) \qquad (4.4)$$

where n_i is the total number of customers in queue i and

$$f_i(n_i) = \sum_{T} g_i(y_i) \qquad (4.5)$$

where $T = \{n_{i1}+n_{i2}+\ldots+n_{iR} = n_i\}$.

Theorem. For every queueing network QN as defined there exists an equivalent queueing network QN′ with R′=1 such that for any aggregate states A and A′, P[A] = P[A′].

Proof. Substituting (4.2) into (4.5) we obtain

$$f_i(n_i) = \sum_{T} n_i! \prod_{r=1}^{R} (1/n_{ir}!)\beta_{ir}^{n_{ir}}. \qquad (4.6)$$

We note that (6) is a multinomial expansion, thus

$$f_i(n_i) = (\beta_{i1}+\beta_{i2}+\ldots+\beta_{iR})^{n_i} \qquad (4.7)$$

We can define a single type for QN′ such that

$\beta_i' = \sum_{r=1}^{R} \beta_{ir}$. Thus $f_i'(n_i) = f_i(n_i)$ by (4.7).

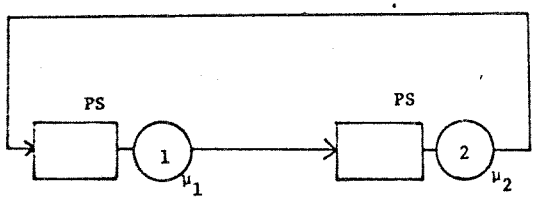
From (4.4) P[A] = P[A]′.

The implications of this property of ctg queueing network models in local balance is profound: specific patterns in customer behavior have no impact upon the queue-length distributions of the network. The practical implication is that specific job behavior patterns play no role in determining the performance of computer systems which are adequately described by local-balance queueing network models. The importance of this property to the computer system analyst is more fully discussed in the last section of this

chapter.

4.3.1 Example

Consider the queueing network of figure 4.2. The $p(i,r;j,s)$, e_{ir}, M, and L are defined in the preceding section. The queueing discipline is PS for both queues. β_1 and β_2 are defined in the proof. We observe that the marginal aggregate state distribution for both the original and equivalent networks are identical.



$p(1,1:2,1)=1/2$, $p(1,1:2,2)=1/2$, $p(1,2:2,1)=1$, $p(1,2:2,2)=0$

$p(2,1:1,1)=1$, $p(2,1:1,2)=0$, $p(2,2:1,1)=1/3$, $p(2,2:1,2)=2/3$.

$e_{11}=1/2$, $e_{12}=1/2$, $e_{21}=1/4$, $e_{22}=3/4$.

$\mu_{11}=1$, $\mu_{12}=2$, $\mu_{21}=2$, $\mu_{22}=3$.

$\beta_{11}=1/2$, $\beta_{12}=1/4$, $\beta_{21}=1/8$, $\beta_{22}=1/4$.

Figure 4.2: An Example of CTG Equivalence

TABLE 4.1: State Probabilities for the Example of Figure 4.2

State description: (y_1,y_2), $y_1=(n_{11},n_{12})$, $y_2=(n_{21},n_{22})$ where n_{ij} is the number of customers of type j in queue i. M=2.

n_{11},n_{12}	n_{21},n_{22}	$g(y_1)\ g(y_2)$	$s(y_1)\ s(y_2)$
2 0	0 0	β_{11}^2	.25
0 2	0 0	β_{12}^2	.0625
1 1	0 0	$2\beta_{11}\beta_{12}$	.25
1 0	1 0	$\beta_{11}\beta_{21}$	.0625
1 0	0 1	$\beta_{11}\beta_{22}$	.125
0 1	1 0	$\beta_{12}\beta_{21}$	.03125
0 1	0 1	$\beta_{12}\beta_{22}$	.0625
0 0	2 0	β_{21}^2	.015625
0 0	1 1	$2\beta_{21}\beta_{22}$	.0625
0 0	0 2	β_{22}^2	.0625

$G = .984375$

Aggregate state probabilities: $P(n_1,n_2)$

n_1	n_2	$P(n_1,n_2)$
2	0	.5625
0	2	.140625
1	1	.28125

Equivalent network $\beta_1 = \beta_{11} + \beta_{12} = 3/4$, $\beta_2 = \beta_{21} + \beta_{22} = 1/4$.

n_1	n_2	$f(n_1)\ f(n_2)$
2	0	.5625
0	2	.140625
1	1	.28125

4.4 The Effect of Service Time Patterns in CTG Queueing Models

In the preceding section we established that specific behavior patterns did not impact queueing network models in local balance. In this section we consider the effect of specific behavior patterns upon non-local balance models. The property violating local balance in the models we consider is that of allowing different service time distributions by customer type in FCFS queues. The practical motivation for considering this particular structure is the speculation that a job's service request distribution may vary according to its past history. Lo [L1] observed that CPU burst time distributions could be characterized as alternating between long-burst and short-burst modes. A possible interpretation of this behavior is that job service requests alternate between long and short modes. We can generalize this property to include both CPU and I/O requests. CPU service disciplines are commonly modeled as PS, while I/O service disciplines are commonly modeled as FCFS. In keeping with the previous section, we evaluate the impact of ctg's with specific sequences of requests to "equivalent" (if the queueing network was in local balance) ctg's which allow arbitrary sequences of requests.

In order to keep the ctg models tractable we must restrict our evaluation to small numbers of queues and job types.

4.4.1 The Models

The two ctg's evaluated are displayed in figure 4.3. In ctg I the customer changes type and resource cyclically. This is

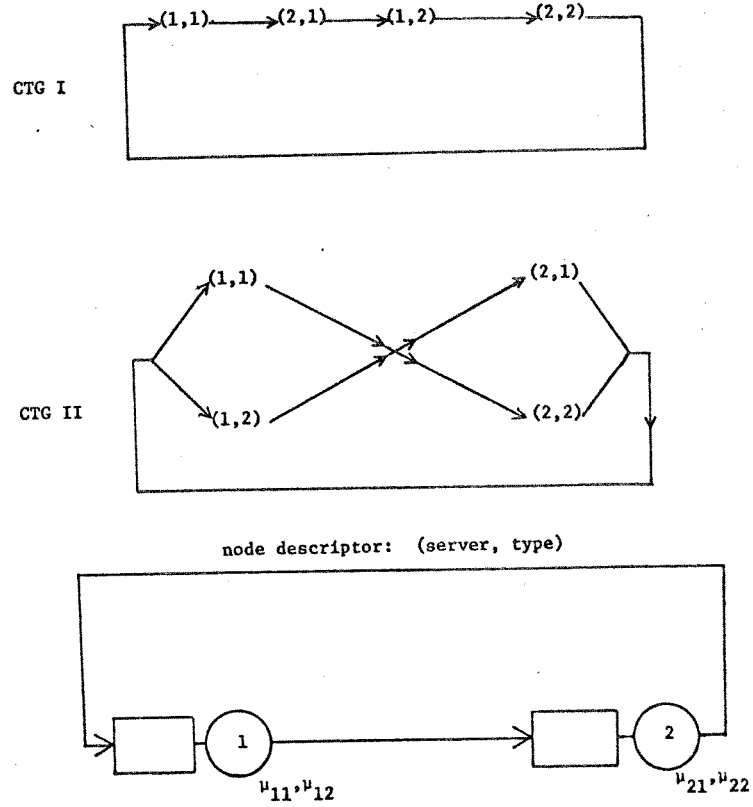


Figure 4.3: Two Customer Task Graph Models (A)

the "patterned" ctg. In ctg II the customer's type is chosen probabilistically while his resource requests are still cyclic. We note the queueing networks associated with these ctg's are identical. Referring to the notation of the previous section, let $\beta_{I,i}$ and $e_{I,i}$ be defined as the composites for server i in ctg I, and similarly in ctg II. Note from the structure of the ctg's that $e_{I,i}=1$ and $e_{II,i}=1/2$, i=1,2 upon inspection. Since the e are defined to within an arbitrary multiplicative constant, for ease of exposition allow $e_{II,i}$ to be redefined as 1 for all i and r. Thus $\beta_{I,i} = \beta_{II,i}$, i=1,2. If the two ctg queueing network models were in local balance they would be equivalent, and hence their performance characteristics would be identical.

The two ctg models are evaluated for different service disciplines (FCFS and PS), for a number of values for μ_{ir} (i=1,2 and r=1,2), and for different numbers of customers. Service times at each server are assumed exponential. For all three models the service discipline of server 2 is FCFS. For model 1 the service discipline of server 1 is FCFS while for models 2 and 3 it is PS.

4.4.2 Solution Method and Validation

The models were solved by an exact matrix method. The number of states for models 1, 2, and 3 are 12, 32, and 26, respectively. A property of model 2, ctg I is that the system is non-ergodic, the state space partitioning itself into two non-intersecting ergodic chains. The steady-state probabilities for states of the larger of the two chains were used in the comparison.

The solution method was validated for all models by setting $\mu_{i1}=\mu_{i2}$ for FCFS server i, bringing the model into local balance. Results were then compared against those obtained from ASQ [K1], a program for the solution of local balance models.

4.4.3 Results

A range of values for the mean services rates μ were chosen for the investigation of equivalence. With no loss of generality μ_{11} was set at 1. Tables 4.2-4.4 display the results of the investigation. An inspection of the tables reveals that the queue-length distributions of the two models usually agree to within 1%, with the greatest disagreement being 13%. Note that server utilization (1-P(2) for server 1 and 1-P(0) for server 2) agree in all cases within 13%.

Note that the queue-length distributions for ctg's I and II of model 1 appear identical. This identity was tested by coding the solution method used for additional precision for model 1. The queue-length distributions obtained for ctg's I and II were identical to within the inherent accuracy of the solution method (at least 10^{-20}).

The close agreement of the queue-length distributions for ctg I and ctg II over the range of parameters leads us to the conclusion that the two ctg models are nearly equivalent.

4.5 The Effect of Branching Patterns in CTG Models

In keeping with the preceding section we consider the impact of specific ctg patterns upon non-local balance models. Non-

TABLE 4.2: COMPARISON OF EQUIVALENCY
FOR MODEL 1, CTG I VS. CTG II

SERVICE DISCIPLINES: SERVER 1 - FCFS, SERVER 2 - FCFS
P(I): PROBABILITY OF I JOBS IN QUEUE 1
NUMBER OF CUSTOMERS: 2

CTG	μ_{11}	μ_{12}	μ_{21}	μ_{22}	P(0)	P(1)	P(2)
I	1.0	1.0	1.0	4.0	.216	.275	.510
II					.216	.275	.510
I	1.0	4.0	1.0	4.0	.371	.258	.371
II					.371	.258	.371
I	1.0	16.0	1.0	4.0	.443	.212	.345
II					.443	.212	.345
I	4.0	1.0	1.0	4.0	.371	.258	.371
II					.371	.258	.371
I	4.0	4.0	1.0	4.0	.649	.228	.123
II					.649	.228	.123
I	4.0	16.0	1.0	4.0	.768	.161	.071
II					.768	.161	.071
I	16.0	1.0	1.0	4.0	.443	.212	.345
II					.443	.212	.345
I	16.0	4.0	1.0	4.0	.768	.161	.071
II					.768	.161	.071
I	16.0	16.0	1.0	4.0	.901	.086	.013
II					.901	.086	.013
I	1.0	1.0	1.0	16.0	.201	.223	.576
II					.201	.223	.576
I	1.0	4.0	1.0	16.0	.345	.212	.443
II					.345	.212	.443
I	1.0	16.0	1.0	16.0	.410	.180	.410
II					.410	.180	.410
I	4.0	1.0	1.0	16.0	.345	.212	.443
II					.345	.212	.443
I	4.0	4.0	1.0	16.0	.619	.190	.190
II					.619	.190	.190
I	4.0	16.0	1.0	16.0	.741	.141	.118
II					.741	.141	.118
I	16.0	1.0	1.0	16.0	.410	.180	.410
II					.410	.180	.410
I	16.0	4.0	1.0	16.0	.741	.141	.118
II					.741	.141	.118
I	16.0	16.0	1.0	16.0	.886	.082	.032
II					.886	.082	.032

TABLE 4.3: COMPARISON OF EQUIVALENCY
FOR MODEL 2, CTG I VS. CTG II

SERVICE DISCIPLINES: SERVER 1 - PS, SERVER 2 - FCFS
P(I): PROBABILITY OF I JOBS IN QUEUE 1
NUMBER OF CUSTOMERS: 2

CTG	μ_{11}	μ_{12}	μ_{21}	μ_{22}	P(0)	P(1)	P(2)	P(3)
I	1.0	1.0	1.0	4.0	.127	.155	.264	.455
II					.135	.158	.247	.460
I	1.0	4.0	1.0	4.0	.284	.149	.284	.284
II					.302	.198	.198	.302
I	1.0	16.0	1.0	4.0	.362	.130	.259	.249
II					.385	.179	.159	.277
I	4.0	1.0	1.0	4.0	.284	.284	.149	.284
II					.302	.198	.198	.302
I	4.0	4.0	1.0	4.0	.620	.216	.115	.050
II					.623	.211	.108	.058
I	4.0	16.0	1.0	4.0	.755	.136	.087	.022
II					.757	.156	.059	.028
I	16.0	1.0	1.0	4.0	.362	.309	.081	.249
II					.385	.179	.159	.277
I	16.0	4.0	1.0	4.0	.755	.189	.034	.022
II					.757	.156	.059	.028
I	16.0	16.0	1.0	4.0	.900	.086	.013	.001
II					.900	.085	.013	.002
I	1.0	1.0	1.0	16.0	.113	.121	.237	.529
II					.126	.131	.203	.536
I	1.0	4.0	1.0	16.0	.249	.081	.309	.362
II					.277	.159	.179	.385
I	1.0	16.0	1.0	16.0	.317	.048	.317	.317
II					.352	.148	.148	.352
I	4.0	1.0	1.0	16.0	.249	.259	.130	.362
II					.277	.159	.179	.385
I	4.0	4.0	1.0	16.0	.574	.173	.160	.094
II					.587	.167	.123	.123
I	4.0	16.0	1.0	16.0	.719	.085	.154	.043
II					.726	.130	.077	.067
I	16.0	1.0	1.0	16.0	.317	.317	.048	.317
II					.352	.148	.148	.352
I	16.0	4.0	1.0	16.0	.719	.197	.042	.043
II					.726	.130	.077	.067
I	16.0	16.0	1.0	16.0	.883	.081	.031	.004
II					.884	.078	.028	.011

TABLE 4.4: COMPARISON OF EQUIVALENCY
FOR MODEL 3, CTG I VS. CTG II

SERVICE DISCIPLINES: SERVER 1 - PS, SERVER 2 - FCFS
P(I): PROBABILITY OF I JOBS IN QUEUE 1
NUMBER OF CUSTOMERS: 3

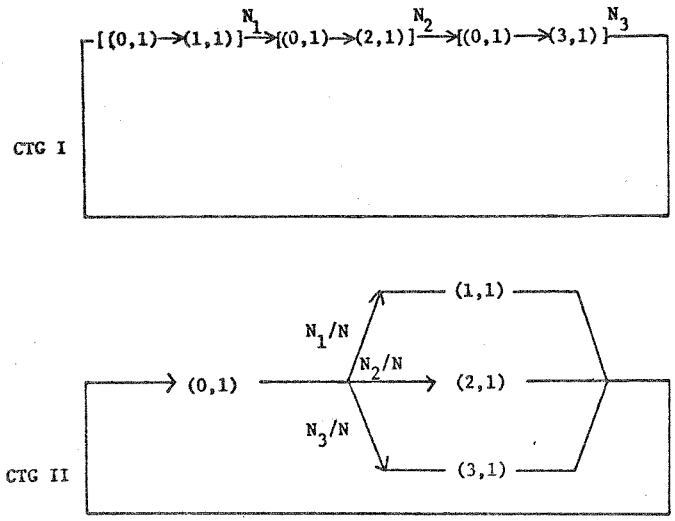
CTG	μ_{11}	μ_{12}	μ_{21}	μ_{22}	P(0)	P(1)	P(2)	P(3)
I	1.0	1.0	1.0	4.0	.134	.157	.250	.459
II					.135	.158	.247	.460
I	1.0	4.0	1.0	4.0	.260	.238	.243	.260
II					.274	.220	.231	.274
I	1.0	16.0	1.0	4.0	.309	.275	.228	.187
II					.332	.239	.215	.214
I	4.0	1.0	1.0	4.0	.284	.207	.224	.284
II					.274	.220	.231	.274
I	4.0	4.0	1.0	4.0	.623	.212	.109	.057
II					.623	.211	.108	.058
I	4.0	16.0	1.0	4.0	.753	.182	.051	.014
II					.755	.170	.056	.019
I	16.0	1.0	1.0	4.0	.348	.214	.205	.233
II					.332	.239	.215	.214
I	16.0	4.0	1.0	4.0	.756	.161	.060	.023
II					.755	.170	.056	.019
I	16.0	16.0	1.0	4.0	.900	.085	.013	.002
II					.900	.085	.013	.002
I	1.0	1.0	1.0	16.0	.124	.129	.211	.535
II					.126	.131	.208	.536
I	1.0	4.0	1.0	16.0	.228	.201	.228	.343
II					.249	.182	.208	.361
I	1.0	16.0	1.0	16.0	.258	.244	.240	.258
II					.294	.205	.208	.294
I	4.0	1.0	1.0	16.0	.262	.166	.199	.373
II					.249	.182	.208	.361
I	4.0	4.0	1.0	16.0	.585	.168	.128	.119
II					.587	.167	.123	.123
I	4.0	16.0	1.0	16.0	.717	.172	.075	.036
II					.721	.147	.078	.053
I	16.0	1.0	1.0	16.0	.318	.178	.187	.318
II					.294	.205	.208	.294
I	16.0	4.0	1.0	16.0	.724	.133	.081	.062
II					.721	.147	.078	.053
I	16.0	16.0	1.0	16.0	.884	.078	.028	.010
II					.884	.078	.028	.011

exponential service distributions in FCFS queues is the property violating local balance in these models. We investigate the effect of deterministic routing through the ctg queueing networks upon the models' equivalence.

The practical motivation for this ctg structure is the observation for some computer systems that disk requests may be clustered in time. One example is a large system in which data bases are tape resident. When a job requiring a data base which is not disk resident is to be run, the appropriate data base must be copied from tape to a single disk--resulting in a preponderant number of accesses to the disk clustered in time. When the data base must be copied back to tape the same behavior is observed. Clustering of device requests is also likely to be observed in data base systems in which data is dumped from disk to tape as part of a rollback/recovery scheme. Deterministic routing is used in the following models to enforce extreme examples of device request clustering.

4.5.1 The Models

The two ctg's evaluated are displayed in Figure 4.4. In ctg I the customer cycles through server 0 and server 1 N_1 times, then cycles through server 0 and server 2 N_2 times, etc. In ctg II the customer chooses server j with probability $N_j/(N_1+N_2+N_3)$, j=1,...,3, upon completion of service at server 0. Using the notation defined in section 4.4, it is intuitively obvious that $e_{I,i} = e_{II,i}$. Let the μ be constant over customer



$N = N_1 + N_2 + N_3$

$[(i,j)\rightarrow(k,\ell)]^n$ is defined as $(i,j)\rightarrow(k,\ell)_1\rightarrow(i,j)\rightarrow(k,\ell)_2\rightarrow\ldots\rightarrow(k,\ell)_n$

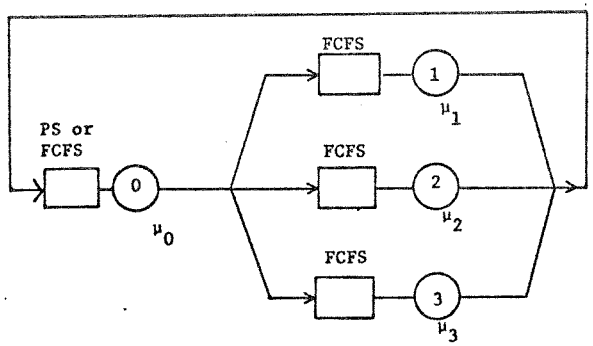


Figure 4.4: Two Customer Task Graph Models (B)

type. Thus the two ctg models would be equivalent if local balance held.

The two ctg models are evaluated for different values of N_i (i=1,...,3), μ_j(j=1,...,3), M (the number of customers), and C_v, the coefficient of variation of the service time distribution for servers 1, 2, and 3. Two values for C_v were chosen for evaluation, 0 and $1/\sqrt{3}$. C_v=0 defines a constant service time and C_v=$1/\sqrt{3}$ lies between 0 and 1, with C_v=1 being a property of the exponential distribution. We note that for C_v=1 the queueing network described is in local balance if its service times are exponential and the two ctg models are known to be equivalent. With no loss of generality μ_0 is set at 1. Servers 1-3 are FCFS while server 0 may be either PS or FCFS.

A central server form for the models was chosen so as to more easily relate the possible behavior patterns for actual systems to the investigation. Thus server 0 represents the CPU, with servers 1-3 representing I/O devices. In the following sections we shall parameterize the model according to assumptions drawn from experience with running computer systems.

4.5.2 Solution Methods, Validation, and an Equivalence Metric

The state space defined by the models was too large for matrix solution methods so a simulation, coded in ASPOL [C9], was used. A Students T test approach [G2] for confidence intervals was used, as the size of the state space made the preferable Crane-Iglehart methods [C13, C14] impractical. Each run iteration of the

simulation model was long enough to insure that all known distribution means (such as device holding times) were within 1% of the simulation input parameters. Values obtained from the simulation are estimated to be within $\pm$ 1% of the actual values with 90% confidence. For $C_v=1/\sqrt{3}$ the actual holding time distribution used is 3-stage Erlangian.

The service discipline for server 0 in the simulation is round-robin with fixed quantum (RRFQ). In the RRFQ discipline the server processes each service request for the duration of a fixed quantum. If the job's service request is complefed during this quantum, it leaves the server, otherwise the job is cycled back to the end of the queue to await its next quantum of service. New arrivals are placed at the end of the queue. The PS discipline is approximated, when appropriate for the models, by setting the quantum to be less (by a factor of 4) than the mean service request time (μ_0^{-1}) for the server. The FCFS discipline is approximated by setting the quantum larger (by a factor of 4) than μ_0^{-1}. Realism was the motivation for using the round-robin discipline for server 0. Actual CPU service disciplines for computer systems are better approximated by round-robin than PS or FCFS, as true PS is impossible to achieve and true FCFS is not desirable (from the results of Sherman, Baskett and Browne [S3]). The service disciplines for servers 1-3 are FCFS, in accordance with their role as I/O devices in the central server model.

The simulation was validated at $C_v=1$ against solutions

obtained via ASQ. Results were in agreement by $\pm$ 1% for all cases considered.

In contrast to the preceding sections, the equivalency metric used was not queue-length--the reason being the awkwardly large amount of data which would result and the greatly increased expense of data gathering in the simulation. An "aggregate" equivalency metric of server 1 utilization was instead chosen. This metric is reasonable since it is also the thruput of server 1 (since μ_0 is set at 1), and represents the work capacity of the system.

4.5.3 Model Parameters and Results

Table 4.5 displays the results of the comparison for different parameterizations of the models. For models 1-7 the discipline for server 0 is PS, for the others it is FCFS. ρ is defined as in chapter 2 and represents the balance between server 0 and the servers 1-3 subsystem. Values for μ_1, μ_2, and μ_3 were chosen to satisfy the relationship $\mu_2^{-1}=\mu_3^{-1}=3\mu_1^{-1}$ and to satisfy the definition of ρ. The factor 3 was selected as the balance between servers since that is typically the disparity in mean access times between disks and drums for accesses to single records. The values for N_1, N_2, and N_3 such that $N_1=N_2=N_3$ reflect equal average frequencies of visits to servers 1-3 (corresponding to $e_1=e_2=e_3$) while $N_1=3N_2=3N_3$ reflects an equal workload balance between servers 1-3 ($\beta_1=\beta_2=\beta_3$). An integer value for each N_i in the table represents a ctg I model, while a dash represents a ctg II model.

TABLE 4.5: Comparison of CTG Models with Branching Patterns

Model	M	μ_1^{-1}	μ_2^{-1}, μ_3^{-1}	C_v	ρ	N_1	N_2	N_3	Server 0 Utilization
1	4	.429	1.29	0	1	1	1	1	.98
						10	10	10	.97
						-	-	-	.97
2	4	.429	1.29	.58	1	1	1	1	.96
						10	10	10	.94
						-	-	-	.95
3	2	.858	2.58	0	2	1	1	1	.58
						10	10	10	.56
						-	-	-	.55
	4	.858	2.58	0	2	1	1	1	.84
						10	10	10	.80
						-	-	-	.79
4	2	.858	2.58	.58	2	1	1	1	.56
						10	10	10	.53
						-	-	-	.55
	4	.858	2.58	.58	2	1	1	1	.79
						10	10	10	.77
						-	-	-	.76
5	4	.556	1.67	.58	1	3	1	1	.97
						30	10	10	.95
						-	-	-	.96
6	2	1.12	3.36	.58	2	3	1	1	.58
						30	10	10	.57
						-	-	-	.56
	4	1.12	3.36	.58	2	3	1	1	.86
						30	10	10	.81
						-	-	-	.83
7	2	1.12	3.36	.58	2	3	1	1	.55
						30	10	10	.55
						-	-	-	.55
	4	1.12	3.36	.58	2	3	1	1	.80
						30	10	10	.79
						-	-	-	.79
8	4	.429	1.29	.58	1	1	1	1	.96
						10	10	10	.96
						-	-	-	.96

Model	M	μ_1^{-1}	μ_2^{-1}, μ_3^{-1}	C_v	ρ	N_1	N_2	N_3	Server 0 Utilization
9	2	.858	2.58	.58	2	1	1	1	.56
						10	10	10	.55
						-	-	-	.54
10	2	1.12	3.36	0	2	3	1	1	.56
						30	10	10	.56
						-	-	-	.56
	4	1.12	3.36	0	2	3	1	1	.82
						30	10	10	.80
						-	-	-	.79
11	2	1.12	3.36	.58	2	3	1	1	.59
						30	10	10	.57
						-	-	-	.56

For models 8-11 all queues are disciplined FCFS.

An inspection of the results reveals that the two ctg models are nearly equivalent, with disparities typically less than 2% and occasionally as great as 3%. It must be remembered that the values displayed in table 4.2 are approximate, and subject to simulation error within the described bounds.

4.6 Conclusions

The conclusion we draw is that specific active resource request patterns by customers have little or no impact upon the accuracy of queueing network models of computer systems. For one class of queueing network models we have proved that patterns play no role in determining the performance measures of the models. Experiments on a wider class of models saw that patterns had negligible impact upon the performance measures.

The practical significance of this conclusion is that the computer system analyst can ignore active resource request patterns over different workloads provided that: (1) the work ascribed to each active resource remains constant, and (2) no scheduling of service requests is made. Obviously, a scheduling scheme which could predict job resource requests by recognizing patterns in job behavior might result in significantly different system performances for different workloads.

Further research investigating the impact of service request patterns quickly diverges according to the nature of patterns and scheduling schemes considered.

V. Summary and Conclusions

This work makes contributions in the area of analytic modeling solution techniques, broadens existing understanding of the robustness of a class of analytical models of computer systems, and executes a case study of workload characterization and program behavior.

Chapter II extends central server models of computer systems to include a passive resource and develops two analytical approximation techniques for their solution. The first solution technique applies to a model assuming exponential service rates. It was discovered independently by a number of authors ([A4], [C10] and [K2]), including this one. The second, and newer, technique is significantly more accurate than the first and applies to the larger class of models assuming non-exponential service rates. The accuracy of the approximations was demonstrated by an extensive study utilizing simulation and (other) analytical methods over a realistic parameter space for computer system models. Our techniques are a step in the direction of more accurate analytical models of computer systems without sacrificing the chief virtues of easy parameterization and solution. Previous models including passive resources were either accurate and expensive (if not intractable) or inaccurate. Of great practical significance is the fact that the solution techniques are easily incorporated into widespread analytical tools (ASQ [K1], for example). The techniques are not limited in application to the Peripheral Processor model considered in the chapter but are applicable to more general queueing

network models in which the level of parallelism in any subsystem is limited. Thus the techniques are a contribution to queueing theory in general.

In chapter III we sought evidence of locality and patterning in file requests to determine if predictive scheduling would be advantageous. The investigation proceeded in two steps: first, an analysis of program behavior to determine the existence and the nature of the patterning; and second, a study to determine if the patterning could be advantageous to a predictive scheduling memory management scheme of the library file system.

Data for our study was extracted from an event trace taken from an actual system in operation. The performance of static and dynamic memory management strategies was studied by trace driven simulation models and an analytical model. Models of the reference generation process (an aspect of workload behavior) were constructed in order to test assumptions about the process. We found that assuming the process to be described by a Markov locality model, while unrealistic, was more accurate than assuming it to be described by an independent reference model. We concluded that despite the slight degree of locality this implied, a static allocation was, in general, superior in performance to a dynamic allocation strategy over the empirical trace for the given system.

This approach embodied two original contributions. First, this is the first investigation of locality in system file references. Previous investigations have been concerned with user file (page) references. Second, we consider the reference trace comprised of

actual system file names, not the address where they reside. The motivation for this approach is shared by other investigators ([M1], for example) who have studied symbolic (name of) array references in user programs; namely that the mapping of names to addresses by a compiler (in our case system memory manager) distorts what patterning there is. The impact of this investigation would have been profound and far-reaching if strong locality had been found in the system file reference trace. The degree of locality the system exhibits limits the impact to the recommendation of a different memory management scheme for the system.

Chapter IV investigates the consequences of specific active resource request sequences of customers in queueing network models. The restriction that requests be probabilistically sequenced is an often raised criticism of a class of queueing network models. We prove this criticism, while intuitive, to be unfounded. We show that the enforcing of any arbitrary sequence of active resource requests for the local balance models has no impact on the performance measures of the models. We conclude from analytical and simulation results that the impact on performance measures of sequencing is negligible for realistic non-local balance models of computer systems.

The significance of this counter-intuitive property is that it helps explain the robustness of simple models which ignore patterns in resource requests. The simple models retain the same performance measures (and are thus approximately equivalent to) models with arbitrarily sequenced requests, providing that the same long-term workload is placed on the resources. The practical impact of these

results to the computer systems analyst/designer is that he can ignore sequencing in the modeling of many systems, confident in the accuracy of "unsequenced" (and simpler) models.

An important area for further research is the generalization of passive resource models and solution techniques to include interrelating passive resources. Different queueing network models of separate passive resources now exist (this work, [T2], and [B6], for example). The possibility of fusing these disparate analytical models into a common model appears promising. To also provide inexpensive solutions to the model is a problem which will not be solved without considerable expense in both time and effort.

Appendix A: Proof of Norton's Theorem for the Central Server Model

The following proof is taken from Chandy, et al. [C3]. Only slightly modifying the notation in [C3] let there be L-1 I/O queues in the central server model. The queues are indexed 1 to L, with the CPU indexed 1. The service rate for the i^{th} queue when there are k customers in the queue is $U_i(k)$, where i=1,...,M, and k=1,...,M. We assume the service times are exponentially distributed. Let p_i be the branching probability from the CPU to I/O queue i, $i \neq 1$. The states of the system are L-tuples, $(n_1,\ldots,n_L)$, where n_i is the number of customers in queue i, including the customer being served. We note that the n_i are non-negative integers and $n_1+\ldots+n_L = M$, the total number of customers. Let $P(n_1,\ldots,n_L)$ be the steady-state probability that the system is in state $(n_1,\ldots,n_L)$. From Gordon and Newell [G1]

$$P(n_1,\ldots,n_L) = g(n_1,\ldots,n_L)/G \tag{1}$$

$$g(n_1,\ldots,n_L) = \begin{cases} \prod_{i=1}^{L} x_i(n_i) & \text{for any feasible state} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

and

$$G = \sum_{\substack{\text{all} \\ \text{feasible} \\ \text{states}}} g(n_1,\ldots,n_L), \text{ a normalization constant.} \tag{3}$$

We define x_i recursively:

$$x_i(0) = 1,\ x_i(k) = x_i(k-1)\cdot y_i/U_i(k) \tag{4}$$

where k=1,...,M and i=1,...,L. The y_i are defined by the structure of the central server model as

$$y_i = \begin{cases} 1 & i=1 \\ p_i & i=2,\ldots,L \end{cases}. \qquad (5)$$

By reviewing the computational techniques of Buzen [B8], we can define a notation which greatly simplifies the proof. Let a "convolution" between two (R+1)-dimensional vectors, $\underline{A} = [a(0),a(1),\ldots,a(R)]$ and $\underline{B} = [b(0),b(1),\ldots,b(R)]$, be the (R+1)-dimensional vector $\underline{C} = [c(0),c(1),\ldots,C(R)]$, where $C(i) = \sum_{j=0}^{R} a(j)\cdot b(R-j)$ and denote this operation by * , so $\underline{C} = \underline{A}*\underline{B}$.

Applying this definition to the problem, we define the (L+1)-dimensional vector $\underline{X}_i = [x_i(0),\ldots,x_i(L)]$, with the x_i defined by (4). Define L+1 vectors $\underline{G}_0, \underline{G}_1,\ldots,\underline{G}_L$ each of dimension M+1:

$$\underline{G}_0 = (1,0,0,\ldots,0) \text{ and}$$

$$\underline{G}_i = \underline{X}_1*\underline{X}_2*\ldots*\underline{X}_i, \quad i=1,\ldots,M. \qquad (6)$$

Thus

$$\underline{G}_i = \underline{G}_{i-1}*\underline{X}_i \text{ for } i=1,\ldots,M. \qquad (7)$$

Let $G_i(r)$ be the r^{th} element of $\underline{G}_i$, r=0,...,L.

From the definition of convolution

$$G_i(r) = \sum_{R} g(n_1,\ldots,n_i,0,\ldots,0), \qquad (8)$$

where $R = \{n_1+\ldots+n_i = r\}$.

From this definition, G of (3) is equal to $G_L(M)$.

Now let $P_i(n)$ be the marginal probability distribution that there are n customers in queue i. Restricting our attention to queue L, we note that

$$P(n_1,\ldots,n_{L-1},n) = g(n_1,\ldots,n_{L-1},0)\cdot X_m(n)/G. \qquad (9)$$

Summing over the set $\{n_1+\ldots+n_{L-1} = M-n\}$

$$P_L(n) = X_L(n)\cdot G_{L-1}(M-n)/G \qquad (10)$$

for n=0,...,M. Note that the marginal probability for any queue i can be obtained by reordering the indexing so that queue i is re-indexed L.

Finally, let T_i be the thruput of queue i, that is, the rate at which customers are serviced and leave queue i. Again restricting our attention to queue M:

$$T_L = \sum_{n=1}^{M} P_L(n)\cdot U_L(n). \qquad (11)$$

From equation (4)

$$X_L(n)\cdot U_L(n) = X_L(n-1)\cdot y_L, \quad n=1,\ldots,M. \qquad (12)$$

From equations (10), (11) and (12)

$$T_L = y_L\cdot \sum_{n=1}^{M} x_M(n-1)\cdot G_{L-1}(M-n)/G$$

$$= y_L\cdot G_L(M-1)/G. \qquad (13)$$

Norton's Theorem for a Closed Network

Consider the above network of L queues. We first determine the queue-length distribution for queue L as a function of its service rate. We next construct an equivalent network consisting of queue L and a composite queue with rate T(n), where n is the number of customers in the composite queue, n=0,...,M. Set T(n) equal to the thruput of queue L when there are n customers in the network and the service time for queue L is set to zero. Let $P_L'(n)$ be the marginal probability of the queue length L in the equivalent network.

Theorem. The queue length distribution for queue L in the equivalent network is the same as in the given network. Or,

$$P_L'(n) = P(n) \text{ for } n=0, 1,\ldots,M. \tag{14}$$

Proof. From equations (1), (2), and (3) it follows that $P_L'(n)$ of M-n customers in the composite queue and n customers in queue L in the equivalent network is proportional to

$$\prod_{j=1}^{n} U_L^{-1}(j) \prod_{k=1}^{M-n} T(k)^{-1} \tag{15}$$

for a detailed derivation see [C3]. When U_L is set to $+\infty$ we see $x_L(n)=1$ for n=0 and $x_L(n)=0$ for $n\neq 0$; thus

$$T(M) = y_L \cdot G_{L-1}(M-1)/G_{L-1}(M). \tag{16}$$

Similarly,

$$T(n) = y_L \cdot G_{L-1}(n-1)/G_{L-1}(n),\ n=1,\ldots,M. \tag{17}$$

Substituting (17) into (15) we find $P_L(n)$ is directly proportional to

$$\frac{X_L(n)}{y_L^M} \cdot \frac{G_{L-1}(M-n)}{G_{L-1}(0)} \text{ for } n=0,1,\ldots,M. \tag{18}$$

Hence,

$$P_L'(n) \alpha X_L(n) \cdot G_{L-1}(M-n) \text{ for } n=0,1,\ldots,M. \tag{19}$$

However, from equation (10)

$$P_L(n) \alpha X_L(n) \cdot G_{L-1}(m-n) \text{ for } n=0,1,\ldots,M. \tag{20}$$

So,

$$P_L'(n) = P_L(n) \text{ for } n=0,1,\ldots,M. \tag{21}$$

Clearly from the theorem and equation (11) $T_L = T_L'$, and similarly the other performance metrics of queue L for both networks are identical.

Appendix B: The Number of States of the Model of Section 2.5.4

We wish to derive the number of states of the model of section 2.5.4. Let L be the number of I/O queues in the L+1 queue network. The I/O queues are assumed FCFS. Let M be the total number of customers (degree of multiprogramming) and let N be the number of PP's available (the level of parallelism in the I/O subsystem). Furthermore, let the service distribution in each I/O queue be represented by a network of T exponential stages. The structure of the network of service stages (hyperexponential, general Erlang, etc.) does not impact the total number of states of the system and so will not be considered. The CPU service rate is assumed exponential.

A state of the system is the (2L+1)-tuple $(n_0, n_1, s_1, n_2, s_2, \ldots, n_L, s_L)$ where n_0 is the number of customers in the CPU, n_i is the number of customers in queue i, and s_i is the index of the stage of service ($1 \leq s_i \leq T$) for the customer in service at queue i. For $n_i=0$, s_i is arbitrarily defined as zero. We wish to find the number of all combinations of members of the (2L+1)-tuple such that $0 \leq n_i \leq m$, $n_0+\ldots+n_L = m$, and $1 \leq s_i \leq T$ (for $n_i > 0$). We note that for T=1, $s_i=1$, i=1,...,L. The total number of states for this case is $\binom{M+L}{M}$, as the total number of ways to distribute M customers among L queues is $\binom{M+L-1}{M}$.

Let us approach the problem by considering the aggregate state descriptor for L I/O queues $(n_1,\ldots,n_L)$, ignoring the number of customers in the CPU and the stage of service in the I/O queues. We know the number of aggregate states to be $\binom{j+L-1}{j}$ for j customers attempting service in the L I/O queue network. We note for $j > N$ there are N customers queued in the I/O subsystem, and j-N customers waiting to be queued. From each aggregate state can be inferred the number of states which comprise the aggregate, namely $T^{(L-i)}$, where is is the count of the n_i such that $n_i=0$, i=1,...,L.

Define S(i,j) as the number of ways of distributing j jobs among L queues with exactly i queues empty. Thus for exactly i empty queues there are L-i queues with one or more customers. Now there are j-(L-i) customers to be allocated among the L-i queues which already contain exactly one customer so

$$S(i,j) = \binom{L}{i}\binom{j-1}{j-L+1} \quad 1 \leq j \leq N$$

$$= \binom{L}{i}\binom{j-1}{N-L+1} \quad N \leq j \leq M,$$

by the constraint that for $N \leq j \leq M$ the number of customers in the queues is N. Summing S(i,j) over j

$$S(i) = \sum_{j=L-i}^{n-1} \binom{L}{i}\binom{j-1}{j-L+1} + \binom{L}{i}\binom{N-1}{N-L+1} \quad (M-N+1).$$

The index j is initialized at L-i because there must be at least L-i customers in the L I/O queues to satisfy the constraint of exactly i empty queues.

Now define R(i) as the number of states of the system with i empty I/O queues comprising the aggregate state. By our previous discussion

$$R(i) = T^{(L-i)}S(i).$$

Note that there is additionally one state where all customers are in the CPU (j=0). So the total number of states is

$$R = 1 + \sum_{i=0}^{L-i} \binom{L}{i} T^{(L-i)} \left[\sum_{j=L-i}^{N-1} \binom{j-1}{j-L+1} + (M - N + 1) \binom{N-1}{N-L+1} \right].$$

Appendix C: Trace Tapes

Table C.1: Environment

TAPE	Tape1	Tape2	Tape3	Tape4
DATE	10/18/74	05/21/75	08/06/75	11/11/75
TIME	15.34.69	15.03.18	15.56.42	22.02.13
MACHINE	6400	6400	6400	6400

Duration of the traces was approximately 30 minutes.

TRACE	TAPE	ENVIRONMENT	REDUCTION
Tape1	Tape1	interactive	multistream
Tape2a	Tape2	interactive	monostream
Tape2b	Tape2	interactive	monostream
Tape3a	Tape3	interactive	monostream
Tape3b	Tape3	interactive	monostream
Tape4	Tape4	batch	multistream
Tape4a	Tape4	batch	monostream
Tape5	composite of tape2a, tape2b, tape3a, and tape3b.		

TABLE C.2: PROGRAM REFERENCES FOR TAPE1 AND TAPE5

	TAPE1			TAPE5	
NAME	SIZE	FREQUENCY	NAME	SIZE	FREQUENCY
CIO	124	1640	CIO	133	33373
2RD	82	529	2CT	470	15118
2CT	470	400	2RD	83	7351
2WD	85	388	2WD	86	6702
1RJ	230	383	1TM	382	3155
1SJ	183	375	1AJ	157	2669
5DG	39	289	2WM	193	2659
CPU	270	268	2PD	183	2258
1DB	153	248	5DG	38	2215
PFM	258	213	5DB	36	2182
2WM	191	198	OPE	72	1751
OPE	72	195	2TS	333	1551
2PD	179	186	PFM	268	1314
5DB	36	159	SNP	55	1239
2TS	351	153	MSG	38	1195
1TM	382	141	2DF	31	924
1AJ	166	136	RFL	110	916
MSG	38	122	CPU	275	789
SIS	166	102	2PU	215	751
RFL	120	78	RTA	52	447
PCC	45	74	LDR	191	234
3EA	187	67	1SR	146	195
SNP	55	51	RCC	39	146
2DF	31	44	3AJ	104	145
2PU	266	42	1RJ	232	137
LDR	199	27	2EF	142	130
1SS	282	20	3CT	39	88
2EF	122	10	PCC	45	72
RCC	39	9	RSF	217	60
3AJ	101	8	1SJ	183	45
5DE	35	6	ERR	289	16
ERR	439	4	2MT	458	12
1CJ	67	3	2JE	21	7
1SR	136	3	1PS	138	7
2JE	15	1	CLO	12	4
1PS	138	1	DMP	359	3
			3EA	187	1
TOTAL NUMBER OF PROGRAMS		36			37
TOTAL SIZE OF PROGRAMS		5752			6012
NUMBER OF REFERENCES		6575			89861

TABLE C.3: PROGRAM REFERENCES FOR TAPE2

	TAPE2A			TAPE2B	
NAME	SIZE	FREQUENCY	NAME	SIZE	FREQUENCY
CIO	133	4165	CIO	133	11407
2RD	83	1274	2CT	470	6026
2CT	470	1203	2RD	83	2159
2WD	86	1197	2WD	86	1886
5DB	36	600	1TM	382	1206
5DG	38	550	2WM	193	1025
1AJ	157	401	1AJ	157	1003
1TM	382	330	2PD	183	711
2WM	193	324	5DG	38	675
OPE	72	288	5DB	36	674
SNP	55	273	OPE	72	594
2PD	183	264	2TS	333	574
2TS	333	262	PFM	268	496
PFM	268	208	MSG	38	469
2PU	215	180	SNP	55	322
CPU	275	180	RFL	110	291
RFL	110	162	2DF	31	280
2DF	31	131	CPU	275	235
MSG	38	119	2PU	215	213
RTA	52	74	RTA	52	188
RCC	39	41	LDR	191	82
1SR	146	28	1SR	146	74
LDR	191	24	3AJ	104	68
1RJ	232	18	2EF	142	46
2EF	142	15	RCC	39	45
3AJ	104	13	1RJ	232	41
PCC	45	5	PCC	45	25
ERR	289	5	RSF	217	22
RSF	217	5	1SJ	183	19
1SJ	183	3	ERR	289	6
DMP	359	1	2JE	21	5
3CT	39	1	1PS	138	5
			CLO	12	3
TOTAL NUMBER OF PROGRAMS		32			33
TOTAL SIZE OF PROGRAMS		5196			4969
TOTAL NUMBER OF REFERENCES		12344			30875

TABLE C.4: PROGRAM REFERENCES FOR TAPE3

TAPE3A			TAPE3B		
NAME	SIZE	FREQUENCY	NAME	SIZE	FREQUENCY
CIO	133	8282	CIO	133	9519
2CT	470	3232	2CT	470	4657
2RD	83	1904	2RD	83	2014
2WD	86	1800	2WD	86	1819
1TM	382	1000	1TM	382	619
2WM	193	784	2PD	183	540
1AJ	157	775	2WM	193	526
2PD	183	743	1AJ	157	490
5DG	38	653	5DB	36	456
OPE	72	581	5DG	38	337
5DB	36	452	2TS	333	291
2TS	333	424	OPE	72	288
PFM	268	416	SNP	55	252
SNP	55	392	MSG	38	227
MSG	38	380	RFL	110	226
2DF	31	290	2DF	31	223
RFL	110	237	PFM	268	194
CPU	275	227	CPU	275	147
2PU	215	217	2PU	215	141
RTA	52	147	3CT	39	83
LDR	191	84	LDR	191	44
1SR	146	50	1RJ	232	44
RCC	39	34	1SR	146	43
1RJ	232	34	3AJ	104	42
2EF	142	30	2EF	142	39
PCC	45	28	RTA	52	38
3AJ	104	22	RCC	39	26
RSF	217	9	RSF	217	24
1SJ	183	9	PCC	45	14
3CT	39	4	1SJ	183	14
ERR	289	3	2MT	458	12
CLO	12	1	DMP	359	2
2JE	21	1	ERR	289	2
1PS	138	1	2JE	21	1
			1PS	138	1
			3EA	187	1
TOTAL NUMBER OF PROGRAMS		34			36
TOTAL SIZE OF PROGRAMS		5008			6000
NUMBER OF REFERENCES		23246			23396

TABLE C.5: PROGRAM REFERENCES FOR TAPE4

TAPE4A			TAPE4			TAPE4(CONTD.)		
NAME	SIZE	FREQ.	NAME	SIZE	FEQ.	NAME	SIZE	FREQ.
CIO	133	9574	CIO	133	22772	RSF	217	106
2MT	458	4384	2RD	83	10796	1SR	146	101
2WD	86	2351	2WD	86	033	RCC	39	98
2RD	83	2173	2MT	458	212	2AM	62	90
2WM	193	872	1DB	153	539	1TD	116	74
2PD	183	780	2PD	183	948	2SP	362	73
MSG	38	468	2WM	193	579	2TJ	26	54
1AJ	157	385	1PS	138	115	2JE	21	53
OPE	72	361	OPE	72	347	2E1	95	49
2TS	333	353	MSG	38	175	2E2	96	49
SNP	55	291	RFL	110	085	3EA	187	35
2DF	31	275	1AJ	157	032	2EF	142	26
RFL	110	270	2TS	333	979	90L	9	24
5DG	38	242	EPR	230	973	9C7	36	22
2PU	215	201	SNP	55	943	CLO	12	18
CPU	275	201	5DG	38	914	2PL	94	17
5DB	36	184	5DB	36	913	DMP	359	9
PFM	268	134	2DF	31	814	2SC	73	4
RTA	52	129	PFM	268	796	2DS	149	4
3AJ	104	116	PAT	93	510	20U	81	2
PCC	45	47	1PL	148	478	ERR	289	2
RCC	39	47	1SS	287	462	2FE	10	2
RSF	217	35	CPU	275	429	9C8	70	1
LDR	191	34	2PU	215	422	9DI	66	1
1SR	146	34	3AJ	104	332			
1RJ	232	34	1RJ	232	314			
1SJ	183	31	RTA	52	313			
2EF	142	11	1SJ	183	309			
3EA	187	7	LDR	191	184			
DMP	359	4	4MD	104	146			
ERR	289	1	PCC	45	124			
TOTAL NUMBER OF PROGRAMS		31						55
TOTAL SIZE OF PROGRAMS		4950						7481
NUMBER OF REFERENCES		24029						71903

Bibliography

[A1] J. W. Anderson, "Primitive Process Level Modeling and Simulation of a Multiprocessing Computer System," Ph.D. Thesis, University of Texas, Austin, Tx., May 1972.

[A2] S. Arora and A. Gallo, "The Optimal Organization of Multiprogrammed Multi-Level Memory," Proc. ACM Workshop on System Performance Evaluation, pp. 104-141, April 1971.

[A3] S. Arora and A. Gallo, "Optimization of Static Loading of Multilevel Memory Systems," J. ACM, Vol. 20, No. 2, pp. 307-319, April 1973.

[A4] B. Avi-Itzhak and D. P. Heyman, "Approximate Queueing Models for Multiprogrammed Computer Systems," Operations Research, Vol. 21, No. 6, pp. 1212-1231, 1973.

[B1] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios, "Open, Closed and Mixed Networks of Queues with Different Classes of Customers," J. ACM, Vol. 22, No. 2, p. 248, April 1975.

[B2] A. P. Batson and R. E. Brundage, "Measurements of the Virtual Memory Demands of ALGOL-60 Programs," Proc. SIGMETRICS 74 Conf. Montreal - Performance Evaluation Review, 3, pp. 121-126, 1974.

[B3] A. Brandwajn, "A Model of a Time Sharing Virtual Memory System Solved Using Equivalence and Decomposition Methods," Acta Informatica, Vol. 4, pp. 11-47, 1974.

[B4] A. Brandwajn, J. P. Buzen, E. Gelenbe, and D. Potier, "A Model of Performance for Virtual Memory Systems," Proc. ACM SIGMETRICS Symp., September 1974.

[B5] R. S. Brice, "A Study of Feedback Coupled Resource Allocation Policies in a Multiprocessing Computer Environment," Ph.D. Thesis, University of Texas, Austin, Tx., August 1973.

[B6] R. M. Brown, "An Analytic Model of a Large Scale Interactive System Including the Effects of Finite Main Memory," M.A. Thesis, Dept. of Computer Sciences, Univ. of Texas, Austin, Tx., 1974.

[B7] J. C. Browne, K. M. Chandy, R. M. Brown, T. W. Keller, D. F. Towsley, and C. W. Dissly, "Hierarchical Techniques for the Development of Realistic Models of Complex Computer Systems," Proc. IEEE, Vol. 63, No. 6, pp. 966-975, June 1975.

[B8] J. P. Buzen, "Queueing Network Models of Multiprogramming," Ph.D. Thesis, Harvard University, Cambridge, Mass., 1971.

[B9] J. P. Buzen, "I/O Subsystem Architecture," Proc. IEEE, Vol. 63, No. 6, pp. 871-878, June 1975.

[C1] K. M. Chandy, "The Analysis and Solutions for General Queueing Networks," Proc. 6th Annual Princeton Conf. Information Sciences and Systems, Princeton Univ., Princeton, N. J., March 1972.

[C2] K. M. Chandy, T. W. Keller, and J. C. Browne, "Design Automation and Queueing Networks: An Interactive System for the Evaluation of Computer Queueing Models," Proc. Design Automation Workshop, June 1975.

[C3] K. M. Chandy, U. Herzog, and L. Woo, "Parametric Analysis of Queueing Network Models," IBM J. Res. Develop., Vol. 19, No. 1, p. 36, 1975.

[C4] K. M. Chandy, U. Herzog, and L. Woo, "Approximate Analysis of General Queueing Networks," IBM J. Res. Develop., Vol. 19, No. 1, p. 43, 1975.

[C5] K. M. Chandy, J. H. Howard, and D. F. Towsley, "Product Form and Local Balance in Queueing Networks," submitted J. ACM.

[C6] W. Chang and D. J. Wong, "Computer Channel Interference Analysis," IBM Sys. J., Vol. 4., No. 2, p. 162, 1965.

[C7] C. R. Chow, "On Optimization of Storage Hierarchies," IBM J. Res. Develop., Vol. 18, No. 3, pp. 194-203, May 1974.

[C8] E. G. Coffman and P. J. Denning, Operating Systems Theory, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1973.

[C9] Control Data Corporation, A Simulation Process-Oriented Language (ASPOL) Reference Manual, C.D.C. Special Support Division, Sunnyvale, Calif., 1972.

[C10] P. J. Courtois, "On the Near-Complete-Decomposability of Networks of Queues and of Stochastic Models of Multiprogramming Computing Systems," Computer Science Dept. Rep. CMU-CS-72-111, Carnegie-Mellon Univ., Pittsburgh, Pa., November 1971.

[C11] P. J. Courtois, "Decomposability, Instabilities, and Saturation in Multiprogramming Systems," Com. ACM, Vol. 18, No. 7, pp. 371-376, July 1975.

[C12] P. J. Courtois, "Error Analysis in Nearly Decomposable Stochastic Systems," Econometrica, Vol. 43, No. 4, p. 691, July 1975.

[C13] M. A. Crane and D. I. Iglehart, "Simulating Stable Stochastic Systems, I: General Multiserver Queues," J. ACM, Vol. 21, p. 103, 1974.

[C14] M. A. Crane and D. I. Iglehart, "Simulating Stable Stochastic Systems, II: Markov Chains," J. ACM, Vol. 21, p. 114, 1974.

[D1] P. J. Denning, "Virtual Memory," Computing Surveys, Vol. 2, No. 3, pp. 153-189, September 1970.

[D2] P. J. Denning and G. S. Graham, "Multiprogrammed Memory Management," Proc. IEEE, Vol. 63, No. 6, 924-939, June 1975.

[F1] J. H. Florkowski, "Extended Analytic Models for Systems Evaluation," IBM Poughkeepsie Lab. Technical Report TR 00.2549, August 1974.

[F2] D. V. Foster, "File Assignment in Memory Hierarchies," Ph.D. Thesis, Dept. of Computer Sciences, Univ. of Texas, Austin, Tx., 1974.

[G1] W. J. Gordon and G. F. Newell, "Closed Queueing Systems with Exponential Servers," Operations Research, Vol. 15, pp. 254-265, March 1967.

[G2] H. Greenberg, Integer Programming, Academic Press, New York, p. 99, 1971.

[H1] U. Herzog, L. Woo, and K. M. Chandy, "Solution of Queueing Problems by a Recursive Technique," IBM J. Res. Develop., Vol. 19, No. 3, pp. 295-300, May 1975.

[H2] J. H. Howard, Jr., "A Large-Scale Dual Operating System," Proc. ACM 1973 Annual Conf., pp. 242-248, 1973.

[H3] J. H. Howard and W. M. Wedel, "The UT-2 Operating System Event Recorder," TSN-37, Computation Center, University of Texas, Austin, Tx., February 1974.

[I1] Information Research Associates (IRA), "A Limiting Capability Analysis of the Cyber 70 ALS," Austin, Tx., 1973.

[J1] J. R. Jackson, "Jobshop-like Queueing Systems," Management Science, Vol. 10, pp. 131-142, 1963.

[K1] T. W. Keller, ASQ Manual, Dept. of Computer Sciences Report TR-27, University of Texas, Austin, Tx., 1973.

[K2] T. W. Keller and K. M. Chandy, "Computer Models with Constrained Parallel Processors," Proc. Sagamore Conference on Parallel Processing, 1974.

[K3] L. Kleinrock, Queueing Systems, Vol. I: Theory, John Wiley and Sons, Inc., New York, 1975.

[L1] T.-C. Lo, "Computer Aids in Computer Systems Analysis," Ph.D. Thesis, University of Texas, Austin, Tx., 1973.

[M1] A. W. Madison and A. P. Batson, "Characteristics of Program Localities," submitted to Comm. ACM, 1975.

[M2] R. R. Muntz, "Analytic Modeling of Interactive Systems," Proc. IEEE, Vol. 63, No. 6, pp. 946-953, June 1975.

[N1] A. S. Noetzel and J. R. Haley, "The Decomposition of a Multiprocessor Event Trace into User Program Resource Demand Patterns," TR-49, Department of Computer Sciences, University of Texas, Austin, Tx., May 1975.

[N2] A. S. Noetzel, "Measurements of Processing Repetition in Interactive Computation," TR-52, Department of Computer Sciences, University of Texas, Austin, Tx., October 1975.

[R1] C. V. Ramamoorthy and K. M. Chandy, "Optimization of Memory Hierarchies in Multiprogrammed Systems," J. ACM, Vol. 17, No. 3, pp. 426-445, July 1970.

[S1] C. H. Sauer, "Configuration of Computing Systems: An Approach Using Queueing Network Models," Ph.D. Thesis, University of Texas, Austin, Tx., 1975.

[S2] G. S. Shedler, "A Cyclic-Queue Model of a Paging Machine," IBM Thomas J. Watson Research Center Report RC 2814, Yorktown Heights, N.Y., 1970.

[S3] S. W. Sherman, F. Baskett, and J. C. Browne, "Trace Driven Modeling and Analysis of CPU Scheduling in a Multiprogramming Systems," Proc. of ACM Workshop on Performance Evaluation, pp. 173-199, Cambridge, Mass., April 1971

[S4] H. A. Simon and A. Ando, "Aggregation of Variables in Dynamic Systems," Econometrica, Vol. 29, pp. 111-138, 1961.

[S5] A. J. Smith, "Analysis of a Locality Model for Disk Reference Patterns," available from the author at Computer Science Division, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Calif., 1975.

[S6] J. Staudhammer, C. A. Combs, and G. Wilkinson, "Analysis of Computer Peripheral Interference," Proc. ACM National Meeting, p. 97, 1967.

[T1] J. E. Thornton, Design of a Computer: The CDC 6600, Scott, Foresman and Co., Glenview, Ill., 1970.

[T2] D. F. Towsley, "Local Balance Models of Computer Systems," Ph.D. Thesis, Dept. of Computer Sciences, University of Texas, Austin, Tx., 1975.

[W1] V. L. Wallace and R. S. Rosenberg, "Markovian Models and Numerical Analysis of Computer System Behavior," Proc. Spring Joint Computer Conference (AFIPS), Vol. 28, Spartan Books, N. Y., 1966.

[W2] T. R. Willemain, "Approximate Analysis of a Hierarchical Queueing Network," Operations Research, Vol. 22, No. 3, p. 522-545, 1974.

[W3] A. C. Williams and R. Bhandiwad, private communication.