1. NETWORKS, QUEUES AND PERFORMANCE MODELLING

Chapter Objectives: To discuss the aims and philosophy of the book as a whole, describe the types of networks that are to be modelled, and introduce the concepts of discrete-time queues and their use as a modelling tool.

1.1 INTRODUCTION

The following pages are about statistical prediction of the behaviour of communication networks for computers and other similar sources of digital information.

The typical sorts of predictions that we would like to make include:

- Which network protocol gives the best delay – throughput characteristic under specified conditions?
- What size buffers must be employed by a network's users to keep the probability of buffer overflow below a particular value?
- What is the maximum number of voice calls that can be accepted by a network in order to keep the voice packet transfer delay bounded?
- How many users can a satellite link support and still maintain a reasonable response time?

These and other related questions can be answered by developing a *performance model* for the network in question, and then solving this to obtain *performance measures.*

In this respect, most performance modelling techniques are based on continuous-time concepts, and these can often be difficult to apply to digitised computer communication networks. Such systems lend themselves in a natural way to a discrete-time approach, where the basic time unit can be used to reflect such things as time slotting in the physical system.

The following can therefore be described as an attempt to fill this gap, and present a unified, discrete-time approach for the modelling of computer communication networks and similar systems.

Before looking at specific application areas, a discrete-time theory for modelling and analysing the networks is first developed. This involves a gradual progression by starting with the necessary probability theory, then moving on through discrete-time Markov chains and discrete-time queues, to discrete-time queueing networks. These latter will be the main theoretical modelling tool that is used.

For applications, we concentrate principally on two basic types of networks: Wide area broadcast networks, such as satellite networks or ground-based radio networks, and local area networks. These latter, as their name implies, service a much smaller area than wide area networks. This area might consist typically of a building or college campus.

These two network types typify the modern trends in the subject area, and the modelling techniques used are generally applicable to other network types.

1.2 NETWORK TYPES

We shall consider a communication or computer network to consist of a population of N geographically distributed *users* connected by a communications facility such as a channel (or a number of channels), arranged according to some *network topology*. In this context, the term 'user' denotes an access point to the communications facility. Located here might be, for example, a number of user-terminals, and information processing devices (such as computers) with which the terminals interact. These devices can be accessed by other users by establishing some form of connection via the network. In this way, the entire catchment of information processing devices is made available to all of the network's users. The general arrangement is shown in Fig. 1.1.

The types of network that will be considered are satellite networks and local area networks. As previously mentioned, these two types of networks typify the modern trends in the subject, and provide a sufficient application area to generalise the associated performance modelling techniques.

In the case of satellite networks, the topology is very simple in that we have N geographically distributed users connected by a single channel which forms the only means of communication between them. All information thus has to be exchanged via the channel. The key feature of a satellite network, at least from a performance modelling point of view, is that the channel has a very large (in communication terms) propagation

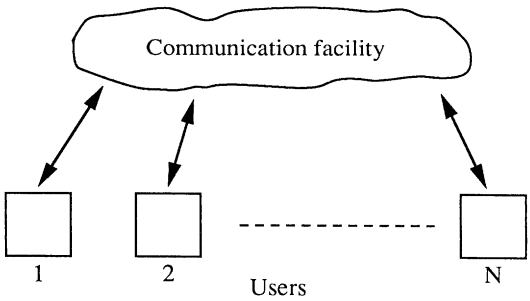

Figure 1.1 Communication or computer network

delay. This is very much in evidence in interactive situations, such as live television interviews via satellite. This delay has to be taken into account in modelling the network and, as will subsequently be seen, can lead to not inconsiderable complications in certain cases.

Local area networks are those that span a geographically small area, such as a building or college campus. We shall consider these to have either a *bus* or *ring* topology. In the case of a bus topology, the N users are tapped at various points onto a length of coaxial cable or other similar bidirectional transmission medium. The system is thus again characterised by a single shared channel, but this time with a very small end-to-end propagation delay due to the small distances involved.

The other major local area network topology is the ring topology, where the N users are connected via interfaces to a unidirectional loop of cable or optical fibre. Again delays are small, but very different operating principles are called for than for the bus topology.

Figure 1.2 summarizes the Topologies for the types of networks to be considered.

1.3 MULTIPLE ACCESS PROTOCOLS

The performance modelling of a network of one of the specified types is inherently concerned with the network's *multiple access protocol.* Multiple access refers to the capability to share a communication facility amongst a population of geographically distributed users. In what follows we shall always consider the communication facility that is to be shared to be the transmission medium (or channel). In this context the multiple access protocol is an algorithm for controlling access to the channel, and to efficiently allocate the channel capacity to the network's users.

In the case of local area networks, the multiple access protocol is also called the *medium access control protocol* [STAL 87], [HAMM 86]. This is usually abbreviated as *MAC protocol.*

A multiple access protocol can be identified as a protocol that overlaps both the Physical Layer and the Data Link Layer of the International Standards Organisation (ISO) Open System Interconnection (OSI) reference model [TANE 88]. In the case of local area networks, a MAC protocol is identified by its own specific layer in the IEEE 802 standards [STAL 87], [HAMM 86].

Figure 1.3 shows a classification of multiple access protocols. The *fixed assignment* methods dedicate a fixed portion of the available channel capacity to each user. The most common forms of this technique are *TDMA* (*Time Division Multiple Access*), *FDMA* (*Frequency Division Multiple Access*), and *CDMA* (*Code Division Multiple Access*).

In *random access* (contention) protocols, the entire bandwidth is presented to the users as a single channel to be accessed randomly. This means that *collisions* of messages can occur, and that colliding messages

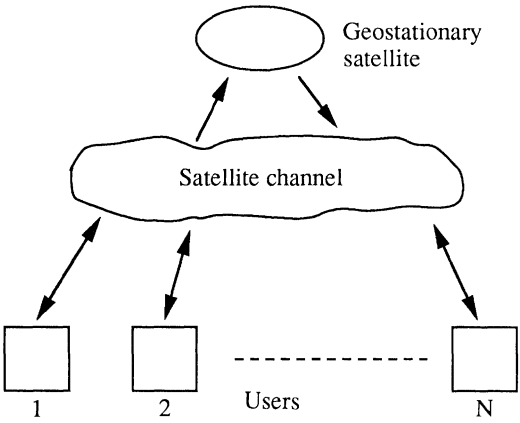

Figure 1.2(a) Satellite

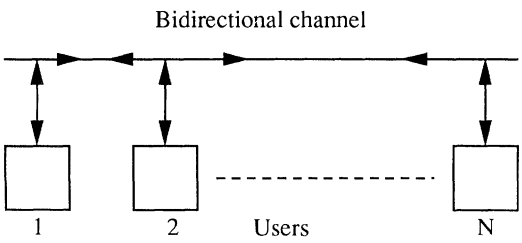

Figure 1.2(b) Bus

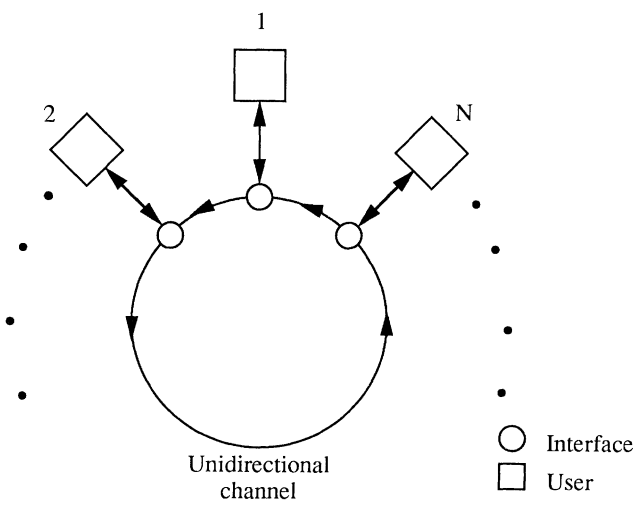

Figure 1.2(c) Ring

Figure 1.2 Network topologies

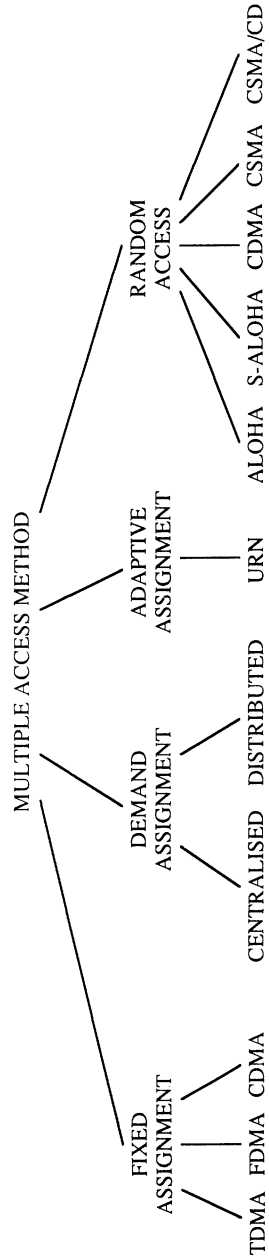

Figure 1.3 Classification of multiple access protocols

must be retransmitted. Random access protocols are mainly concerned with the retransmission scheme used and how the *contention* for the channel can be resolved. Examples of this type protocol are *S-ALOHA* (*Slotted ALOHA*) [ROBE 72], *CSMA* (*Carrier Sense Multiple Access*) and *CSMA/CD* (*Carrier Sense Multiple Access with Collision Detection*) [TOBA 80A].

Demand assignment methods require that explicit control information concerning the users' need for channel capacity is exchanged. The control can be either *centralised* or *distributed*; in a centralised scheme a central controller exists to decide which user should next have the channel access right, whereas with a distributed scheme, each user monitors the requests of all other users, and based on this information executes an identical distributed algorithm to determine who next has access to the channel. In both schemes the control information is exchanged through the channel, which implies additional overhead.

Examples of demand assignment with centralised control are *polling* [MART 72], *probing* [HAYE 78] and *SRMA* (*Split-Channel Reservation Multiple Access*) [TOBA 76B].

Examples of demand assignment with distributed control are the implicit reservation protocol *R-ALOHA* (*Reservation ALOHA*) [CROW 73], and the explicit reservation protocol based in TDMA called *FPODA* (*Fixed Priority Orientated Demand Assignment*) [JACO 78]. Other examples are implicit (virtual) token passing protocols such as the *BRAM* (*Broadcast Recognizing Access Method*) [CHLA 79] and the *slotted ring* [PIER 72], and the explicit token passing protocols such as the *token bus* [STAL 87], and the *token ring* [STAL 87].

The final class of multiple access protocols given in Fig. 1.3 are the *adaptive assignment* protocols. These have not been studied as extensively as the other schemes. Here the access method can change according to the loading offered by the users, the objective being to achieve near optimum performance at all times. The problem here is that the overhead incurred (sometimes a separate signalling channel is required) can often outweigh the advantages. A well known example of an adaptive scheme is the *URN* protocol [KLEI 78], [FUKU 81], so called because its analysis can be reduced to a familiar problem in probability theory in which coloured balls are withdrawn from an urn.

1.4 DISCRETE-TIME QUEUES

One of the major tools that will be used in modelling and analysing the performance of a network is the theory of *discrete-time queues*. The notation that is used to describe such systems is now introduced.

A queueing system can be described in terms of six components;

1. The *arrival process*: This is a stochastic process describing how

customers (customers are usually *messages* in a communications context) arrive into the system.

2. The *service process*: This is a stochastic process describing the length of time that a *server* (servers can be *communication channels* in a communications context) is occupied by a customer; for example, the length of time it takes to transmit a message in a computer communication network.
3. The *number of servers.*
4. The *waiting room*: The limit to the number of customers that can be accommodated to wait for service, including those currently being served.
5. The *customer population*: The limit to the total number of customers, including potential ones. This can be considered as part of the arrival process, since the number of potential customers can change the rate of arrivals.
6. The *service discipline*: The rules for deciding which customer or customers, within a queueing system, to serve.

The standard notation for specifying the first five of the above components is *Kendall's notation* [KEND 53], which consists of a series of letters and numbers separated into fields by a forward slash. In general, this notation can be represented as $A/B/c/n/p$, where A and B describe the arrival and service processes, respectively, c gives the number of servers, n the waiting room, and p the customer population. The final two fields are optional, and if omitted are assumed, by default, to be infinite. Clearly, c, n and p can take on only positive integer values including infinity. The letters which describe the arrival and service processes are chosen from a small set of descriptors. The most common ones are:

D: This stands for *deterministic*, which implies constant interarrival or service times.

M: This stands for *Markovian* (or *memoryless*). In a continuous system, $A = M$ implies that *interarrival times are exponentially distributed*, so that the arrival process is *Poisson*. Similarly, service times are exponentially distributed if $B = M$. In a discrete-time system, $A = M$ implies that *interarrival times are geometrically distributed*. If $B = M$ then service times are also geometrically distributed.

G: This stands for *generally distributed*, which means that no restrictions are placed on the type of distribution that can be used.

Other descriptions are often used in discrete-time systems, such as *Geo* to represent geometrically distributed interarrival times. We shall retain the use of *M* in this book however. Provided that a queue is explicitly described as either continuous or discrete-time, no ambiguity should arise.

A common extension of the above is to attach a superscript to a descriptor to indicate multiple (batch) arrivals. We shall use a_n to denote batch size of the nth arrival, and d_n to denote the batch size of the nth

departure (service completion). In discrete-time systems a_n denotes the number of arrivals in the nth time unit and d_n the number of departures in the nth time unit. If a_n, $d_n \in \{0, 1\}$, $n = 0, 1, 2, \ldots$, then the superscript will be omitted.

To give some examples of the above notation, a continuous-time $M/M/1$ queue has a Poisson arrival process, exponentially distributed service times, a single server, infinite waiting room, and an infinite customer population. A discrete-time $M/M/1$ queue has geometrically distributed interarrival times with either zero or one arrival permitted in a time unit. This implies that the arrival process is a *Bernoulli process*. Also, this queue has a geometric service time distribution, a single server with either zero or one departure in a time unit, infinite waiting room, and an infinite customer population. A discrete-time $M^{a_n}/M/1$ queue is the same as a discrete-time $M/M/1$ queue except that the number of customers arriving in time unit n is a sequence of independent and identically distributed (i.i.d.) random variables $\{a_n, n = 0, 1, 2, \ldots\}$, the distribution of which can be specified. Note that the interarrival times of the batches is still geometrically distributed.

The sixth component describing a queueing system, the service discipline, is not included in the Kendall notation. Typical service disciplines are first-come first-served (FCFS), last-come first-served (LCFS), and processor sharing (PS). The first two have their obvious meanings, while the PS discipline implies that in each time unit a unit of service is distributed equally amongst all the customers in the queue.

Unless otherwise implied, the service discipline will be taken to be FCFS, which perhaps needs some clarification in the context of a discrete-time system when multiple (batch) arrivals can occur in a time unit. The a_n customers arriving in time unit n are assumed to randomly order themselves (within the batch) in serial fashion, and are then queued behind the a_{n-1} customers that arrived in the previous time unit. In this way, when the queueing discipline is FCFS all customers arriving in a given time unit begin service before any customer that arrives in the next time unit.

As a simple example of a discrete-time queue, consider a FCFS discrete-time $M/D/1$ system. This might, for example, be used to model a user's buffer in a computer communication network. According to our notation, the M indicates that this queue has geometrically distributed interarrival times, with, at most, one customer permitted to arrive in a time unit. The D indicates that customer service times are constant, with, at most, one customer permitted to depart in a time unit. Also, the queue has a single server, infinite waiting room, and a potentially infinite customer population.

Let a single new customer arrive in a time unit with probability α, and no new customer arrive with probability $1 - \alpha$. Furthermore, let us assume that the (constant) service time of a customer is one time unit. If we define the state of the system as the number of queued customers left to be served,

then if the queue starts off empty, the only possible states are 0 and 1. Then if x_n describes the state of the queue during time unit $n = 0, 1, 2, \ldots$, we can formally describe the system's behaviour by the sequence of i.i.d. random variables $\{x_n = x,\ n = 0, 1, 2, \ldots\}$, with $x_0 \equiv 0$ the *initial state*, and $x \in \{0, 1\}$ the *state space*.

If the queue is in state 0 it is empty, and makes a transition to state 1 at the next time unit with probability α. When the queue is in state 1, if no arrival occurs then the queue will make the transition to state 0 at the next time unit, and if an arrival occurs the queue will remain in state 1 at the next time unit, since the arriving customer will exactly balance the customer that is served. Note that in the above it is usual to assume that arrivals occur just after the beginning of a time unit, and that departures (service completions) take place just before the end of a time unit.

Even if the system has an initial state that is different from 0, say a long initial queue of customers, with probability 1 it will always eventually return to state 0, and once that state has been reached, the only other state that can be reached is state 1. In this case any state other than 0 or 1 is called *transient* in that the system can only assume such a state once, and can never return to that state once it has left it.

In most instances, we are interested in the long run performance of a system, rather than its transient performance. In other words, we are interested in the behaviour of a system when observed over a very long period of time, and, in particular, the probability of finding a system in a particular state when observed at a random time unit. Such probabilities are the *equilibrium probabilities*. The equilibrium probability of any transient states is clearly 0.

In the discrete-time $M/D/1$ queue, it is rather obvious that the equilibrium probabilities of states 0 and 1 are $1 - \alpha$ and α respectively.

There are, of course, formal ways of calculating equilibrium probabilities, or at least obtaining approximations for them. Such calculations are central to the theme of performance evaluation of systems such as computer communication networks, since most performance measures of interest can be derived directly from the equilibrium probabilities.

Also, it is worth noting that the behaviour of queueing systems can be conveniently represented by *state transition diagrams*. That for the discrete-time $M/D/1$ queue is shown in Fig. 1.4, where the nodes in the

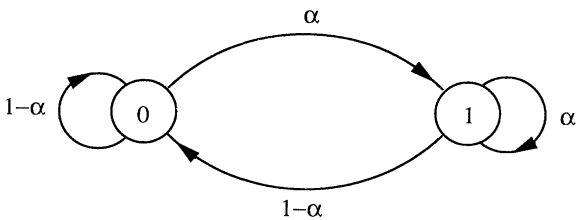

Figure 1.4 State transition diagram for a discrete-time M/D/1 queue

diagram represent states, and the directed arcs between nodes represent possible state transitions, and, as such, are labelled with the *transition probabilities*. This diagram has been derived on the basis of the state being observed just after the arrivals, this convention being somewhat flexible in a discrete-time system.

Had the state been observed at the time unit boundaries (just before the arrivals and just after the departures), which is another possible convention, then state 1 becomes transient in that once the system is in state 0 it can never leave that state. This can be seen in that if an arrival takes place just after the state has been observed as $x_n = 0$, then this customer must depart just before the state is observed at time $n + 1$, resulting in $x_{n+1} = 0$. This is a phenomenon that is unique to discrete-time systems, and illustrates the importance of having a sufficiently small time unit in relation to the interarrival and service times of customers, otherwise information concerning the system's behaviour can be lost. In the limit, when the time unit tends to zero, we have the familiar continuous-time systems.

1.5 PERFORMANCE MEASURES

There are many important measures that can be used for assessing the performance of a communication or computer network. Among these are throughput, average message delay, probability of message loss, reliability, and adaptability to variations in traffic.

The two main performance measures however are *throughput*, defined as the number of successfully transmitted messages per mean transmission time of a message, and the *message delay*, defined as the time interval, in units of the average transmission time of a message, from the moment a message is generated, to the instant it is correctly received. These will be adopted as the key performance measures for all the networks considered. It should be noted that it is sometimes more convenient to express these performance measures in units of *slots* rather than average message transmission times, where a slot is the basic, fixed time unit in terms of which all other times, such as message transmission times, channel propagation delays, and so on, are measured.

The mean values of throughput and message delay will be given the symbols S and W respectively.

A third performance measure that will often be used is the *probability of buffer overflow*, B. This is defined as the fraction of messages that are lost due to no buffer space being available at the time of their arrival. This measure can be important in assessing the transmission quality for certain types of data such as voice or video.

As a simple example of calculating these measures, we can consider the discrete-time $M/D/1$ queue explained in Section 1.4 as a model for a user's buffer in a computer communication network. The mean throughput for

this user will then simply be the fraction of time the user spends transmitting messages. In other words, it will be the fraction of time that the queueing system spends serving customers. Noting that a customer will be served in a time unit (henceforth called a slot) only if the queue is in state 1, then if Π_i is the equilibrium probability that the queue is in state i, clearly the mean throughput is

$$S = \Pi_1 = \alpha \tag{1.1}$$

This is in units of messages per slot, or messages per mean message transmission time, this latter being equal to one slot in this particular case.

The mean message delay can be calculated by first finding the average number of messages stored in the system, and then using the famous *Little's result* [LITT 61]. This result will be considered later, but simply states that the average number of customers stored in a queueing system, L, is equal to the product of the average rate of customer arrivals, λ, and the average time a customer spends in the system, W. This latter, of course, is the mean waiting time or delay through the system. We thus have

$$L = \lambda W \tag{1.2}$$

Clearly, for the discrete-time $M/D/1$ queue we must have

$$L = 0.\Pi_1 + 1.\Pi_1 = \alpha \tag{1.3}$$

and

$$\lambda = \alpha \tag{1.4}$$

This gives the rather trivial result that the mean message delay is equal to 1 slot, which is somewhat obvious from intuition. Also obvious is that the probability of buffer overvlow is zero for this system. This would be true, even if the buffer had a finite capacity.