Chapter 1

M/M/1 QUEUE

I'm sure that I've never been in a queue as slow as this.
Any Customer, Anywhere, Anytime

Nobody goes there anymore. It's too crowded.
Yogi Berra

The M/M/1 queue, the simplest and most elementary of all queues, is covered it here in some detail. But what we discuss differs from that covered in the usual first course in queueing theory, and we use different techniques to accomplish our goals. Our purpose is threefold. First, we want to connect Chapter 1 with queueing theory and familiarize the reader with our terminology. Second, we want to set up points of view and techniques that are used in later chapters when LAQT is finally introduced. Third, we want to reinforce the view that the behavior of a queueing system in the transient or small time region may be important more often than we have thought heretofore, and that it is possible to study that region realistically and perform calculations relatively easily, in fact, in some cases with the same ease (or difficulty) as with the steady state.

All systems treated in this book are ***closed***. That is, there is always a fixed number of customers in the system. Each system is made up of two subsystems that interact with each other exclusively by exchanging customers. If N, the fixed number of customers, is large enough, we show that one of the subsystems must become saturated ($\mathbb{P}\mathbf{r}$(subsystem is idle) $\rightarrow 0$). It then becomes a steady source of customers to the other subsystem. ***Open systems***, then, are those where N is so large that one of the subsystems is continuously fed by the other which is at full capacity (almost) all the time. We make this clear in what follows.

2.1 Steady-State M/M/1-Type Loops

Consider the system shown in Figure 2.1.1. It is made of two subsystems, called S_1 and S_2. At any time, S_1 has n customers, S_2 has k customers, and the system as a whole has $N = n + k$ customers. In this chapter both S_1 and S_2 are memoryless and thus have exponential service time pdfs of the form $\mu \exp(-\mu x)$ and $\lambda \exp(-\lambda x)$, respectively (which from a formal point of view means that each external state has only one internal state, but more of that in Chapter 3). The system is completely specified at any time if n and k are known. Since N is fixed, k is known if n is known, so the states of the system can be labeled by $n = 0, 1, 2, \ldots N$ (i.e., there are $N + 1$ states).

The notation $M_2/M_1/1/\,/N$ corresponds to Figure 2.1.1 in the following way. First assume that S_1 has a shorter mean service time than S_2. The first symbol $[M_2]$ indicates that S_2 is memoryless or Markovian or exponential, or equivalently, has only one internal state. M_1 says the same thing about S_1. The third position, containing the number "1", means that S_1 can serve only one customer at a time. The space between the third and fourth slashes tells us that there is no limit as to how many customers can be in the queue at S_1. If there had been a number there, S_1 would have had a ***finite waiting room*** or ***finite buffer***. We look at this *slot* when discussing the *customer loss* problem. The last symbol N indicates that there are a total of N customers in the system. Some books assume that S_2 has N identical servers, so all customers at S_2 can be served simultaneously, as in the ***machine minding model*** (also known as ***machine repairman model***) or in a ***time-sharing system***. This is discussed in detail in Section 2.1.4, and again in Section 6.3.5.

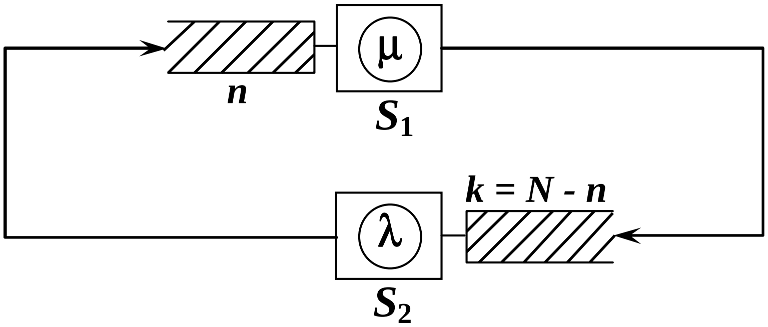

Figure 2.1.1: Closed loop made up of two subsystems, S_1 and S_2. The number of customers at S_1 (including the one in service) is n, and k is the number at S_2. Their sum $N = k + n$ is fixed, thus the system is closed.

Recall from Equations (1.3.2) that the completion rate matrix, $\boldsymbol{M}$, is diagonal, where M_{ii} is the rate at which the system leaves state i given that it is in state i. Here i stands for the integer pair $(n, N-n)$, so, for instance, for $n = 0$, all customers are at S_2, and because only one can be served at a time, $M_{00} = \lambda$. Similarly, when all the customers are at S_1 ($n = N$), no customers can be served at S_2, so $M_{NN} = \mu$. However, for n in between, both subsystems are servicing customers, so the total departure rate is the sum of two service rates, namely, $M_{ii} = \mu + \lambda$. We prove this by deriving the density function for the first subsystem to complete service. First let $R_1(x) = \exp(-\mu x)$ be the probability that S_1 will still be unchanged at time x. Similarly, let $R_2(x) = \exp(-\lambda x)$. Then $R_1(x)R_2(x) = \exp[-(\mu+\lambda)x]$ is the probability that both S_1 and S_2 are unchanged at time x. Next define

$$B_<(x) := 1 - R_1(x)R_2(x)$$

as the probability that at least one of the subsystems has done something by time x. Then

$$b_<(x) := \frac{d}{dx}B_<(x) = (\mu+\lambda)e^{-(\mu+\lambda)x}$$

is the desired pdf. Therefore the process in which one of two things can happen is exponentially distributed, with service (departure in this case) rate $(\mu + \lambda)$.

In summary, the completion rate matrix looks like

$$\boldsymbol{M} = \begin{bmatrix} \lambda & 0 & 0 & \cdots & 0 \\ 0 & \mu+\lambda & 0 & \cdots & 0 \\ 0 & 0 & \mu+\lambda & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & \mu \end{bmatrix}. \tag{2.1.1a}$$

The transition matrix $\boldsymbol{P}$ from Equations (1.3.1) has the following values. For $n = 0$, the only thing that can happen is for a customer to leave S_2 and go to S_1, so $P_{01} = 1$. Similarly, $P_{N,N-1} = 1$. For all other n, one of two things could happen. Either a customer could leave S_2 and go to S_1, or the reverse. In the first case the system would go from state n to $n+1$, and in the other case the system would go from n to $n-1$. The probability that one would happen over the other is proportional to the separate subsystems' (servers') service rates, μ and λ. In other words, $P_{n,n+1} = \lambda/(\mu+\lambda)$. We show this by evaluating the probability that S_2 will finish before S_1. This will occur if S_2 finishes around time t $[b_2(t)dt]$ while S_1 is still running $[R_1(t)]$ for any $t > 0$ (integrate over t). This gives us

$$\mathbf{Pr}(S_2 \text{will finish before } S_1) = \int_0^\infty b_2(t)R_1(t)dt$$

$$= \int_0^\infty \lambda e^{-\lambda\, t} e^{-\mu\, t} dt = \lambda \int_0^\infty e^{-(\mu+\lambda)t} dt = \frac{\lambda}{\mu+\lambda}. \tag{2.1.1b}$$

What we have just shown is important enough to be summarized in a theorem.

> **Theorem 2.1.1:** Let X_1 and X_2 be independent random variables having exponential distribution functions with rates μ and λ, respectively. Then the PDF for the first one to finish, given that both have already started, but have not finished, by time $x = 0$, is also exponentially distributed, with parameter $\mu + \lambda$. That is, let
>
> $$X = \min[X_1, X_2].$$
>
> Then
>
> $$\Pr(X < x) := B_<(x) = 1 - e^{-(\lambda+\mu)x},$$
>
> and
>
> $$b_<(x) = (\mu+\lambda)e^{-(\lambda+\mu)x}.$$
>
> Furthermore, $\Pr(X_2 < X_1)$ is given by (2.1.1b). Because both X_1 and X_2 are exponentially distributed, these results do not depend upon which server started first. ■

The entire $\boldsymbol{P}$ matrix is the following.

$$\boldsymbol{P} = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \frac{\mu}{\mu+\lambda} & 0 & \frac{\lambda}{\mu+\lambda} & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{\mu}{\mu+\lambda} & 0 & \frac{\lambda}{\mu+\lambda} & \cdots & 0 & 0 & 0 \\ : & : & : & : & \cdots & : & : & : \\ 0 & 0 & 0 & 0 & \cdots & 0 & \frac{\lambda}{\mu+\lambda} & 0 \\ 0 & 0 & 0 & 0 & \cdots & \frac{\mu}{\mu+\lambda} & 0 & \frac{\lambda}{\mu+\lambda} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 \end{bmatrix}. \tag{2.1.1c}$$

Finally, $\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{I} - \boldsymbol{P})$ can easily be calculated to give us

$$\boldsymbol{Q} = \begin{bmatrix} \lambda & -\lambda & 0 & 0 & \cdots & 0 & 0 & 0 \\ -\mu & \mu+\lambda & -\lambda & 0 & \cdots & 0 & 0 & 0 \\ 0 & -\mu & \mu+\lambda & -\lambda & \cdots & 0 & 0 & 0 \\ : & : & : & : & \cdots & : & : & : \\ 0 & 0 & 0 & 0 & \cdots & \mu+\lambda & -\lambda & 0 \\ 0 & 0 & 0 & 0 & \cdots & -\mu & \mu+\lambda & -\lambda \\ 0 & 0 & 0 & 0 & \cdots & 0 & -\mu & \mu \end{bmatrix}. \tag{2.1.1d}$$

This procedure of calculating $\boldsymbol{Q}$ in two steps rather than directly, as is usually done, seems cumbersome, but its utility becomes clear in later chapters.

$\boldsymbol{Q}$ matrices of the form in (2.1.1d) (i.e., those that are tridiagonal) generate what are known as ***birth-death processes***. In general, if the states can be linearly ordered, and transitions only occur between neighboring states (i.e., given that the system is in state n, it can only go to $n-1$, n, or $n+1$), then we have a birth-death process. This can be generalized in the following way. Suppose that the states of the system can be partitioned into subsets that are linearly ordered as $\{\Xi_o, \Xi_1, \Xi_2, \ldots, \Xi_{n-1}, \Xi_n, \Xi_{n+1}, \ldots\}$. If transitions can only occur between adjacent sets, we have a ***Quasi Birth-Death* (QBD) *process*** [?]. The $\boldsymbol{Q}$ matrix for a QBD process looks like (2.1.1d), except that each of the elements is itself a matrix. All the processes discussed in this book are QBD. This means leaving out such topics as bulk arrival processes, a typical topic in other queueing theory books.

2.1.1 Time-Dependent Solution for $N = 2$

The time-dependent solution for $N = 1$ was actually done in Exercise 1.3.3. The next simplest nontrivial case is $N = 2$. Here

$$\boldsymbol{Q} = \begin{bmatrix} \lambda & -\lambda & 0 \\ -\mu & \mu+\lambda & -\lambda \\ 0 & -\mu & \mu \end{bmatrix}. \tag{2.1.2}$$

Obviously, $\boldsymbol{\epsilon'}$ ($\boldsymbol{\epsilon} = [1,1,1]$) is a right eigenvector of $\boldsymbol{Q}$ with eigenvalue 0, and it is not hard to find its companion, the left eigenvector with eigenvalue 0 [i.e., $\boldsymbol{\pi}(2)\boldsymbol{Q} = \mathbf{o}$]. One proves by direct substitution that

$$\boldsymbol{\pi}(2) = \frac{1}{1+\rho+\rho^2}[1,\ \rho,\ \rho^2],$$

where $\rho = \lambda/\mu$ and $\boldsymbol{\pi}\,\boldsymbol{\epsilon'} = 1$. The components of the total probability vector $[\boldsymbol{\pi}(2)]_j$ are the steady-state probabilities of finding $(j-1)$ customers at S_1. Put colloquially, after a long time, a random observer who may come along will find $j-1$ customers at S_1 with probability $[\boldsymbol{\pi}(2)]_j$. The eigenvalues of $\boldsymbol{Q}$ satisfy the polynomial equation coming from Equations (1.3.6),

$$\phi(\beta) = \beta^3 - 2(\mu+\lambda)\beta^2 + (\mu^2 + \mu\lambda + \lambda^2)\beta = 0. \tag{2.1.3a}$$

The roots of this equation are (for convenience we let the indices take on values 0 to $N = 2$ rather than the convention used in Chapter 1)

$$\begin{aligned} \beta_o &= 0 \\ \beta_1 &= \mu(1+\rho+\sqrt{\rho}) \\ \beta_2 &= \mu(1+\rho-\sqrt{\rho}). \end{aligned} \tag{2.1.3b}$$

β_o is the root corresponding to the steady-state solution, whereas $\beta_1$and β_2 moderate the transient behavior. Now $\beta_2 < \beta_1$, so the relaxation time from Equations (1.3.11) is $1/\beta_2$. Because the time units are arbitrary, we must establish some comparison to learn something from the formula. One convenient time unit to use in this case is the mean time for a single customer to go around the loop once, unimpeded. A simple way to do this is to let $1/\mu + 1/\lambda = 1$; then, from Equations (1.3.11),

$$RT(\rho) = \frac{\rho}{(1+\rho)(1+\rho-\sqrt{\rho})}.$$

In this case it should be easy to see that RT is maximal when $\rho = 1$ and that $RT(\rho) = RT(1/\rho)$. We examine the general case in Section 2.2, but we note that these results are typical.

Exercise 2.1.1: For a cycle time of 1 ($1/\mu + 1/\lambda = 1$) show that the formula above is true, and draw a graph of RT versus ρ. When is RT a maximum? Prove that $RT(\rho) = RT(1/\rho)$.

Exercise 2.1.2: Find all the left and right eigenvectors of $\boldsymbol{Q}$ and verify that Equations (1.3.8a) are satisfied. Construct $\boldsymbol{G}(t)$ from (1.3.9a), and then $\boldsymbol{\pi}(t;2)$, where $\boldsymbol{\pi}(0;2)$ is one of [1 0 0], or [0 1 0], or [0 0 1].

2.1.2 Steady-State Solution for Any *N*

The steady-state solution for the M/M/1/ /N queue is, of course, well known and is shown in every book that discusses queueing theory to any extent. We discuss it briefly here to show how one goes from closed to open systems. Our assumption in this section is that S_2 is load independent. That is, the service rate of S_2 is the same irrespective of how many customers are in its queue.

From (1.3.9) and (1.3.10), the steady-state solution of our loop satisfies $\boldsymbol{\pi}\,\boldsymbol{Q} = \mathbf{o}$, which from (1.3.2c) is the same as $\boldsymbol{\pi}\,\boldsymbol{M} = \boldsymbol{\pi}\,\boldsymbol{M}\,\boldsymbol{P}$. (See Theorem 1.3.3 for a summary.) These equations are referred to as the steady-state ***balance equations***. In the notation of Chapter 1, the left-hand side ($\pi_i\,\mu_i$) is interpreted as the probability rate of leaving state i, and the right-hand side is the probability rate of entering state i. And, of course, they are equal when a system reaches its steady state.

At this point it is advantageous for us to change our notation, to be consistent with succeeding chapters, where $\boldsymbol{\pi}$ takes on a different meaning. The abstract state i stands for there being $n = i - 1$ customers at S_1, we therefore define the following.

Definition 2.1.1

$r(n;N)$:= steady-state probability that there are n customers at S_1, where N is the (fixed) number of customers in the system overall. Then $r(n;N)$ replaces $[\boldsymbol{\pi}(N)]_i$ $\quad(n = i - 1)$ everywhere. □

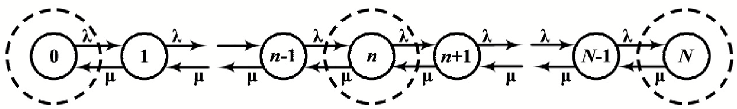

Figure 2.1.2: State transition rate diagram for an M/M/1/ /N queue, representing the probability rate of going from the tail to the head of each arrow. The three closed, dashed curves correspond to the three equations of (2.1.4a).

For the M/M/1/ /N queue, these equations become, using (2.1.1d),

$$\begin{aligned} \lambda r(0;\,N) &= \mu\, r(1;N), \\ (\mu+\lambda) r(n;\,N) &= \lambda r(n-1;\,N) + \mu\, r(n+1;\,N), \\ \mu\, r(N;\,N) &= \lambda\, r(N-1;\,N), \end{aligned} \tag{2.1.4a}$$

where $0 < n < N$. It is common to represent these equations graphically by what are called ***state transition rate diagrams*** (or simply ***transition diagrams***), as shown in Figure 2.1.2. Each arrow corresponds to going from the state represented by the circle at the tail to the state represented by the circle at the head, with probability rate equal to the probability of being at the tail times the rate corresponding to the arrow. Every closed curve encompassing part of the graph represents a valid balance equation, where the sum of the rates represented by the arrows going into the loop equals the sum of the rates leaving the loop. In particular, each closed loop enclosing only one state (circle) yields one of the equations in (2.1.4a).

In any case, the solution to (2.1.4a) is well known to be

$$r(n;N) = \frac{\rho^n}{K(N)}, \qquad 0 \le n \le N, \tag{2.1.4b}$$

where

$$K(N) := \sum_{n=0}^{N} \rho^n = \frac{1-\rho^{N+1}}{1-\rho} \qquad (\rho \neq 1). \tag{2.1.4c}$$

The proof follows by substituting (2.1.4b) into (2.1.4a). Equation (2.1.4c) follows from the requirement that $\sum_{n=0}^{N} r(n;N) = 1$. For future reference, observe that $K(N)$ satisfies the recurrence relation

$$K(N) = 1 + \rho K(N-1). \tag{2.1.4d}$$

When $\rho = 1$, $r(n;N) = 1/(N+1)$ for all n. That is, the steady-state probability for all queue lengths is the same. Yet if the system initially had all its customers at S_1, it would be a long time indeed before a majority of them would be found at S_2. Of course, for very large N, and after a long period of time, we are unlikely to find the system in any particular state. Thus the steady-state solution, if anything, is warning our random observer to be wary of any conclusions concerning the behavior of a system that are based on short-term observations. We look at this again in Section 2.3.

Example 2.1.1: In Figure 2.1.3 we have plotted the steady-state queue length probabilities for the M/M/1//20 queue for various values of ρ. Notice that when $\rho < 1$, $r(n;20)$ is a monotonically decreasing function of n, and when $\rho > 1$, it is a monotonically decreasing function of $N - n$. As you might expect, the curves labeled $\rho = 0.5$ and $\rho = 2$ are mirror images of each other. The most significant feature of

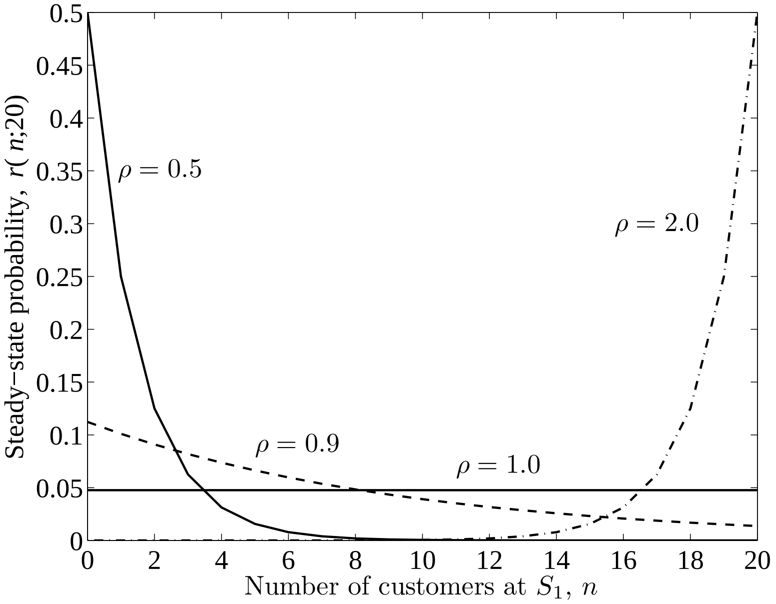

Figure 2.1.3: Steady-state probabilities $r(n;20)$ that there will be n customers at or in S_1, for $\rho = 0.5, 0.9, 1,$ and 2. The curves for $\rho = 0.5$ and 2 are mirror images of each other. Also, the curve for $\rho = 1$ is a constant; that is, all queue lengths are equally likely. These observations are not necessarily true for more general queues. Equations (2.1.4b) and (2.1.4c or d) are used to compute the values plotted.

these curves is that they are so broad, particularly when ρ is near 1. It is best to think of $r(n;N)$ as being the fraction of time that n customers will be at S_1 over a very very long period of time. ▲[†]

What is often of interest in closed systems is the activity of each of the servers. The probability that a server is busy is equivalent to the fraction of time it is busy over a long period of time. This, in turn, determines the amount of "work" done per unit time by that server. Now suppose that customers somehow enter our closed loop, travel around until they have received a total of T_i units of service from S_i $(i = 1, 2)$, and then leave, being replaced instantly by a statistical clone. By definition, $T_1/T_2 = \rho$. Next define the steady-state probabilities.

Definition 2.1.2

$P_i(N)$:= *steady-state probability that* S_i, $i = 1, 2$, *is busy, given that there are* N *customers in the loop.* Then

$$\Lambda(N) := \frac{P_i(N)}{T_i} \tag{2.1.5a}$$

is the rate at which customers enter and leave the loop, and is independent of i. $\Lambda(N)$ can be referred to as the ***system throughput***. In fact, this formula is valid for networks of any number of servers, as long as customers do not have the option of using a different server if the one they want is busy, and if the servers are not load dependent. □

$P_1(N)$ is 1 minus the probability that S_1 is idle, so from (2.1.4b) with $n = 0$, and (2.1.4d),

$$P_1(N) = 1 - r(0;N) = 1 - \frac{1}{K(N)} = \frac{K(N)-1}{K(N)} = \rho\frac{K(N-1)}{K(N)}\cdot \tag{2.1.5b}$$

Similarly, from (2.1.4c),

$$P_2(N) = 1 - r(N;N) = \frac{K(N)-\rho^N}{K(N)} = \frac{K(N-1)}{K(N)}\cdot \tag{2.1.5c}$$

Then, because $\rho = T_1/T_2$, we show that the throughput as seen at S_1 is the same as that seen at S_2:

$$\Lambda(N) = \frac{P_1(N)}{T_1} = \frac{1}{T_2}\frac{K(N-1)}{K(N)} = \frac{P_2(N)}{T_2}\cdot \tag{2.1.5d}$$

Example 2.1.2: We can understand the throughput behavior by looking at Figure 2.1.4, which shows $\Lambda(N)$ as a function of N for several values of ρ. Note that $\Lambda(N;\rho) = \Lambda(N;1/\rho)$. In all cases, $\Lambda(N)$ saturates as N becomes increasingly large, and we see behavior typical of even more complicated queueing systems. That is, $\Lambda(N+1) > \Lambda(N)$ for all N, but

$$[\Lambda(N+2) - \Lambda(N+1)] < [\Lambda(N+1) - \Lambda(N)].$$

This is the law of diminishing returns. "Adding yet one more customer to the system will increase throughput, but the increase will not be as much as it was in adding the previous customer." Finally,

$$\lim_{N\to\infty}[P_1(N) + P_2(N)] = 1 + \rho, \qquad \text{for } \rho \le 1.$$

That is, in general, only one server will saturate, and the other will be busy only a fraction of the time. Only when $\rho = 1$ will both servers approach full capacity with ever-increasing N. ▲

Exercise 2.1.3: Prove that the limit given in the preceding equation is indeed true. What is the limit when ρ is greater than 1? Also prove that $\Lambda(N;\rho) = \Lambda(N;1/\rho)$ when $T_1 + T_2 = 1$.

[†] Symbol ▲ designates the end of the example

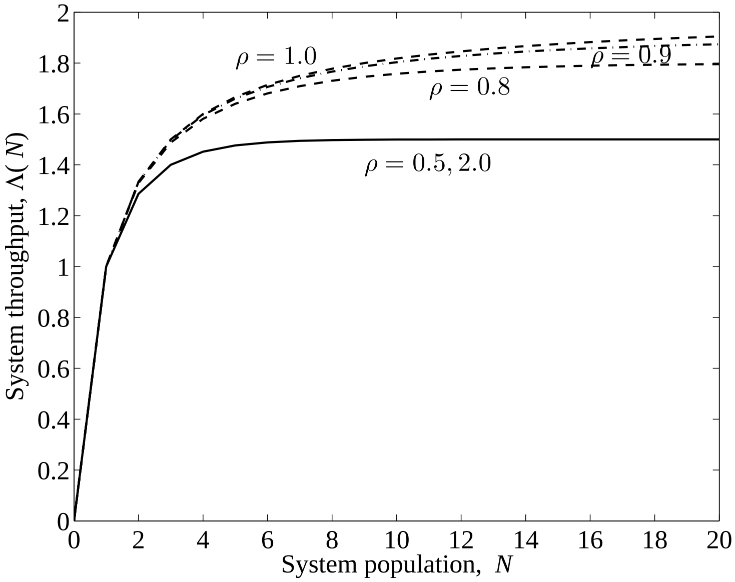

Figure 2.1.4: Throughput for steady-state M/M/1//N queues, where the total resource time needed for a customer to go around once is $T_1 + T_2 = 1$. The curves for $\rho = 0.5$ and $\rho = 2$ are identical because ρ and $1/\rho$ yield the same system with S_1 and S_2 interchanged. All the curves will saturate (become horizontal) if N is made large enough. Use Equations (2.1.5c), (2.1.5d), and (2.1.4d).

2.1.3 Open M/M/1 Queue ($N \to \infty$)

We can find the open system solution by doing the following. When $\rho < 1$, Equations (2.1.4) retain their meaning for large N. In this case,

$$\lim_{N\to\infty} K(N) = \frac{1}{1-\rho},$$

so

$$r(n) := \lim_{N\to\infty} r(n;N) = (1-\rho)\rho^n \tag{2.1.6a}$$

and

$$\lim_{n\to\infty} r(n) = 0.$$

That is, when N is very large, the probability that S_2 will be idle is negligible, so it is continually serving customers whose interdeparture times are exponentially distributed. Each new customer starts up in the same way the previous one did, so S_2 becomes a steady Poisson process of arrivals to S_1. Thus we have the equivalent of an open M/M/1 queue, with a mean queue length of

$$\bar{q}_s := \sum_{n=1}^{\infty} n\, r(n) = (1-\rho)\sum_{n=1}^{\infty} n\,\rho^n = \frac{\rho}{1-\rho}\cdot \tag{2.1.6b}$$

When $\rho > 1$ it follows from (2.1.4c) that $1/K(N)$ becomes vanishingly small for very large N, and thus for small n, $r(n;N)$ is essentially zero. Now S_1 is never idle and becomes a Poisson source for S_2. One would expect a certain duality between S_1 and S_2, which indeed is the case. Simply interchange 1 and 2, and thus replace ρ by $1/\rho$.

It is also interesting to evaluate the asymptotic throughput of our loop. We are thus interested in [from (2.1.5d)]

$$\lim_{N\to\infty} \Lambda(N) = \frac{1}{T_2}\frac{K(N-1)}{K(N)}\cdot$$

We have already noted that when $\rho < 1$, $K(N)$ approaches $(1-\rho)^{-1}$, but from (2.1.4c), $K(N)$ grows as ρ^N when ρ is greater than 1. This leads easily to the following limiting values.

$$\lim_{N\to\infty} \Lambda(N) = \frac{1}{T_2} \qquad \text{for } \rho \le 1$$

and

$$\lim_{N\to\infty} \Lambda(N) = \frac{1}{T_2}\frac{1}{\rho} = \frac{1}{T_1} \qquad \text{for } \rho \ge 1.$$

In other words, we have proven what should be obvious. The throughput of the system is bounded by the maximal throughput of the slower server, the ***bottleneck***. The two equations can be summarized by

$$\lim_{N\to\infty} \Lambda(N) = \min\left(\frac{1}{T_1}, \frac{1}{T_2}\right) \cdot \tag{2.1.6c}$$

A perhaps more interesting question to answer is: how long will a customer be at S_1, both waiting for and being served? This turns out to be easy to answer once the mean queue length is known. The relevant expression, ***Little's formula***, which we introduced in (1.1.2), existed for many years before being proven under certain conditions by J. D. C. Little in 1961 [**?**]. Recall that it is valid for any subsystem that has been in operation long enough so that the number of customers who have come and gone is far greater than the number presently there or who were there originally. Restated simply,

$$\bar{q}_s = \Lambda \bar{T}_s, \tag{2.1.7a}$$

where Λ is the mean arrival rate to (and departure rate from) the subsystem and $\bar{T}_s$ is the mean time spent there by each customer. In our case, $\Lambda = \lambda$ and $\rho = \lambda/\mu$, so from (2.1.6b), we have proven (1.1.4b)

$$\bar{T}_s = \frac{\bar{q}_s}{\lambda} = \frac{\bar{x}}{1-\rho}, \tag{2.1.7b}$$

where $\bar{x} = 1/\mu$ is the mean service time of S_1. Note that if $\rho = 0$ (no customers waiting at all), the mean time a customer remains in the system is the expected $\bar{x}$, and as with the mean queue length, the time a customer must wait grows unboundedly as ρ approaches 1.

It is useful to tighten up our terminology somewhat. Often, one wishes to make a distinction between the time spent waiting for service and the time in service. We use the term ***system time*** or ***total time*** spent in, say, S_1 as the time spent by a customer from the moment he enters S_1's queue until he leaves that subsystem. In a closed loop, this also corresponds to the time interval from the moment the customer leaves S_2 until he returns. For that reason, this time interval is also called the ***response time*** for S_1. We use the three terms interchangeably, tending to prefer the first two when discussing open systems, whereas the latter tends to be used more in dealing with time-sharing systems.

In many applications, the time spent being served is considered useful, and only the time spent waiting in the queue is wasted. This time is called both ***queueing time*** and ***waiting time***. We try to use the latter term, for there is some ambiguity here when load-dependent servers are considered (see the following section and Section 5.4), or when we consider "generalized M/G/C systems" in Chapter 6, for then it is not always clear when waiting ends and service begins. We often talk about ***queue length***, or the *number of customers in the queue*, and when we do, we invariably mean "the number of customers at, or in, S_i," that is, including those being served.

If only one customer can be served at a time, and the performance of S_1 is the same no matter how many customers are in its queue, the steady-state mean system time $\bar{T}_s$ and mean waiting time $\bar{T}_w$ are related by the simple relation

$$\bar{T}_s = \bar{T}_w + \bar{x}_1. \tag{2.1.7c}$$

From Little's formula, the number in the queue and the number in S_1 are related by the slightly strange formula

$$\bar{q}_s = \bar{q}_w + \rho. \tag{2.1.7d}$$

The reason ρ appears instead of 1 is that sometimes there is no one waiting when someone is being served. It is pleasant to realize that (2.1.7c) and (2.1.7d) are true for any distribution, but the reader should be careful to observe the restrictions as stated in the beginning of this paragraph.

2.1.4 Buffer Overflow and Cell Loss for M/M/1/N Queues

An important problem in designing systems with queues involves deciding how much space should be provided to accommodate waiting customers. We look at this issue in two ways. First consider that a ***waiting room*** is made up of a ***primary buffer*** that can accommodate, say N_1 customers, and a secondary buffer, or ***backup buffer***, that can hold as many as needed. An example of this might be a cache interfacing a bulk storage device with a communications channel. Then the question is the following.

(1) What is the probability that an arriving customer will not be able to fit into the primary buffer, or in other words, will the ***buffer overflow***?

One could instead assume that there is only a primary buffer, with no backup. Then an arriving customer, seeing a full buffer, would give up and disappear, or what is mathematically equivalent, return to the queue at S_2. The question then is the following.

(2) What is the probability that an arriving customer will be rejected from the queue at S_1?

The first case corresponds to an M/M/1 queue, and the second corresponds to an M/M/1/N_1/N queue. The latter expression requires some interpretation. If $N_1 < N$ and we are to assume that customers arriving at a full queue would have to instantly return to the end of the queue at S_2, then S_2 is always busy, so N might just as well be ∞. For this reason, the M/M/1/N queue is considered to be open even though the population at S_1 is always less than or equal to N_1.

If, on the other hand, $N_1 \geq N$ the buffer can never be full to an arriving customer. Therefore,

$$\mathrm{M/M/1}/N_1/N \equiv \begin{cases} \mathrm{M/M/1}/N_1 & \text{for} \quad N_1 < N \\ \mathrm{M/M/1}//N & \text{for} \quad N_1 \geq N \end{cases}$$

In general, the solutions for M/G/1//N loops are very similar to those for M/G/1/N queues. The difference becomes significant in Section 5.3 when we compare the G/M/1//N and G/M/1/N queues, but we give a short explanation here. When a customer arrives at an M/M/1/N queue that already has N customers, the arriving customer is turned away. Each subsequent arrival will be turned away until S_1 has a completion. Given that the arrival process is a Poisson process, the time for the next arrival is exponentially distributed, but now starting at the time of the departure, having no memory of the previous arrival. The M/M/1//N loop behaves in the following way. If all N customers are at S_1, there can be no further arrivals until S_1 has a completion. After such a completion, S_2 can service its new arrival, thereby preparing a new arrival for S_1. We see that shutting off the arrival process has the same effect as turning away arrivals, but only if the arrival process is memoryless, that is, Poisson.

Cases (1) and (2) both talk about an ***arriving customer***, whereas we have given solutions for a random observer [$r(n)$]. Therefore we must introduce some new variables.

Definition 2.1.3

$a(n;\, N) :=$ *probability that a customer arriving at* S_1 *in an M/M/1//N loop will see n customers already in the queue, including the one in service.* By this definition, it must be that

$$a(N;\, N) = 0.$$

After all, the arriving customer is one of the N customers in the system, so he can see at most $N-1$ customers before him at S_1. □

We give similar definitions for the M/M/1/N queue, using ***f*** (for ***finite buffer***) as the distinguishing marker.

Definition 2.1.4

$r_f(n;\, N)$:= probability that a random observer will see n customers at or in S_1 for an M/M/1/N queue.
$a_f(n;\, N)$:= probability that a customer arriving at an M/M/1/N queue will see n customers already in the queue, including the one in service. By this definition, $a_f(N;\, N)$ is the probability that an arriving customer will be turned away (i.e., the ***customer loss*** probability). If we were dealing with the tranmission of packets or cells in telecommunications we would call this ***packet loss probability*** or ***cell loss probability***. □

In Chapters 4 and 5 we give a more rigorous argument for the following equations, but for the systems of interest here the following arguments are sufficient. Because we are looking at Poisson arrivals, each arriving customer has no knowledge of when the previous customer arrived, therefore he will see the same thing that a random observer does, except that he cannot see N customers already in the queue. Therefore,

$$a(n;\, N) = \begin{cases} c(N)\, r(n;\, N) & \text{for } 0 \le n < N \\ 0 & \text{for } n = N \end{cases} .$$

The sum of the $a(n;\, N)$'s must be 1, therefore it follows that $c(N) = (1-\rho^{N+1})/(1-\rho^N)$. We can now summarize the steady-state properties of the M/M/1//N queue in the following theorem.

Theorem 2.1.2: The steady-state probabilities of finding n customers in an M/M/1//N loop are given by Equations (2.1.4), namely,

$$r(n; N) = \frac{\rho^n}{K(N)}, \qquad 0 \le n \le N,$$

where $K(N) = N+1$ for $\rho = 1$, and for $\rho \ne 1$

$$K(N) := \sum_{n=0}^{N} \rho^n = \frac{1-\rho^{N+1}}{1-\rho} = 1 + \rho K(N-1).$$

The probability that a customer arriving at S_1 will find n customers already there is given by

$$a(n;\, N) = \frac{1-\rho}{1-\rho^N}\rho^n \quad \text{for } 0 \le n < N, \tag{2.1.8a}$$

and $a(N;\, N) = 0$. For $\rho = 1$, $a(n;\, N) = 1/N$. In other words, $a(n;\, N) = r(n;\, N-1)$ for $n < N$.

The probability that an arriving customer will see $N_1 \le n < N$ or more customers already in the queue (***overflow probability***) is given by:

$$P_o(N_1;\, N) := \sum_{n=N_1}^{N-1} a(n;\, N) = \frac{\rho^{N_1} - \rho^N}{1-\rho^N}\cdot \tag{2.1.8b}$$

For $\rho < 1$ the open M/M/1 queue steady-state probabilities, from (2.1.6a) and (2.1.8a), are

$$a(n) = r(n) = \lim_{N\to\infty} r(n;\, N) = (1-\rho)\rho^n. \tag{2.1.8c}$$

Here, the arriving customer and the random observer see the same queue lengths.

The mean queue length, from (2.1.6b) is

$$\bar{q}_s = \frac{\rho}{1-\rho}$$

and the mean system time, from (2.1.7b), is

$$\bar{T}_s = \frac{\bar{x}}{1-\rho},$$

where $\rho = \lambda/\mu$ and $\bar{x} = 1/\mu$. Also,

$$P_o(N_1) := \lim_{N\to\infty} P_o(N_1;\, N) = \rho^{N_1}. \tag{2.1.8d}$$

We have used the subscript ***o*** to denote primary buffer ***o***verflow. ■

The steady-state solutions for the M/M/1/N_1 queue are easy to write down, because an arriving customer in a Poisson arrival process sees the same thing as the random observer, even if the finite buffer is full. Therefore, we have the following.

Theorem 2.1.3: Systems with finite buffers have the following probabilities.

$$a_f(n;\, N_1) = r_f(n;\, N_1) = r(n;\, N_1) = \frac{1-\rho}{1-\rho^{N_1+1}}\rho^n. \tag{2.1.9a}$$

These equations are valid for all ρ. The probability that an arriving customer will find the buffer full, and be turned away is given by:

$$P_f(N_1) = a_f(N_1;\, N_1) = \frac{1-\rho}{1-\rho^{N_1+1}}\rho^{N_1}. \tag{2.1.9b}$$

In telecommunications systems, this is known as the ***cell loss probability*** or ***packet loss probability***.

The mean queue length, $\bar{q}_f(N_1)$, is

$$\bar{q}_f(N_1) := \sum_{n=1}^{N_1} n\, r_f(n;\, N_1)$$

$$= \frac{\rho}{1-\rho}\left[\frac{1+N_1\rho^{N_1+1}-(N_1+1)\rho^{N_1}}{1-\rho^{N_1+1}}\right]. \tag{2.1.9c}$$

Note that $\bar{q}_f(N_1)$ does not blow up at $\rho = 1$. In fact $\bar{q}_f(N_1|\rho = 1) = N_1/2$, and $P_f(N_1|\rho = 1) = 1/(N_1+1)$. In other words, a relatively small loss of cells can yield a manageable size queue. (Recall that the mean queue length for a queue with an infinite buffer, where no losses are allowed, is infinite when $\rho = 1$).

Let $T_f(N)$ be the random variable denoting the system time for a customer that is not rejected. Then

$$\mathbb{E}[T_f(N_1)] = \frac{1/\mu}{1-\rho}\left[\frac{1+N_1\rho^{N_1+1}-(N_1+1)\rho^{N_1}}{1-\rho^{N_1}}\right]. \tag{2.1.9d}$$

This last equation requires some explanation which we give in the following proof. ■

Proof: In order to get (2.1.9d) from (2.1.9c) using Little's formula, one must use the *effective* arrival rate to the queue. That is, one must include only those customers that are not turned away. That is,

$$\lambda_f(N_1) := \frac{\lambda}{1-Pff(N_1)} = \frac{1-\rho^{N_1}}{1-\rho^{N_1+1}}\,\lambda.$$

Then (2.1.9d) follows from

$$\mathbb{E}[T_f(N_1)] = \frac{\bar{q}_f(N_1)}{\lambda_f(N_1)}.$$

An alternate proof is given as an exercise.

Note that the effective arrival process is no longer a Poisson process. In fact, it's no longer a renewal process. Observe that the customers are not thrown away randomly. If one is thrown away, then the next one is also likely to be lost. $\boxed{\textbf{QED}}$[†]

Exercise 2.1.4: Using (2.1.9b) and given a fixed value for the probability p_ℓ of customer loss, show that ρ must always be less than $1/(1-p_\ell)$ in order that $P_f(N) \leq p_\ell$, no matter how large N is. [See Equations (2.1.10) for a general proof.]

Exercise 2.1.5: One can derive (2.1.9d) directly from the definition of $a_f(n;\, N_1)$. The service distribution is exponential here, therefore the mean time remaining for the customer in service at the moment a new customer arrives is $1/\mu$, the same as from the beginning of service. If a customer arrives with $n < N$ already in the queue, then he must expect to wait $[(n+1)/\mu]$ units of time until all those in front and he himself are served. The probability that he will find n in the queue, given that he will be accepted, is given by $a_f(n;\, N_1 \,|\, \text{accepted}) := a_f(n;\, N_1)/[1 - P_f(N_1)]$ Then

$$\bar{T}_f(N_1) = \sum_{n=o}^{N_1-1} \left[\frac{n+1}{\mu}\right] a_f(n;\, N_1 \,|\, \text{accepted}).$$

Use this expression to derive (2.1.9d)

Before closing this section we compare $P_o(N_1)$ and $P_f(N_1)$ and discuss their uses and significance. First note that $P_f(N_1) < P_o(N_1)$ for every ρ, remembering that $P_o(N_1)$ is not defined for $\rho \geq 1$. The reason should be clear, because the finite buffer system throws away customers, and thus processes fewer of them than the overflow system for any given arrival rate. In exchange for this, the mean queue length and the mean waiting time for the customers is considerably reduced. For instance, let $\rho = 1$ and $N_1 = 10$. Then the mean queue length in the back-up buffer of the M/M/1 queue is infinite, but $\bar{q}_f(10) = 5$. This can be evaluated from (2.1.9c) by using L'Hospital's rule, or by recognizing that $a_f(n;\, N_1;\, \rho = 1) = 1/(N_1 + 1)$ $\forall\, n$. We see, then, by throwing away one customer in 11, one allows the others to get decent service; that is, $\bar{T}_f(10;\, \rho = 1) = 5.5/\mu$.

Maximum Cell Loss

Finite buffers can be a useful solution for systems where not all customers must be served. For instance, one may throw away 10% of the packets carrying telephone messages or video data over telecommunications networks, and still be able to recognize the audio or video signal. But as ρ approaches 2, half the customers

[†] These letters stand for the time-honored Latin phrase *Quod Erat Demonstrandum*, whose translation is "which was to be demonstrated." $\boxed{\textbf{QED}}$ designates the end of a proof.

have to be rejected, a circumstance that is not acceptable even for these examples. In any case, as ρ becomes larger, $\bar{q}_f(N_1)$ approaches N_1.

Suppose that a system can tolerate a maximum fractional loss of p_ℓ. Then there exists a maximum ρ_m above which even an infinite buffer will be inadequate. Consider a very large interval of time Δ. During that period, a total of $N_a(\Delta)$ customers will have arrived at the server, while $N_s(\Delta)$ customers will have been served. Their difference $N(\Delta)$ is the number waiting, or thrown away. In the limit as Δ becomes unboundedly large, all three must become unboundedly large. Because we are assuming a finite buffer, if the arrival rate is greater than the service rate, $N(\Delta)/\Delta$ must be the rate at which customers are lost. Therefore, $N(\Delta)/N_a(\Delta)$ must be the fraction that are lost. But

$$\lambda := \lim_{\Delta\to\infty} \frac{N_a(\Delta)}{\Delta} \quad \text{and} \quad \mu := \lim_{\Delta\to\infty} \frac{N_s(\Delta)}{\Delta}.$$

Therefore,

$$p_\ell > \lim_{\Delta\to\infty} \frac{N(\Delta)}{N_a(\Delta)} = \lim_{\Delta\to\infty} \frac{N_a(\Delta) - N_s(\Delta)}{N_a(\Delta)} = 1 - \frac{\mu}{\lambda} = 1 - \frac{1}{\rho}, \tag{2.1.10a}$$

and solving for ρ,

$$\rho < \rho_m = \frac{1}{1 - p_\ell}. \tag{2.1.10b}$$

Note that $\rho_m > 1$. Thus, as ρ approaches ρ_m, the buffer size needed to keep losses below p_ℓ goes to infinity, and excessive losses cannot be prevented. This is true for all load-independent single-server queues. For load-dependent, or multiple-server queues the concept of utilization must be generalized, but then an appropriate bound can be derived.

In many applications, no amount of loss is acceptable, as in the transmission of data or text over a communications channel. The formulas for $P_o(N_1)$ and $P_f(N_1)$ show that both are proportional to ρ^{N_1}, so to reduce the loss or overflow, one can either increase the buffer size, or decrease ρ by replacing the server with a faster one. If delay is not the critical factor, then increasing the buffer's capacity may be the cheaper solution. For instance, by doubling N_1, one gets $P(2\,N_1) \approx P(N_1)^2$. If $P(N_1)$ is already small, say 0.01, then $P(2\,N_1) \approx .0001$, very small indeed. Thus one often solves such problems by throwing buffer space at it. In Chapter 4 we show that for certain kinds of service time distributions, this solution will not work.

Exercise 2.1.6: Draw curves of $\bar{q}_f(N_1)$ as a function of $\rho = 0 \to 2$ for $N_1 = 10,\ 20,$ and 40. Include $\bar{q}_s$ for $\rho = 0 \to 1$ for comparison.

Exercise 2.1.7: Suppose that a router has enough space to hold 20 packets, and that $\rho = 0.9$. What percentage of packets will be lost if there is no backup buffer? By how much must the service rate $[\mu]$ be increased to reduce losses by a factor of 10? How much buffer space must be added for the same reduction? Redo the problem for overflow to a backup buffer.

2.1.5 Load-Dependent Servers

The solutions for the M/M/1 queue can be extended without much difficulty to the M/M/C//N, and even somewhat more general, queues. Suppose that there are C identical exponential servers in S_1, each with service rate μ, feeding off a single queue. That is, as long as there are $n \geq C$ customers at S_1, all of the servers will be active, and as long as $n \leq C$, none of the customers will be waiting to be served. As we already know, if several exponential servers are busy, the probability rate for something to happen is the sum of their service rates. Therefore, we can define a service rate for S_1 that depends on the number of customers there. That is, let $\mu(n)$ be the service rate of S_1 when there are n customers there; then

$$\mu(n) = \begin{cases} n\,\mu & \text{for } n \leq C \\ C\,\mu & \text{for } n \geq C. \end{cases} \tag{2.1.11a}$$

We think of S_1 as a load-dependent server. Actually, the formulas we derive in this section do not depend on the explicit form we have just given the μ's; thus we can immediately generalize, and let $\mu(1)$, $\mu(2)$, and so on, be any positive numbers. The reader may think of S_i as a ***multiple server*** subsystem, or as a single server whose service rate changes (not necessarily by integral units) with change of queue length. See the end of this section for further notational discussion.

Another formulation, which we adopt here, is to introduce the ***load-dependence factor*** $\alpha_1(n)$, which is the ratio of service rates $\mu(n)$ and $\mu(1)$. By definition, $\mu(1) := \mu$, $\alpha_1(1)$ always equals 1, and $\alpha_1(n) = \mu(n)/\mu$, which for a subsystem with C identical servers gives the following.

$$\alpha_1(n) = \begin{cases} n & \text{for } n \leq C \\ C & \text{for } n \geq C. \end{cases} \tag{2.1.11b}$$

Clearly, $\mu(n) = \alpha_1(n)\mu$. Similarly, we can view S_2 as a load-dependent server, with load-dependence factor $\alpha_2(n)$. Then $\lambda(n) = \alpha_2(n)\lambda$. Next look at Figure 2.1.2. The arrow going from n to $n-1$ corresponds to the probability rate of going from n to $n-1$, which can happen only if there is a completion at S_1. The rate for this to happen is $\mu(n)$. Similarly, the arrow going from n to $n+1$ corresponds to an arrival from S_2, whose rate must be $\lambda(N-n)$. Then all the arrows pointing to the left should be labeled (reading from right to left)

$$\mu(N),\ \mu(N-1),\ \ldots,\ \mu(n+1),\ \mu(n),\ \mu(n-1),\ \ldots,\ \mu(1),$$

and those pointing to the right are labeled (reading, this time, from left to right)

$$\lambda(N),\ \lambda(N-1),\ \ldots,\ \lambda(N-n+1),\ \lambda(N-n), \lambda(N-n-1),\ \ldots,\ \lambda(1).$$

Before solving for the M/M/C//N loop, let us review the meaning of a *state transition-rate diagram.* If, as in Figure 2.1.2, a single node is encircled, the sum of the probability rates entering the circle minus the sum of those leaving must be zero in the steady state. Suppose, instead, that two adjacent nodes are enclosed together. Then the arrows connecting them would not be included in the balance equations. But this would yield the same as one would get by adding the single equations together. After all, each of the two arrows appears in each equation, once as leaving one node, and once as entering the other, canceling out when the two equations are added. In general, then, we can say that for *any* closed curve, what goes in must equal what goes out for the steady state to occur. Now consider the closed curve that encompasses all nodes from 0 to n. Only one arrow goes in, and one arrow goes out, so we have the simple set of first-order difference equations:

$$\lambda(N-n)r(n;N) = \mu(n+1)r(n+1;N) \quad \text{for } 0 \leq n < N. \tag{2.1.12a}$$

In particular,

$$r(1;N) = \frac{\lambda(N)}{\mu(1)}r(0;N) \tag{2.1.12b}$$

and

$$r(2;N) = \frac{\lambda(N-1)}{\mu(2)} r(1;N) = \frac{\lambda(N)\,\lambda(N-1)}{\mu(1)\,\mu(2)} r(0;N). \tag{2.1.12c}$$

Next, following the notation of [**?**], let $\rho = \lambda/\mu$, $\beta_i(0) := 1$, and for $n > 0$,

$$\beta_i(n) := \alpha_i(n)\beta_i(n-1) = \alpha_i(1)\alpha_i(2)\cdots\alpha_i(n). \tag{2.1.13a}$$

For a subsystem with C identical servers, we have

$$\beta_i(n) := \begin{cases} n! & \text{for } n \le C \\ C!\,C^{n-c} & \text{for } n \ge C. \end{cases} \tag{2.1.13b}$$

Then with only a little trickery, the general solution becomes

$$r(n;\,N) = \frac{1}{K(N)} \frac{\rho^n}{\beta_1(n)\,\beta_2(N-n)}, \tag{2.1.14a}$$

where, owing to the fact that the sum of probabilities must be 1,

$$K(N) := \sum_{n=0}^{N} \frac{\rho^n}{\beta_1(n)\beta_2(N-n)}. \tag{2.1.14b}$$

The reader may recognize this as a discrete convolution of the reciprocals of the μ's and λ's.

Next consider a generalization of the throughput as defined in (2.1.5a). The probability that S_1 is busy no longer can yield the throughput, because its service rate depends on n. Therefore, it is somewhat more difficult to express for a load-dependent server, but turns out to be just as simple to compute. The rate at which S_1 serves customers depends on the distribution of the number in the queue. Then $\Lambda(N)$ is a weighted average of the $\mu(n)$'s:

$$\Lambda(N) = \sum_{n=1}^{N} \mu(n) r(n;N) = \sum_{n=1}^{N} \mu\,\alpha_1(n) r(n;N) = \frac{\mu}{K(N)} \sum_{n=1}^{N} \frac{\alpha_1(n)\rho^n}{\beta_1(n)\beta_2(N-n)}.$$

But $\alpha_1(n)/\beta_1(n) = 1/\beta_1(n-1)$ and $\mu\,\rho = \lambda$, so (change the summation variable from n to $n-1$)

$$\Lambda(N) = \frac{\lambda}{K(N)} \sum_{n=1}^{N} \frac{\rho^{n-1}}{\beta_1(n-1)\beta_2(N-n)} = \frac{\lambda K(N-1)}{K(N)}. \tag{2.1.15a}$$

This is identical to the throughput for the load-independent system described in (2.1.5d) with $\lambda = 1/T_2$, except that now $K(N)$ does not satisfy (2.1.4d). There is no simple recursive relationship among the $K(N)$s for arbitrary β's.

There are three different ways to "open up" our load-dependent system, two of which yield equivalent results. For the first way, merely let $\beta_2(n) = 1$ for all n. Then, if $\lambda/\mu(N)$ is less than 1 for large N, S_2 is a Poisson source to S_1 and we have the standard M/M/C queue when $\beta_1(n)$ satisfies (2.1.13b). That is, from Equations (2.1.14),

$$K := \lim_{N\to\infty} K(N) = \sum_{n=0}^{\infty} \frac{\rho^n}{\beta_1(n)} \tag{2.1.15b}$$

and

$$r(n) := \lim_{N\to\infty} r(n;N) = \frac{1}{K} \frac{\rho^n}{\beta_1(n)}. \tag{2.1.15c}$$

Actually, one can make a somewhat more general statement. If

$$\lambda_\infty := \lim_{N\to\infty} \lambda(N)$$

exists and $\lambda_\infty/\mu(N)$ is less than 1 for large N, everything still holds except that now $\rho = \lambda_\infty/\mu$.

A second approach is to argue that $\lambda(n)$ is really a function of N and n by way of their difference, $N - n$. That is, let

$$\bar{\lambda}(n) := \lim_{N\to\infty} \lambda(N-n)$$

and

$$K = \sum_{n=0}^{\infty} \frac{\rho^n}{\beta_1(n)\bar{\beta}_2(n)},$$

where $\bar{\alpha}_2(n) := \bar{\lambda}(n)/\bar{\lambda}(1)$, $\bar{\beta}_2(0) := 1$, and

$$\bar{\beta}_2(n) := \bar{\alpha}_2(n)\,\bar{\beta}_2(n-1).$$

The $\bar{\alpha}_2$s can be interpreted as a slowdown of the arrival process because of the increasing queue length, so this is referred to as an M/M/C queue with ***discouraged arrivals***. This may be a misnomer in some countries where consumer goods are scarce. In those places, we are told, arrival rates to queues actually increase with queue length. Mathematically, because K in this case is not a convolution, β_1 and $\bar{\beta}_2$ can be combined into a single load-dependent factor. However, for more general queues (e.g., M/G/C and G/M/C) the two must still be kept separate. The third view, which ends up being the same as the first, considers all customers, while they are at S_2, to act independently. That is, each customer spends a random amount of time at S_2, with mean Z, and then, independently of the other customers, goes to S_1. The completion rate is exactly $(N-n)/Z$. Z is called the ***think time***, or ***delay time***, and S_2 is called a ***think stage*** or ***time-sharing stage*** or ***delay stage***, as well as some other names. Clearly, as N goes to infinity, the arrival rate grows unboundedly, thereby swamping S_1. In reality, there never are an infinite number of potential customers, but there may be so many and they may stay at S_2 so long that n (the number at S_1) is always small compared to N, so the departure rate from S_2 is more or less constant. In mathematical terms, let Z grow unboundedly with N, and let

$$\lambda_\infty = \lim_{N\to\infty} \frac{N}{Z}\cdot$$

This yields the same solution as case 1.

In all these cases we can make a statement that generalizes (2.1.6c). Let μ_∞ be the limiting value of $\mu(N)$; then

$$\lim_{N\to\infty} \Lambda(N) = \min(\mu_\infty, \lambda_\infty)\,. \tag{2.1.16}$$

Once again, the throughput of the system is bounded by the maximal capacity of its slowest server.

Example 2.1.3: The simplest example of a load-dependent queue is the M/M/2 queue. In this case, $\beta_1(n) = 2^{n-1}$, $\bar{\beta}_2(n) = 1$,

$$r(0) = \frac{2-\rho}{2+\rho},$$

and

$$r(n) = \frac{2}{K}\left(\frac{\rho}{2}\right)^n \quad \text{for } n > 0,$$

where

$$K = 2\,\frac{2+\rho}{2-\rho}\cdot$$

We leave it for the reader in Exercise 2.1.8 below to show that

$$\bar{q}_s = \frac{4\rho}{4-\rho^2}, \quad \text{and} \quad \bar{T}_s = \frac{4\bar{x}}{4-\rho^2}\cdot$$

Note that the queue doesn't blow up until ρ approaches 2. ▲

Finally, let us consider our open M/M/C queue, and let C go to infinity. Then S_1 is a place where customers arrive randomly, "hang around" for a while, $[1/\mu]$, and then leave. The number present at any time is distributed according to the Poisson distribution. Because $\beta_1(n) = n!$,

$$K = \sum_{n=0}^{\infty} \frac{\rho^n}{n!} = e^{\rho},$$

leading to

$$r(n) = \frac{\rho^n}{n!} e^{-\rho}. \tag{2.1.17}$$

This is just one of the many derivations of the Poisson distribution that start from different assumptions.

Observe that all the formulas are valid whether or not $\alpha(n)$ and $\mu(n)$ satisfy Equations (2.1.11). If they do, we retain the notation "M/M/C//N loop," including the system with a time-sharing subsystem, for which we use the notation "M/M/∞//N" or "M/M/C//C." If we wish to look at systems in which the α's are not necessarily integers but instead satisfy a weakened version of Equations (2.1.11), namely, for $n \leq C$, $\alpha_1(n) =$ anything > 0, but

$$\alpha_1(n) = \alpha_1(C) \quad \text{for } n \geq C,$$

then we would refer to it as a ***generalized M/M/C//N loop***. If the α's can be anything whatsoever, we use the notation, "M/M/X/ /N *loop*." To maintain a connection with the outside literature, we refer to all of these generically as "M/M/*C-type systems*," or, *systems with load-dependent servers*." We also adhere to this notation in dealing with more general distributions in Sections 4.4.4 and 5.4, and Chapter 6 (e.g., G/M/X and M/G/C queues). In Chapter 6, we also introduce the generalized M/G/C system.

Exercise 2.1.8: Consider systems (A) through (D) as described below. What are the formulas for their respective system times? Call them T_A, T_B, T_C, and T_D, respectively. Assume that the service rate for the base server is $\mu = 1$. Plot the four system times on the same graph as a function of $\lambda = \rho$, for $0 \leq \lambda < 2$. Of course, T_A blows up at $\lambda = 1$, but the other three have the same maximal capacity, and blow up at $\lambda = 2$.

Even the simple M/M/1 queue can have realistic practical applications. We present one in the following exercises. In most facilities it is generally true that the demand for a critical resource will always increase in time. This can be viewed in our simple world as an arrival rate λ that increases monotonically with time. Inevitably then, the system time, as given by (2.1.7b) will become intolerably long. Call this *System* (A). This leads to two questions: How can the service be improved? And for what value of λ should the improvement be implemented? We consider two possible changes for improvement: either add a second server, or replace the existing one by another that is faster. For simple analysis we assume that the new server is twice as fast. In the latter case, this is still an M/M/1 queue where $\mu \Rightarrow 2\mu$. Call this *System* (D). In the former case consider two possible implementations. Arriving customers come to a dispatching point, and are then randomly assigned (with equal probability) to either of the two servers, where they then queue for service. It can be shown that this is equivalent to having each server see a Poisson arrival stream, but with arrival rate $\lambda/2$. This yields two M/M/1 queues. Call this *System* (B). Finally, for *System* (C), customers queue up at the dispatching point, and are assigned to a processor as soon as it becomes idle. This is the M/M/2 described in this section. In Chapter 5 we present another dispatching option.

Exercise 2.1.9: Using the results of the previous exercise, show that $T_A > T_B > T_C > T_D$ for all $0 < \lambda < 2$. In fact, show that:

$$T_B = 2\,T_D \quad \text{and} \quad T_C = T_D + \frac{1}{2+\lambda}.$$

We seem to have shown that "twice as fast is always better than twice as much," but remember, we have only shown this for Poisson arrivals to exponential servers. In Chapter 6 we show that if the squared coefficient of variation, C_v^2 is large, this is not necessarily the case.

We have seen *how* a system might be improved, and now we look at the question as to *when* it would be cost effective to do so.

Exercise 2.1.10: Suppose that one single-speed server costs C dollars per hour to rent, and that each customer is paid S dollars per hour. Assume that when a customer is waiting for, or receiving service, his time is being wasted. Then at all times, on average, there are $\bar{q}_s$ customers wasting their time. The total cost then can be given as

$$\$_A = C + S\,\bar{q}_A(\lambda) \quad \text{and} \quad \$_I = 2\,C + S\,\bar{q}_I(\lambda),$$

where $I \in \{B,\, C,\, D\}$. We are assuming that the double-fast server costs as much as two single servers. Clearly, for λ very small, $\$_A$ is smaller than the other three, and it doesn't pay to upgrade. But $\$_A$ blows up at $\lambda = 1$ whereas the others don't blow up until $\lambda = 2$. Therefore, the curves must cross somewhere for $0 < \lambda < 1$. This must be true for any values of C and S. In fact, the crossing point depends only on their ratio, $r = S/C$. Make a graph of the four $\$_I$s for $0 \le \lambda < 1$ for $r = 0.1,\ 1.0,\ 10.0$, showing the crossings in each case. What are the values of λ_I at those points? Now draw three curves on the same graph of λ_I versus r.

2.1.6 Departure Process

Let us now consider one last steady-state process before moving on to the transient behavior of the M/M/1 queue. Suppose that an observer is sitting just downstream from S_1, measuring the time between departures, without knowing the state of the system. What would she expect to see? In other words, given that a customer has just left, what is the time until the next one leaves S_1? We are asking for the distribution of *interdeparture times.* First we give some appropriate definitions.

Definition 2.1.5

*$X_d(N)$:=r.v. denoting the time between departures for a steady-state M/M/1//N queue (**interdeparture times**).*

$$X_d := \lim_{N\to\infty} X_d(N)$$

$b_d(t;\, N) := b_{X_d(N)}(t)$ = density function for the process. □

This question was originally considered by P. J. Burke [**?**] and is easy enough to find out once we accept a theorem about M/M/1 queues that is be proven in Section 4.1.3, Theorem 4.1.4. This theorem states that for both open and closed M/M/1 queues, and more generally, M/G/1 (but *not* G/M/1) queues, the steady-state probability that a departing customer will leave n fellow customers behind at S_1 is the same as the steady-state probability of finding n there, except that he will never leave N customers behind, because he, at least, must be at S_2. Let $d(n;N)$ be this probability; then from (2.1.4b) we can write

$$d(n;N) = \frac{\rho^n}{c(N)}, \tag{2.1.18}$$

where $c(N)$ is found by summing over n, from 0 to $N-1$. Thus $c(N) = K(N-1)$ from (2.1.4c).

Now, as long as S_1 is busy, the density function for the departure of the next customer is simply the same as the pdf of S_1 (i.e., $\mu e^{-\mu\, t}$). But if S_1 is idle, our downstream observer must wait first for a customer to finish being served at S_2 and then be processed by S_1. This is the ***convolution*** of the two pdfs:

$$[b_1 \times b_2](t) := \int_{\mathrm{o}}^{t} b_1(s) b_2(t-s) ds = \int_{\mathrm{o}}^{t} b_1(t-s) b_2(s) ds,$$

which for two exponential distributions yields

$$\begin{aligned}[b_1 \times b_2](t) :&= \int_{\mathrm{o}}^{t} \mu e^{-\mu s} \lambda e^{-\lambda(t-s)} ds = \mu\lambda e^{-\lambda\, t} \int_{\mathrm{o}}^{t} e^{-(\mu-\lambda)s} ds \\ &= \frac{\mu\lambda}{\mu-\lambda}\left(e^{-\lambda\, t} - e^{-\mu\, t}\right).\end{aligned}$$

The overall distribution is the weighted average of the two possibilities. Recall that $\rho = \lambda/\mu$; then

$$\begin{aligned}b_d(t;N) &= d(0:N)[b_1 \times b_2](t) + [1 - d(0;N)]\mu e^{-\mu\, t} \\ &= \frac{1-\rho}{1-\rho^N}\frac{\lambda}{1-\rho}\left(e^{-\lambda\, t} - e^{-\mu\, t}\right) + \left(1 - \frac{1-\rho}{1-\rho^N}\right)\mu e^{-\mu\, t}.\end{aligned}$$

We can regroup the terms to get the following simple form.

$$b_d(t:N) = \frac{1}{1-\rho^N}\lambda e^{-\lambda\, t} - \frac{\rho^N}{1-\rho^N}\mu e^{-\mu\, t}. \tag{2.1.19a}$$

For the closed loop, the departure process is *not* a Poisson process, because the interdeparture times are not exponentially distributed. For the open queue, where $\rho < 1$ and $N \to \infty$, $b_d(t)$ *is* exponential. The mean time between departures is easy enough to get:

$$\mathbb{E}[X_d(N)] = \int_{\mathrm{o}}^{\infty} t\, b_d(t:N)\, dt = \frac{1-\rho^{N+1}}{1-\rho^N}\frac{1}{\lambda}. \tag{2.1.19b}$$

We leave it to the following exercise to show that $\mathbb{E}[X_d(N)]$ is the reciprocal of the mean throughput given by (2.1.5d).

Exercise 2.1.11: Verify that (2.1.19b) is true, and show that $\mathbb{E}[X_d(N)] = 1/\Lambda(N)$.

Either from (2.1.19b) or from (2.1.6c), we have

$$\lim_{N\to\infty} \mathbb{E}[X_d(N)] = \max\left(\frac{1}{\lambda}, \frac{1}{\mu}\right). \tag{2.1.20a}$$

For the open queue, if ρ is less than 1, $(\lambda < \mu)$, the mean departure rate from S_1 is the same as the mean arrival rate. But if ρ is greater than 1, the mean departure rate is governed by the service rate of S_1. We can now prove the well-known result, first given by P. J. Burke in 1956, that the departures from an open M/M/1 queue are exponentially distributed. Simply let N go to infinity on (2.1.19a),

$$\begin{aligned} b_d(t) := \lim_{N\to\infty} b_d(t;\, N) &= \lambda\, e^{-\lambda\, t} \quad \text{for } \rho < 1 \\ &= \mu\, e^{-\mu x} \quad \text{for } \rho > 1. \end{aligned} \tag{2.1.20b}$$

As long as ρ is less than 1, it is as though S_1 did not exist (exponential in → exponential out). We also see once again that S_2, with its unbounded number of customers, is a Poisson source for S_1. But if ρ is greater than 1, S_1 releases customers at its service rate and becomes a Poisson source for S_2. The symmetry of our loop would require this, anyway.

We must emphasize that this result (exponential in → exponential out, for an open, unsaturated M/M/1 queue) is indeed extraordinary. It is also valid for load-dependent (i.e., M/M/C) queues. Note however that it is not true for first-come first-served M/G/1 queues or even G/M/1 queues. It is not even true for closed M/M/1//N loops. We must be careful not to generalize too quickly from what we learn about the M/M/1 queue.

2.2 Relaxation Time for M/M/1/ /*N* Loops

For the rest of this chapter we examine systems for which not enough time has elapsed to declare that a system is in its steady state. We call this time range the ***transient region***. In principle we would like to solve the Chapman-Kolmogorov equations (1.3.2b), but in practice, if N is large, this is not an easy task. Aside from the M/M/1 queue, there are very few known analytic solutions to this equation. A rather ingenious solution for the open M/M/1 queue, where N is infinite, is given in [**?**]. That may well be the only explicit solution for an infinite state-space, transient queueing system in existence. But even the existence of that solution does not help much, because it is so difficult to evaluate or interpret.* Therefore we must find some simpler ways of parameterizing transient behavior. For our initial view, we remind the reader of the discussion about relaxation times in Section 1.3.3 and (1.3.11).

In general, finding the eigenvalues of a matrix is not a trivial task, particularly if one wants to express them in terms of unspecified parameters rather than numerically. If the dimension of the matrix is small enough, as with (2.1.2) and (2.1.3), the eigenvalues can be found by straightforward, if tedious, methods. In the case of our $\boldsymbol{Q}$, one of the eigenvalues is zero, thus the characteristic equation can be written as degree N rather than $N+1$, the size of $\boldsymbol{Q}$. It is well known that no general formula (such as the quadratic equation) exists for the roots of polynomials of degree greater than four, nor can one ever be found. (If you have ever used the cubic or quartic formulas to get analytic expressions, you might be inclined to say that even four is too big.) Therefore, unless one is “lucky” (as with the zero eigenvalue), the task is hopeless for $N > 4$.

By a fortuitous stroke of good fortune, because the $\boldsymbol{Q}$ of (2.1.1c) is so repetitive, $\phi_N(\beta) = |\boldsymbol{Q} - \beta \boldsymbol{I}|$ satisfies a recurrence relation in N which turns out to be similar to that satisfied by Chebyshev polynomials of the second kind, from which all the eigenvalues can be obtained. The details can be found in [**?**]. As always, $\beta_o = 0$, and

$$\beta_k = \mu + \lambda + 2\sqrt{\mu\,\lambda}\,\cos\frac{k\pi}{N+1} \qquad \text{for } k = 1, 2, 3, \ldots, N. \tag{2.2.1a}$$

The smallest β is β_N, which therefore must be $1/RT$. As in Exercise 2.1.1, it is convenient to express the relaxation time in units of the time it takes a lone customer to make one cycle $(1/\mu + 1/\lambda)$. Then,

*Takacs actually supplies two different forms for the solution, neither of which is easy to evaluate. Most texts list the second form, which involves an infinite sum of Bessel functions, but the first form turns out to be more useful (particularly in the region where the time parameter is neither very small nor very large) if one is comfortable with numerical integration.

recalling that $\rho = \lambda/\mu$, and $\cos[\pi N/(N+1)] = -\cos[\pi/(N+1)]$, we get the following expression for the ***normalized relaxation time***.

$$T(\rho,\, N) := \frac{\mu\lambda}{\mu+\lambda} RT = \frac{\rho}{(1+\rho)} \left(1 + \rho - 2\sqrt{\rho}\,\cos\frac{\pi}{N+1}\right)^{-1}. \tag{2.2.1b}$$

T is invariant to the replacement of ρ with $1/\rho$; that is, $T(\rho, N) = T(1/\rho, N)$.

Next, we look at $T(\rho, N)$ when N is very large. For $\rho \neq 1$, $T(\rho, N)$ has a finite limit as N goes to infinity. Thus the relaxation time for an open system (normalized so that $1/\mu + 1/\lambda = 1$) is

$$T(\rho) \;:= \lim_{N\to\infty} T(\rho, N) = \frac{\rho}{(1+\rho)(1-\sqrt{\rho})^2} = T(1/\rho). \tag{2.2.2a}$$

It is not hard to show that $T(\rho)$ approaches $0.5/(1-\rho)^2$ when ρ is close to 1 [**?**]. As so often happens, $\rho = 1$ must be treated as a special case. We can either set $\rho = 1$, or let $N \to \infty$, but not both at the same time. $T(1, N)$ goes to infinity as $\mathrm{O}(N^2)$. We show this by setting ρ equal to 1 in (2.2.1b) to get

$$T(1, N) = \frac{1}{2}\left(2 - 2\cos\frac{\pi}{N+1}\right)^{-1} = \frac{1}{4}\left(1 - \cos\frac{\pi}{N+1}\right)^{-1},$$

and then use Maclaurin's expansion for $\cos x$ [$\cos x = 1 - x^2/2 + \mathrm{O}(x^4)$]:

$$\begin{aligned} T(1, N) &= \frac{1}{4}\left[\frac{1}{2}\left(\frac{\pi}{N+1}\right)^2 + \mathrm{O}\left(\frac{1}{N^4}\right)\right]^{-1} \\ &= \frac{1}{2}\left(\frac{N+1}{\pi}\right)^2 \left[1 + \mathrm{O}\left(\frac{1}{N^2}\right)\right]. \end{aligned} \tag{2.2.2b}$$

Naturally, the relaxation time for an open system ($N = \infty$) is infinite when $\rho = 1$. That is, the system never reaches a steady state.

Example 2.2.1: Figure 2.2.1 summarizes what we have said about relaxation times. What is most important is to observe that as systems get bigger (in this case, N larger) and more saturated (ρ close to 1), the time it takes to approach the steady-state solution grows as well. This puts into question the steady-state solution as a description of systems that are in existence for relatively short times. ▲

Example 2.2.2: Figure 2.2.2 presents the same information in a different way. Now N varies for fixed $\rho = 0.5, 0.9, 0.95$, and 1. As with the throughput curves, $T(\rho, N) = T(1/\rho, N)$, so $\rho = 2$ yields the same curve as $\rho = 1/2$. As $N \to \infty$, each curve approaches its limit as given by (2.2.2a), except, of course, for $\rho = 1$, which has no limit. ▲

Clearly, if ρ is close to 1, the relaxation time can be very large. However, if ρ is very small (or very large), $T(\rho, N)$ is small. This may be an underestimate of how long it takes a system to come close to its steady state. If all customers are initially at the slower server, very few completions would have to occur to approach the steady state, because very few customers are ever likely to be at the faster server at any one time. Even so, the mean time for one slow server completion (in units of the cycle time) is $1/(1+\rho)$, which (for small ρ) is $1/\rho$ times larger than $T(\rho, N)$. On the other hand, if all the customers are initially at the faster server, the steady state cannot be approached until almost all of them have been served at least once. The mean time for this is of the order of $\rho N/(1+\rho)$. The two conditions together imply that

$$0 \leq RT \leq \frac{N}{\rho} T(\rho, N), \quad \text{for } \rho < 1. \tag{2.2.2c}$$

RT could be 0 if the system were initially in its steady state, which means that all queue lengths are possible from the beginning (i.e., we do not know anything).

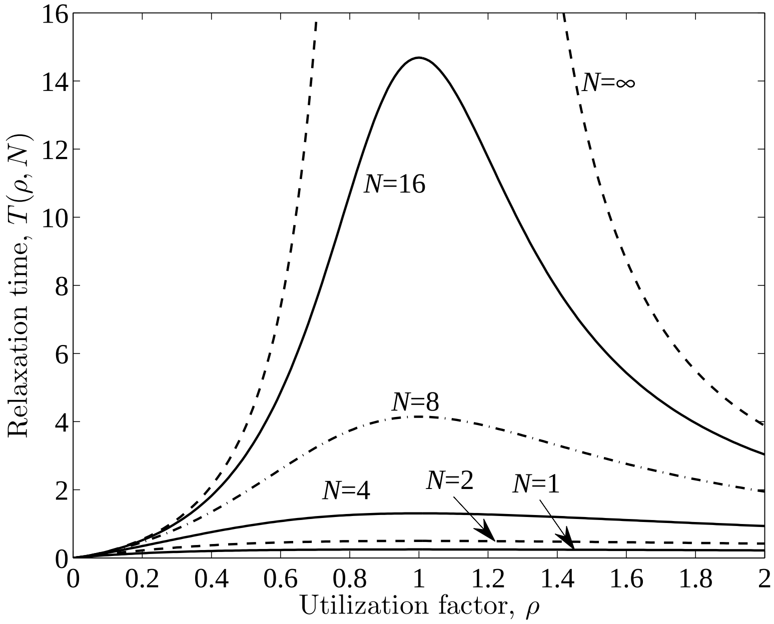

Figure 2.2.1: Relaxation time as a function of ρ for M/M/1/ /N queues, as given by (2.2.1b). $T(\rho, N)$ is in units of cycle time for one customer. All curves peak at $\rho = 1$, whereas at $\rho = 1$, $T(1, N)$ goes to infinity as N becomes increasingly large. For all values of ρ, the relaxation time increases with N.

2.3 Other Transient Parameters

In this section we introduce alternative ways (other than RT) of examining the transient region. We are pleased to find that some of the objects we needed for the steady-state solution are also used here. As with every Markov chain, only one thing at a time can happen in a queueing network; the evolution of the system in time is marked by a discrete sequence of events. We call the interval after one event up to and including the next event an ***epoch***. This deviates from conventional use. Feller [?] prefers to use *epoch* to mean the time the event occured (not the interval). Sometimes, the time between events is called the ***sojourn time***.

Such sequences, or epochs, can be represented by *time-dependent state transition diagrams*. The technique described here is easily generalized to include nonexponential and even more general service centers, and that is done in succeeding chapters.

2.3.1 Mean First-Passage Times for Queue Growth

As a first application, we examine the time it takes for a queue to grow from 0 to some integer n. Such processes are referred to as *first passages*, and the average times for such events to take place are called ***mean first-passage times***, or simply ***first-passage times***. The points at which a Markov chain reaches each length for the first time are called ***ladder points***. All the things that happen from the time the queue reaches j to the time it reaches $j+1$ is said to have "occurred during the j-th epoch."

Looking at Figure 2.1.1, suppose that initially all the customers are at S_2; then in mean time $1/\lambda$ the first event occurs, corresponding to an arrival to S_1 (epoch 0 has ended). After that, one of two events can occur: either the customer at S_1 returns to S_2, or another customer from S_2 goes to S_1. The sequence of possible events grows factorially after that, and it becomes thoroughly impractical to enumerate all of them. However, if in any sequence the system returns to a state it was in previously, a recursive relation

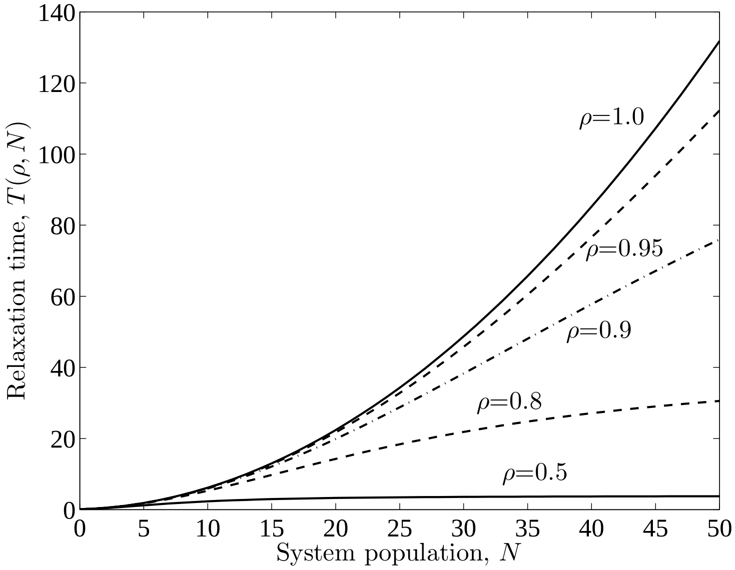

Figure 2.2.2: Relaxation times as a function of system population N for M/M/1/ /N loops. The RT's for ρ and $1/\rho$ are identical; therefore, we only show curves for $\rho \leq 1$. As $N \to \infty$, all curves except that for $\rho = 1$ will saturate.

can be set up that may be solvable. This is known as a ***regenerative process*** [**?**]. We show how this works in this section and use it frequently in subsequent chapters. To apply this method, one must start with single jumps. So we define

Definition 2.3.1

$\tau_u(n)$:= ***mean first-passage time*** *for the queue at* S_1 *to go from* n *to* $n+1$. The n-th epoch begins with n customers at S_1. Customers may leave and arrive in arbitrary order, but eventually there will be $n+1$ customers at S_1 for the first time (end of epoch n and beginning of epoch $(n+1)$. The mean time for this to happen is $\tau_u(n)$. The subscript ***u*** stands for ***u***p. (In subsequent sections we will have occasion to use ***d*** for ***d***own and ***m*** for ***m***ax.) □

Consider Figure 2.3.1. The circles on the lowest horizontal line correspond to the set of states the system can be in initially, which in the present case is labeled by the number of customers at S_1. The second horizontal line represents the state the system is in after one transition. The average time elapsed between the two lines depends on the initial state. Thus if the system started with all customers at S_2 [$n = 0$], the mean time for the first transition would be $1/\lambda$. Similarly, if all customers were initially at $S_1[n = N]$, the average time elapsed would be $1/\mu$. For all other initial states, the time would be $1/(\mu+\lambda)$. A straight arrow corresponds to a single direct transition, with the probability that it will occur written near it. For instance, the system can go from $n \to n+1$ in one step, with probability $\lambda/(\mu+\lambda)$, with a mean time delay of $1/(\mu+\lambda)$. A wavy arrow corresponds to the sum of all possible ways the system can get from the tail to the head for the first time, irrespective of the number of transitions taken. Thus the arrow labeled "$\tau_u(n)$" includes not only the direct transition ($n \to n+1$), but also ($n \to n-1 \to n \to n+1$), and ($n \to n-1 \to n-2 \to n-1 \to n \to n-1 \to n \to n+1$) and the infinite number of other sequences that eventually lead to $n+1$.

Our ability to represent an infinite number of sequences by a single symbol is the key to setting up a soluble set of recursive relations. If the system starts with n at S_1, an event will occur in mean time $1/(\mu+\lambda)$. That event can be one of two things. Either the queue will go directly to $n+1$, or it will drop to $n-1$, in which case it will take time $\tau_u(n-1)$ to get back to n, and a further $\tau_u(n)$ to finally get to

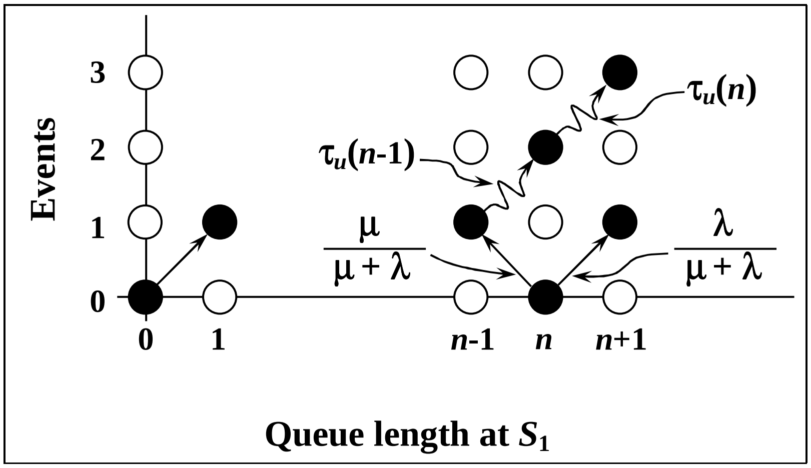

Figure 2.3.1: Time-dependent state transition diagram for a closed M/M/1/ /N loop, describing the mean time $[\tau_u(n)]$ for a queue to grow by one customer. See text for details.

$n+1$. Mathematically we can write

$$\tau_u(n) = \frac{\lambda}{\mu+\lambda} \cdot \frac{1}{\mu+\lambda} + \frac{\mu}{\mu+\lambda}\left[\frac{1}{\mu+\lambda} + \tau_u(n-1) + \tau_u(n)\right],$$

where $\tau_u(0) = 1/\lambda$. For convenience, drop the subscript u when no confusion is likely to arise. The two terms without a τ in them combine to yield the following.

$$\tau(n) = \frac{1}{\mu+\lambda} + \frac{\mu}{\mu+\lambda}[\tau(n-1) + \tau(n)]. \tag{2.3.1a}$$

We interpret this as follows. It takes a mean time of $1/(\mu+\lambda)$ for something to happen. If the event was an arrival, we are done. The probability that it was not an arrival is $\mu/(\mu+\lambda)$, in which case the queue will have dropped back to $n-1$ and take a mean time of $[\tau(n-1)+\tau(n)]$ to first get back to n and then to $n+1$. Note that $\tau(n)$ appears on both sides of the equation, indicating that the system got back to where it started, and that is what we mean by a regenerative process. We derive equations for more complicated processes in just this way, so the reader should expect to return to this section for reference.

We next solve for $\tau(n)$ and get the following recursive equation.

$$\tau(n) = \frac{1}{\lambda}[1 + \mu\tau(n-1)], \quad \text{for } n > 0, \tag{2.3.1b}$$

and $\tau(0) = 1/\lambda$. By direct substitution into (2.3.1b) it follows that $\tau(1) = (1+1/\rho)/\lambda$ and $\tau(2) = (1+1/\rho+1/\rho^2)/\lambda$. One can guess that the general expression for $\tau(n)$ is

$$\tau(n) = \frac{1}{\lambda}\sum_{j=o}^{n} \frac{1}{\rho^j}, \tag{2.3.2a}$$

which can be proven by induction to be the solution of (2.3.1b). [†]. Equation (2.3.2a) is the well-known

[†] We interject a word or two about "guessing." If science were merely a sequence of deductions, we all would have already been replaced by computers. Research is a creative process. The imaginative scientist, mathematician, or engineer plays with the tools of the trade and regularly makes guesses at what is correct. (These guesses are often credited to intuition.) Most guesses that prove wrong never come to public light. You, the reader, only see the successes and thus may think that there is some secret process going on to which you will never be privy. Nonsense. The creative person who plays long enough with the relevant material will ultimately make many correct guesses. Remember, proof by induction does not require that we defend the source of the guess. It must only prove that the guess is correct (if it is).

(certainly by now) geometric series for which a closed-form expression exists.

$$\tau(n) = \frac{1}{\lambda}\sum_{j=0}^{n}\frac{1}{\rho^j} = \frac{1/\mu}{1-\rho}\left(\frac{1}{\rho^{n+1}} - 1\right) \quad \text{for } \rho \neq 1 \tag{2.3.2b}$$

and

$$\tau(n) = \frac{n+1}{\mu} \quad \text{for } \rho = 1. \tag{2.3.2c}$$

We are now ready to find the time it takes for a queue to grow to a given length.

Definition 2.3.2

$t_u(0 \to n)$:= *mean first-passage time for the queue at* S_1 *to grow from* 0 *to* n *customers.* The queue could drop to 0 many times before finally reaching the goal. □

This parameter satisfies the following.

$$t_u(0 \to n) = \sum_{j=0}^{n-1}\tau_u(j). \tag{2.3.3a}$$

After substituting Equations (2.3.2) into the above (and omitting the subscript u), the explicit expressions follow.

$$t(0 \to n) = \frac{1/\mu}{1-\rho}\left(\frac{1}{\rho^n}\frac{1-\rho^n}{1-\rho} - n\right) \quad \text{for } \rho \neq 1 \tag{2.3.3b}$$

and

$$t(0 \to n) = \frac{n(n+1)}{2\mu} \quad \text{for } \rho = 1. \tag{2.3.3c}$$

Equations (2.3.3) can be thought of as the mean rate at which a queue grows in time. For instance, we see from (2.3.3c) that for $\rho = 1$ and large n, $t(0 \to n)$ grows as n^2. We can get a different insight to this process by thinking of $t(0 \to n)$ as the independent variable. Then we see that n grows as the square root of t. This is quite similar to behavior of a ***random walk*** process, and is in fact a special type of random walk with a barrier. [**?**] considers such processes to be ***renewal processes***.

For $\rho < 1$, (2.3.3b) implies that $\mu\, t(0 \to n)$ approaches $(1/\rho)^n/(1-\rho)^2$ as n gets increasingly large. Considering n as the dependent variable, it follows that n grows as the $\log t$. This is indeed an extremely slow growth rate, for although all queue lengths are possible, when ρ is less than 1, long queue lengths take exponential time to be reached even once.

Finally, for $\rho > 1$, (2.3.3b) implies that $t(0 \to n)$ and n grow proportionally. This actually makes intuitive sense, whereas the two previous examples are a consequence of statistical fluctuations. Clearly, the arrival rate exceeds the service rate, so with every passing unit of time, customers who have yet to be served accumulate at S_1 in proportion to the difference between the arrival and service rates, namely $\mu(\rho - 1)$. Examples for all three cases are shown in Figure 2.3.2. Asymptotic behavior can be summarized by the following equations.

$$n(t) \to \frac{\log(\mu\, t)}{\log(1/\rho)} \quad \text{for } \rho < 1, \tag{2.3.4a}$$

$$n(t) \to \sqrt{2\mu\, t} \quad \text{for } \rho = 1, \tag{2.3.4b}$$

$$n(t) \to \mu\, t(\rho - 1) + \frac{1}{\rho - 1} \quad \text{for } \rho > 1. \tag{2.3.4c}$$

These three asymptotic forms are quite different, yet if ρ is close to 1, $\mu\, t$ must be rather large before the three will look considerably different.

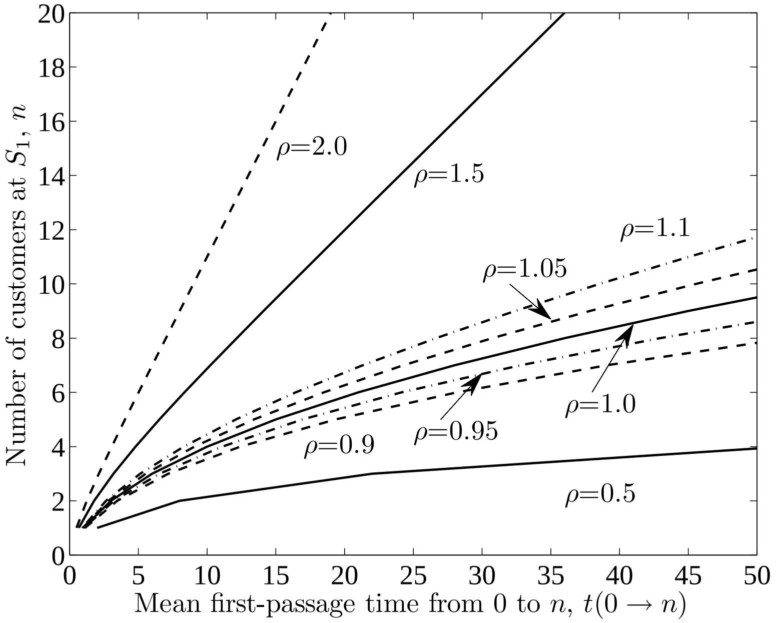

Figure 2.3.2: Number of customers versus mean first-passage time for the queue at S_1 to grow from 0 to n, $t(0 \rightarrow n)$, as given by Equations (2.3.3). Equations (2.3.4) show that when $\rho < 1$, n grows as $\log t$, but when $\rho > 1$, n grows linearly with t. Yet t must be very large for this behavior to become apparent if ρ is close to 1.

Example 2.3.1: It can be seen from Figure 2.3.2 that the closer ρ is to 1, the larger μt will be before (2.3.4a) or (2.3.4c) deviate from (2.3.4b). An interesting consequence of this is the following. In taking data of such a system (or an ensemble of such systems), an observer cannot measure very accurately what ρ is, without waiting an extremely long time. Also, note that even after 50 cycle times, the queue has not come anywhere near its steady-state mean queue length for $\rho > 0.9$. ▲

Exercise 2.3.1: An interesting variation of $t(0 \rightarrow n)$ is to find the mean number of arrivals before the queue reaches its steady-state mean queue length for the first time. Here ρ must be less than 1, and $\lambda t(0 \rightarrow n)$ is that quantity, for any n. Let n be $\bar{q}$ from (2.1.6b) and draw a curve of $\lambda t(0 \rightarrow \bar{q})$ versus ρ, for ρ between 0 and 1. How do these results compare with Figures 2.2.2 and 2.3.2?

2.3.2 *k*-Busy Period

A much-used view of queueing systems that does not require waiting for the steady-state is the ***busy period***. By definition, a busy period begins when a customer arrives at an empty subsystem and ends when a customer leaves behind an empty subsystem. Put differently, the busy period is the interval between idle periods. In general, one can imagine starting with k customers at S_1 and then have customers come and go until, eventually, the queue drains. This is known as the ***k-busy period***, with $k = 1$ being simply the busy period. A good insight into system behavior can often be gained by taking data over several busy periods, and comparing with analytical results. Unlike the steady state, each period has a well-defined beginning and end.

2.3.2.1 Mean Time of a Busy Period

The first parameter we consider is the mean time for the busy period. This can be calculated in a manner very similar to the preceding section. Whereas in that section we were interested in queue growth, here we are interested in queue-length reduction. We use the same symbols as before [τ and $t(0 \to n)$, etc.], and when a distinction between the two types is necessary, we use subscripts u for "up" and d for "down". Otherwise, the subscripts are omitted.

In analogy with Section 2.3.1, with the apparent added restriction that the queue never exceeds N, define the following.

Definition 2.3.3

$\tau_d(n; N)$:= ***mean first-passage time*** *for the queue at* S_1 *to* ***drop*** *from* n *to* $n-1$, *in an* M/M/1/ /N *loop*. Given that there are only N customers in the system, the queue can never exceed N. The process begins with n customers at S_1 and ends when the queue reaches $n-1$ for the first time and could have risen to N any number of times in that period of time. □

This actually is exactly analogous to Definition 2.3.1, because $\tau_u(n)$ includes the self-evident constraint that the queue can never drop below 0.

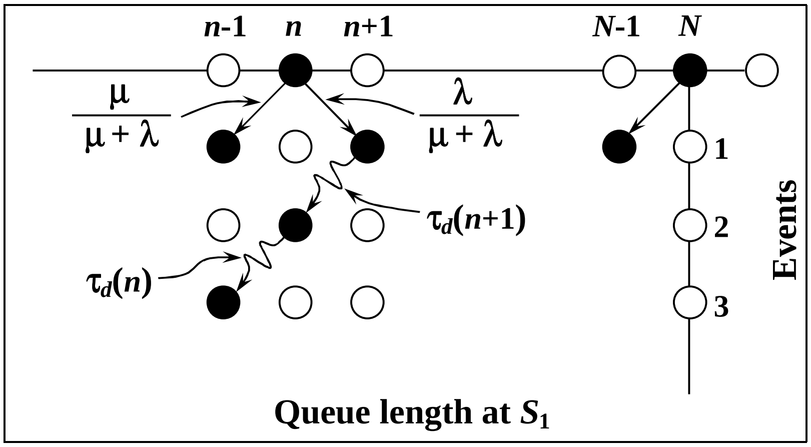

Figure 2.3.3: Time-dependent state transition diagram for a closed M/M/1/ /N loop describing the mean time [$\tau_d(n)$] for a queue to decrease by 1 customer. $\tau_d(1)$ is the mean busy period. See text for full details.

Figure 2.3.3 is similar to Figure 2.3.1, but now the τ_d-s are pointing toward lower lengths. As before, in mean time $1/(\mu+\lambda)$, something happens, and if that something is not a departure, then with probability $\lambda/(\mu+\lambda)$ it is an arrival that raises the queue length by 1, after which it will drift back down to n in time $\tau_d(n+1; N)$, and finally, to $n-1$ in further time, $\tau_d(n; N)$. This leads to (dropping the subscripts d)

$$\tau(n; N) = \frac{1}{\mu+\lambda} + \frac{\lambda}{\mu+\lambda}[\tau(n+1; N) + \tau(n; N)], \tag{2.3.5a}$$

where $\tau(N; N) = 1/\mu$. Making the substitution $\rho = \lambda/\mu$ and the usual rearrangements, we get

$$\mu\tau(n; N) = 1 + \rho\mu\tau(n+1; N). \tag{2.3.5b}$$

Directly substituting into (2.3.5b) for $n = N-1$ and $N-2$, it follows that $\mu\tau(N-1; N) = 1+\rho$, and $\mu\tau(N-2; N) = 1+\rho+\rho^2$. One can easily guess, and prove by induction, that

$$\tau(N-k;\, N) = \frac{1}{\mu}\sum_{i=0}^{k}\rho^i = \frac{1-\rho^{k+1}}{\mu(1-\rho)} \quad \text{for } \rho \neq 1 \tag{2.3.6a}$$

and

$$\tau(N-k;\, N) = \frac{k+1}{\mu} \quad \text{for } \rho = 1, \tag{2.3.6b}$$

where $k = N - n$. It is clear that when $\rho \geq 1$, τ_d grows unboundedly with N (and k), but when $\rho < 1$, then

$$\tau_d(n) := \lim_{N\to\infty} \tau_d(n;\, N) = \frac{1/\mu}{(1-\rho)}\cdot \tag{2.3.6c}$$

We see then, that for an open system, the mean time for a queue to drop by 1 is the same for all n, a result that some might call obvious.

By definition, the mean time for a busy (1-busy) period is the same as the mean time to eventually go from $n = 1$ to $n = 0$. The *k-busy time* is defined as follows.

Definition 2.3.4

$t_d(k \to 0;\, N) :=$ *the mean time for the* ***k-busy period*** *of an* M/M/1/ /N *loop*. The process begins with k customers at S_1, and ends when there are 0 customers there for the first time. □

First we have

$$t_d(1 \to 0;\, N) = \tau(1;\, N) = \frac{1-\rho^N}{\mu(1-\rho)} \quad \text{for } \rho \neq 1 \tag{2.3.7a}$$

and

$$t_d(1 \to 0;\, N) = \frac{N}{\mu} \quad \text{for } \rho = 1. \tag{2.3.7b}$$

As with the τ_ds, when $\rho \geq 1$, the mean extent of the busy period grows unboundedly with N, but when $\rho < 1$, the limit for $t_d(1 \to 0;\, N)$ exists and approaches [the same as (2.3.6c)]

$$t_d := t_d(1 \to 0) = t_d(1) = \frac{1/\mu}{(1-\rho)}. \tag{2.3.7c}$$

This expression looks familiar. It tells us that the mean busy period for an open M/M/1 queue is the same as its mean system time as given by (2.1.7b). Actually, (2.3.7c) gives the mean time of a busy period for all open M/G/1 queues (but not G/M/1 queues), whereas the expression for the mean system time for M/G/1 queues [see (4.2.6e) and (4.2.6f)], is more complicated.

An expression for t_d can be derived in the following way. Any single server queue (open or closed) will alternate between busy and idle periods. Let T_i and X_i be the lengths of the i-th busy and idle periods, respectively. Then

$$R_b(m) := \frac{\sum_{i=1}^m T_i}{\sum_{i=1}^m (T_i + X_i)} \tag{2.3.7d}$$

is the fraction of time S_1 is busy during the first m cycles. As m gets very large, $(\sum T_i/m)$ approaches t_d, $(\sum X_i)/m$ approaches the mean idle time (call it t_I), and R_b approaches $1 - r(0,\, N) = \Pr(S_1$ is busy). When we put this all together, we get

$$t_d = t_I \frac{1 - r(0,\, N)}{r(0,\, N)}\cdot \tag{2.3.7e}$$

For every open single-server queue ($N \to \infty$), $r(0,\, N) \to 1 - \rho$, and for Poisson arrivals, $t_I = 1/\lambda$. All this yields (2.3.7c).

In direct analogy with Equations (2.3.3) we see that the mean time for the k-busy period is

$$t_d(k \to 0; N) = \sum_{j=1}^{k} \tau_d(j; N) = \frac{1/\mu}{1-\rho} \sum_{j=1}^{k} (1 - \rho^{N-j+1}),$$

which after some straightforward manipulation yields

$$\mu\, t_d(k \to 0; N) = \frac{k}{1-\rho} - \frac{\rho^{N-k+1}}{(1-\rho)^2} + \frac{\rho^{N+1}}{(1-\rho)^2} \quad \text{for } \rho \neq 1 \tag{2.3.8a}$$

and

$$\mu\, t_d(k \to 0; N) = kN - \frac{k(k-1)}{2} \quad \text{for } \rho = 1. \tag{2.3.8b}$$

As with the τ_ds for open systems, the k-busy period is infinite when $\rho \geq 1$, but when $\rho < 1$,

$$\mu\, t_d(k \to 0) = \frac{k}{1-\rho}. \tag{2.3.8c}$$

This makes sense, because it takes a time $1/[\mu\,(1-\rho)]$ [or what is the same thing, $\rho/[\lambda(1-\rho)]$] for an open queue to drop by 1, so if there were k customers to start with, it should take k times $\lambda\rho/(1-\rho)$ to drop to 0.

2.3.2.2 Probability That Queue Will Reach Length *k*

Although the time for a busy period may be important, it is by no means the only parameter worth examining. From an experimental point of view, it is easy to measure, for instance, the number of busy periods in which a given queue length was reached or the maximum queue length reached. It is desirable, therefore, to be able to compute these quantities as well.

By now we should be getting pretty good at working with time-dependent state transition diagrams. Unfortunately we now have a new complication. All objects we looked at previously in this section were certain to happen. The busy period was certain to end (if $\rho \leq 1$), and all queue lengths will occur eventually. But now we have to worry whether a busy period will end before reaching a given queue length. Such processes are known as ***taboo processes*** (it is taboo - or tabu - to reach that given length) which we now define.

Definition 2.3.5

Let Ξ be the set of all possible states of a system. Let Ξ_1 and Ξ_2 be disjoint proper subsets of Ξ. That is, $\Xi_1 \cap \Xi_2 = \emptyset$ (empty). Also, let $\Xi_1 \cup \Xi_2 \subset \Xi$ (proper subset). That is, $\Xi_3 := \Xi - [\Xi_1 \cup \Xi_2] \neq \emptyset$ (not empty). In other words, Ξ_1, Ξ_2, and Ξ_3 form a ***partition*** of Ξ (every $s \in \Xi$ is in one, and only one of the Ξ_is). A ***taboo process*** is one that starts in some state $s_i \in \Xi_3$, and ends when the system finds itself in some state $s_f \in \Xi_1 \cup \Xi_2$. The process succeeded if $s_f \in \Xi_1$, and failed if $s_f \in \Xi_2$ (the taboo states). We are usually interested in $\mathbf{Pr}(s_f \in \Xi_1 \,|\, s_i \in \Xi_3)$ (i.e., the probability that the outcome was *good*). If Ξ_2 is empty then $\mathbf{Pr}(\cdot) = 1$, unless there is no way to get from s_i to Ξ_1, in which case $\mathbf{Pr}(\cdot) = \infty$, because by our definition, the process never ends. □

The next processes are examples of taboo processses.

The procedure for calculating probabilities for queue changes is similar to that for calculating the mean time for the change to occur. First we must calculate the probabilities for one step at a time, and then take the product of the probabilities (note that we take the sum of the step times) for the complete process. First define the following.

Definition 2.3.6

$W_u(n) :=$ *probability that the queue at* S_1 *will go from* n *to* $n+1$ *during a busy period* (i.e., without going to 0). The process begins with n customers at S_1, and ends when the queue (including the active customer) either reaches $n+1$ or 0. The queue can fall and rise any number of times before the process ends.

This is a taboo process where $\Xi = \{s \,|\, 0 \leq s < \infty\}$, $\Xi_1 = \{s \,|\, s > n\}$, $\Xi_2 = \{0\}$ and $\Xi_3 = \{s \,|\, 0 < s \leq n\}$. The process starts with $s_i = n \in \Xi_3$, and ends when $s_f = n+1 \in \Xi_1$ (good) or when $s_f = 0 \in \Xi_2$ (bad). So $W_u(n) = \mathbf{Pr}(s_f \in \Xi_1 \,|\, s_i = n)$. The reader should decide if the taboo concept is helpful for understanding particular processes. □

The queue either goes up [with probability $\lambda/(\mu+\lambda)$], or goes down [$\mu/(\mu+\lambda)$], in which case it must eventually get back to n without first going to 0 [$W_u(n-1)$], and then get to $n+1$, [$W_u(n)$, another regenerative process]. The equation describing this is

$$W_u(n) = \frac{\lambda}{\mu+\lambda} + \frac{\mu}{\mu+\lambda}[W_u(n-1)W_u(n)]. \tag{2.3.9a}$$

This reorganizes to

$$W_u(n) = \rho[1+\rho - W_u(n-1)]^{-1}, \tag{2.3.9b}$$

where $W_u(1) = \lambda/(\mu+\lambda) = \rho/(1+\rho)$. Our “great” experience with these things allows us to guess and prove by induction, with $K(0)=1$, that

$$W_u(n) = \rho\frac{K(n-1)}{K(n)}, \tag{2.3.9c}$$

where $K(n)$ was defined in (2.1.4c) and satisfies the recursive and explicit formulas

$$K(n) = \sum_{j=0}^{n} \rho^j = 1 + \rho K(n-1) = \frac{1-\rho^{n+1}}{1-\rho} \quad \text{for } \rho \neq 1 \tag{2.3.10a}$$

and

$$K(n) = n+1 \quad \text{for } \rho = 1. \tag{2.3.10b}$$

We will not always be so fortunate to find explicit expressions for more complicated queues.

As the final effort of this section, we calculate the probability that the queue will get at least to k during a busy period. This is the same as the following.

Definition 2.3.7

$W_u(1 \to k) :=$ *probability that the queue at* S_1 *will go from* 1 *to* k *before going to* 0. The process begins with one customer at S_1 and ends when the queue (including the active customer) reaches either k or 0. This is another taboo process. □

Then $W_u(1\to 1) = 1$, and for $k > 1$,

$$W_u(1\to k) = \prod_{n=1}^{k-1} W_u(n) := W_u(1)W_u(2)\cdots W_u(k-1), \tag{2.3.11a}$$

which due to (2.3.9c) gives us

$$W_u(1\to k) = \frac{\rho}{K(1)}\rho\frac{K(1)}{K(2)}\rho\frac{K(2)}{K(3)}\cdots\rho\frac{K(k-2)}{K(k-1)}\cdot$$

As long as ρ does not equal 1, this conveniently simplifies to

$$W_u(1\to k) = \frac{\rho^{k-1}}{K(k-1)} = \frac{(1-\rho)\rho^{k-1}}{1-\rho^k}\cdot \tag{2.3.11b}$$

For $\rho = 1$ we get the much simpler expression

$$W_u(1\to k) = \frac{1}{k} \quad (\rho = 1). \tag{2.3.11c}$$

Note that (2.3.11a, b, and c) are valid for any customer population as long as $k \leq N$. Thus they are valid for open systems as well. Observe that as might be expected if $\rho \leq 1$, then $W_u(1\to k)$ approaches 0 as k gets increasingly large. However, if $\rho > 1$, then

$$\lim_{k\to\infty} W_u(1\to k) = \lim_{k\to\infty} \frac{(1-\rho)\rho^{k-1}}{1-\rho^k} = 1 - \frac{1}{\rho}\cdot \tag{2.3.11d}$$

In other words, for an open system with $\rho > 1$, the probability that the queue will grow to infinity without the busy period ever ending is $1 - 1/\rho$. That is, the probability that a busy period will end is $1/\rho$. A process that is not guaranteed to end is sometimes referred to as having a ***defective probability distribution*** [**?**]. When $\rho = 1$, we have the interesting apparent contradiction that each busy period will surely end $[1 - W_u(1 \to \infty) = 1]$, but on average it will take an infinite amount of time to do so.

2.3.2.3 Maximum Queue Length During a Busy Period

The last property that we study in this chapter is the probability that S_1's maximum queue length in a busy period will be k. Call this $W_m(k; N)$, where N is the total number of customers in the system. To evaluate this, we not only use the W_u's of the preceding section, but we also evaluate the probabilities of coming down without ever exceeding $k < N$. So, define the following.

Definition 2.3.8

$W_d(n, k; N)$ = *probability that the queue at* S_1 *will go from* n *to* $n - 1$ *without exceeding* k*, where* $N \geq k \geq n > 0$. The process begins with n customers at S_1 and ends when the queue either reaches $n-1$ or $k+1$. Put differently, $W_d(n, k; N)$ is also the probability that the queue will reach $n-1$ before going to $k+1$. For $k = N$, then, $W_d(n, N; N) = 1$, because it is certain that the queue will eventually drop by 1 from any n. This is yet another taboo process, where $\Xi_1 = \{j \,|\, j < n\}$, $\Xi_2 = \{j \,|\, k < j \leq N\}$, and $\Xi_3 = \{j \,|\, n \leq j \leq k\}$. □

Next we recognize that for $k < N$,

$$W_d(k, k; N) = \frac{\mu}{\mu + \lambda} = \frac{1}{1 + \rho}\cdot \tag{2.3.12a}$$

For $n < k$, the recursive formulas are exactly analogous to (2.3.9), namely

$$W_d(n, k; N) = \frac{\mu}{\mu + \lambda} + \frac{\lambda}{\mu + \lambda}[W_d(n + 1, k; N)W_d(n, k; N)],$$

which leads to

$$W_d(n, k; N) = [1 + \rho - \rho W_d(n + 1, k; N)]^{-1}. \tag{2.3.12b}$$

The usual guess and proof by induction gives us an explicit expression for $W_d(n, k; N)$:

$$W_d(n, k; N) = \frac{K(k - n)}{K(k - n + 1)} \quad \text{for } k < N. \tag{2.3.12c}$$

Notice that this expression is independent of N, as long as $k < N$. For $k = N$ it is clear that $W_d(N, N; N) = 1$, because the queue cannot grow beyond N. It follows from (2.3.12b) that if $W_d(n + 1, N; N) = 1$, then $W_d(n, N; N)$ must also equal 1. Therefore,

$$W_d(n, N; N) = 1 \quad \text{for } 1 \leq n \leq N. \tag{2.3.12d}$$

This merely states the obvious, that a closed system will experience every queue length with certainty (not once, but over and over), and of course, irrespective of what ρ is. It is nice to know that our mathematics sometimes produces the expected. Remember, though, that (2.3.12d) is not necessarily true of open systems.

Exercise 2.3.2: Given Equations (2.3.10) and (2.3.12a), prove by induction that (2.3.12c) is the unique solution of (2.3.12b).

Our next task is to calculate the object in the following definition.

Definition 2.3.9

$W_d(k \to 0; N)$:= *probability that the queue at* S_1 *will drop from* $k \to 0$ *without ever exceeding* k*, in an* M/M/1//N *loop*. The process begins with k customers at S_1, and ends when it reaches either $k+1$ or 0. □

This must be the product of the probabilities of events cascading downward one stepat a time. Therefore, given that $K(0) = 1$, this is

$$W_d(k \to 0; N) = \prod_{n=1}^{k} W_d(n, k; N) = \frac{K(k-1)}{K(k)} \frac{K(k-2)}{K(k-1)} \cdots \frac{K(1)}{K(2)} \frac{K(0)}{K(1)}.$$

All but one of the terms cancel, leaving us with the simple formula

$$W_d(k \to 0; N) = \frac{1}{K(k)} = \frac{1-\rho}{1-\rho^{k+1}}, \tag{2.3.13a}$$

for $k = 1, 2, 3, \cdots, N-1$, with

$$W_d(N \to 0; N) = 1. \tag{2.3.13b}$$

This last equation must be true. Because it is impossible for the queue to exceed N, it must drain eventually.

Our final exercise is to calculate the probability described in this section's title. Clearly, this is equal to the probability that the queue at S_1 will reach k $[W_u(1 \to k)]$ and then drop to 0 without ever exceeding k $[W_d(k \to 0; N)]$. Therefore, we define for the M/M/1//N queue as follows.

Definition 2.3.10

$W_m(k; N)$:= *probability that the queue at* S_1 *will reach a maximum of* k *during a busy period for an* M/M/1//N *queue*. The process begins with 1 customer at S_1, and ends when there are either $k+1$ or 0 customers there. The process is a success only if it ends with 0 customers, and the queue reaches k at least once during the interval. □

This turns out to be

$$\begin{aligned} W_m(k; N) &= W_u(1 \to k) W_d(k \to 0; N) \\ &= \frac{\rho^{k-1}}{K(k-1)} \frac{1}{K(k)} \quad \text{for } 1 \leq k < N \end{aligned} \tag{2.3.14a}$$

and

$$W_m(N; N) = W_u(1 \to N) = \frac{\rho^{N-1}}{K(N-1)}. \tag{2.3.14b}$$

Note that $W_m(k, N)$ does not depend on N as long as $k < N$; thus we can write that

$$W_m(k; N) = W_m(k; \infty) \quad \text{for } k < N.$$

The queue at S_1 must grow to some maximum length during a busy period, therefore it must follow that

$$\sum_{k=1}^{N} W_m(k; N) = 1. \tag{2.3.15}$$

This is shown to be true by recognizing that because $K(n) = 1 + \rho K(n-1)$,

$$W_m(k;N) = \frac{\rho^{k-1}}{K(k-1)K(k)} = \frac{\rho^{k-1}}{K(k-1)} - \frac{\rho^k}{K(k)}. \tag{2.3.16a}$$

Clearly, in validating that (2.3.16a) satisfies (2.3.15), the negative term of $W_m(k;N)$ exactly cancels the positive term of $W_m(k+1;N)$, and given that $W_m(N;N)$ has only a positive term, all terms cancel except the positive part of $W_m(1;N)$, which is $\rho^o/K(0) = 1$.

Equation (2.3.16a) tells us something else, which we should have suspected in the first place. Notice from (2.3.11b) that

$$W_m(k;N) = W_u(1 \to k) - W_u(1 \to k+1), \tag{2.3.16b}$$

but still, it is nice to know that we have derived it.

Example 2.3.2: As our truly final example for this chapter, we observe how $W_m(k;N)$ behaves when both k and N are very large. This is shown in Figure 2.3.4 for $N = 10$ and various values of ρ. Clearly, when $\rho < 1$, W_m goes to 0 as ρ^k. That is, the probability of reaching long queues becomes highly unlikely. Now, if $\rho = 1$, then $W_m(k;N) = 1/k(k+1)$ for $k < N$ and $W_m(N;N) = 1/N$. Thus very large queue lengths can be expected during a busy period, in fact, so large that it may take forever for some busy periods to end. ▲

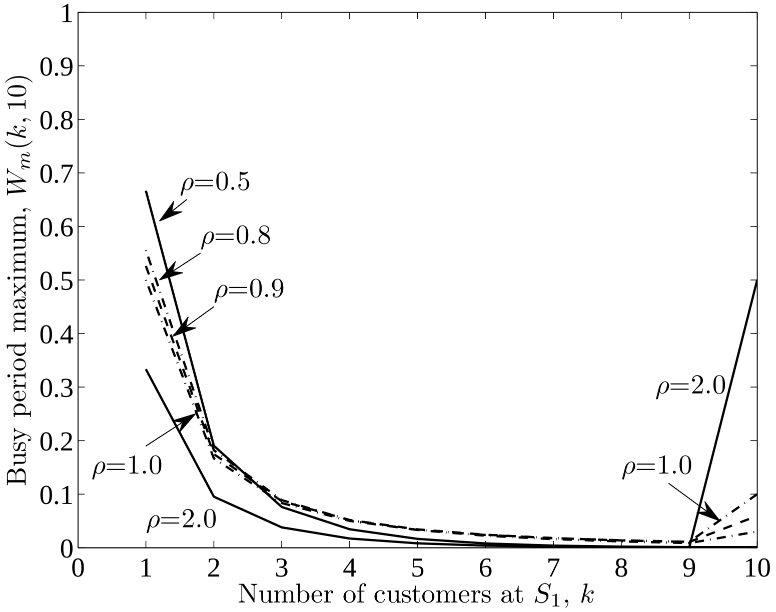

Figure 2.3.4: Probability $W_m(k;10)$ that the queue at S_1 will reach a maximum of k during a busy period of an M/M/1//10 loop. Curves for ρ = 0.5, 0.8, 0.9, 1.0, and 2.0 are displayed. All the curves decrease for increasing k, except at $k = 10$. Given that $W_m(k;10) = W_m(k;\infty)$ for all $k < 10$, $W_m(10;10)$ corresponds to the probability that the open queue will exceed a length 9 during a busy period. At $k = 1, W_m(1;10)$ decreases with ρ, but at $k = 10$, the reverse is true.

Exercise 2.3.3: Evaluate $W_m(k;\infty)$ and $W_m(N;N)$ for all k for $N = 5$ and 20, and $\rho = 0.1$, 0.5, 0.9, 1, 1.1, and 2. Make sure that your numbers satisfy (2.3.15). How do your numbers compare with Figure 2.3.4?

Perhaps the most interesting results for maximum queue length occur for $\rho > 1$. In this case $W_m(k;N)$ goes to 0 as $1/\rho^k$, just as it does for $\rho < 1$. But $W_m(N;N)$ approaches the finite limit, $1 - 1/\rho$. This, of course, is the probability that the busy period will never end in an open system. For those busy periods that do end (the probability of which is $1/\rho$), $W_m(k;\infty)$ is still the correct probability that k will be the maximum queue length.