QUEUES / Causes and Cures

by Edward L. Fritz

At one time or another, all of us have stood in a queue. A queue is a waiting line. We have waited in supermarket lines, at the bank, at the Post Office, at a toll booth, or in a restaurant. Frequently we may have thought that somebody should do something to cut down our delays.

Some of us might even be aware of the vast body of literature that exists on the subject of queuing, telephone companies throughout the world have pioneered in the development of many mathematical models. The high quality of telephone service, which we take for granted in our daily lives, is directly attributable to the proper design of trunk lines based on queuing considerations.

Others have built on this solid foundation. However, most of the writing in technical journals is couched in mathematical formulae that are too abstract for immediate application. This is an attempt to clarify the various concepts and to bring to the surface those elements of queuing theories that can be applied to design problems in computers or computer installations.

Some queuing problems are obvious -- lines of people -- such as the examples given above. Other problems are more subtle -- people need a service but wander off when the facility is busy. Some examples of these subtle queues are: a desk calculator used by many people, an office duplicating or reproduction machine, a key punching machine, a computer where programmers wait for debug runs. In each of these cases, if the facility is busy, a person may while away his waiting time at some other activity until the facility is free.

A third class of waiting problems arises when work itself is waiting to be done by people or machines. For example, paper work at a typist's desk, material waiting to be loaded at a warehouse, programs batched at a computer are all queues. Problems awaiting executive decisions or technical solutions also are sitting in queues.
In every one of these examples, delays give rise to some cost -- either direct or indirect. On the other hand, an attempt to minimize delays also involves a cost. Therefore, a realistic solution should balance the delay cost against the outlay needed for changing procedures or adding facilities. I will attempt to provide here, at least an awareness of the problem, a measure of consequences, and an outline of the alternatives for solution.

Ed Fritz is a Principal Analyst in our Washington Scientific Office. He has had more than 16 years experience in systems analysis, reliability studies and operations research.

From 1950 to 1956, he participated in a series of studies for the Civil Aeronautics Authority. These studies concerned traffic landing patterns, enroute air traffic, and terminal airport communication.

From 1956 to 1963 Ed was a consultant at General Electric, specializing in the area of reliability measurement. He was concerned with investigating and measuring the reliability of many hardware systems and subsystems. He developed new techniques for measuring system reliability and assessing the reliability of complex space vechicles.

From 1963 to 1965 Ed headed the Operations Analysis Department at ITT's Information System Division. There he developed criteria for measuring the reliability of the U. S. Navy's Tactical Data System.

Ed's published papers include "Information Theory in Air Traffic Control"; "Empirical Entropy-A Study of Information Flow," Control Systems Laboratory, University of Illinois; "Transient vs. Steady State Delays"; "Bayesian Reliability Estimates," plus a large number of internal working papers and analysis reports at Franklin Institute and General Electric.

Ed holds graduate degrees in Mathematics and Statistics from the University of Michigan. He joined CUC in 1965 as Technical Director in the Washington Office.

In the computing industry, both the monitor system designers and machine users are becoming aware of the need to understand queues. The former, because multi-programming queues arise at several points in the machine (e.g. disc arm, drum access, printer, etc.) and the latter because (as shall be explained later) the goals of efficient use of computers are in conflict with higher thruput of problems. Indeed, as machines get faster and the number of jobs which are to be processed increases (e.g. Time Sharing), computing systems will more and more have to be designed with user queues in mind - like telephone systems.

State of the Art

Queuing models are concerned with systems where "customers" demand service more or less at random. For example, a computer facility may handle an average of, say, fifty programs a day but the number arriving during any interval of time (9:00 -- 9:30) is known only on a probability basis.

In general the most important characteristics of a system with queues are: the average arrival rate (a) and the average service rate (s). Another vital factor is the number of parallel service channels (c). The ratio $\frac{a}{cs} = u$ is defined as the utilization rate per channel.

A couple of examples should clarify the significance of these terms. Suppose that, on the average, a computer facility receives eight programs an hour and, with one computer, can handle an average of ten per hour. Then the utilization rate is $u = \frac{8}{(1)(10)} = .80$. In other words, the computer will be busy 80% of the time. On the other hand, consider another facility with an input rate of 24 per hour but with three identical machines, each of which can handle ten programs per hour. Once again, the utilization rate per computer is $u = \frac{24}{3(10)} = .80$. Each computer is still busy 80% of the time but, as we shall see shortly, the second system provides a better quality of service.

Although the above parameters -- arrival rate, service rate, and number of channels -- define a particular system, the actual queue size depends on several other considerations. For example, any kind of rational scheduling will reduce waiting time over the case of pure random arrivals. Similarly, the more equal the jobs are in length, the shorter the waiting times. The worst kind of service time distribution is the negative exponential in which there are many short service times but a few long ones. This kind of distribution is typical of telephone calls, service at banks, and at computer installations where all jobs are processed on a first-come-first-served basis.

This last case arises when many short assembly and debugging runs are combined with longer production runs. Another factor in assessing waiting time is queue discipline -- whether items are served on a first-come-first-served basis or if priorities are assigned in one of many ways. Finally, the question of single vs. batch processing modifies the behavior of the queues.

Simplified Design Approach

A number of books and articles have been published on these and other variations. However, the greater the detail we go into when specifying a particular system, the more complex the mathematics. And it becomes harder to evaluate all the alternatives. In this article I have selected one mathematical model to illustrate some of the principles of congestion and queuing. At least it should provide a feel for how waiting lines build up. At best it can serve as a rough design tool for modifying or installing a particular system. The basic techniques can be applied to many uses, including computer networks, time sharing, multi-processing, and multi-programming besides better allocation of peripheral equipment.

Description of Model

The model I have selected has the following characteristics:

1. New jobs arrive at random times.
2. Job lengths are exponentially distributed (many short ones and a few long ones).
3. Priorities are given on a first-come-first-served basis.
4. Each job is completed before the next is started by one machine.
5. Any number of machines can be used to provide parallel service of several jobs simultaneously.
6. There is no balking or reneging if too many jobs are in queue.
7. The total number of customers or users is large (20 or more.)

This model was chosen for two sound reasons. First, it results in the worst delays possible for a given configuration, utilization (u) and channels (c). In a sense this permits the worst case design. The second reason is that this model is the most tractable mathematically. Although other models have been, or can be, solved in principle, the evaluation of the formulae is too complex to be included in this general survey of this field. The problems associated with time sharing and partial processing of many programs simultaneously are particularly interesting. But the results, of course, depend on the particular techniques and priorities indigenous to each machine. There is much to be done in this field.

One further simplification is possible with this worst case approach. There is no need to determine the average arrival rate and the average service rate separately and their

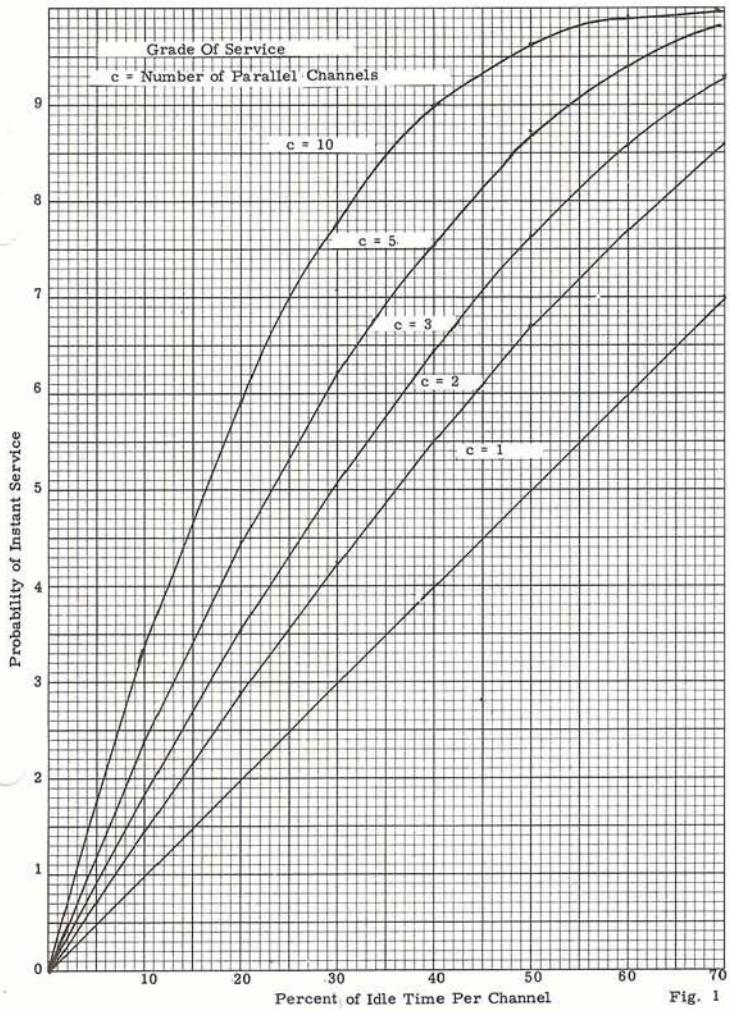

Fig. 1

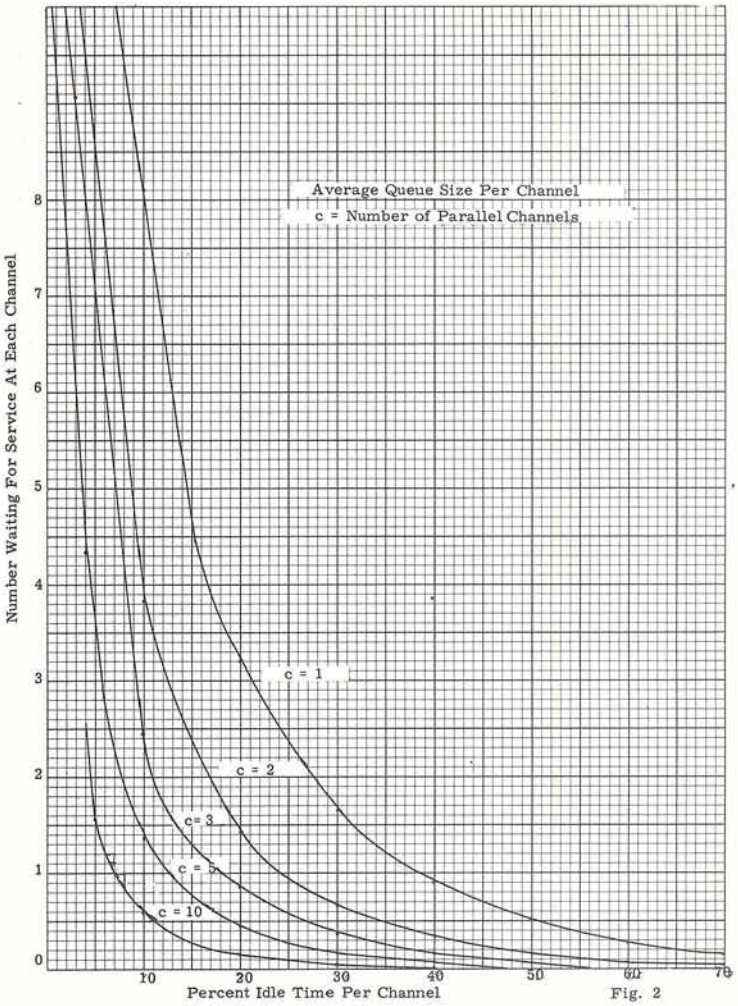

Fig. 2

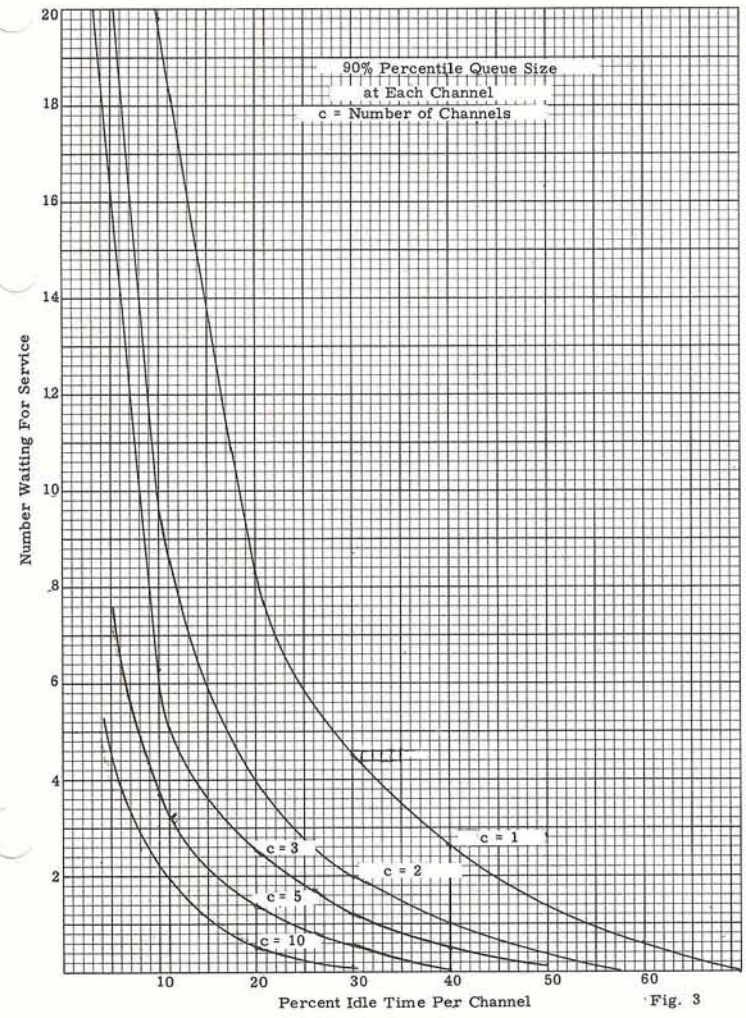

Fig. 3

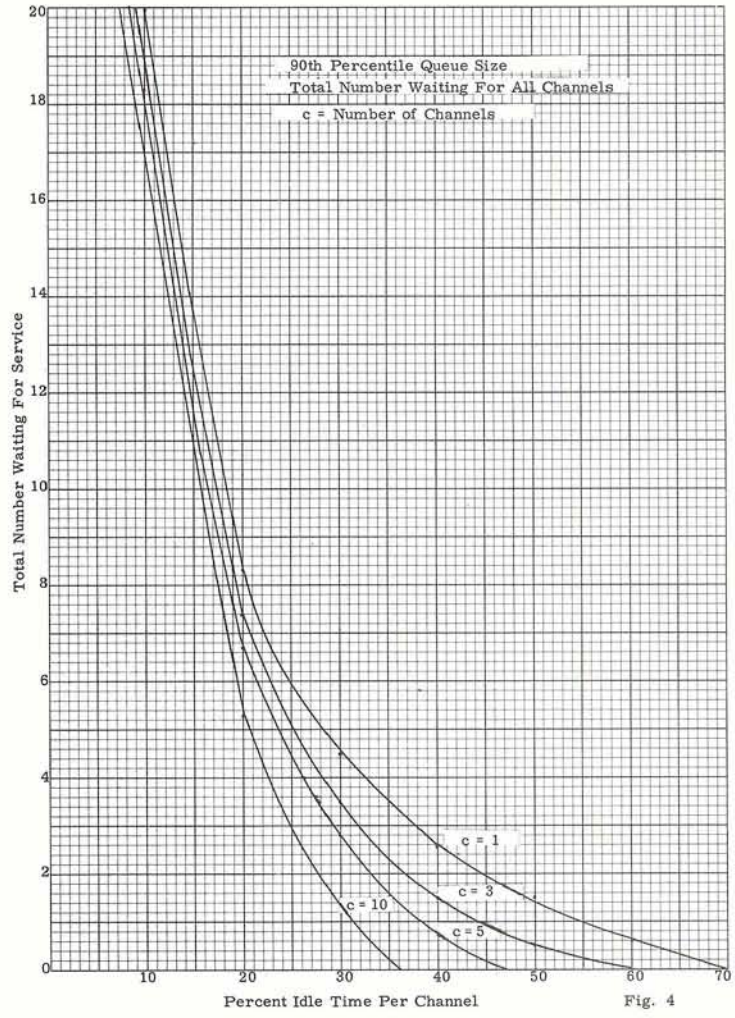

Fig. 4