QUEUING ANALYSIS

William Stallings

This document available at WilliamStallings.com/StudentSupport.html

Queuing analysis is one of the most important tools for those involved with computer and network analysis. It can be used to provide approximate answers to a host of questions, such as:

- What happens to file retrieval time when disk I/O utilization goes up?
- Does response time change if both processor speed and the number of users on the system are doubled?
- How many lines should a time-sharing system have on a dial-in rotary?
- How many terminals are needed in an on line inquiry center, and how much idle time will the operators have?

The number of questions that can be addressed with a queuing analysis is endless and touches on virtually every area in computer science. The ability to make such an analysis is an essential tool for those involved in this field.

Although the theory of queuing is mathematically complex, the application of queuing theory to the analysis of performance is, in many cases, remarkably straightforward. A knowledge of elementary statistical concepts (means and standard deviations) and a basic understanding of the applicability of queuing theory is all that is required. Armed with these, the analyst can often make a queuing analysis on the back of an envelope using readily available queuing tables, or with the use of simple computer programs that occupy only a few lines of code.

The purpose of this paper is to provide a practical guide to queuing analysis. A subset, although a very important subset, of the subject is addressed. In the final section, pointers to additional references are provided. An annex to this paper reviews some elementary concepts in probability and statistics.

WHY QUEUING ANALYSIS?

There are many cases when it is important to be able to project the effect of some change in a design: either the load on a system is expected to increase or a design change is contemplated. For example, an organization supports a number of terminals, personal computers, and workstations on a 100-Mbps local area network (LAN). An additional department in the building is to be cut over onto the network. Can the existing LAN handle the increased workload, or would it be better to provide a second LAN with a bridge between the two? There are other cases in which no facility exists but, on the basis of expected demand, a system design needs to be created. For example, a department intends to equip all of its personnel with a personal computer and to configure these into a LAN with a file server. Based on experience elsewhere in the company, the load generated by each PC can be estimated.

The concern is system performance. In an interactive or real-time application, often the parameter of concern is response time. In other cases, throughput is the principal issue. In any case, projections of performance are to be made on the basis of existing load information or on the basis of estimated load for a new environment. A number of approaches are possible:

1. Do an after-the-fact analysis based on actual values.
2. Make a simple projection by scaling up from existing experience to the expected future environment.
3. Develop an analytic model based on queuing theory.
4. Program and run a simulation model.

Option 1 is no option at all: we will wait and see what happens. This leads to unhappy users and to unwise purchases. Option 2 sounds more promising. The analyst may take the position that it is impossible to project future demand with any degree of certainty. Therefore, it is pointless to attempt some exact modeling procedure. Rather, a rough-and-ready projection will provide ballpark estimates. The problem with this approach is that the behavior of most systems

under a changing load is not what one would intuitively expect. If there is an environment in which there is a shared facility (e.g., a network, a transmission line, a time-sharing system), then the performance of that system typically responds in an exponential way to increases in demand.

Figure 1 is a typical example. The upper line shows what happens to user response time on a shared facility as the load on that facility increases. The load is expressed as a fraction of capacity. Thus, if we are dealing with a input from a disk that is capable of transferring 1000 blocks per second, then a load of 0.5 represents a transfer of 500 blocks per second, and the response time is the amount of time it takes to retransmit any incoming block. The lower line is a simple projection[1] based on a knowledge of the behavior of the system up to a load of 0.5. Note that while things appear rosy when the simple projection is made, performance on the system will in fact collapse beyond a load of about 0.8 to 0.9.

Thus, a more exact prediction tool is needed. Option 3 is to make use of an analytic model, which is one that can be expressed as a set of equations that can be solved to yield the desired parameters (response time, throughput, etc.). For computer, operating-system, and networking problems, and indeed for many practical real-world problems, analytic models based on queuing theory provide a reasonably good fit to reality. The disadvantage of queuing theory is that a number of simplifying assumptions must be made to derive equations for the parameters of interest.

The final approach is a simulation model. Here, given a sufficiently powerful and flexible simulation programming language, the analyst can model reality in great detail and avoid making many of the assumptions required of queuing theory. However, in most cases, a simulation model is not needed or at least is not advisable as a first step in the analysis. For one thing, both existing measurements and projections of future load carry with them a certain margin of error. Thus, no matter how good the simulation model, the value of the results are limited by the quality of the input. For another, despite the many assumptions required of queuing theory, the results that are produced often come quite close to those that would be produced by a more careful simulation analysis. Furthermore, a queuing analysis can literally be accomplished in a matter of minutes for a well-defined problem, whereas simulation exercises can take days, weeks, or longer to program and run.

Accordingly, it behooves the analyst to master the basics of queuing analysis.

QUEUING MODELS

The Single-Server Queue

The simplest queuing system is depicted in Figure 2. The central element of the system is a server, which provides some service to items. Items from some population of items arrive at the system to be served. If the server is idle, an item is served immediately. Otherwise, an arriving item joins a waiting line[2]. When the server has completed serving an item, the item departs. If there are items waiting in the queue, one is immediately dispatched to the server. The server in this model can represent anything that performs some function or service for a collection of items. Examples: a processor provides service to processes; a transmission line provides a transmission service to packets or frames of data; an I/O device provides a read or write service for I/O requests.

---

[1] In fact, the lower line is based on fitting a third-order polynomial to the data available up to a load of 0.5.

[2] The waiting line is referred to as a queue in some treatments in the literature; it is also common to refer to the entire system as a queue. Unless otherwise noted, we use the term *queue* to mean waiting line.

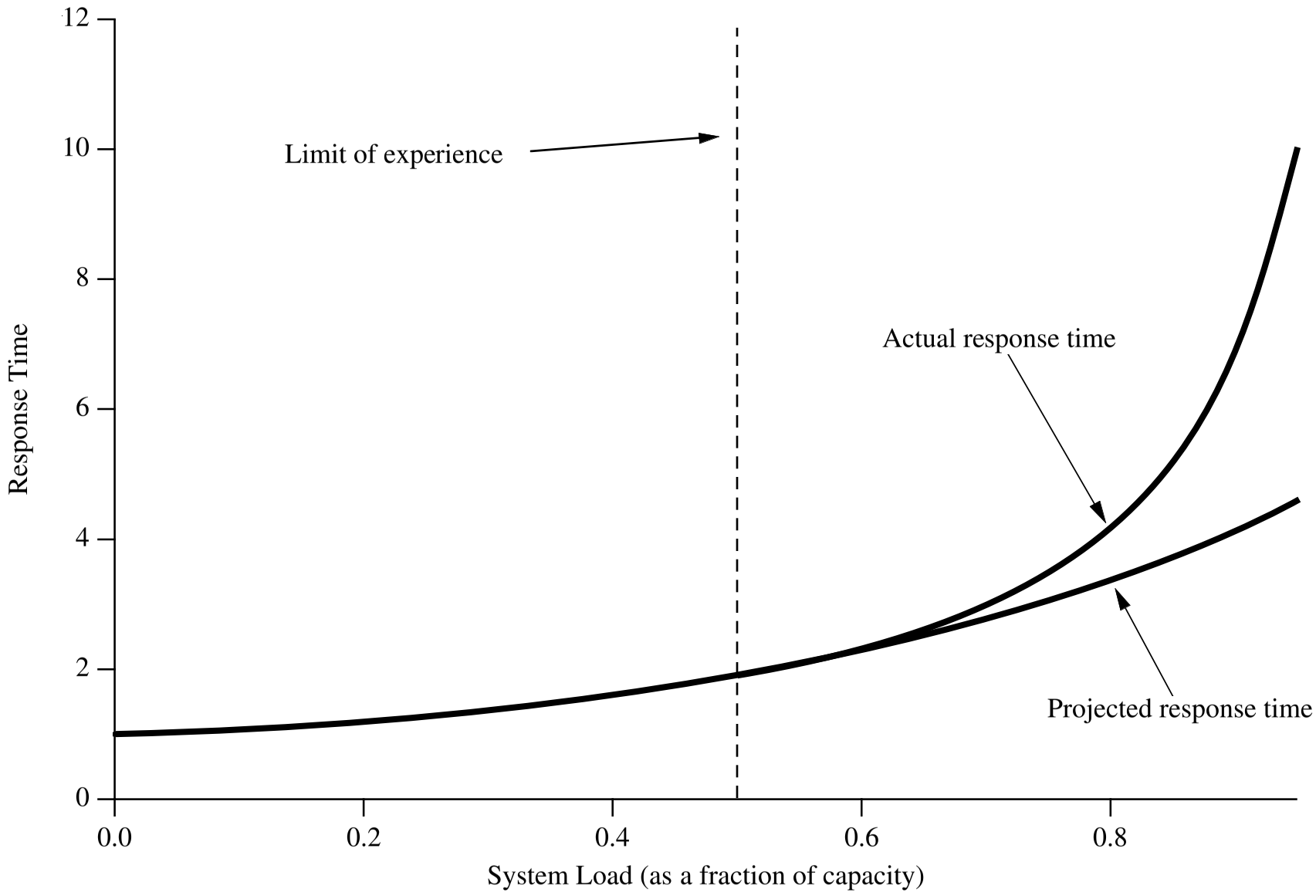


Figure 1 Projected Versus Actual Response Time

Queue Parameters

Figure 2 also illustrates some important parameters associated with a queuing model. Items arrive at the facility at some average rate (items arriving per second) λ. At any given time, a certain number of items will be waiting in the queue (zero or more); the average number waiting is *w*, and the mean time that an item must wait is T_w. T_w is averaged over all incoming items, including those that do not wait at all. The server handles incoming items with an average service time T_S; this is the time interval between the dispatching of an item to the server and the departure of that item from the server. Utilization, ρ, is the fraction of time that the server is busy, measured over some interval of time. Finally, two parameters apply to the system as a whole. The average number of items resident in the system, including the item being served (if any) and the items waiting (if any), is *r*; and the average time that an item spends in the system, waiting and being served, is T_r; we refer to this as the mean residence time.[3]

If we assume that the capacity of the queue is infinite, then no items are ever lost from the system; they are just delayed until they can be served. Under these circumstances, the departure rate equals the arrival rate. As the arrival rate, which is the rate of traffic passing through the system, increases, the utilization increases and with it, congestion. The queue becomes longer, increasing waiting time. At $\rho = 1$, the server becomes saturated, working 100% of the time. Thus, the theoretical maximum input rate that can be handled by the system is:

$$\lambda_{max} = \frac{1}{T_s}$$

However, queues become very large near system saturation, growing without bound when $\rho = 1$. Practical considerations, such as response time requirements or buffer sizes, usually limit the input rate for a single server to 70-90% of the theoretical maximum.

To proceed, we need to make some assumption about this model:

- **Item population:** Typically, we assume an infinite population. This means that the arrival rate is not altered by the loss of population. If the population is finite, then the population available for arrival is reduced by the number of items currently in the system; this would typically reduce the arrival rate proportionally.
- **Queue size:** Typically, we assume an infinite queue size. Thus, the waiting line can grow without bound. With a finite queue, it is possible for items to be lost from the system. In practice, any queue is finite. In many cases, this will make no substantive difference to the analysis. We address this issue briefly, below.
- **Dispatching discipline:** When the server becomes free, and if there is more than one item waiting, a decision must be made as to which item to dispatch next. The simplest approach is first-in, first-out; this discipline is what is normally implied when the term *queue* is used. Another possibility is last-in, first-out. One that you might encounter in practice is a dispatching discipline based on service time. For example, a packet-switching node may choose to dispatch packets on the basis of shortest first (to generate the most outgoing packets) or longest first (to minimize processing time relative to transmission time). Unfortunately, a discipline based on service time is very difficult to model analytically.

---

[3] Again, in some of the literature, this is referred to as the mean queuing time, while other treatments use mean queuing time to mean the average time spent waiting in the queue (before being served).

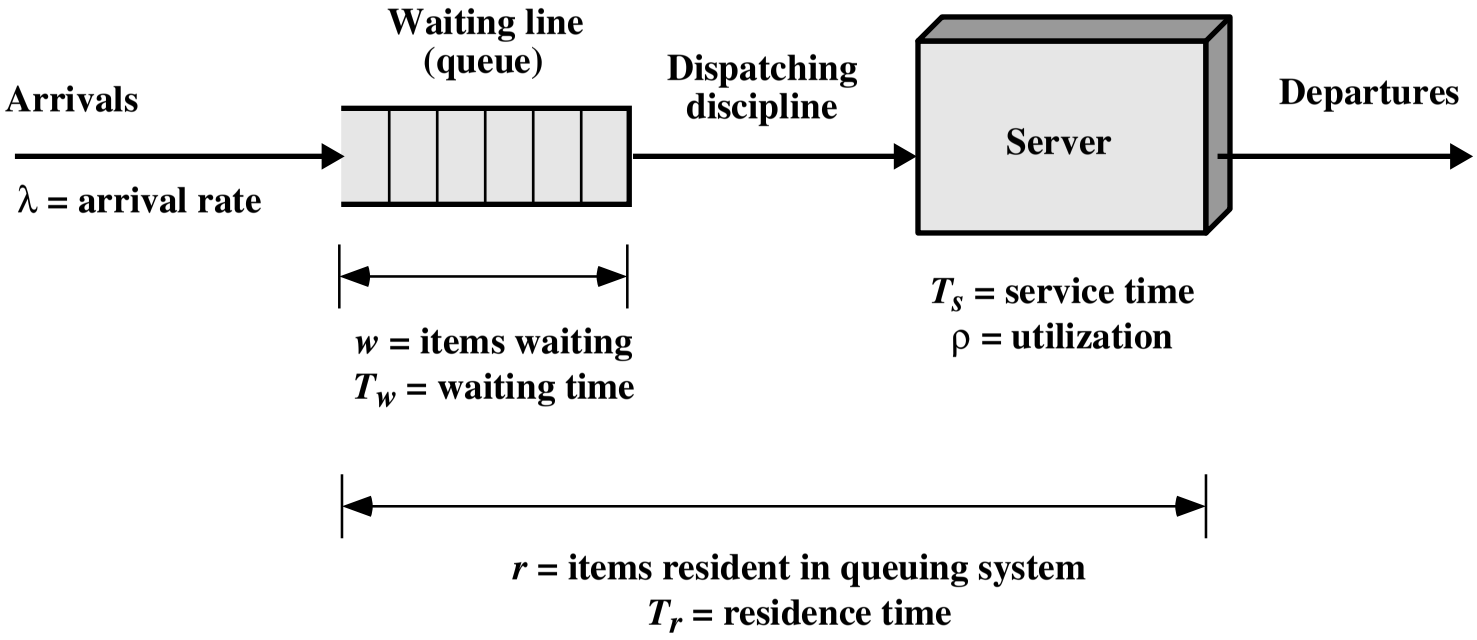


Figure 2 Queuing System Structure and Parameters for Single-Server Queue

Table 1 summarizes the notation that is used in Figure 2 and introduces some other parameters that are useful. In particular, we are often interested in the variability of various parameters, and this is neatly captured in the standard deviation.

The Multiserver Queue

Figure 3a shows a generalization of the simple model we have been discussing for multiple servers, all sharing a common queue. If an item arrives and at least one server is available, then the item is immediately dispatched to that server. It is assumed that all servers are identical; thus, if more than one server is available, it makes no difference which server is chosen for the item. If all servers are busy, a queue begins to form. As soon as one server becomes free, an item is dispatched from the queue using the dispatching discipline in force.

With the exception of utilization, all of the parameters illustrated in Figure 2 carry over to the multiserver case with the same interpretation. If we have N identical servers, then ρ is the utilization of each server, and we can consider $N\rho$ to be the utilization of the entire system; this latter term is often referred to as the traffic intensity, u. Thus, the theoretical maximum utilization is $N \times 100\%$, and the theoretical maximum input rate is:

$$\lambda_{\max} = \frac{N}{T_s}$$

The key characteristics typically chosen for the multiserver queue correspond to those for the single-server queue. That is, we assume an infinite population and an infinite queue size, with a single infinite queue shared among all servers. Unless otherwise stated, the dispatching discipline is FIFO. For the multiserver case, if all servers are assumed identical, the selection of a particular server for a waiting item has no effect on service time.

By way of contrast, Figure 3b shows the structure of multiple single-server queues. As we shall see, this apparently minor change in structure has a significant impact on performance.

Basic Queuing Relationships

To proceed much further, we are going to have to make some simplifying assumptions. These assumptions risk making the models less valid for various real-world situations. Fortunately, in most cases, the results will be sufficiently accurate for planning and design purposes.

There are, however, some relationships that are true in the general case, and these are illustrated in Table 2. By themselves, these relationships are not particularly helpful.

Assumptions

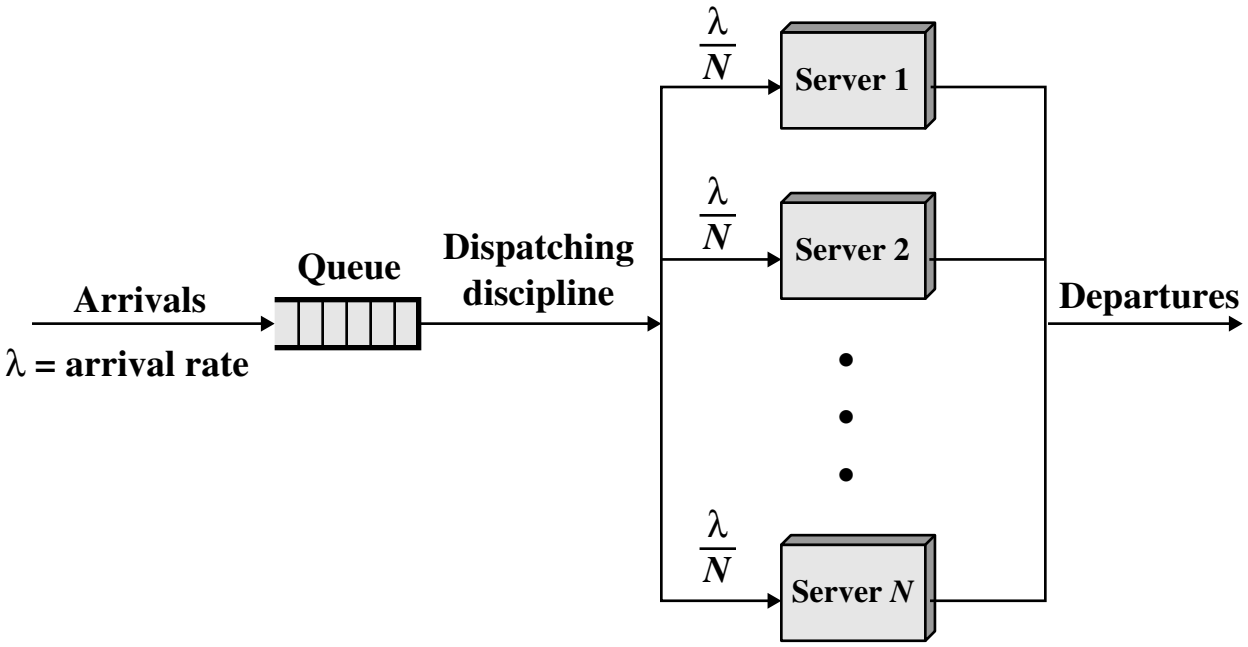
The fundamental task of a queuing analysis is as follows: Given the following information as input:

- Arrival rate
- Service time

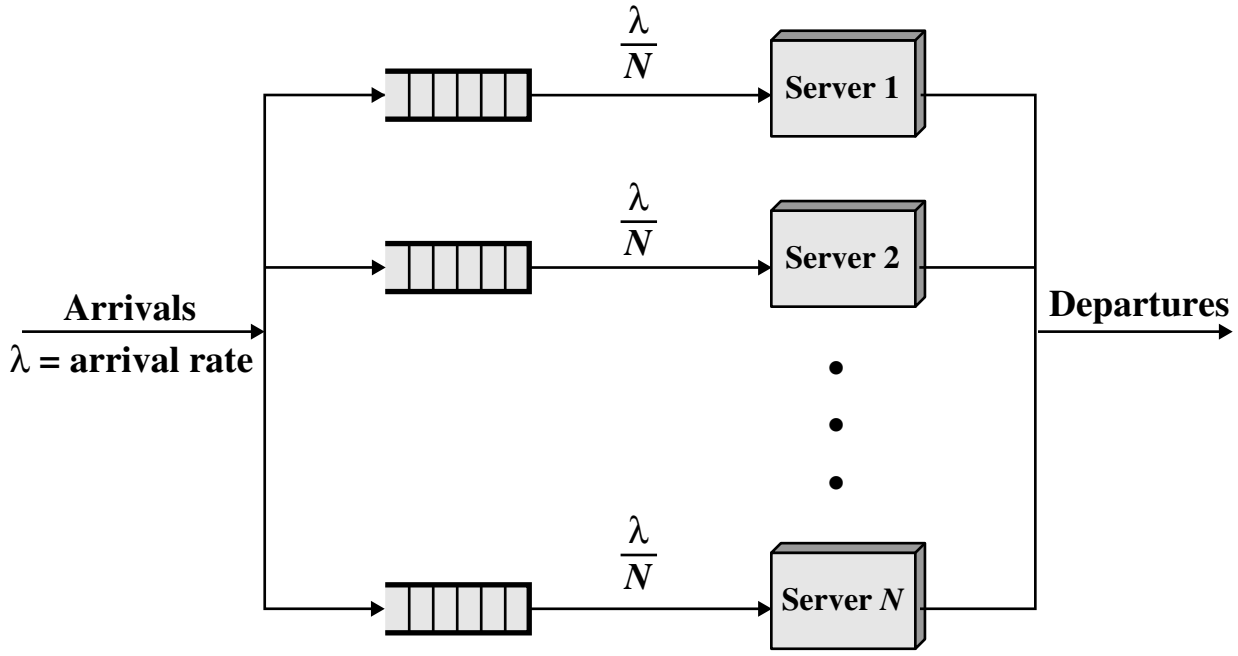
Provide as output information concerning:

- Items waiting
- Waiting time
- Items in residence
- Residence time.

What specifically would we like to know about these outputs? Certainly we would like to know their average values (w, T_w, r, T_r). In addition, it would be useful to know something about



(a) Multiserver queue



(b) Multiple Single-server queues

Figure 3 Multiserver Versus Multiple Single-Server Queues

Table 1 Notation for Queuing Systems

λ	=	arrival rate; mean number of arrivals per second
T_s	=	mean service time for each arrival; amount of time being served, not counting time waiting in the queue
σ_{Ts}	=	standard deviation of service time
ρ	=	utilization; fraction of time facility (server or servers) is busy
u	=	traffic intensity
r	=	mean number of items in system, waiting and being served (residence time)
R	=	number of items in system, waiting and being served
T_r	=	mean time an item spends in system (residence time)
T_R	=	time an item spends in system (residence time)
σ_r	=	standard deviation of r
σ_{Tr}	=	standard deviation of T_r
w	=	mean number of items waiting to be served
σ_w	=	standard deviation of w
T_w	=	mean waiting time (including items that have to wait and items with waiting time = 0)
T_d	=	mean waiting time for items that have to wait
N	=	number of servers
$m_X(y)$	=	the yth percentile; that value of y below which x occurs y percent of the time

Table 2 Some Basic Queuing Relationships

General	Single Server	Multiserver
$r = \lambda T_r$ Little's formula	$\rho = \lambda T_s$	$\rho = \frac{\lambda T_s}{N}$
$w = \lambda T_w$ Little's formula	$r = w + \rho$	$u = \lambda T_s = \rho N$
$T_r = T_w + T_s$		$r = w + N\rho$

their variability. Thus, the standard deviation of each would be useful $\left(\sigma_w, \sigma_{T_w}, \sigma_r, \sigma_{T_r}\right)$. Other measures may also be useful. For example, to design a buffer associated with a bridge or multiplexer, it might be useful to know for what buffer size the probability of overflow is less than 0.001. That is, what is the value of *M* such that Pr[items waiting < *M*] = 0.999?

To answer such questions in general requires complete knowledge of the probability distribution of the arrival rate and service time. Furthermore, even with that knowledge, the resulting formulas are exceedingly complex. Thus, to make the problem tractable, we need to make some simplifying assumptions.

The most important of these assumptions is that the arrival rate obeys the Poisson distribution, which is equivalent to saying that the interarrival times are exponential, which is equivalent to saying that the arrivals occur randomly and independent of one another. This assumption is almost invariably made. Without it, most queuing analysis is impractical. With this assumption, it turns out that many useful results can be obtained if only the mean and standard deviation of the arrival rate and service time are known. Matters can be made even simpler and more detailed results can be obtained if it is assumed that the service time is exponential or constant.

A convenient notation has been developed for summarizing the principal assumptions that are made in developing a queuing model. The notation is X/Y/N, where X refers to the distribution of the interarrival times, Y refers to the distribution of service times, and N refers to the number of servers. The most common distributions are denoted as follows:

G = general independent arrivals or service times
M = negative exponential distribution
D = deterministic arrivals or fixed length service.

Thus, M/M/1 refers to a single-server queuing model with Poisson arrivals and exponential service times.

SINGLE-SERVER QUEUES

Table 3a provides some equations for single server queues that follow the M/G/1 model. That is, the arrival rate is Poisson and the service time is general. Making use of a scaling factor, *A*, the equations for some of the key output variables are straightforward. Note that the key factor in the scaling parameter is the ratio of the standard deviation of service time to the mean. No other information about the service time is needed. Two special cases are of some interest. When the standard deviation is equal to the mean, the service time distribution is exponential (M/M/1). This is the simplest case and the easiest one for calculating results. Table 3b shows the simplified versions of equations for the standard deviation of *r* and T_r, plus some other parameters of interest. The other interesting case is a standard deviation of service time equal to zero, that is, a constant service time (M/D/1). The corresponding equations are shown in Table 3c.

Figures 4 and 5 plot values of average queue size and residence time versus utilization for three values of σ_{T_S}/T_S. This latter quantity is known as the **coefficient of variation**, and gives a normalized measure of variability. Note that the poorest performance is exhibited by the exponential service time, and the best by a constant service time. Usually, one can consider the exponential service time to be a worst case. An analysis based on this assumption will give conservative results. This is nice, because tables are available for the M/M/1 case and values can be looked up quickly.

What value of σ_{T_S}/T_S is one likely to encounter? We can consider four regions:

- **Zero:** This is the rare case of constant service time. For example, if all transmitted messages are of the same length, they would fit this category.

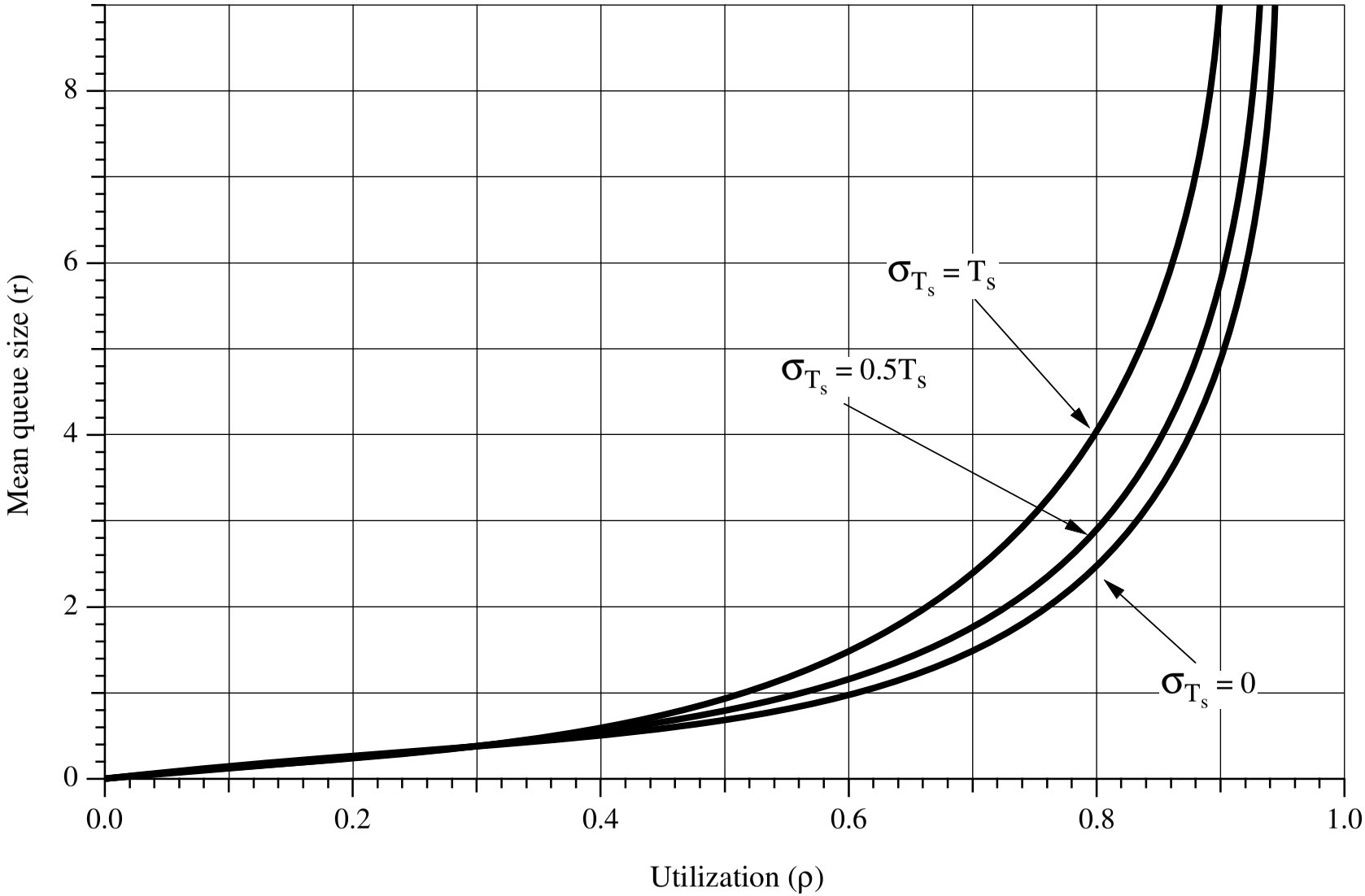


Figure 4 Mean Queue Size for Single-Server Queue

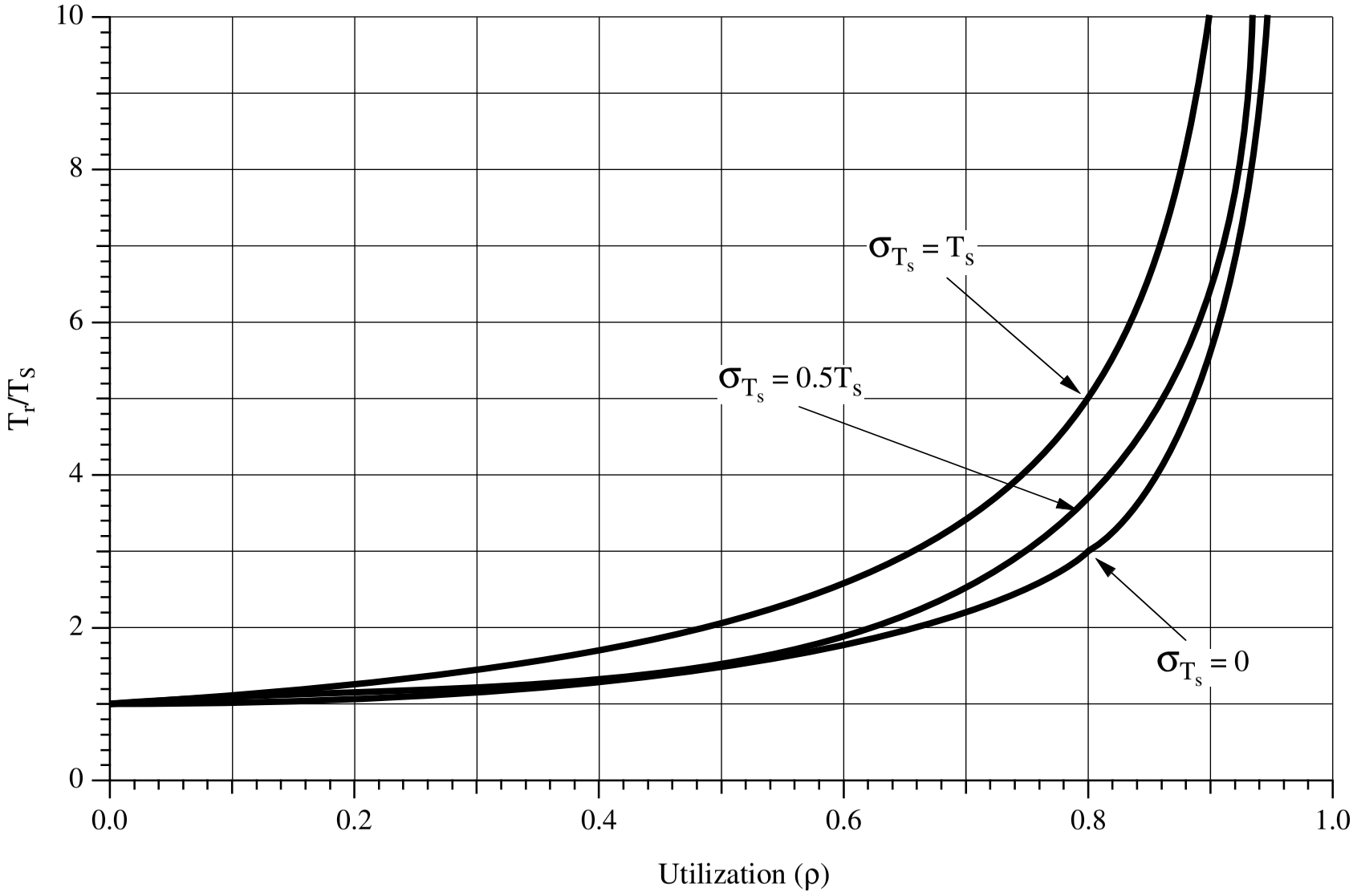


Figure 5 Mean Residence Time for Single-Server Queue

Table 3 Formulas for Single-Server Queues

Assumptions:

1. Poisson arrival rate.
2. Dispatching discipline does not give preference to items based on service times.
3. Formulas for standard deviation assume first-in, first-out dispatching.
4. No items are discarded from the queue.

(a) General Service Times (M/G/1)

$$A = \frac{1}{2}\left[1 + \left(\frac{\sigma_{T_s}}{T_s}\right)^2\right]$$

$$r = \rho + \frac{\rho^2 A}{1-\rho}$$

$$w = \frac{\rho^2 A}{1-\rho}$$

$$T_r = T_s + \frac{\rho T_s A}{1-\rho}$$

$$T_w = \frac{\rho T_s A}{1-\rho}$$

(b) Exponential Service Times (M/M/1)

$$r = \frac{\rho}{1-\rho} \qquad w = \frac{\rho^2}{1-\rho}$$

$$T_r = \frac{T_s}{1-\rho} \qquad T_w = \frac{\rho T_s}{1-\rho}$$

$$\sigma_r = \frac{\sqrt{\rho}}{1-\rho} \qquad \sigma_{T_r} = \frac{T_s}{1-\rho}$$

$$\Pr[R = N] = (1-\rho)\rho^N$$

$$\Pr[R \le N] = \sum_{i=0}^{N} (1-\rho)\rho^i$$

$$\Pr[T_R \le T] = 1 - e^{-(1-\rho)t/T_s}$$

$$m_{T_r}(y) = T_r \times \ln\left(\frac{100}{100-y}\right)$$

$$m_{T_w}(y) = \frac{T_w}{\rho} \times \ln\left(\frac{100\rho}{100-y}\right)$$

(c) Constant Service Times (M/D/1)

$$r = \frac{\rho^2}{2(1-\rho)} + \rho$$

$$w = \frac{\rho^2}{2(1-\rho)}$$

$$T_r = \frac{T_s(2-\rho)}{2(1-\rho)}$$

$$T_w = \frac{\rho T_s}{2(1-\rho)}$$

$$\sigma_r = \frac{1}{1-\rho}\sqrt{\rho - \frac{3\rho^2}{2} + \frac{5\rho^3}{6} - \frac{\rho^4}{12}}$$

$$\sigma_{T_r} = \frac{T_s}{1-\rho}\sqrt{\frac{\rho}{3} - \frac{\rho^2}{12}}$$

- **Ratio less than 1:** Because this ratio is better than the exponential case, using M/M/1 tables will give queue sizes and times that are slightly larger than they should be. Using the M/M/1 model would give answers on the safe side. An example of this category might be a data entry application for a particular form.
- **Ratio close to 1:** This is a common occurrence and corresponds to exponential service time. That is, service times are essentially random. Consider message lengths to a computer terminal: a full screen might be 1920 characters, with message sizes varying over the full range. Airline reservations, file lookups on inquires, shared LAN, and packet-switching networks are examples of systems that often fit this category.
- **Ratio greater then 1:** If you observe this, you need to use the M/G/1 model and not rely on the M/M/1 model. A common occurrence of this is a bimodal distribution, with a wide spread between the peaks. An example is a system that experiences many short messages, many long messages, and few in between.

The same consideration applies to the arrival rate. For a Poisson arrival rate, the interarrival times are exponential, and the ratio of standard deviation to mean is 1. If the observed ratio is much less than 1, then arrivals tend to be evenly spaced (not much variability), and the Poisson assumption will overestimate queue sizes and delays. On the other hand, if the ratio is greater than 1, then arrivals tend to cluster and congestion becomes more acute.

MULTISERVER QUEUES

Table 4 lists formulas for some key parameters for the multiserver case. Note the restrictiveness of the assumptions. Useful congestion statistics for this model have been obtained only for the case of M/M/N, where the exponential service times are identical for the *N* servers.

NETWORKS OF QUEUES

In a distributed environment, isolated queues are unfortunately not the only problem presented to the analyst. Often, the problem to be analyzed consists of several interconnected queues. Figure 6 illustrates this situation, using nodes to represent queues and the interconnecting lines to represent traffic flow.

Two elements of such a network complicate the methods shown so far:

- The partitioning and merging of traffic, as illustrated by nodes 1 and 5 respectively in the figure.
- The existence of queues in tandem, or series, as illustrated by nodes 3 and 4.

No exact method has been developed for analyzing general queuing problems that have the above elements. However, if the traffic flow is Poisson and the service times are exponential, an exact and simple solution exists. In this section, we first examine the two elements listed above, and then present the approach to queuing analysis.

Partitioning and Merging of Traffic Streams

Suppose that traffic arrives at a queue with a mean arrival rate of λ, and that there are two paths, A and B, by which an item may depart (Figure 7a). When an item is serviced and departs the queue, it does so via path A with probability P and via path B with probability $(1 - P)$. In general, the traffic distribution of streams A and B will differ from the incoming distribution. However, if the incoming distribution is Poisson, then the two departing traffic flows also have Poisson distributions, with mean rates of $P\lambda$ and $(1 - P)\lambda$.

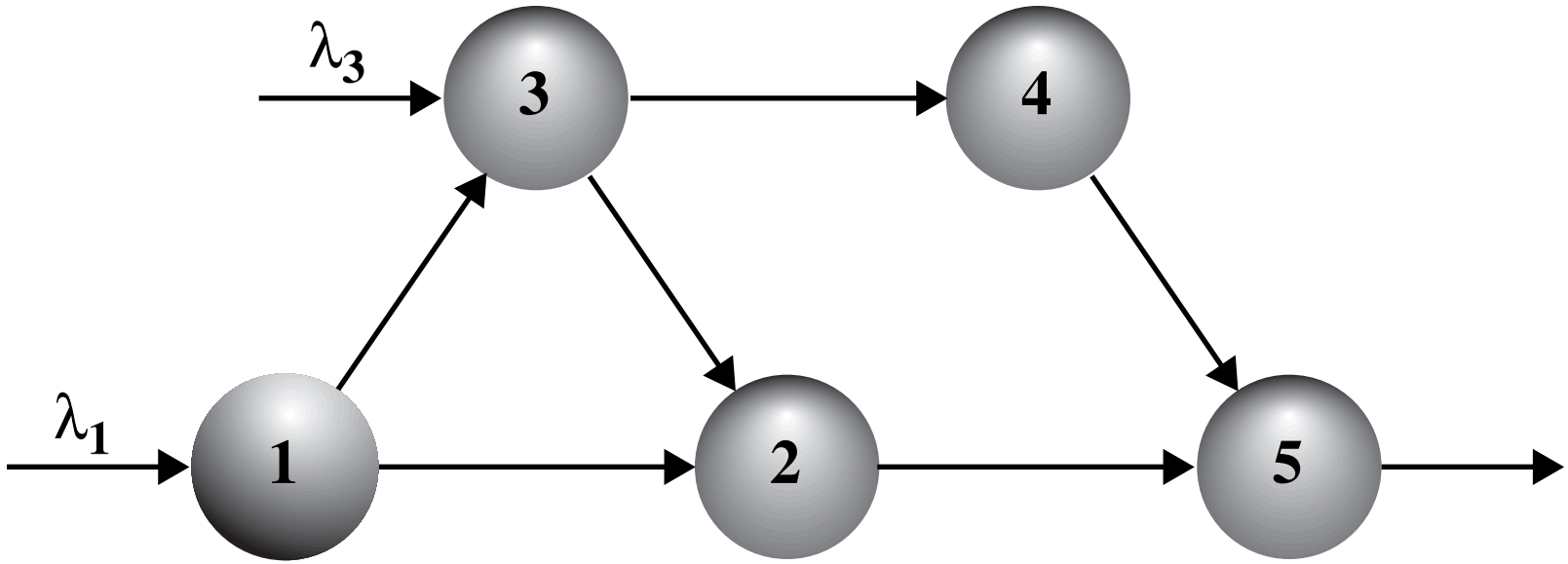
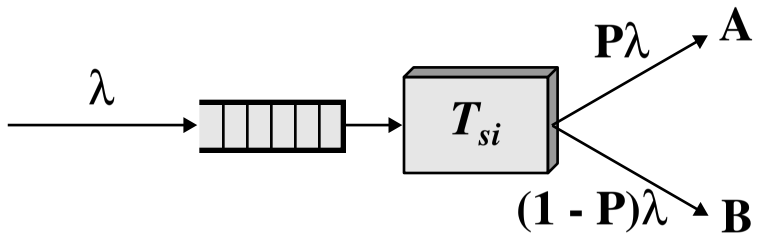
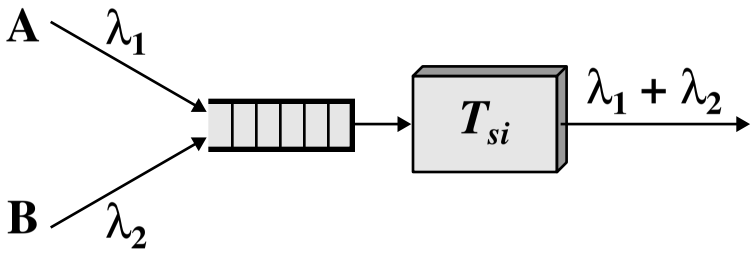


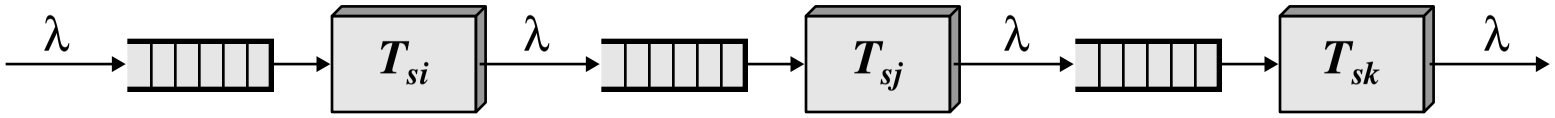
Figure 6 Example of a Network of Queues



(a) Traffic partitioning



(b) Traffic merging



(c) Simple tandem queue

Figure 7 Elements of Queuing Networks

Table 4 Formulas for Multiserver Queues (M/M/N)

Assumptions:	1. Poisson arrival rate.
	2. Exponential service times
	3. All servers equally loaded
	4. All servers have same mean service time
	5. First-in, first-out dispatching
	6. No items are discarded from the queue

$$K = \frac{\sum_{I=0}^{N-1} \frac{(N\rho)^I}{I!}}{\sum_{I=0}^{N} \frac{(N\rho)^I}{I!}} \qquad \text{Poisson ratio function}$$

$$\text{Erlang-C function} = \text{Probability that all servers are busy} = C = \frac{1-K}{1-\rho K}$$

$$r = C\frac{\rho}{1-\rho} + N\rho \qquad w = C\frac{\rho}{1-\rho}$$

$$T_r = \frac{C}{N}\frac{T_s}{1-\rho} + T_s \qquad T_w = \frac{C}{N}\frac{T_s}{1-\rho}$$

$$\sigma_{T_r} = \frac{T_s}{N(1-\rho)}\sqrt{C(2-C) + N^2(1-\rho)^2}$$

$$\sigma_w = \frac{1}{1-\rho}\sqrt{C\rho(1+\rho-C\rho)}$$

$$\Pr[T_W > t] = Ce^{-N(1-\rho)t/T_s}$$

$$m_{T_w}(y) = \frac{T_s}{N(1-\rho)}\ln\left(\frac{100C}{100-y}\right)$$

$$T_d = \frac{T_s}{N(1-\rho)}$$

A similar situation exists for traffic merging. If two Poisson streams with mean rates of λ_1 and λ_2 are merged, the resulting stream is Poisson with a mean rate of $\lambda_1 + \lambda_2$.

Both of these results generalize to more than two departing streams for partitioning and more then two arriving streams for merging.

Queues in Tandem

Figure 7c is an example of a set of single-server queues in tandem: The input for each queue except the first is the output of the previous queue. Assume that the input to the first queue is Poisson. Then, if the service time of each queue is exponential and the waiting lines are infinite, the output of each queue is a Poisson stream statistically identical to the input. When this stream is fed into the next queue, the delays at the second queue are the same as if the original traffic had bypassed the first queue and fed directly into the second queue. Thus the queues are independent and may be analyzed one at a time. Therefore, the mean total delay for the tandem system is equal to the sum of the mean delays at each stage.

This result can be extended to the case where some or all of the nodes in tandem are multiserver queues.

Jackson's Theorem

Jackson's theorem can be used to analyze a network of queues. The theorem is based on three assumptions:

1. The queuing network consists of *m* nodes, each of which provides an independent exponential service.
2. Items arriving from outside the system to any one of the nodes arrive with a Poisson rate.
3. Once served at a node, an item goes (immediately) to one of the other nodes with a fixed probability, or out of the system.

Jackson's theorem states that in such a network of queues, each node is an independent queuing system, with a Poisson input determined by the principles of partitioning, merging, and tandem queuing. Thus each node may be analyzed separately from the others using the M/M/1 or M/M/N model, and the results may be combined by ordinary statistical methods. Mean delays at each node may be added to derive system delays, but nothing can be said about the higher moments of system delays (e.g., standard deviation).

Jackson's theorem appears attractive for application to packet-switching networks. One can model the packet-switching network as a network of queues. Each packet represents an individual item. We assume that each packet is transmitted separately and, at each packet-switching node in the path from source to destination, the packet is queued for transmission on the next length. The service at a queue is the actual transmission of the packet and is proportional to the length of the packet.

The flaw in this approach is that a condition of the theorem is violated: namely, it is not the case that the service distributions are independent. Because the length of a packet is the same at each transmission link, the arrival process to each queue is correlated to the service process. However, Kleinrock [KLEI76] has demonstrated that, because of the averaging effect of merging and partitioning, assuming independent service times provides a good approximation.

Application to a Packet-Switching Network

Consider a packet-switching network, consisting of nodes interconnected by transmission links, with each node acting as the interface for zero or more attached systems, each of which functions as a source and destination of traffic. The external workload that is offered to the network can be characterized as:

$$\gamma = \sum_{j=1}^{N} \sum_{k=1}^{N} \gamma_{jk}$$

where

γ = total workload in packets per second
γ_{jk} = workload between source j and destination k
N = total number of sources and destinations

Because a packet may traverse more than one link between source and destination, the total internal workload will be higher than the offered load:

$$\lambda = \sum_{i=1}^{L} \lambda_i$$

where

λ = total load on all of the links in the network
λ_i = load on link i
L = total number of links

The internal load will depend on the actual path taken by packets through the network. We will assume that a routing algorithm is given such that the load on the individual links, λ_i, can be determined from the offered load, γ_{jk}. For any particular routing assignment, we can determine the average number of links that a packet will traverse from these workload parameters. Some thought should convince you that the average length for all paths is given by:

$$E[\text{number of links in a path}] = \frac{\lambda}{\gamma}$$

Our objective is to determine the average delay, T, experienced by a packet through the network. For this purpose, it is useful to apply Little's formula (Table 2). For each link in the network, the average number of items waiting and being served for that link is given by:

$$r_i = \lambda_i T_{ri}$$

where T_{ri} is the yet-to-be-determined residence time at each queue. Suppose that we sum these quantities. That would give us the average total number of packets waiting in all of the queues of the network. It turns out that Little's formula works in the aggregate as well[4]. Thus, the number of packets waiting and being served in the network can be expressed as γT. Combining the two:

$$T = \frac{1}{\gamma} \sum_{i=1}^{L} \lambda_i T_{ri}$$

To determine the value of T, we need to determine the values of the individual delays, T_{ri}. Because we are assuming that each queue can be treated as an independent M/M/1 model, this is easily determined:

---

[4] In essence, this statement is based on the fact that the sum of the averages is the average of the sums.

$$T_{ri} = \frac{T_{si}}{1 - \rho_i} = \frac{T_{si}}{1 - \lambda_i T_{si}}$$

The service time T_{si} for link i is just the product of the data rate on the link, in bits per second (B_i) and the average packet length in bits (M). Then:

$$T_{ri} = \frac{\frac{M}{R_i}}{1 - \frac{M\lambda_i}{B_i}} = \frac{M}{B_i - M\lambda_i}$$

Putting all of the elements together, we can calculate the average delay of packets sent through the network:

$$T = \frac{1}{\gamma} \sum_{i=1}^{L} \frac{M\lambda_i}{B_i - M\lambda_i}$$

EXAMPLES

Let us look at a few examples to get some feel for the use of these equations.

Database Server

Consider a LAN with 100 personal computers and a server that maintains a common database for a query application. The average time for the server to respond to a query is 0.6 seconds, and the standard deviation is estimated to equal the mean. At peak times, the query rate over the LAN reaches 20 queries per minute. We would like to answer the following questions:

- What is the average response time ignoring line overhead?
- If a 1.5-second response time is considered the maximum acceptable, what percent growth in message load can occur before the maximum is reached?
- If 20% more utilization is experienced, will response time increase by more or less than 20%?

Assume an M/M/1 model, with the database server being the server in the model. We ignore the effect of the LAN, assuming that its contribution to the delay is negligible. Facility utilization is calculated as:

$\rho = \lambda T_S$
= (20 arrivals per minute)(0.6 seconds per transmission)/(60 sec/min)
= 0.2

The first value, average response time, is easily calculated:

$T_r = T_S/(1 - \rho)$
= $0.6/(1 - 0.2) = 0.75$ seconds

The second value is more difficult to obtain. Indeed, as worded, there is no answer because there is a nonzero probability that some instances of response time will exceed 1.5 seconds for any

value of utilization. Instead, let us say that we would like 90% of all responses to be less than 1.5 seconds. Then, we can use the equation from Table 3b:

$$m_{T_r}(y) = T_r \times \ln(100/(100-y))$$

$$m_{T_r}(90) = T_r \times \ln(10) = \frac{T_s}{1-\rho} \times 2.3 = 1.5 \text{ seconds}$$

We have $T_S = 0.6$. Solving for ρ yields ρ = 0.08. In fact, utilization would have to decline from 20% to 8% to put 1.5 seconds at the 90th percentile.

The third part of the question is to find the relationship between increases in load versus response time. Because a facility utilization of 0.2 is down in the flat part of the curve, response time will increase more slowly than utilization. In this case, if facility utilization increases from 20% to 40%, which is a 100% increase, the value of T_r goes from 0.75 seconds to 1.0 second, which is an increase of only 33%.

Tightly-Coupled Multiprocessor

Let us consider the use of multiple tightly-coupled processors in a single computer system. One of the design decisions had to do with whether processes are dedicated to processors. If a process is permanently assigned to one processor from activation until its completion, then a separate short-term queue is kept for each processor. In this case, one processor can be idle, with an empty queue, while another processor has a backlog. To prevent this situation, a common queue can be used. All processes go into one queue and are scheduled to any available processor. Thus, over the life of a process, the process may be executed on different processors at different times.

Let us try to get a feel for the performance speed-up to be achieved by using a common queue. Consider a system with five processors and that the average amount of processor time provided to a process while in the Running state is 0.1 sec. Assume that the standard deviation of service time is observed to be 0.094 sec. Because the standard deviation is close to the mean, we will assume exponential service time. Also assume that processes are arriving at the Ready state at the rate of 40 per second.

Single-Server Approach

If processes are evenly distributed among the processors, then the load for each processor is 40/5 = 8 processes per second. Thus,

$$\rho = \lambda T_S = 8 \times 0.1 = 0.8$$

The residence time is then easily calculated:

$$t_r = \frac{T_s}{1-\rho} = \frac{0.1}{0.2} = 0.5 \text{ sec}$$

Multiserver Approach

Now assume that a single Ready queue is maintained for all processors. We now have an aggregate arrival rate of 40 processes per second. However, the facility utilization is still 0.8 ($\lambda T_S/M$). To calculate the residence time from the formula in Table 4, we need to first calculate the Erlang C function. If you have not programmed the parameter, it can be looked up in a table under a facility utilization of 0.8 for 5 servers to yield C = 0.554. Substituting,

$$t_r = (0.1) + \frac{(0.544)(0.1)}{5(1-0.8)} = 0.1544$$

So the use of multiserver queue has reduced average residence time from 0.5 sec down to 0.1544 sec, which is greater than a factor of 3. If we look at just the waiting time, the multiserver case is 0.0544 seconds compared to 0.4 seconds, which is a factor of 7.

Although you may not be an expert in queuing theory, you now know enough to be annoyed when you have to wait in a line at a multiple single-server queue facility.

Calculating Percentiles

Consider a configuration in which packets are sent from computers on a LAN to systems on other networks. All of these packets must pass through a router that connects the LAN to a wide-area network and hence to the outside world. Let us look at the traffic from the LAN through the router. Packets arrive with a mean arrival rate of 5 per second. The average packet length is 144 octets, and it is assumed that packet length is exponentially distributed. Line speed from the router to the wide-area network is 9600 bps. The following questions are asked:

1. What is the mean residence time in the router?
2. How many packets are in the router, including those waiting for transmission and the one currently being transmitted (if any), on the average?
3. Same question as (2), for the 90th percentile.
4. Same question as (2), for the 95th percentile.

λ	=	5 packets/sec	
T_S	=	(144 octets × 8 bits/octet)/9600 bps = 0.12 sec	
ρ	=	$\lambda T_S = 5 \times 0.12 = 0.6$	
T_r	=	$T_S/(1 - \rho) = 0.3$ sec	Mean residence time
r	=	$\rho/(1 - \rho) = 1.5$ packets	Mean queue length

To obtain the percentiles, we use the equation from Table 3b:

$$\Pr[R = N] = (1 - \rho)\rho^N$$

To calculate the yth percentile of queue size, we write the preceding equation in cumulative form:

$$\frac{y}{100} = \sum_{k=0}^{m_r(y)} (1-\rho)\rho^k = 1 - \rho^{1+m_r(y)}$$

Here $m_r(y)$ represents the maximum number of packets in the queue expected y percent of the time. That is, $m_r(y)$ is that value below which R occurs y percent of the time. In the form given, we can determine the percentile for any queue size. We wish to do the reverse: given y, find $m_r(y)$. So, taking the logarithm of both sides:

$$m_r(y) = \frac{\ln\left(1 - \frac{y}{100}\right)}{\ln \rho} - 1$$

If $m_r(y)$ is fractional, take the next higher integer; if it is negative, set it to zero. For our example, $\rho = 0.6$ and we wish to find $m_r(90)$ and $m_r(95)$:

$$m_r(90) = \frac{\ln(1-0.90)}{\ln(0.6)} - 1 = 3.5$$

$$m_r(95) = \frac{\ln(1-0.95)}{\ln(0.6)} - 1 = 4.8$$

Thus, 90% of the time there are fewer than 4 packets in the queue, and 95% of the time there are fewer than 5 packets. If we were designing to a 95th percentile criterion, a buffer would have to be provided to store at least 5 packets.

OTHER QUEUING MODELS

In this paper, we have concentrated on one type of queuing model. There are in fact a number of models, based on two key factors:

- The manner in which blocked items are handled
- The number of traffic sources

When an item arrives at a server and finds that server busy, or arrives at a multiple-server facility and finds all servers busy, that item is said to be blocked. Blocked items can be handled in a number of ways. First, the item can be placed in a queue awaiting a free server. This is referred to in the telephone traffic literature as *lost calls delayed*, although in fact the call is not lost. Alternatively, no waiting line is provided. This in turn leads to two assumptions about the action of the item. The item may wait some random amount of time and then try again; this is known as *lost calls cleared*. If the item repeatedly attempts to gain service, with no pause, it is referred to as *lost calls held*. The lost calls delayed model is the most appropriate for most computer and data communications problems. Lost calls cleared is usually the most appropriate in a telephone switching environment.

The second key element of a traffic model is whether the number of sources is assumed infinite or finite. For an infinite source model, there is assumed to be a fixed arrival rate. For the finite source case, the arrival rate will depend on the number of sources already engaged. Thus, if each of L sources generates arrivals at a rate λ/L, then when the queuing facility is unoccupied, the arrival rate is λ. However, if K sources are in the queuing facility at a particular time, then the instantaneous arrival rate at that time is $\lambda(L - K)/L$. Infinite source models are easier to deal with. The infinite source assumption is reasonable when the number of sources is at least 5 to 10 times the capacity of the system.

RECOMMENDED READING

A good practical reference is [TANN95]; it provides detailed guidance for the application of queuing analysis plus a number of worked-out examples. The book also contains a disk with an extensive library of subroutines in Pascal for calculating the characteristics of many queuing situations. Another excellent practical reference is [GUNT98]. [MART93] provides a good overview of queuing theory and contains a number of graphs and tables that can be used to perform quick queuing analyses.

For those who wish to delve more deeply into queuing theory, a host of books is available. Some of the more worthwhile ones are the following. A good text that covers queuing theory and its application to computers and communications is [MOLL89]. [STUC85] is an excellent treatment that focuses on data communications and networking. The classic treatment of queuing theory for computer applications, with a detailed discussion of computer networks, is found in [KLEI75] and [KLEI76]. Perhaps the best book on performance modeling, covering queuing analysis and simulation, is [JAIN91].

A good elementary introduction to statistics is [PHIL92]. For a more detailed and rigorous treatment, there are numerous texts; one that is particularly well suited for self-study is [BULM79]. The U.S. government publishes two excellent guides to the practical application of statistics. [NBS63], which is still available, contains tables, formulas, and examples that aid in determining the proper procedure for estimating values from samples and in evaluating the results. [LURI94] contains detailed step-by-step procedures for performing various statistical tests, as well as a certain amount of tutorial information on the procedures.

BULM79 Bulmer, M. *Principles of Statistics.* New York: Dover, 1979.

GUNT98 Gunther, N. *The Practical Performance Analyst.* New York: McGraw-Hill, 1998.

JAIN91 Jain, R. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.* New York: Wiley, 1991.

KLEI75 Kleinrock, L. *Queueing Systems, Volume I: Theory.* New York: Wiley, 1975.

KLEI76 Kleinrock, L. *Queueing Systems, Volume II: Computer Applications.* New York: Wiley, 1976.

LURI94 Lurie, D., and Moore, R. *Applying Statistics.* U.S. Nuclear Regulatory Commission Report NUREG-1475. (Available from the Government Printing Office, GPO Stock Number 052-020-00390-4).

MART93 Martine, R. *Basic Traffic Analysis.* Englewood Cliffs, NJ: Prentice-Hall, 1993.

MOLL89 Molloy, M. *Fundamentals of Performance Modeling.* New York: Macmillan, 1989.

NBS63 National Bureau of Standards. *Experimental Statistics.* NBS Handbook 91, 1963. (Available from the Government Printing Office, GPO Stock Number 003-003-00135-0.)

PHIL92 Phillips, J. *How to Think About Statistics.* New York: Freeman, 1992.

STUC85 Stuck, B, and Arthurs, E. *A Computer and Communications Network Performance Analysis Primer.* Englewood Cliffs, NJ: Prentice-Hall 1985.

TANN95 Tanner, M. *Practical Queueing Analysis.* New York: McGraw-Hill, 1995.

Recommended Web site:

- **Myron Hlynka's Queueing Theory Page** (www2.uwindsor.ca/~hlynka/queue.html): Includes FAQ, examples, links to other queuing sites, even queuing theory employment opportunities.

ANNEX A JUST ENOUGH PROBABILITY AND STATISTICS

Measures of Probability

A continuous random variable X can be described by either its **distribution function** $F(x)$ or **density function** $f(x)$:

distribution function: $F(x) = \Pr[X \le x]$ $\quad F(-\infty) = 0;$ $\quad F(\infty) = 1$

density function: $f(x) = \dfrac{d}{dx} F(x)$ $\quad F(x) = \int_{-\infty}^{x} f(y)\,dy$ $\quad \int_{-\infty}^{\infty} f(y)\,dy = 1$

For a discrete random variable, its probability distribution is characterized by

$$P_X(k) = \Pr[X = k] \qquad \sum_{\text{all } k} P_X(k) = 1$$

We are often concerned with some characteristic of a random variable rather than the entire distribution, such as the mean value:

$$E[X] = \mu_X = \int_{-\infty}^{\infty} x f(x)\,dx \qquad \text{continuous case}$$

$$E[X] = \mu_X = \sum_{\text{all } k} k \Pr[x = k] \qquad \text{discrete case}$$

Other useful measures:

Second moment:

$$E[X^2] = \int_{-\infty}^{\infty} x^2 f(x)\,dx \qquad \text{continuous case}$$

$$E[X^2] = \sum_{\text{all } k} k^2 \Pr[x = k] \qquad \text{discrete case}$$

Variance: $\quad \mathrm{Var}[X] = E\left[(X - \mu_X)^2\right] = E[X^2] - \mu_X^2$

Standard deviation $\quad \sigma_X = \sqrt{\mathrm{Var}[X]}$

The variance and standard deviation are measures of the dispersion of values around the mean.

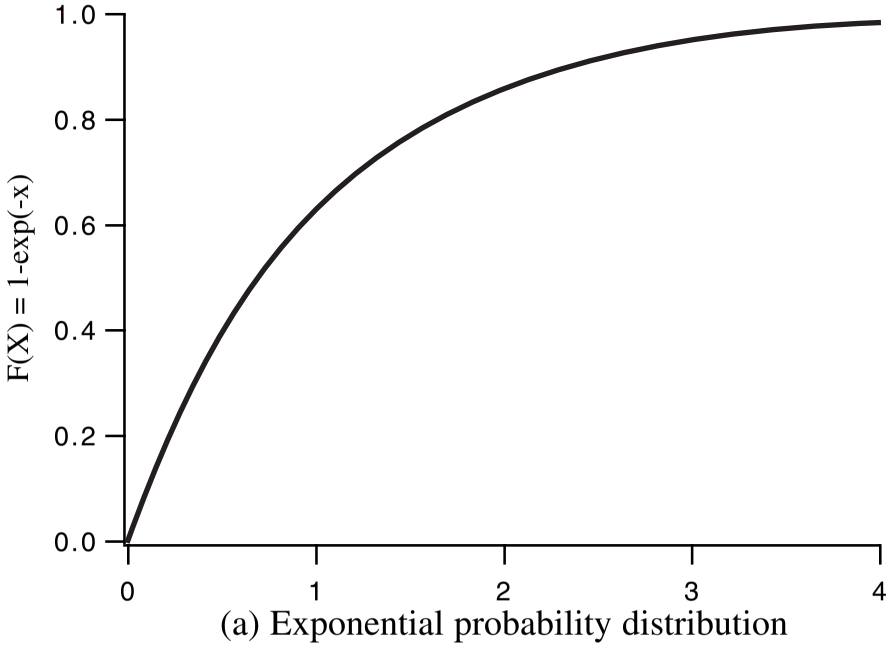
The Exponential and Poisson Distributions

The exponential distribution with parameter λ ($\lambda > 0$) is given by (Figures 8a and 8b):

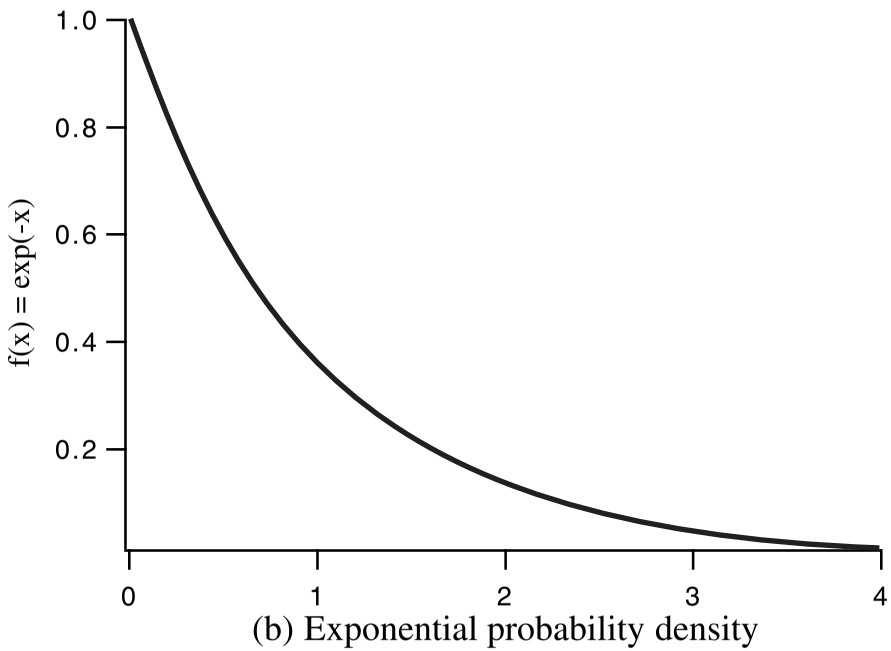
$$F(x) = 1 - e^{-\lambda x} \qquad \text{distribution}$$

$$x \ge 0$$

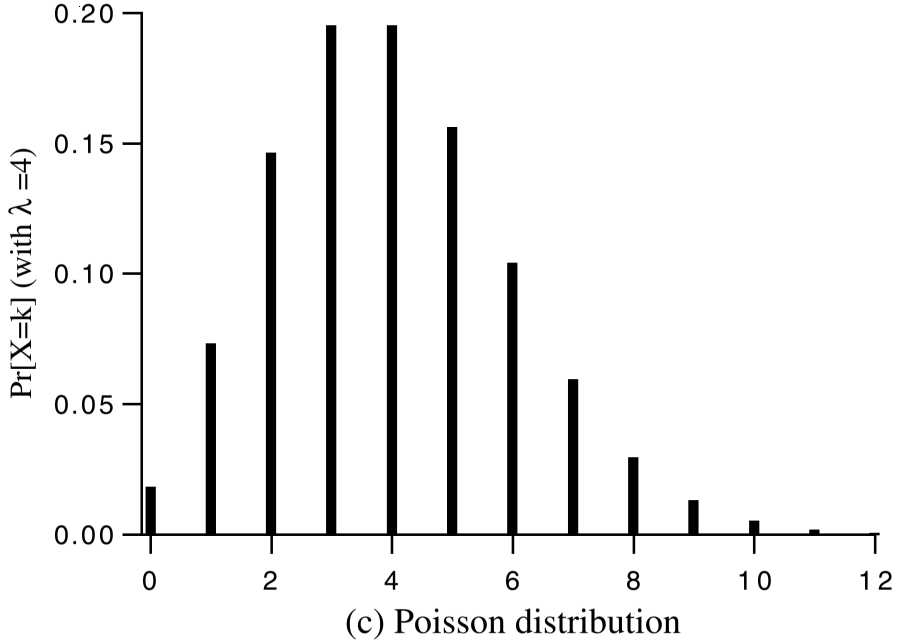
$$f(x) = \lambda e^{-\lambda x} \qquad \text{density}$$



(a) Exponential probability distribution



(b) Exponential probability density



(c) Poisson distribution

Figure 8 Some Probability Functions

The exponential distribution has the interesting property that its mean is equal to its standard deviation:

$$E[X] = \sigma_X = \frac{1}{\lambda}$$

When used to refer to a time interval, such as a service time, this distribution is sometimes referred to as a random distribution. This is because, for a time interval that has already begun, each time at which the interval may finish is equally likely.

This distribution is important in queuing theory because we can often assume that the service time of a server in a queuing system is exponential. In the case of telephone traffic, the service time is the time for which a subscriber engages the equipment of interest. In a packet-switching network, the service time is the transmission time and is therefore proportional to the packet length. It is difficult to give a sound theoretical reason why service times should be exponential, but the fact is that in most cases they are very nearly exponential. This is good news because it simplifies the queuing analysis immensely.

Another important distribution is the Poisson distribution (Figure 8c) with parameter λ ($\lambda > 0$), which takes on values at the points 0, 1,…:

$$\Pr[X = k] = \frac{\lambda^k}{k!} e^{-\lambda} \qquad k = 0,1,2\ldots$$

$$E[X] = \mathrm{Var}[X] = \lambda$$

If $\lambda < 1$, then $\Pr[X = k]$ is maximum for $k = 0$. If $\lambda > 1$ but not an integer, then $\Pr[X = k]$ is maximum for the largest integer smaller than λ; if λ is a positive integer, then there are two maxima at $k = \lambda$ and $k = \lambda - 1$.

The Poisson distribution is also important in queuing analysis because we must assume a Poisson arrival pattern to be able to develop the queuing equations. Fortunately, the assumption of Poisson arrivals is usually valid.

The way in which the Poisson distribution can be applied to arrival rate is as follows. If items arrive at a queue according to a Poisson process, this may be expressed as:

$$\Pr[k \text{ items arrive in time interval T}] = \frac{(\lambda T)^k}{k!} e^{-\lambda T}$$

Expected number of items to arrive in time interval $T = \lambda T$

Mean arrival rate, in items per second = λ

Arrivals occurring according to a Poisson process are often referred to as random arrivals. This is because the probability of arrival of an item in a small interval is proportional to the length of the interval and is independent of the amount of elapsed time since the arrival of the last item. That is, when items are arriving according to a Poisson process, an item is as likely to arrive at one instant as any other, regardless of the instants at which the other customers arrive.

Another interesting property of the Poisson process is its relationship to the exponential distribution. If we look at the times between arrivals of items T_a (called the interarrival times), then we find that this quantity obeys the exponential distribution:

$$\Pr[T_a < t] = 1 - e^{-\lambda t}$$

$$E[T_a] = \frac{1}{\lambda}$$

Thus, the mean interarrival time is the reciprocal of the arrival rate, as one would expect.

Sampling

To perform a queuing analysis, we need to estimate the values of the input parameters, specifically the mean and standard deviation of the arrival rate and service time. If we are contemplating a new system, these estimates may have to be based on judgment and an assessment of the equipment and work patterns likely to prevail. However, it will often be the case that an existing system is available for examination. For example, a collection of terminals, personal computers, and host computers are interconnected in a building by direct connection and multiplexers, and it is desired to replace the interconnection facility with a local area network. To be able to size the network, it is possible to measure the load currently generated by each device.

The measurements that are taken are in the form of samples. A particular parameter, for example, the rate of packets generated by a terminal or the size of packets, is estimated by observing the number of packets generated during a period of time.

To estimate a quantity, such as the length of a packet, the following equations can be used:

sample mean: $$\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$$

sample variance: $$S^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(X_i - \overline{X}\right)^2$$

sample standard deviation: $$S = \sqrt{S^2}$$

where

N = sample size
X_i = ith item in the sample

To estimate the arrival rate from a sample, we can use the following:

$$\overline{\lambda} = \frac{N}{T}$$

where N is the number of items observed in a period of time of duration T. Another approach is to consider each arrival time as a sample and calculate the sample mean and sample standard deviation as above.

When we estimate values such as the mean and standard deviation on the basis of a sample, we leave the realm of probability and enter that of statistics. This is a complex topic that will not be explored here, except to provide a few comments.

It is important to note that the sample mean and sample standard deviation are themselves random variables. For example, if you take a sample from some population and calculate the sample mean, and do this a number of times, the calculated values will differ. Thus, we can talk

of the mean and standard deviation of the sample mean, or even of the entire probability distribution of the sample mean.

It follows that the probabilistic nature of our estimated values is a source of error, known as sampling error. In general, the greater the size of the sample taken, the smaller the standard deviation of the sample mean, and therefore the closer that our estimate is likely to be to the actual mean. By making certain reasonable assumptions about the nature of the random variable being tested and the randomness of the sampling procedure, one can in fact determine the probability that a sample mean or sample standard deviation is within a certain distance from the actual mean or standard deviation. This concept is often reported with the results of a sample. For example, it is common for the result of an opinion poll to include a comment such as: "The result is within 5% of the true value with a confidence (probability) of 99%."

There is, however, another source of error, which is less widely appreciated among non-statisticians, namely bias. For example, if an opinion poll is conducted, and only members of a certain socio-economic group are interviewed, the results are not necessarily representative of the entire population. In a communications context, sampling done during one time of day may not reflect the activity at another time of day. If we are concerned to design a system that will handle the peak load that is likely to be experienced, then we should observe the traffic during the time of day that is most likely to produce the greatest load.