THE ANALYSIS AND SOLUTIONS FOR GENERAL QUEUEING NETWORKS

K. M. Chandy
University of Texas at Austin

TR-44

Introduction

A queueing network is a network of arbitrarily connected queues. In analogy to a computer system we shall refer to a queue of "jobs." Each queue of jobs is serviced by one or more "devices." A job that finishes service in one queue may leave the network or join another queue. In an open network jobs enter the network from exterior sources, and leave the network and go to external sinks. In a closed network, the number of jobs in the network is a constant; jobs do not enter or leave the network. The number of jobs in a closed network will be called the level of multiprogramming.

a) Contributions of this paper

(i) Local Balance

Queueing networks have been widely used in modeling computer systems /1, 2, 3, 4, 5, 6, 7, 8/. Steady state probabilities for specific networks have been obtained by solving balance equations. The balance equations equate the rate at which a Markov process enters a state to the rate at which the process leaves that state. The number of states and the complexity of the balance equations increases rapidly with the level of multiprogramming. We shall hereafter refer to these balance equations as global balance equations. We define a new set of balance equations called local balance equations which equate the rate at which jobs enter and leave a single queue of the network. In other words, local balance is concerned with a local area: a queue isolated from the rest of the network. Local balance is a sufficient but not necessary condition for global balance. We shall show that it is possible to trivially analyze certain classes of networks using the concepts of local balance. Networks where local balance does not hold will also be discussed.

(ii) Exponential Service Distributions and Poisson Sources

Networks in which all devices have exponential service distributions and sources (if any) are Poisson, and in which queues have infinite capacity can be analyzed using local balance. Jackson's results /6/ and those of Gordon and Newell /7/ follow easily from local balance.

(iii) Geometric Service Distributions and Binomial Sources

These networks can also be analyzed using local balance.

(iv) Processor Sharing Discipline

A processor sharing discipline is the limiting case of a round-robin fixed quantum discipline, as the time quantum and switch times get arbitrarily small. We have shown that for computing steady-state probabilities, utilizations throughput etc., in a network, a queue with a processor-sharing discipline and any service distribution with a rational Laplace transform, behaves in exactly the same way as a queue with a first-come-first-served discipline and an exponential service distribution with the same mean. This is an important result because it shows, that for the computation of steady-state probabilities, the only relevant parameter in a service distribution with a rational Laplace transform is the mean value, provided a processor-sharing discipline is used. Median, mode, variance, Laplace transform, etc., are totally irrelevant. Baskett/1/ and Palacios /5/ have obtained the same result but for very specific and limited

*This research was supported in part by NSF Grant GJ 492, and GJ1084 networks. Buzen [3] has discussed the importance of studying the processor-sharing discipline.

(v) Last Come First Served Preemptive Resume Discipline

In this discipline a job that enters a queue, preempts the job (if any) which is currently getting service; the new job immediately starts getting service. This discipline is similar to many priority schemes in which the latest job has the highest priority. We have shown that a queue in a network, with this service discipline, and any service distribution which has a rational Laplace transform, behaves in the same way, as far as utilizations and throughput are concerned, as a queue with an exponential service distribution. In this case too, the only relevant parameter of the service distribution is the mean. This discipline is potentially very important because the change in throughput obtained by using this discipline is the same as from processor sharing. It also has less overhead than round-robin fixed quantum, in some systems.

(vi) Design Automation, Algebraic Symbol Manipulation and Queueing Theory

We have partially completed a program which will accept as input, a network, with the parameters of the network specified algebraically or numerically. For instance, the expected service time of a device may be specified by an algebraic term (such as 1/u) or as a number. The program will output relevant characteristics of the network, such as device utilizations, either as an algebraic expression, or as a graph. Though the types of networks which will be analyzed by the system, will be limited initially, to the classes of networks discussed above, we hope to generalize our system as we get additional theoretical results on different kinds of service disciplines. However, the system will be able to analyze most of the models studied in Baskett [1], [2], Buzen, [3], [4], Palacios [5], Jackson [6], and Gordon and Newell [7]. A description of a first pass at such a system will appear in [9].

Analysis of Closed Networks with Exponential Service Distributions

Consider a closed queueing network in which all jobs enter at an origin, pass through a network, reach a destination and are recycled back to the origin, see fig. 1. Let p_{ij} be the probability that a job which finishes service in queue i then joins queue j. Let there be M queues in all. Let BTERM(i)--Branch TERM for queue i--be the probability that a given job passes through queue i on its passage from the source to the sink. Clearly $\text{BTERM}(i) = \sum_j \text{BTERM}(j) \cdot p_{ji}$.

Let the level of multiprogramming be N. A state of this system is an M-tuple $(n_1, n_2, .., n_M)$ where n_i is the number of jobs waiting for, or getting service in queue i. Of course, $n_1 + .. + n_M = N$. The system can transit out of this state into $(n_1, .., n_i - 1, .., n_j + 1, .., n_M)$ if $p_{ij} > 0$. The system can transit into this state from $(n_1, .., n_i + 1, .., n_j - 1, n_M)$ if $p_{ij} > 0$. Global balance is concerned with all the (M) different ways a state may be entered, and the (M) different ways in which a state may be departed from.

Local balance is concerned with a single queue in the network. Consider a state $S_i = (k_1, .., k_{i-1}, k_i + 1, k_{i+1}, .., k_M)$ where $k_1 + .. + k_M + 1 = N$. We shall refer to the k_r jobs in queue r, r = 1, .., i, .., M as ceteris paribus jobs, or "stay-put" jobs. We shall call the additional job in queue i, the peripatetic job. These designations are somewhat arbitrary, but useful in understanding local balance. We shall assume that k_r, r = 1, .., M are non-negative.

Consider the rate at which the system leaves S_i DUE TO THE MOVEMENT OF A JOB OUT OF QUEUE i. Thus we are only considering transitions out of S_i into S_j where $S_j = (k_1, .., k_{j-1}, k_j + 1, k_{j+1}, .., k_M)$. Note that in these transitions the ceteris paribus jobs stay put, while the peripatetic job jumps out of queue i and into another queue. If queue i is served by a single device (with an exponential service distribution), with expected service time $1/u_i$, and if the steady-state probability of being in state S_i is $P(S_i)$, then the rate at which the system transits out of S_i due to the movement of a job out of queue i is $P(S_i) \cdot u_i$.

Consider the rate at which the process enters S_i due to the movement of a job <u>into</u> queue i. Thus we are considering transitions from S_j into S_i where, once again, the ceteris paribus jobs stay put and the peripatetic job jumps from queue j into queue i. The net rate at which the process enters S_i due to the movement of a job into queue i is $\sum_j P(S_j) \cdot u_j \cdot p_{ji}$

> Local Balance: The rate at which the process enters a given state due to the movement of a job into a given queue is equal to the rate at which the process leaves that state due to the movement of a job out of that queue.

Clearly local balance is a sufficient but not necessary condition for global balance. In our example local balance is

$$\sum_j P(S_j) \cdot u_j \cdot p_{ji} = P(S_i) \cdot u_i$$

The Ceteris Paribus Equation

Let $R(,\ldots,)$ be the functional form of the steady-state probabilities, i.e.

$$P(n_1, \ldots, n_M) = R(n_1, \ldots, n_M) \text{ if } (n_1, \ldots, n_M) \text{ is a feasible state}$$
$$= 0 \text{ if } (n_1, \ldots, n_M) \text{ is infeasible}$$

We will show that for many systems, the rate at which the process leaves state S_i due to the movement of a job out of queue i is $E \cdot \mathrm{BTERM}(i)$, where $E = R(k_1, \ldots, k_M)$. E is called the <u>ceteris paribus</u> term. E has the same functional form as the steady state probabilities but of course $(k_1, \ldots, k_M)$ is not a feasible state since $k_1 + \ldots + k_M \neq N$. The ceteris paribus term is concerned with the ceteris paribus jobs. In our example $P(S_i) \cdot u_i = E \cdot \mathrm{BTERM}(i)$. This equation will be referred to as the <u>ceteris paribus equation.</u>

Lemma: A sufficient condition for global balance is that the ceteris paribus equation hold for all queues in the network.

Proof: If the ceteris paribus equation holds then local balance holds, since the left hand side of the local balance equation is $\sum_j E \cdot \mathrm{BTERM}(j) \cdot p_{ji}$ and the right hand side is $E \cdot \mathrm{BTERM}(i)$, and we know that $\sum_j \mathrm{BTERM}(j) \cdot p_{ji} = \mathrm{BTERM}(i)$.

Let $\mathrm{PTERM}(i) = 1/u_i$, where PTERM is the processing term for device i. Then in a closed network, the functional form:

$$R(n_1, \ldots, n_M) = \mathrm{NORM} \prod_{i=1}^{M} \{\mathrm{PTERM}(i).\mathrm{BTERM}(i)\}^{n_i}$$

satisfied the ceteris paribus equation, and hence is the steady-state probability.

We will now present the functional forms for steady-state probabilities for different kinds of networks. Lack of space prevents us from deriving the functional form from the ceteris paribus equation in each case; however, it is easy to verify that the ceteris paribus equation is satisfied. We will only consider closed networks. The extension to open networks is straightforward and is found in /10/.

(i) <u>Geometric Service Distributions, Closed Network</u>

Let the probability that the service time in device i will be exactly k units long be $(1-P_i)^{k-1} P_i$. Let p_{ij} be the probability that a job which finishes service on queue i joins queue j. Let BTERM(i) be defined as before. Then $R(n_1, \ldots, n_M) = \mathrm{NORM} \prod_{i=1}^{M} \{\mathrm{PTERM}(i) \cdot \mathrm{BTERM}(i)\}^{n_i}$ where $\mathrm{PTERM}(i) = 1/P_i$.

(ii) <u>Exponential Service Distributions, Closed Network</u>

Same form as above, with $\mathrm{PTERM}(i) = 1/u_i$.

<u>Exponential Network Distributions</u>

A service time distribution may be represented as a network of interconnected exponential stages, where the time spent by a job in the ith stage has an exponential distribution with mean value $1/v_i$. A job starts at an <u>entry point</u>, traverses the network, and then finishes its service when it reaches the <u>exit point</u> (fig. 2). The network may have several branches, and the path that a job traces through the program may vary with each service. We will assume that there are no loops in the network; a job can visit a stage at most once on its path from the entry point to the exit point. Each stage in the network distribution may correspond to a task in a program, and a branch point in the network distribution may correspond to a conditional branch in the program. On the other hand, the stages and branches of the network distribution may be quite arbitrary and have nothing to do with program structure. Distributions which can be represented as a finite network of interconnected exponential stages will be called (EN) (Exponential Network) distributions. Note that the class of EN distributions is very rich and includes hypo- and hyper-exponential distributions. Any distribution with a rational Laplace transform is an EN distribution.

Processor Sharing

If a device is being processor shared, then if there are K jobs awaiting service from the device, then each of the K jobs is serviced at 1/K times the rate of a single job serviced on the device by itself. Consider a closed queueing network in which all devices are processor-shared and all service distributions are EN distributions. A state of the vector is a set of M vectors $S = (y_1, .., y_i, .., y_M)$, where $y_i = (x_{i1}, .., x_{ij}, .., x_{iQ})$ where x_{ij} is the number of jobs in stage j of queue i. Let $K_i = x_{i1} + .. + x_{iQ_i}$. Then all K_i jobs are being simultaneously processed by the device.

The local balance equation is in terms of stages rather than queues: the rate at which the process enters a given state due to the movement of a job into a given stage of a given queue is equal to the rate at which the process leaves that state due to the movement of a job out of that stage of that queue. The rate at which the process gets out of state S due to the movement of a job out of stage j of queue i is $P(S) \cdot (x_{ij} \cdot v_{ij}/ K_i)$.

Let BTERM(i) --the Branch TERM for queue i-- be defined as before, i. e., $\text{BTERM}(i) = \sum_j \text{BTERM}(j) \cdot p_{ji}$. Let ATERM(i, j) be the probability that a job gets to stage j of queue i on its traversal from the entry point to the exit point of the EN service distribution of queue i. Let p_{ijk} be the probability that a job goes to stage k of queue i after finishing service on stage j of queue i. Then $\text{ATERM}(i, j) = \sum_k \text{ATERM}(i, k) \cdot p_{ikj}$ For ease of exposition we shall label the entry point of all EN distributions with a 0, and define $\text{ATERM}(i, 0) = 1$.

The ceteris paribus equation for EN service distributions is : the rate at which the process leaves a state due to the movement of a job out of stage j of queue i is $E \cdot \text{BTERM}(i) \cdot \text{ATERM}(i,j)$. It is easy to verify that the ceteris paribus equation is a sufficient condition for local balance.

The following steady-state probabilities satisfy the ceteris paribus equation:

$$P(y_1, .., y_M) = \text{NORM} \prod_{i=1}^{M} \frac{K_i!}{x_{i1}! \cdots x_{iQ_i}!} \text{BTERM}(i)^{K_i} \prod_{j=1}^{Q_i} \left\{\text{ATERM}(i, j) \cdot \text{PTERM}(i, j)\right\}^{x_{ij}}$$

where $K_i = x_{i1} + .. + x_{iQ_i}$ is the total number of jobs sharing device i, and $\text{PTERM}(i,j) = (1/v_{ij})$ is the expected time in stage j of queue i. Note that the form of steady-state probabilities is very similar to that given for exponential service distributions, and is quite simple. Furthermore, summing over all values of x_{ij}, $j = 1, .., Q_i$ such that $\sum_{j=1}^{Q_i} x_{ij} = K_i$, we find that the probability of K_i jobs in queue i, $i = 1, .., M$ is

$$\text{NORM} \prod_{i=1}^{M} \left\{\text{PTERM}(i) \cdot \text{BTERM}(i)\right\}^{K_i} \text{ where}$$

$\text{PTERM}(i) = \sum_{j=1}^{Q_i} \{\text{ATERM}(i, j) \cdot \text{PTERM}(i, j)\}$, which is the expected service time of device i. In other words, the steady-state probability is the <u>same as in an exponential service distribution with the same mean</u>.

Last Come First Served Preemptive Resume

The states of the network are a set of M vectors $S = (y_1, .., y_M)$, where $y_i = (z_{i1}, .., z_{in_i})$ is a stack of jobs awaiting service in queue i. z_{ij} is the stage in which the jth job in the stack is in. The topmost job in the stack is currently being serviced full time. Thus if $y_i = (2, 1, 2)$ then there are three jobs in queue i; the job that is currently getting service is in stage 2 of the EN distribution, the next job in the stack is in stage 1 and the lowest job in the stack is also in stage 2. If a new job should enter queue i, before the topmost job in the stack finishes service, then the topmost job is preempted, and the new job enters the stack at the top.

The rate at which the process leaves state S due to the movement of a job out of queue i is P(S).u where 1/u is the expected time in stage z_{i1}. Steady-state probabilities which satisfy the ceteris paribus equation are

$$P(y_1,..,y_M) = \text{NORM} \prod_{i=1}^{M} \text{BTERM}(i)^{n_i} \prod_{j=1}^{n_i} \{\text{ATERM}(z_{ij}).\text{PTERM}(z_{ij})\}$$

Once again, summing over all possible distinct sequences $(z_{i1},.., z_{in_i})$, we find that the steady-state probability of n_1 jobs in queue 1, and n_2 jobs in queue 2, and .., n_M jobs in queue M is $\text{NORM} \prod_{i=1}^{M} \{\text{BTERM}(i).\text{PTERM}(i)\}^{n_i}$, where PTERM(i) is the mean service time of device i. Once again, the steady-state probabilities are the same as for an exponential service distribution, with mean PTERM(i).

Processor-sharing and last come first served preemptive resume disciplines force nonexponential service distributions to behave like exponential distributions. This may explain why analysts have found models of round robin fixed quantum disciplines to be somewhat insensitive to service distributions.

Networks Without Local Balance

Unfortunately many networks do not satisfy local balance. For instance, if a device has a hyperexponential service distribution, and a first come first served service discipline, then it is easy to show that local balance does not hold. These networks must be analyzed using global balance. The analysis can be automated, but is extremely slow, and sometimes intractable. Extensions of this work to other types of networks is found in /10/.

Applications

Many of the models of multiprogrammed computer systems which have appeared in the literature are specific cases of the networks analyzed here. The concept of local balance and ceteris paribus has pedagogic significance: it has been used in an undergraduate first course in systems modeling. It was also used in computing the improvement in throughput obtained by parallel processing in a multiprocessing environment /11/.

REFERENCES

1. Baskett, F. The dependence of computer system queues upon processing time distribution and central processor scheduling, Proc. 3rd. Sym on Op. Sys. Principles, Stanford U., Oct.'71,
2. _____., Mathematical models of multiprogrammed computer systems. TSN-17, Comp. Center, U. Texas.
3. Buzen, J. Analysis of system bottlenecks using a queueing network model. ACM-SIGOPS Workshop on System Performance Evaluation, N.Y., Apr. '71.
4. _____,, Queueing network models of multiprogramming. Ph.D. dissertation, Harvard U. August '71.
5. Palacios, F. "Processor sharing models with queues." Master's thesis, U. Texas, '72.
6. Jackson, J.R. Jobshop-like queueing systems. Manag. Sci., 10 Oct. '63, 131-142.
7. Gordon, W.J., & G.F. Newell, Closed queueing systems with exponential servers. Oper. Res. 15, Apr. '67, 254-265.
8. Arora, S.R. & A. Gallo, The optimal organization of multi-programmed multi-level memory. ACM-SIGOPS Workshop on Sys. Perf. Eval. N.Y. Apr.'71.
9. Chandy,K.M. et al. Design automation and queueing networks. to be presented at the 9th Annual Design Automation Workshop, Dallas, June, '72.
10. Chandy, K.M., Exponential and processor sharing queueing network models of computer systems. Submitted to IEEE-Trans. Comp. January '72.
11. Chandy, K.M. & J. Hogarth. The improvement in throughput obtained by parallel processing. Submitted to Parallel Proc. Workshop. Seattle'72.

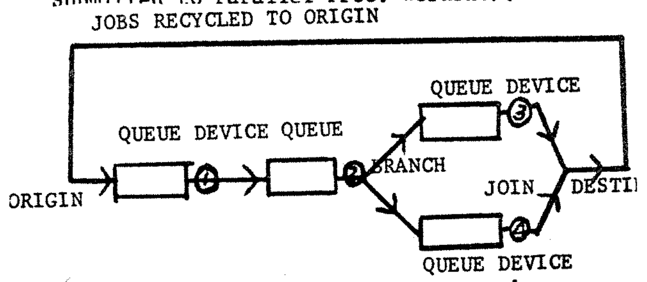
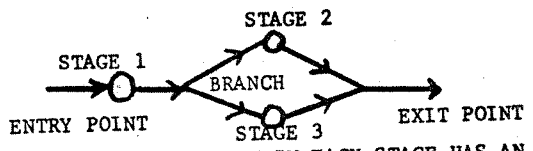


Fig 1: A Closed Queueimg Network



THE TIME SPENT IN EACH STAGE HAS AN EXPONENTIAL SERVICE DISTRIBUTION.

Fig 2: An Exponential Service Dist.