DEPARTAMENTO DE INFORMÁTICA
E.T.S.E.

Grado de Ingeniería Telemática
Planificación de Redes

Lesson 1
Performance Evaluation

Prof. Juan Manuel Orduña Huertas

Contents

1 INTRODUCTION. OBJECTIVES

Computer networks are becoming a key element for the success of any company of a certain size. The introduction of computers in all areas of daily life has meant that practically every business activity uses computers. The expansion of the internet and the possibilities that the interconnection of computers offer the user have made computer networks a fundamental element in many companies and organisations.

Computer networks are built to support very diverse functions, from text processing or computer-aided design to email or image and video transmission. And the success or failure of the network depends on it being able to support each of these functions in real time. However, the term ‘real time’ depends on each application, and can vary from milliseconds to hours.

It is demonstrated that staff productivity decreases rapidly if the applications they use do not respond quickly. The user feels frustrated, the attitude toward the system deteriorates, and trust in the network decreases. This can slow down business activity. If customers perceive that they are not being served promptly, it can lead to the loss of customers, and so disappears the competitiveness that the network promised. Therefore, we must ensure that the network can respond quickly to the needs of *all* network users.

The benefits that the network offers users must be maintained at all times, and particularly when it is most needed (for example, at the end of the month, at the end of fiscal deadlines, Monday mornings, etc.). These features must also be maintained when new users are added to the network, when a new department is incorporated, when a new application is introduced for all users, or when new printers are installed, for example. Remote users must also receive good response times.

The subject **Network Planning** aims to ensure that the student can measure the performance of an existing network, and based on the analysis of these benefits, design the necessary changes to adapt to a new situation.

The first question that appears when planning a network is: why? What advantages can be obtained from planning or predicting the behaviour of a network? Some of these advantages (although not all) are detailed below:

- Keep response time short. The user is inconvenienced if the server fails or if the network fails. To be able to blame the server, you have to be able to calculate the response time and/or the latency of the network.
- To be able to provide an adequate bandwidth that allows users to increase their productivity. As we have already mentioned, if the user perceives the system as slow, it demotivates and decreases productivity.
- Plan the future growth of the network with security. If we predict the benefits of the network we can modify it knowing that the resulting network will provide the desired services.
- Ensure the implementation of new applications in the existing network.
- Validate response times of new network designs.
- Detect and resolve system bottlenecks.
- Choose safely between different network applications, depending on the requirements of each.
- Choose the most appropriate network technology.

1.1 Types of network performance evaluation

To properly plan networks, it is necessary to evaluate (determine or measure) the performance that these networks will yield. There are different approaches to measure the performance of a network: analysis; simulation; and experimentation or monitoring of the network. Analysis consists of using a mathematical model of the network (more or less precise) to estimate performance. Analysis provides approximate numbers about network performance with minimal effort, and gives indications of how different factors affect performance without the network itself being built. Analysis enables one-time evaluation of whole families of networks with configurations and variant parameters by solving a set of equations that predict the behaviour of the entire family. However, analysis usually involves approximations that can affect the accuracy of the results.

One step beyond analysis to achieve more accurate results is network simulation. This is usually done to validate the results of the analysis, and provides a more accurate estimate of network performance, although it requires more time to generate such an estimate. In addition, each simulation only evaluates a network configuration, a specific traffic pattern, and a point or load state of the network. A simulation also gives a summary of results, without giving direct indications of which factors led to those results. To obtain these indications, the simulator must be correctly instrumented. A simulator is as accurate as the model that simulates. For example, a simulator that properly models the timing and arbitration of a router and channel delays, will provide very accurate results, while a simulator that ignores router timing and arbitration will give inaccurate results.

Finally, once a network is built, experimentation with the network or network monitoring can be used to measure the real network performance and validate simulation models. Of course, at this stage it can be very difficult to make changes in the network if problems are found in the performance.

2 NETWORK PERFORMANCE MEASUREMENTS

There are many different ways to measure and present the benefits of a particular network. Let's first present the basic general definitions, and then we will study the main concepts from a standard network measurement instrumentation scheme.

2.1 General basic measures

The basic performance measures that are common to the three types of network performance evaluation are the following:

Bandwidth: This concept is sometimes called signalling speed, and it is the rate or speed at which the bits can be transmitted by the communication medium. The bandwidth is determined by the physical layer and the link layer of the network infrastructure. Thus, for example, an Ethernet network has a bandwidth of 10 Mbps and a T1 link has a bandwidth of 1.536 Mbps.

Productivity or throughput: Productivity or throughput is the amount of data (free of errors) that a network can transmit in a unit of time (for example, packets per second or *pps*, frames per second or *fps*, bits per second or *bps*). Productivity can be measured for the entire network, for a specific link, or even for a specific session.

Capacity: The capacity of a network, or a link, is the amount of data that the network or link can theoretically serve in a unit of time. Ideally, capacity and productivity are identical, but in reality they are not for a variety of reasons. In theory, productivity should increase as the load (the traffic injected into the network) increases, until reaching the maximum capacity for the network. However, productivity increases linearly, but reaches a peak that is below capacity and then begins to fall, as Graph 2.1 shows. This is basically due to factors such as the inability of intermediate devices to process information fast enough and so causing added delays. If we are expressing productivity in terms of user data (instead of bits per second transmitted), then overheads, or overheads imposed by different protocols, will make productivity less than capacity.

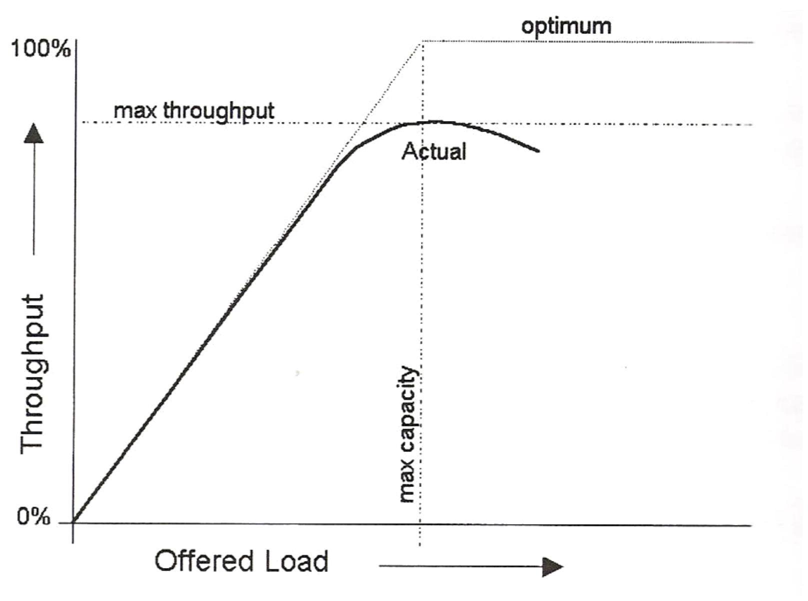


Figure 2.1: Differences between capacity and productivity

Latency: Latency is the cumulative delay or end-to-end delay that a packet experiences when it traverses the network. Each component of the network in the propagation path of the packet will contribute to the total latency. It should be noted that latency and productivity are **NOT** the same. It is preferable to have the same productivity in two different media or networks and with very different latencies. That is, the data can be sent with the same transmission speed, but in one of the media it can arrive later. Bandwidth and latency are combined to determine the total productivity of the communication channel. The network contributes to latency in several ways:

- Propagation delay: the theoretical time it takes for each bit to propagate the entire length of the transmission medium. Typically, the propagation speed (the inverse of this delay) is a value close to the speed of light in a vacuum, so that this delay does not contribute significantly to the latency of the network.

- Transmission delay: The time it takes a packet to traverse a particular medium. This is determined by the speed of propagation of the medium and the size of the package.
- Switching delay: The time it takes the switching device (router, switch, etc.) from when it receives the packet from an input link until it puts the packet on the output link network. This delay will depend on the switching technique used by the network (store & forward, virtual cut-through, wormhole, etc.).
- Processing delay: The time it takes for the switching device to decide what to do with it. This delay includes searches in routing tables, filtering, modification of headers, error detection, jump account increments, encapsulation, etc.

Some applications (such as multimedia) are notoriously sensitive to latency. In other applications, the limit of latency bearable by the user is greater, but in general, it is usually around 100 or 200 ms. Of course, real-time video applications are very sensitive to latency.

Jitter: This is a measure of latency variability. If there is no jitter it means that latency is constant. For example, jitter in a telephone conversation causes ‘pops’ and ‘clicks’ audible by the user. Many multimedia applications are designed to minimise jitter. The most common technique is to buffer incoming data, and remove it from that buffer at a constant speed.

Medium access delay: This delay can be important in systems that involve an LAN, if they use a CSMA/CD medium access protocol or similar. This delay increases as the number of stations contending through the medium increases.

Accuracy: This term is sometimes used to express the difference between the data offered to the network and the data received from the network. Although both should be identical, noise, electromagnetic interference, crosstalk, voltage spikes, and device failures can alter data in transit over the network. Normally these differences are treated in the protocols of the upper layers such as TCP, but the effects of these errors imply retransmission, and therefore, inefficiency, or reduction of productivity. For wide area links, the accuracy is expressed as the error rate, or *bit error rate (BER)*. In local area networks, certain BER are not used, but as a general rule, the acceptable limit is 1 byte every 10^6 bytes.

Message size: The size of the message unit is a very important parameter, since it affects network performance. Normally, the unit message is called the *protocol data unit or PDU*. But the PDU of the network varies in size and format according to the underlying protocols used. So, when trying to understand the flow of traffic between different networks, the correct measurement is the *maximum transmission unit or MTU* allowed in any of the intermediate channels of the network, whether it is formed by local area networks, or by wide area networks. The MTU determines the maximum packet size that can be transmitted over a particular link, and therefore that link cannot carry in its original form a packet that is longer than the MTU of the medium. If a packet exceeds the size in a circuit or intermediate channel, it must be segmented into smaller packets. If the system does not support fragmentation, the package must be discarded and an alternative route found. Segmentation and reassembly can degrade the performance of the network, depending on the relative differences in MTUs, since each fragment must contain valid headers.

Payload: The actual amount of data that can be transferred over a particular channel. It is a significant factor in determining performance perceived by users. The payload is limited by two factors: the MTU of the medium, and the overhead of the protocol in each packet. So, for example, for an Ethernet network that uses TCP/IP to transfer data, the maximum payload will be

$$\begin{aligned} Payload &= MTU - (TCP_{overhead} + IP_{overhead} + MAC_{overhead}) \\ &= 1.518 - (20 + 20 + 18) \\ &= 1460\ bytes \end{aligned}$$

Response time: Usually the user works with applications that run on client computers, move information over the network, and access server systems. Using these applications, the user expects the response time of the system to be predictable and tolerable. Non-tolerable or unpredictable response times generate a dissatisfied user who makes mistakes and eventually prefers not to use the system, regardless of where the failure is located. If a transaction is particularly complex and requires a long response time, then it is better to perform it in the background, releasing the system for new use instantly. In addition to being tolerable and predictable, the long-term variation of the response time is also a factor to be taken into account for user satisfaction. The mean and the variance of the response time must be almost invariant in time. It is not acceptable to have a good response time for seven hours, and an unpredictable response times during the last hour of the workday (for example when backups or jobs in batch mode are performed). Finally, users are interested in the total response time and not how it is distributed. It may or may not be taken into account if the response time is affected by the time the user takes to think, by the latency of the client computer, by the delay of the network, or by the response time of the client server. Users want an oversized system that is always perceived to offer good benefits.

Figure 2.2 shows how the response for a well-sized system should appear. In this graph the abscissa axis shows the percentage of active users, and the ordinate axis shows the response time in seconds. A well-dimensioned system is one whose curve never exceeds the levels of the curve labelled 'real system'. In this system, when connecting 100 % of users the response time grows in a more or less tolerable and predictable way.

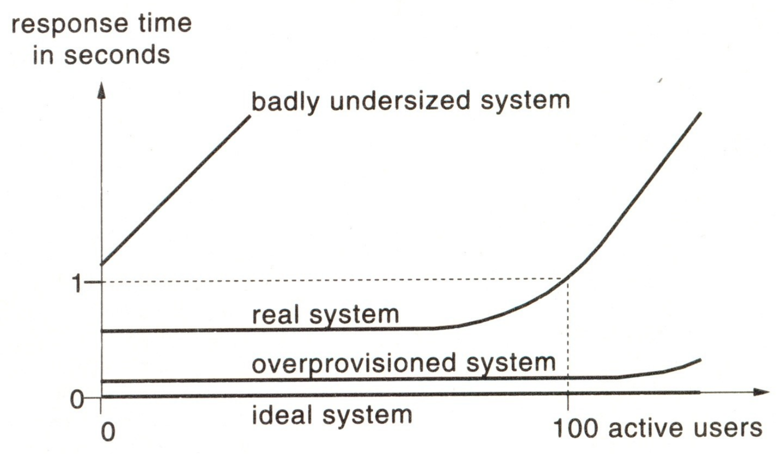


Figure 2.2: Response times depending on the percentage of active users on the system

2.2 Performance measures in interconnection network simulators

In the previous section we talked about the measures of benefits that can be obtained from real networks or systems, generally networks based on the TCP/IP protocol. In this section, we study the performance of switching-based interconnection networks based on the study of a particular network. Our direct interconnection network (each computing node is connected to a router) has a topology shown in Figure 2.3, the *2-cube-4-ary* topology or 2-D torus. In this topology, the routers or nodes of the network are numbered by their coordinates in each dimension (and they are represented by circles in the figure). Since there are 16 nodes arranged in two dimensions, and in each dimension there are four nodes (four nodes per dimension and two dimensions result in 4^2 = 16 nodes) the figure shows 16 circles numbered with 2 digits, from 00 to 33 (representing the coordinates from (0,0) to (3,3)). In this figure, a pair of channels (one in each direction) is represented by a line joining each pair of nodes. Thus, each node is connected by eight channels, one towards and the other from its four neighbours (north, south, east and west). In the network shown in Figure 2.3 exactly 16 unidirectional channels (8 bidirectional) cross the bisector or central line of the topology. If we assume 16 bit channels and a transmission speed of 1Gb / s, then the bandwidth of the bisection would be 256 Gb / s.

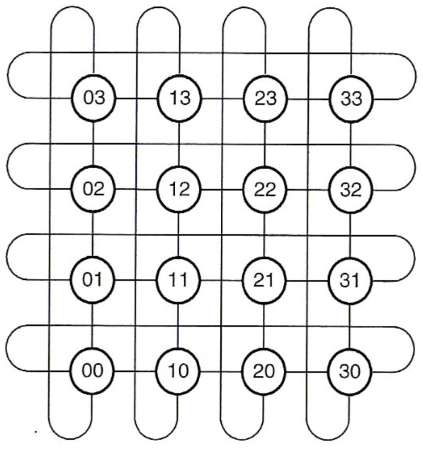


Figure 2.3: 2-cube-4-ary network topology

The performance of an interconnection network as shown in Figure 2.3 is mainly described by the latency curve versus offered traffic, like that shown in Figure 2.4. To draw a particular latency curve versus offered traffic, you must also specify the traffic pattern (for example, random traffic). Although the latency curves versus offered traffic gives the most accurate perspective of the ultimate performance of an interconnection network, they do not have closed expressions or functions that define them, and are usually found by simulation of discrete events.

The **zero-load latency** is the lower bound of the average latency of a packet that traverses the network. The zero load hypothesis is that a packet will never have to contend for any network resource with other packets. Under this hypothesis, the average latency of a packet consists of the serialisation latency plus the jump latency. For example, the network in Figure 2.3 shows packets of length L = 512 bits, channels with a bandwidth of b = 16 Gbits / s and random traffic, and so serialisation latency will be L/b = 32 ns. The

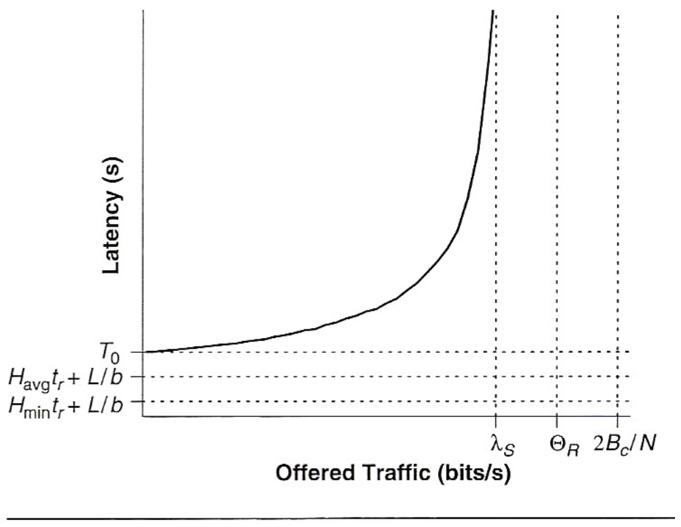


Figure 2.4: Latency curve versus offered traffic

lowest jump latency for random traffic occurs with minimum routing and gives an average number of hops of H_{min} = 2. For a router latency of t_r = 10 ns, the minimum jump latency will be $H_{min}t_r$ = 20 ns. This gives a lower latency level of $H_{min}t_r + L/b$ = 32 + 20 = 52 ns of average latency for a packet across the network, based only on topology, packing, and the net traffic pattern. The addition of the average number of jumps H_{avg} of the real routing algorithm used in the network gives a more adjusted level of packet latency, since $H_{avg} \leq H_{min}$. Finally, the flow control used by the network can further reduce the benefits on these dimensions given by the topology and routing. So, for example, if the network uses flow control *Store & Forward* the latency at zero load will be $H_{min}t_r \times L/b$ instead of $H_{min}t_r + L/b$. The real latency T_0 incorporates the constraints of the topology and the real performance of routing and flow control.

A similar approach gives us higher levels of network productivity. While each source offers a particular amount of traffic to the network, the productivity or **accepted traffic** is the traffic rate (bits / sec) that is delivered to the destination terminals. For our network example, the bisection was 16 channels, with a total bandwidth of B_c = 256 Gb / s. With a random traffic pattern, half of the traffic will cross the bisection of the network, and therefore the total traffic generated cannot exceed $2B_c$. Thus, traffic per node cannot exceed $2B_c/N$ = 32 Gb / s. But this dimension assumes that the traffic is perfectly balanced (distributed) between all the channels of the bisection. That is, a particular routing algorithm R cannot exceed this level and the productivity or throughput taking into account the routing algorithm (which we will denote Θ_R) can be lower if it imbalances the traffic, imposing a bound minor ($\Theta_R \leq 2B_c/N$). Finally, if the flow control results in idle channels due to resource dependency, then the **network saturation productivity** λ_S can be significantly less than Θ_R.

2.2.1 Network instrumentation

We usually study two (or three) of the basic measures seen in the previous section: throughput and latency (and in some cases fault tolerance). Although we have already defined these measurements, their exact definition depends to a large extent on how they are measured, or more specifically, on the measurement system. Figure **??** shows the scheme of what would be a standard measurement system for an interconnection network. To measure the performance of a network, an *instrumentation terminal* must be connected to each terminal or port on the network. This instrumentation includes a generator or source of packets that generates packets according to a determined traffic pattern, with a distribution of packet length and with a determined distribution of time between packets. By separating the packet source from the network there is a *source queue* of depth or infinite size. The source queues are not part of the simulated network, but serve to isolate network traffic processes properly said. Between the source packet and the source queue, an input packet measurement process counts the injected packets and measures the start or start time of each packet injected. It is important that this measurement process be placed *before* the source queue, so that packets that have been generated but not yet injected into the network are considered, and that the latency of the packet includes the time they spend in the source queue. In the same way, a complementary process in each exit terminal counts the packages and records their final arrival time.

The productivity or *throughput* is measured by counting the packets that arrive at each of the outputs, and the latency is measured by subtracting the initial time from the final or arrival time of each package. This measurement configuration *in open loop* enables controlling the traffic parameters independently of the network itself. Without the input queues, a packet source may attempt to inject a packet when the network terminal cannot accept traffic (for example, when the input buffers are full). In this case, the generation of traffic produced by the source would be affected by the network itself, and it would not be the original traffic pattern.

Closed-loop measurement systems (where the network influences the generated

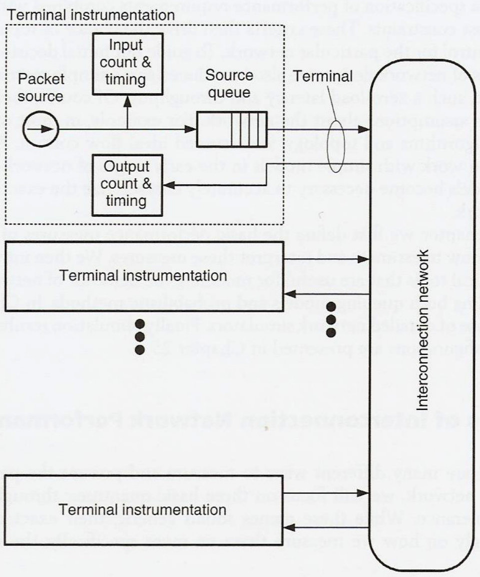


Figure 2.5: Schematic of standard measurement system for a network

traffic) are useful for measuring overall system performance. For example, the performance of a multicomputer can be estimated through a simulation where the instrumentation terminals are replaced by simulations of the multicomputer processes. The generated traffic will consist of the inter-process messages generated by the execution of the applications. A typical application of this type of measurement would be the study of the sensitivity of the execution time of the application for the network parameters such as bandwidth, routing algorithms, or flow control.

We will usually want to know the performance in a *steady-state* of the network. This is the performance yielded with stationary traffic after it has reached the equilibrium, that is, when the average length of the network queues has reached a stable state. To measure steady state performance, the simulation must be done in three phases: preheating; measuring; and draining. We will simulate N_1 preheating clock cycles of the network without taking measurements, then N_2 clock cycles with measurement of performance, and during the draining phase we will simulate the network with enough cycles for *all* of the measured packages to reach their destinations. Throughput will be measured with respect to all packets collected at destination during the measurement phase, while latency will be measured with respect to all generated packets during the measurement phase (even if they arrive in the draining phase).

2.2.2 Throughput

Throughput is the rate or speed at which packets are delivered by the network for a particular traffic pattern. It is measured by counting the packets arriving at their destination during a time interval for each flow (origin-destination pair) of the traffic pattern, and calculating from these flows the total fraction of the traffic pattern injected. Thus, the thoughput or **accepted traffic** by the network is different from the demand or **offered traffic** to the network by the traffic pattern, which is the rate at which the packets are generated by the package sources.

To understand how productivity is related to demand, the throughput in the graphs (accepted traffic) is usually drawn according to the demand or accepted traffic, as shown in the Graph 2.6. This graph shows the throughput versus traffic offered for an 8-ary 2D mesh network under complementary bit traffic and with routing in order of dimension, and showing the average and minimum productivity of all origin-destination pairs. Both the productivity and the offered traffic (abscissa axis) are shown as a fraction of the capacity, and therefore go from 0 to 1.

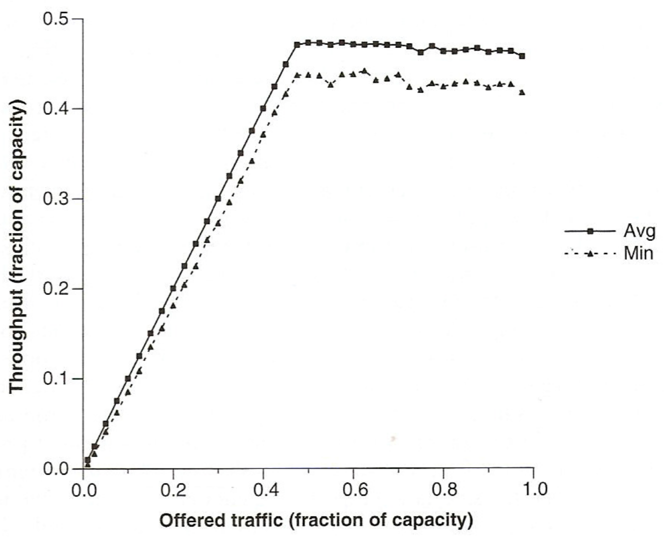


Figure 2.6: Throughput versus offered traffic for an 8-ary 2D mesh network under complementary bit traffic using dimension-order routing

For offered traffic levels lower than the *saturation*, the productivity of the network is equal to the traffic demand required by the traffic pattern, and therefore the productivity curve is a straight line. But if the offered traffic continues to increase, **saturation** is reached, which is the highest level of traffic offered for which productivity equals demand. When the demand (offered traffic) increases beyond the saturation point, the network cannot distribute the packets as fast as they are created (at least for that traffic pattern). Above saturation, a *stable* network continues to provide the same maximum productivity as at the saturation point, while in a *unstable* network, productivity falls as the offered traffic increases. Thus, the network in Figure 2.6 is stable with a saturation point of 43 % of capacity. However, Figure 2.7 shows an unstable network that although it is also saturated at 43 % of capacity also shows a sharp drop in productivity just after the saturation point.

In some unstable networks it also happens that the productivity remains constant beyond the point of saturation, but what happens is that the distribution of these packages no longer follows the specified traffic pattern. For this reason, the specified pattern is applied to all the inputs of the network and the accepted traffic for each flow is measured separately to correctly measure the throughput of these networks.

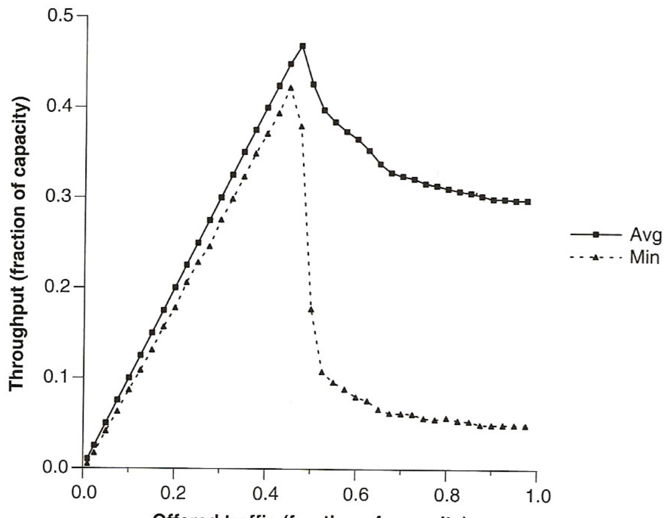


Figure 2.7: Typical productivity of an unstable network

2.2.3 Latency

Latency is the time required for a packet to cross the network from the source node to the destination node or terminal. In the evaluation made previously we have set the latency at zero load. But that latency ignores the latency due to the contention of the network, that is, to the competition between two or more packets for the same shared resource of the network (router, link, etc.). Once we include the contention latency, by simulation or by modelling, then the latency becomes a function of the offered traffic, and it is useful to draw this function or curve. To measure the latency we use the instrumentation of the network that appears in Figure 2.5. Typically, the offered traffic α is swept from $\alpha = 0$ to the saturation productivity, $\alpha = \Theta$. The latency is infinite and cannot be measured for traffic levels $\alpha > \Theta$. For each packet, the latency is measured from the time instant in which the first bit is put on the network until the time instant in which the last bit leaves the network. The global latency is reported as the average latency of all the packets that have circulated on the network. However, in some cases, it is also useful to obtain latency statistics for individual flows. Figure 2.8 shows an example of latency graph versus offered traffic.

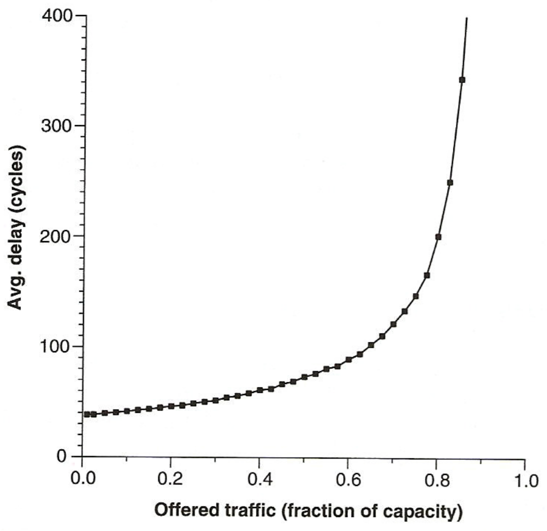


Figure 2.8: Latency vs. traffic offered for an 8-ary 2-D mesh network under uniform traffic with dimensionally ordered routing

The latency for the offered traffic shows a distinctive form that begins asymptotically to zero load latency, and whose slope increases up to the vertical asymptote of saturation throughput. For low traffic levels, latency approaches zero load latency. As traffic increases, the delay produced by contention causes the latency to increase, since the packets have to wait in buffers and channels.

Although it is generally useful to look at the average latency of the packets, it can also be indicative to look at the distribution of overall latency, or over a subset of the packets. For example, if virtual channels are used, some channels may be reserved for high priority traffic. In this case, the latency curves separated for normal and high priority traffic can give an idea of the effectiveness of the prioritisation scheme.

2.2.4 Common errors in instrumentation

Performance measurements are often obtained with a wrong instrumentation of the network. Two of the most common errors in this regard are shown in Figure 2.9. In Figure 2.9 a) a network instrumentation is shown without a source queue, so that if the packet source generates a new packet just when the injection port to the network is busy (there is traffic circulating through that node at that moment), then the source must discard the package or delay its injection. In the latter case, the injection of subsequent packets is also delayed. In either case, the traffic pattern is affected by the network contention.

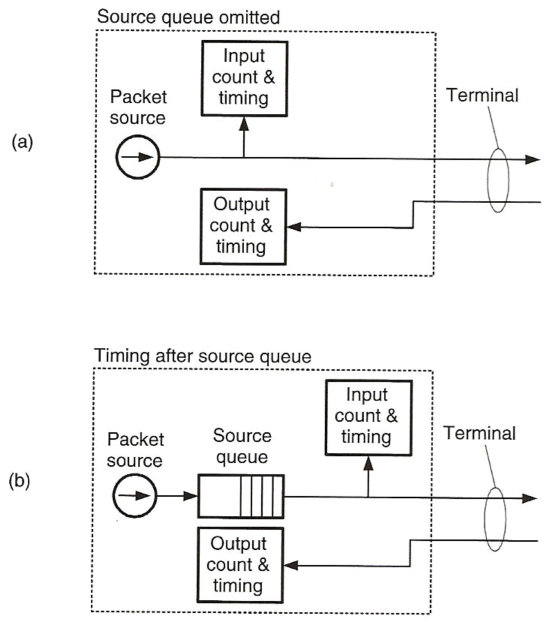


Figure 2.9: Common errors in network instrumentation: a) Absence of source queue b) wrong measurement

Figure 2.9 b) shows another case where there is a source queue to store the packages generated by the source, but the count of the number of packages and the timestamp of when they were generated is made at the exit of the source queue – instead of the entry. That is, it is done at the moment in which they are injected into the network, not when they were generated. In both errors, the consequence is that the traffic pattern for which the network's performance is intended to be measured is modified.

2.3 Performance measurements in existing networks and systems

2.3.1 Utilisation

The use of resources is an often measured system performance metric (from the point of view of the system itself). The use of a resource is defined as the percentage of time that the resource is being used. Given that multiple resources are involved in a system, we will divide the analysis of this measure according to the resource in question.

Primary and secondary memory

The physical memory of the system is used by the operating system kernel, its data structures, and various buffers. The rest of the physical memory is used by applications for their code, stack, data structures, etc. When the physical memory is full, the operating system usually offers the virtual memory subsystem, which enables writing some pages of the memory of an application to the disk, and freeing that memory for another application. However, when the memory demand is excessive, the disk area dedicated to these functions (swap area) may also become saturated.

The percentage of the maximum input/output rate is also a measure of disk utilisation. Disk drivers can specify the number of inputs/outputs per second, the average search time, or the sustained rate of data transfer.

As the disk access (input/output) rate approaches its maximum, queues will form in the memory area corresponding to the disk controller. These queues introduce delays in the response time.

The percentage of storage capacity used is also another measure of disk capacity.

This measure affects the performance of the system when the available space becomes too fragmented or when the existing files are also fragmented. This is because the number of searches that the heads must perform increases, and this increases disk response time.

CPU and the network

The use of the CPU is given either as CPU utilisation, or as *load factor*. The load factor is defined as the average number of processes in the execution queue, normalised to a time interval such as 1, 5, or 15 minutes.

This is because utilisation measured as a percentage does not give an idea of CPU performance beyond 100 %, where it is perfectly possible for a CPU to work, particularly a multicore CPU. In this zone, load factor is the best metric.

Finally, the use of the network is measured either as percentage of utilisation, or packets per second.

2.3.2 Collision rate

Obviously, collisions occur in those networks that use protocols where collisions can occur. The best known of these protocols is CSMA/CD, as used in IEEE 802.3 networks. The number of collisions that occur in a network can easily be measured by a network analyser or a segment monitor. The SNMP agent (from *simple network management protocol*) can also provide the number of collisions produced in each of its ports, while most operating systems provide the number of collisions in which they have participated.

The number of collisions occurred is not useful if the number of packets sent is not also provided. This will give us the percentage of collisions, through the simple formula.

$$\frac{collisions}{sent_packets} \times 100; = \ \%collisions \tag{2.1}$$

It should be remembered that collisions are normal in protocols with collisions. They do not cause data loss, and if they do not occur in bulk they do not significantly affect packet latency. A multiple collision is defined as a collision involving more than two frames. All network cards increase the value of this statistic, but the network analyser does not distinguish this type of collision.

Performance degradation produced by collisions is accentuated when a frame encounters between 10 and 15 collisions. Upwards from 16 successive collisions are considered as excessive retries. According to the 802.3 standard, in that case the card must discard the frame and try to send the next frame of the buffer. As this frame is considered lost at the link level, the transport layer will be responsible for detecting that such a frame is missing. So, for example, since the minimum default time for the TCP timer on HP-UX machines is one second, the system using this transport connection experiences an additional latency of one second. No data is lost, but the performance is degraded.

In addition, there are *late collisions*. These are collisions that occur after the sending station has been transmitting for 51.2 microseconds, and they cannot occur if the topology of the network complies with the physical level specifications (segment length, number of repeaters, etc.). The network card that detects a late collision increases a local variable. This statistic indicates that a violation of the specification has been foreseen, and serves to alert the administrator of the network.

Table 2.1 shows some of the measures that can be obtained in network cards and network analysers:

Measurement		**Description**
Transmitted frames	*	Correctly transmitted frames
Frames not transmitted		Frames not transmitted due to resource limitations
Transmission collisions	*	Total number of collisions
First transmission collisions		Frames that produced one collision
Additional collisions		Frames that produced more than one collision
Excessive retries		Frames discarded for 16 collisions
Delayed transmissions		Times that the network was busy when listening to the access medium.
Carrier losses		Times that the network interface was not able to listen to its own frame
Late collisions	*	Collisions detected after 51.2 microseconds

* Means measures also offered by a network analyser

Table 2.1: Measures offered by network interfaces and analysers

2.3.3 Protocol overhead

The overhead introduced by the protocols is necessary for network operation, but the addition of headers and queues at each OSI level reduces the efficiency of data transfer, especially when only a few bytes of data are sent in a packet. The mechanism for receipt acknowledgement of reliable protocols also adds latency and traffic. Finally, fragmentation and reassembly of data also adds overhead.

The reason for calculating the overhead introduced by the protocols is to calculate the network efficiency. For example, let us consider the telnet protocol, which sends single-byte data packets with TCP / IP over Ethernet. Each pressed key is sent to the server, which returns a frame with the same character and includes a piggybacking TPC ACK. Finally, the sender recognises the TCP ACK sending the same frame to the server. In total, 3 Ethernet frames are sent for each key pressed. As the Ethernet frames have a minimum size of 64 bytes (excluding the preamble), we have three frames of which the first two have one byte of data. The third does not carry data. Therefore, the efficiency is only 2/192 = 1.04 %. What happens is that users cannot write very fast (300 beats per minute, at most) and therefore the pressure on the network is not high.

The mechanisms of acknowledgment of the transport layer also reduce productivity. For example, in the IPX protocol the client must wait for the server to respond, and the response is typically 576 bytes. If packets have to traverse several routers, the propagation delay still further reduces productivity.

2.3.4 Rate of losses and retransmissions

The link and network layers can discard or lose packets (although there is no transmission error, buffers can be full, processors may respond to higher priority interruptions, etc.). Therefore, the transport layer is designed to temporarily store unrecognised data and retransmit it if necessary. The rate of loss of the lower layers, therefore, has a certain impact on the retransmission rate of the transport layer.

For example, let an Ethernet network transmit 3,290 packets per second correctly and experience 2,060 collisions per second. The probability that a packet or frame will experience a collision will be

$$p; = \frac{collisions}{collisions\ +\ sent_packets}; = \frac{2.060}{2.060\ +\ 3.290}; = \ 0,38; = \ 38\% \tag{2.2}$$

and the probability of losing a frame due to a 16 successive collisions will be

$$p^{16}\ =\ 2,355 \cdot 10^{-7} \tag{2.3}$$

As you can see, this value is very small. However, other reasons can cause an Ethernet network to lose frames, such as lack of resources in network cards. Ethernet switches can also drop frames. These switches have queues to store frames that compete for the same output port. With high transmission speeds and a small tail size, the switch will discard frames.

Another example is the ATM switches, which also incorporate frame output queues that compete for the same output port. Cells marked as disposable will be lost when the buffers overflow. The loss of a single cell will also require the transmission of the entire data segment.

Bridges and routers can also discard packets when internal buffers overflow. This case is especially likely when the WAN line is congested, and the transport layer timer reaches its limit and retransmits the packet.

The actual loss rate of an Ethernet or ATM switch, bridge, and routers can be estimated using queue theory or simulation methods. We will examine these techniques in the next lessons.

2.3.5 Maximum bandwidth and transmission rates

The maximum transmission rates (measured in *packets per second (PPS))* is a useful measure to highlight potential bottlenecks, and ensure that the capacity or bandwidth allocated is adequate for a given application. It is calculated as

$$PPS_{max} = Channel\_speed/(8bits * PDU_{Size})$$

where *channel _speed* would be the bandwidth or capacity of the channel, measured in bits per second, and the *PDU* is the *packet data unit* or the number of bytes that has the format of packages used in the network layer. Thus, for a circuit with a bandwidth of 64 kbps and with fixed packets of 128 bytes, we would have

$$\begin{aligned} PPS_{max} &= 64000/(8bits * 128) \\ &= 62,5\ pps \end{aligned}$$

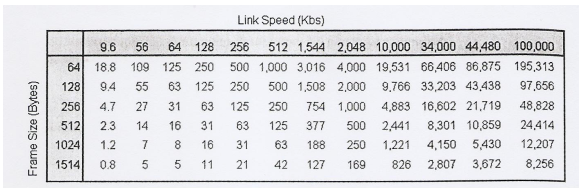
However, here we do not have the protocol overhead that we saw before, when determined by the MAC header or due to encapsulation. Thus, assuming an encapsulated

PPP (point-to-point protocol), we have to count an additional 4 bytes of the PPP header and the loss of the MAC header, and therefore the calculations would be

$$\begin{aligned} PPS_{max} &= 64000/(8bits * 114) + (8 * 4) \\ &= 67,797\, pps \end{aligned}$$

As the size of the packages is reduced, the effect of the encapsulation is more important. Thus, for example, with a 64-byte packet and the PPP protocol, the maximum rate would be 125 pps without the effects of the encapsulation, and 148 pps with the correct encapsulation. The difference can be significant; however, it seem otherwise as large packages are more efficient. For example, assuming 1500-byte Ethernet frames on a 64 kbps line, the maximum packet rate is 5.33 pps. But if we assume that they are TCP / IP packets, the total encapsulation overhead is 44 bytes (20 bytes of TCP header, 20 bytes of IP header and 4 bytes of PPP header). This represents 2.95 % of a frame of 1500 bytes, but 81.48 % of a 64-byte frame.

Figure 2.10 shows a table with the relationship between link speed and maximum packet rate for a given range of packet sizes. So, for example, if we have an application that can happily generate Ethernet frames of 1000 bytes at 150 pps, then it is impossible to work well with a leased 64 kbps line, and we need a T1 or E1 line, or we could use compression hardware between the link routers.



Link Speed (Kbs)

Frame Size (Bytes)	9.6	56	64	128	256	512	1,544	2,048	10,000	34,000	44,480	100,000
64	18.8	109	125	250	500	1,000	3,016	4,000	19,531	66,406	86,875	195,313
128	9.4	55	63	125	250	500	1,508	2,000	9,766	33,203	43,438	97,656
256	4.7	27	31	63	125	250	754	1,000	4,883	16,602	21,719	48,828
512	2.3	14	16	31	63	125	377	500	2,441	8,301	10,859	24,414
1024	1.2	7	8	16	31	63	188	250	1,221	4,150	5,430	12,207
1514	0.8	5	5	11	21	42	127	169	826	2,807	3,672	8,256

Figure 2.10: Link saturation matrix

2.4 Bottleneck detection

The benefits are measured to plan the capacity of the network and detect bottlenecks. Normally users have their own ”performance measures”. When the system as a whole does not work well, you always hear the same complaint: “The network is slow”

When certain performance problems appear in a network, it is necessary to use a methodical approach. For example, you have to discover, if possible, the following:

- Who experiences the same problem?
- Which users do not have this problem?
- When does the problem occur?
- When does the problem not occur?
- What has changed in the client system?
- When did the problem appear for the first time?
- Was there a time when this problem did not occur?

Next, we must analyse the differences:

- What is different between users who have and those who do not have the problem?
- What is the difference in the location of the problem?
- What is different when the problem does not occur?

Answers to these questions provide essential information to discover possible causes of the problem. Figure 2.11 shows several examples of potential bottlenecks and performance issues that may appear in a given interconnection topology.

In general terms, we should look for the following:

- Shared services (centralised server farms)
- Multi-user applications and shared databases
- Low speed network cards
- Shared LAN segments
- Limited bandwidth in WAN links

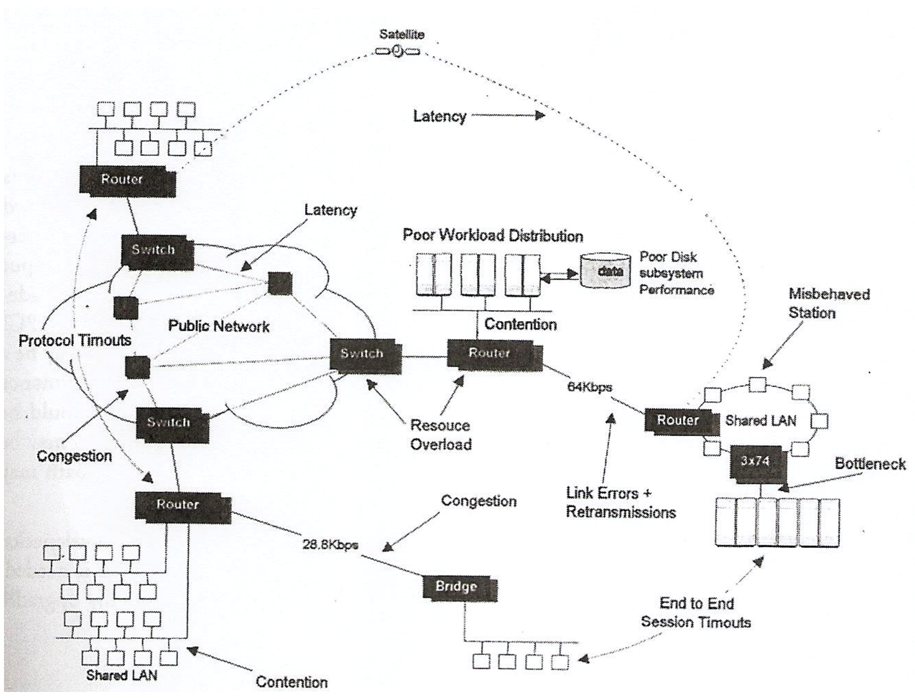


Figure 2.11: Potential bottlenecks

- Switching and routing main components
- Firewalls

If you have traffic analysis data or simulation studies, these will help areas of possible stress in the design of the network. It is advisable to have a global vision of the implications of any potential change before solving a local problem, since we could be transferring the problem to another part of the network. Consider the following example: suppose that users complain about the response time of a database server. The server is linked to an Ethernet network of 10 Mbps, and this network has an average usage of 65 % with peaks of 90 %. The monitoring of the network reveals that the application's timeouts are causing excessive retransmissions, responsible in turn for a large part of the load. It would seem appropriate to change the network in question from 10 to 100 Mbps, and yet the problem worsens, with slower response times and even occasional server crashes. A more detailed and detailed study reveals that the server did not have the required update for the CPU, and part of the server's memory has been removed without your knowledge. In addition, 60 users from another department have been using this server instead of their local server due to a routing error. As a result, both the CPU and the server memoirs were saturated: the server could not process all of the requests, and as the bandwidth of the network increased, they received more requests, which made the problem worse.

This type of problems can be avoided by studying the problem end-to-end, analysing the complete path between the client application and the server application, taking into account response times as well as retransmissions, lost frames, erroneous packets, etc. The slowest device on the route must be identified, and attention must be given to shared routes.

2.4.1 Clients and servers

As we have seen in the previous example, bottlenecks are not always in the network. Both clients and servers are subject to potential bottlenecks in the CPU, memory, input/output (network and disk) and the system bus. When designing scalable networks with a server-oriented application scenario, server performance is inherently crucial, since server performance must scale as the network grows, while clients do not need it. Many servers require transmission of 300 or 400 Mbps for their network cards. Although today almost all new processors are 64-bit architecture, there are still many computers with 32-bit bus architecture and inadequate productivity for gigabit applications per second. In a distributed application scenario bottlenecks may be more difficult to find, but at least it must be determined if the bottleneck is in the application or the computer, since in that case any improvements in the network are a waste of time and money.

2.4.2 Network performance

Data transmission over the network is limited by two factors: bandwidth and latency. However, the components of the network (routers, bridge switches, etc.) are limited by their internal resources (CPU, memory access speed, etc.). In a large interconnected network topology, these factors can be important and significantly contribute to end-to-end latency.

We will use a simple model to illustrate the effects of network latency in the transmission of a large segment of information, divided into two blocks of data, each sized W bytes (where W in the smallest transport window size, or the size of the application block). Suppose that the file is transmitted over a network whose bandwidth is Bw from the server to the client. Each transmitted block is followed by an acknowledgment from the client to the server (we will ignore sliding windows for the sake of simplicity). The transmission speed can be computed as

$$Datarate \quad = \quad W/(W/Bw + 2 * T_n + T_s + T_c)$$

where

T_n is the network latency

T_s is the server thinking time

T_c is the client thinking time

W/Bw is the required time for transmitting a data block over the transmission medium

Therefore, the data rate is the size of the block divided by the total time needed to send, receive, and recognise the block. However, the network latency T_n appears twice in the equation (each block is sent and recognised). If we wish to study the latency, we can consider the processing times as negligible:

$$Datarate \quad = \quad W/(W/Bw + 2 * T_n)$$

This simple equation calculates the productivity of the network in terms of effective data rate. For example, if we have an E1 link (2,048 Mbps) with a network delay of 10 ms and hosts use a block size of 1500 bytes (12000 bits) then the effective productivity of the network will be:

$$\begin{aligned} Datarate &= 12.000/(12.000/2.048.000 + 2 * 0,010) \\ &= 12.000/0.0258593 \\ &= 464.050 kbps \end{aligned}$$

This represents only 23 % of the available bandwidth. Applications that move large amounts of data (such as web / HTTP applications) are especially vulnerable to network latency.

2.4.3 Effects of contention and protocol overhead in shared networks

An Ethernet network at 10 Mbps has a maximum capacity of 14,880 frames per second, since the minimum frame size is 64 bytes, with a 64-bit preamble and a frame spacing of 9.6 ms. Therefore, 10,000,000 / [(64 x 8) + 64 +96] = 14,880.9 frames per second. For a maximum packet size (1,514 bytes) the maximum data rate is 814.8 frames per second. Therefore, as Ethernet headers are a very small proportion of the total size, the protocol efficiency for maximum size frames is greater (99.08 %), than the efficiency of 78.1 % for minimum size frames.

However, these calculations assume that the transmitter has exclusive access to the medium. In shared-media Ethernet networks, performance drops significantly as more stations are connected to the network. Some IEEE studies suggest that when a shared network with 50 nodes is approximately at 37% of the load, then the frames experience more delay than the token ring frames, and this is due to increased collisions. This study compared an Ethernet at 10Mbps with a Token-Ring at 10 Mbps with 128-byte frames. This has led to a widespread misconception that Ethernet begins to degrade performance when it is at 37% load.

2.4.4 Effects of buffering on WAN links

Since the bandwidth of WAN links is expensive, it is common to buy wide area links. In a routed network connection, the routers in the WAN interface must handle LAN traffic bursts directed to remote sites. There is usually a large speed difference between the LAN and WAN interfaces of the routers, and if all the LAN traffic had to be passed to the WAN link, then most would be lost, or much more capacity would be required from the WAN links. Fortunately, only a small part requires crossing to the WAN (usually, less than 20 %, although this is increasing because of distributed computing). Since LAN traffic is in bursts, buffers must be used in each WAN interface. Router buffers (queues) are used to standardise traffic to bursts in relatively slow WAN circuits or links. There must be enough buffers for the worst of the bursts, because otherwise packets will be discarded and this will cause retransmissions and the load on the link will further increase. An average use of the 70 % link is ideal to allow bursts to be handled without degrading performance, but this is done at the expense of wasting bandwidth (the WAN link) which is very expensive.

The other factor to consider when there are large differences between LAN and WAN speeds is the time that the packets pass in the buffers, that is, the contribution of the queues to the total end-to-end latency. Additionally, packets that spend a lot of time in queue ‘age’ and can be discarded erroneously, causing in turn retransmissions that worsen the situation.

3 NETWORK SIMULATION

Network modelling and analysis cannot model many aspects of networks because there are simply situations that are too complex to be expressed under those models, such as network contention. In these cases, simulation is an essential tool. However, simulation is a double-edged sword, because although it can provide excellent models of very complex network designs, both simulators and simulations can be equally complex. Therefore, not all simulators need to model with the same level of detail the entrails of the network, the details of the microarchitecture of the router, etc. Additionally, as important as adequately choosing the modelling accuracy is the task of properly selecting the workload that the network will support for the input to network simulation.

3.1 Levels of detail

The use of an extremely precise simulator is the safest approach, but it is also expensive in terms of the time required for both the design of the simulator and the execution of simulations. Therefore, an efficient approach is to choose a level of simulation detail that captures the important aspects of the behaviour of the network and avoids simulating unnecessary details. The following list of detail levels lists the typical ranges of different simulators:

Interface level: Models network interfaces and provides package delivery. Generates simple approximations of the packet latency based only on distance travelled

Capacity level: Adds simple constraints on the capacity of resources, such as channel bandwidth or limits on total number of packets in transit. Containment or contention for resources can affect the latency of the packages.

Flit level: The utilisation of the resources is followed at the flit level, requiring a detailed modelling of most of the components of the network router, including buffers, switches, and arbitration. The latency of the packets is accurately modeled in terms of flit times or router cycles.

Hardware level: Details of the hardware implementation are added to the simulator, showing silicon area and timing information. The latencies are expressed in absolute terms of time.

The *interface level* provides only the functionality of the network combined with packet delivery. At this level it is also known as *behavioural* simulation. This type of simulation is useful in the early stages of design where aspects such as consistency protocols need to be tested.

3.2 Workload

In a network simulator, the term *workload* refers to the traffic pattern (in packets) that will be applied to the terminals over time. Understanding and modelling the load enables us to design and evaluate topologies and routing functions. The most realistic traffic patterns or loads are the *application-driven workloads* generated directly by customers when they use the network. For example, in an interconnection network of a shared memory machine, the network traffic consists of the coherence messages exchanged by the nodes of the network. Therefore, if we model the network and the application that is running on the processors, our load will be exactly that required by the application. This ‘total system full simulation’ approach is called *execution-driven workload.* The downside is that any change in the design of the network affects the network and the workload.

An alternative to the simultaneous simulation of the network and its clients is to capture the sequence of messages that clients send over the network during an execution of the application and then ‘play’ that message sequence during network simulation. This is what is known as *trace-driven workload.* These traces can be captured either from a running system or from a run-directed simulation.

Although the workloads handled by the application are accurate, it is often difficult to achieve complete coverage of the expected traffic with these methods alone, as many factors influence a particular instance or execution of an application. Alternatively, we can use a *synthetic load* or traffic pattern of known characteristics but with variable parameters that can be adjusted (to some extent) to the particular characteristics of an application.

3.3 Simulator design. Network simulators

There are two basic approaches to the design of a network simulator: simulators *based on cycles (cycle-based simulators)* and simulators *based on events (event-driven simulators)*.

In the simulators based on cycles, time works in two phases, one associated with the global reading of the state of the network, and another associated with the writing of that state. Although the definition of the functionality of each state is weak or diffuse, the critical invariant is that all the functions of each phase can be evaluated in any order without changing the results of the simulation.

Alternatively, event-based simulations are not associated with a global clock, and this allows much more flexibility in the models. These simulators are based on *events*, which are data structures with three fields: time (clock cycle) when the event occurs; an action associated with that event; and data. The simulation is made by creating an *event queue* of all pending events, ordered by their execution or occurrence instants. The pending event with the lowest instant (the current one) is removed from the list and the corresponding action or function is invoked. The actions associated with the events can update the state of the network, as well as generate new future events that will occur as a result of the actions executed at the current time.

4 TOOLS FOR MONITORING (PERFORMANCE EVALUATION OF EXISTING NETWORKS)

Planning a network requires measuring or estimating the volume of network traffic. In the previous chapter we discussed which measures are the most important. In this chapter we describe the tools and techniques that the network administrator can use to obtain these measures **from an existing network**.

4.1 Reference data about performance

The reference data on application features are taken with trace tools while a client-server application is tested. A user executes the same transaction repeatedly, either for a certain time or a fixed number of times. These parameters are set and the user is catalogued as light, medium, or heavy. Traces of network traffic are captured from this session, package by package. This data is the most valuable for planning the capacity of a network, because it is reproducible and documented. The traces can be reused in a simulation and in addition the traffic statistics can be extracted for analysis.

However, certain rules must be followed to obtain these traces. Ideally, the traces should be free of any other network traffic to preserve the timing of the data. That is, an isolated LAN must be used to obtain the traces.

The traces can be reused as long as the test conditions persist. If the application, the network technology or the operating system changes, new traces should be used. Some statistics can be obtained from the traces, such as average transaction length, average number of bytes, average network utilisation, total number of packets, average packet rate (both client and server), average time of client processing, average length of data, or average server response time.

4.2 SNMP

In the 1990s the *simple network management protocol (SNMP)* was developed to manage open systems. Virtually all network devices today (hubs, switches, bridges, routers, printers, workstations, etc.) support (or can support) SNMP. SNMP is used to collect information on the features of network devices. SNMP basically consists of a manager and an agent. There is a network management application (NMA, from *network management application*) that operates on the network management station (NMS), which will be one of the stations connected to the network. This application builds an SNMP GET request and transmits it as a UDP datagram over the IP network. This datagram is transmitted as *best effort* traffic (as best you can) to the managed device. The SNMP agent of this device constructs an SNMP RESPONSE that crosses the reverse path to the NMA.

The SNMP agents perceive the data of the management information base (MIB) as a tree structure grouped into functional branches. The de facto standard established by industry, MIB-2, is supported by all modern devices. Figure 4.1 shows the logical groups in which the benefit variables are grouped. The most interesting groups are the interface, IP, TCP and UDP, because they contain counters on features that we can read.

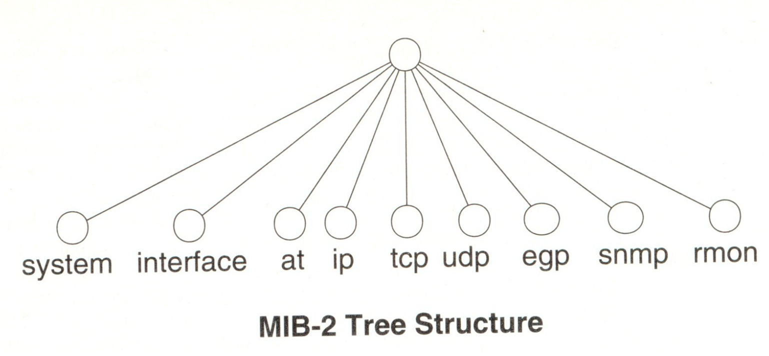


Figure 4.1: MIB-2 tree structure

Additional MIBs have been developed for hubs, terminals, FDDI, and ATM. Manufacturers often provide manufacturer-specific MIBs. Network management applications come with a set of MIBs from various manufacturers. The network management stations provide MIB compilers that accept MIB files in ASCII format and provide "MIB viewers" to study the MIB tree structures, and establish which performance parameters a specific MIB supports.

SNMP distinguishes between GAUGE and COUNTER type variables. GAUGE variables are non-cumulative variables, such as a car's speedometer. For example, the use of the CPU is a GAUGE type variable. The COUNTER type variables always increase monotonically since they are cumulative. They only decrease when they reach their maximum value and are set to zero. For example, the number of bytes received by an interface is a COUNTER type variable.

Normally, the network management station provides a graphical interface to establish the collection of historical data of SNMP variables. The name of the network device, the MIB variable to be sampled, and the sampling frequency have to specified. After a period of time (which can even be months), performance graphs (daily, weekly, etc.) can be obtained for a device or a group of devices. As an example, Figure 4.2 shows the usage (in bits / second) of a serial line of a router over a period of several weeks.

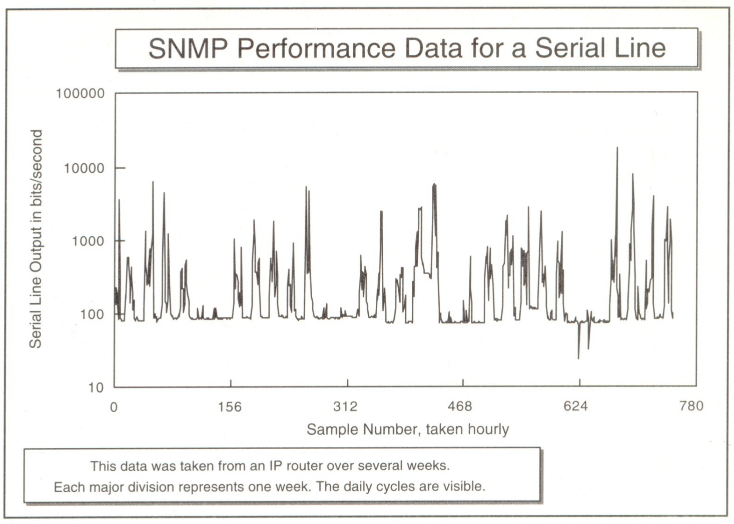


Figure 4.2: SNMP data about the serial line of a router

The performance data of an interface is one of the COUNTER-type variables, so to obtain the graph of Figure 4.2 a spreadsheet is used to log the differences between successive samples and calculate the values of bits per second.

The main question to be solved is which variables, among the hundreds of existing

SNMP MIB variables, should be sampled, and which network device should be sampled. Table 4.1 shows some useful indicators.

Measure	**Description**	**Device**
Perf. indicator	Bytes sent/received	Hub, switch, bridge,etc
	Unicast frames sent/rec.	
	Broadcast/multicast frames	
	CPU utilisation	
	Re-sending rate	Bridges, routers
Deterioration indicators	Ethernet collisions	Hub, switch, etc.
Performance	TCP retransmissions	Server
Problem indicators	CRC errors	All
	Carrier losses	
	Interface disconnection	
	Number Ethernet excessive retries (+16)	

Table 4.1: Measures offered by network interfaces and analysers

4.3 LAN analysers and monitors

LAN analysers basically consist of a computer with a network card (Ethernet, token ring, FDDI, etc.) with an ”analyser” application that establishes a filter and a set of TCP / IP utilities. The analyser’s network card simply operates in promiscuous mode, receiving all the frames regardless of the destination address or any other criteria. The frames that meet the conditions specified by the filter are stored in the computer.

When the network analyser is used to capture traces of benchmarks, the MAC addresses of the client and the server are set as filters, to capture that traffic exclusively and so avoid exceeding analyser capacity.

The network analysers usually come with statistical packages to keep track of usage, number of frames, collisions, etc. The information can be seen as histograms, tables, graphs, etc. This data can be used to analyse performance evolution, or to detect inaccessible operating conditions.

Segment monitors are network analysers for sections of the network. They are usually incorporated into hubs or other devices. RMON (from *Remote Monitoring MIB*) provides a standard of both data structure and command structure to remotely control the segment monitors.The RMON MIB defines several groups of measures, and is integrated into the MIB-2. Figure 4.3 illustrates this structure.

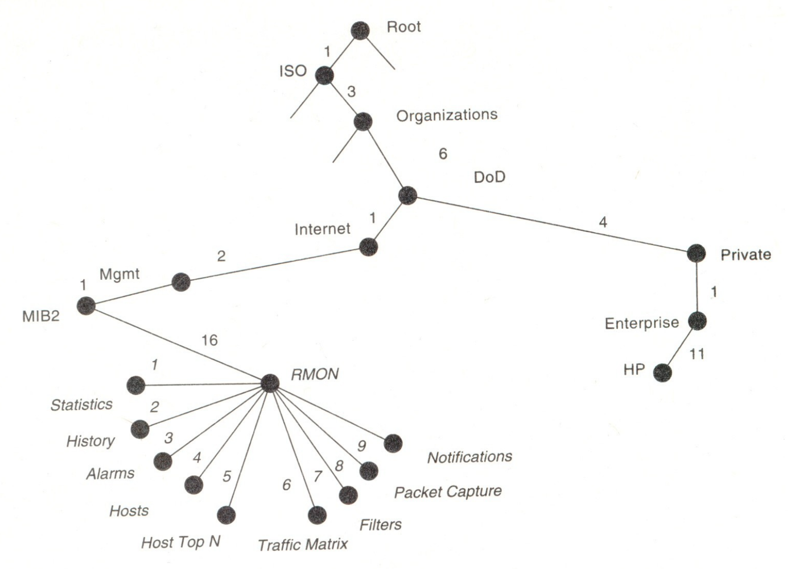


Figure 4.3: RMON MIB structure

In particular, group "statistics", "history" and "packet capture" yield data for predicting the performance of existing networks so that the rest of the groups fit better into the operational use of the network.

4.4 UNIX functions for performance measurement

UNIX is one of the most common operating systems in use today. In addition, in recent years the expansion of the Linux system (which is the UNIX for PC architecture) has caused the definitive expansion of this system. UNIX is multi-user and multi-tasking, and its many features include packet tracking. This functionality can be used to characterise network features. While the client or server application is running on a UNIX operating system, the sent packets can be captured and stored. The schema of the information flow is shown in Figure 4.4

However, it must be noted that under normal conditions, a UNIX system will operate in real time, and therefore the additional overhead that packet capture introduces may

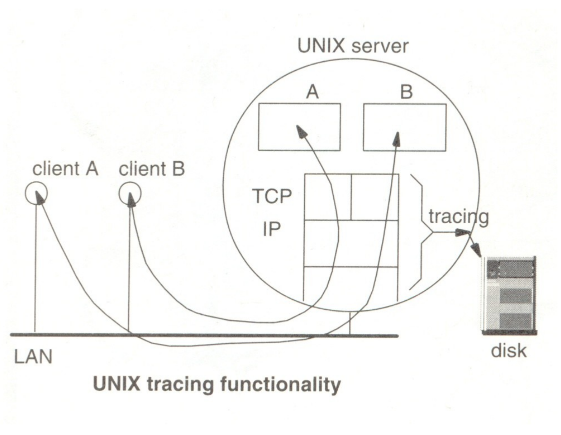


Figure 4.4: Packet tracing in UNIX

introduce jitter into the captured data. The less busy the system, the more accurate will be the data.

The advantage of using the UNIX tracking functions are numerous:

- Totally free
- All packages have a time stamp.
- Source-destination conversations can be preserved
- The utility *awk* enables processing the tracked data a posteriori.
- Systems with multiple cards (linked to multiple networks) can track all the packets concurrently.

A very efficient utility that incorporates UNIX or LINUX is the command *ping*. The name of the command, "ping" comes from ***p*****acket** ***i*****nter** ***n*****et** ***g*****roper** and it is distributed with UNIX or LINUX. The ping command is used to send packets to a remote system to verify that it is accessible. This program simply builds an ICMP loopback request (*internet control message protocol*) for an IP address, and injects it into the network card. The network transports the datagram to its destination, and it issues a response. The sequence number of the ping response calculates the round-trip delay. For more information, in a UNIX or LINUX terminal you can invoke the manual using the "man ping" command. The ping command looks like this:

```
PING correo.uv.es (147.156.1.72) from 147.156.16.72 : 56(84) bytes of data.
64 bytes from correo.uv.es (147.156.1.72): icmp_seq=1 ttl=253 time=1.25 ms
64 bytes from correo.uv.es (147.156.1.72): icmp_seq=2 ttl=253 time=1.59 ms
64 bytes from correo.uv.es (147.156.1.72): icmp_seq=3 ttl=253 time=1.35 ms
64 bytes from correo.uv.es (147.156.1.72): icmp_seq=4 ttl=253 time=1.64 ms
64 bytes from correo.uv.es (147.156.1.72): icmp_seq=5 ttl=253 time=0.834 ms
64 bytes from correo.uv.es (147.156.1.72): icmp_seq=6 ttl=253 time=1.35 ms
64 bytes from correo.uv.es (147.156.1.72): icmp_seq=7 ttl=253 time=1.39 ms

--- correo.uv.es ping statistics ---
7 packets transmitted, 7 received, 0% loss, time 6061ms
rtt min/avg/max/mdev = 0.834/1.348/1.647/0.246 ms
```

The performance of a network can be monitored by continuously using pings to different segments of the network that run in the background. This would directly and continuously measure response time.

The command `netstat` extracts kernel statistics about the network cards and the IP, ICMP, TCP, and UDP protocols – of all the elements that have participated in network communications.

The command `netstat -i` provides statistics for the network card, while the command `netstat -s` provides statistics for each of the protocols mentioned above. As an example, the output of the command `netstat -i` would be the following:

```
Kernel Interface table
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR   TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0   1500   0   69957      0      0      0    4293      0      0      0 BMNRU
lo    16436   0    5172      0      0      0    5172      0      0      0 LRU
```

A complementary utility for these commands is the program `awk`. This program enables processing the information obtained and extracting the numerical values from the output of the previous commands. So, for example, when executing the commands

```
ping -i 0.2 -c 200 -s 256 post.uv.es > pingtime.txt
awk -F = '{print $2 $4}' <pingtime.txt | sed 's/[a-z]//g' > pingdata
```

we generate a file called “pingdata ” that contains for the output line obtained with the ping command the numerical values of the sequence field and the round-trip delay. This file can be interpreted in a spreadsheet.

The Gnuplot utility enables generating graphs like the one in Figure 4.5, which shows the round-trip delay experienced by each of the packets sent using the “ping” command.

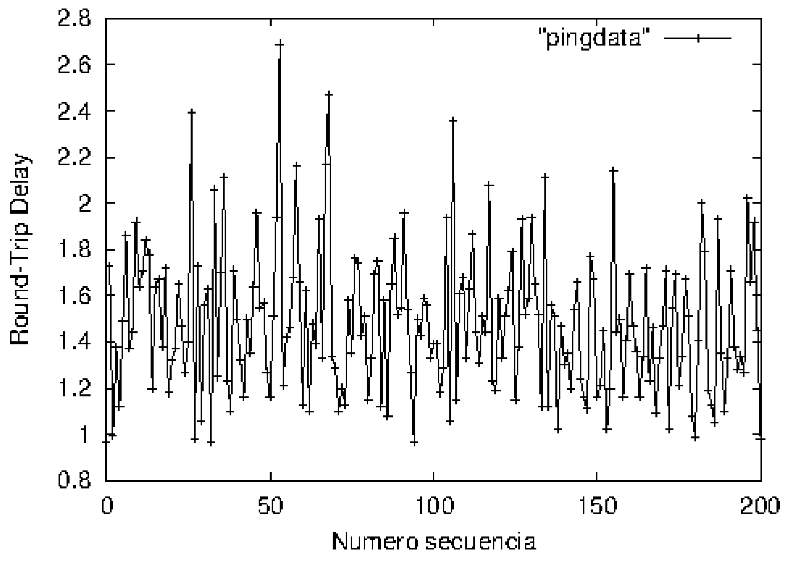


Figure 4.5: Latencies experienced by packets sent with the ping command

The command `gnuplot dibuja` can be used to obtain this graphic, where the file *dibuja* should contain the following gnuplot commands:

```
set terminal postscript eps 22
set output 'pingdata.ps'
set xlabel 'Numero secuencia'
set ylabel 'Round-Trip Delay'
plot  'pingdata' with linespoints

set terminal x11
set xlabel 'Numero secuencia'
set ylabel 'Round-Trip Delay'
plot  'pingdata' with linespoints
pause -1
```

4.4.1 Sampling frequency of performance data

How often should performance data be collected and stored? This is an important question, and it can affect the network itself. For example, most instrumentation enables sampling the network every second. However, the volume of data collected at this sampling frequency would be huge if long-term data was saved.

The data sampling frequency determines the resolution of performance data and storage requirements. If a network analyser captures 14,000 frames in a second, the short-term measures will have a high resolution and a large variation. However, if these 14,000 samples are taken over one minute, the variation will be smaller, but the volume of data stored will be 1/60 of the volume obtained if we take 14,000 in one second. The statistical service data taken from client-server benchmarks should be as reliable as possible, and should be taken once per second.

Network performance statistics can have a periodic component. If so, the sampling period of the benefit data must be less than half of that period (so that the sampling theorem is met). The period of the data should not be divisible by the sampling period. Otherwise, the samples will always be taken at the peaks, midpoints, or minimum points of the data cycles. Thus, for example, if the cycle of using a serial line is known to be 20 minutes, the samples should be taken in a range of less than 10 minutes, and in a sampling period such that 20 was not a multiple. For example, 7 minutes would be a good sampling interval.

Bibliography

[1] John Blommers, *‘Practical Planning for Network Growth’*, Hewlett-Packard Professional Books. Ed. Prentice-Hall, 1996.

[2] William J. Dally and Brian Towles, *‘Principles and Practices of Interconnection Networks’*, Morgan & Kaufmann Publishers (Elsevier), 2004. ISBN: 13-978-0-12-200751-4

[3] Gilbert Held, *‘Enhacing LAN Performance’*, Ed. Wiley & Sons, 3¿¿ ed., 2000.

[4] Tony Kenyon, ‘High Performance Data Network Design’, Digital Press (Butterworth-Heinemann), 2002. ISBN: 978-1-555558-207-4

DEPARTAMENTO DE INFORMÁTICA
E.T.S.E.

Grado de Ingeniería Telemática
Planificación de Redes

Lesson 3
Network Modelling

Prof. Juan Manuel Orduña Huertas

Contents

1 INTRODUCTION

Network modelling is based on the *queue theory*, which is also a discipline that studies congestion and how to deal with it. This technique enables predicting delays, minimising delays, estimating queue length, estimating the number of servers needed, etc. The typical applications of queuing theory are vehicle traffic, consumer service (for example, the number of windows necessary in a bank branch, the number of cash tills in a supermarket, the number of hospital beds or taxis in a city, etc.) and in our case, communications.

Queue theory is a subdiscipline of the most general mathematical discipline of probability. However, *teletraffic* is a special application of queuing theory for a special set of communication problems, particularly in the field of telephone communications.

Telecommunications, data communications, and computers themselves are examples of systems where the number of available resources is less than the number of entities that demand those resources. In turn, this implies that some users will have to wait until other users release the resources, or they can be denied service. Therefore, queue theory is fundamental in the design and management of such systems.

Consider the design of a trunk between two telephone switches (or between two ATM switches). There may be hundreds, or (in the case of telephone switches) thousands, of users connected to each of the two switches. It would be unreasonable to build thousands of trunks between the two switches. Suppose that, on average, only 5 % of the users of each switch require the service simultaneously. And suppose that only 50 % of the users of a switch wish to communicate with a user of the other switch. If each switch has 5000 subscribers, 250 users connect to it simultaneously as media. And of these, only 125 connect to the other switch. The same happens in the other switch, with which we have an average of 250 users using the trunk. The system would be designed with these requirements in mind, let's say$E$.

However, at a particular time or interval a certain number of customers above the average may demand the service. These extra clients will have to wait, since the system was designed only to accommodate E simultaneous users. Unfortunately, even if at another time there are fewer than E customers the remaining capacity cannot be accumulated for when there are more customers. An illustrative example could be a hairdresser with three assistants. If at 10 o'clock in the morning there are only two clients and the third assistant is unoccupied, this surplus resource does not count when four clients arrive later.

Intuitively, it can be observed that the greater the variation in the arrival of customers, the longer will be the system delay. If the average E is 20 and the variation is 2, at some point the number of customers could be 22 and in another it could be 18. Therefore, the overload and delay for those customers would be small. The variation in the influx of customers is what is called *arrival distribution* and consists of a random variable that can vary its value as a function of time.

In addition to the variation in the arrival of customers, the delay introduced for each client is also a function of the service time. Returning to the example of the hairdresser, if a haircut lasts 10 minutes then a fourth client will have to wait 10 minutes maximum (maybe less if a haircut was already halfway finished). However, if a haircut takes 60 minutes, a fourth client may have to wait 60 minutes. The time a customer needs to use a service is called *service distribution*, and it is another random variable.

A *queue model* is a mathematical abstraction of situations from the real world of this type. The goal of the queue models is to provide analytical expressions that determine the benefits for a given flow of customers (where the term 'client' can be applied to telephone calls, data packets, ATM cells, or LAN network packets) through the queue. The queue models enable us to analyse and plan the benefits that a network with certain resources would have if the traffic followed a certain behaviour. Queue models are based on birth-death processes that we will analyse later.

2 BASIC CONCEPTS

2.1 Probabilistic distributions

In a random or probabilistic queuing system, customer arrivals or service time are not known a priori. Therefore, the behaviour of these quantities must be specified by a **probability distribution**. In a probabilistic case you should not expect a single or possible result, and a range of possibilities may materialise. The probabilistic distribution lists each of these results and the probability that it will materialise.

There are two types of arrival and service distributions: discrete and continuous. **Discrete distributions** are those in which the sample space is composed of a countable set (set of integers). A **continuous distribution** is one where the population is composed of an infinite set of samples.

Of course, specifying the type of distribution is not enough, and every one of the parameters that describe that distribution must be specified.

2.1.1 Set functions and axioms of probability

The bases of the theory of probability are formed by three concepts of theoretical sets: the first is the set of all the possible experimental results Ω, such as 1,2,3,4,5,6 when throwing a dice. The second is the set of events$\mathbf{A}$. A$event$ is defined as a subset of the set of all possible outcomes. For example, the even number when throwing a die. The following operations are defined on the set of events:

1. **Add-on:**AC is the event that the A event does not occur. It is the set of all experimental results in Ω that are not in A. A^C is called *complement of A*.
2. **Intersection:**$A \bigcap B$ is the event that the A and B events occur. This is the set of experimental results that are common to A and B.
3. **Union:**$A \bigcup B$ is the event that the A or B events occur. It is the set of the experimental results that are in A, B, or both.
4. **Inclusion:** $A \subset B$ is when an event A implies that event B occurs.

The set of events **A** is closed in these operations, that is, when performing these operations on the members included in A no new members appear. So, if A and B are events, then $A\ bigcap\ B$, $A \bigcup B$, ..., are also events. Two sets of events are called **disjoints** if $A \bigcap B = \emptyset$

Let $P(A)$ be the *probability of event A*. $P(A)$ is a function on the set of events A that satisfies the following five axioms

1. There is a non-empty set of experimental results Ω and a set of A events defined on the set of experimental results.
2. For any event $A \in \mathbf{A}$, $P(A) \geq 0$
3. For the set of all possible experimental results Ω,$P(\Omega) = 1$
4. If the events in A and in B are disjointed or mutually exclusive, $A\ bigcap\ B = \emptyset$, then it follows that $P(A \bigcup B) = P(A) + P(B)$.
5. For an infinite countable set $A_1, A_2, \ldots$ such that $A_i \bigcap A_j = \emptyset$ for $i \neq j$, we have that

$$P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$$

All the properties of probability can be derived from these five axioms. Some of the most important are the following:

1. For any event A it is true that$P(A) \leq 1$.
2. $P(A^C) = 1 - P(A)$.

3. $P(A \bigcup B) = P(A) + P(B) - P(A \bigcap B)$ for arbitrary sets A and B.
4. $P(A) \leq P(B)$ for $A \subset B$.

Example: *Let us assume a deck of cards, where there are 52 cards, divided into four suits: diamonds, hearts, clubs, and spades. In each suit, the cards are numbered from 2 to 10, plus the jack, queen, king, and ace. In a poker hand, where five cards are held and none is changed, what is the probability of having five cards of the same suit or having a run composed of 10, jack, queen, king, and ace of different suits?*

Answer: *The probability of drawing a particular card will be* $1/52$. *Therefore, the probability of drawing a particular subset of 5 cards will be* $1/52 \times 1/51 \times 1/50 \times 1/49 \times 1/48$ = *3,20641* $\times 10^{-9}$.

The question we are asked is the union of two sets (all of the same suit **or** *a 10 and the rest picture cards). Therefore, the problem must be solved by applying property 3. We need, three steps to obtain the solution:*

Let's first compute the probability of the intersection of both sets, that is, that it is a 'straight flush': 10, jack, queen, king and ace, all of the same suit. Since there are 4 mutually exclusive suits, by axiom 4 we must add the 4 probabilities that they are of a particular suit, but since all the suits have the same probability, it results in a multiplication:

$P(straight\ colour) = 4 \times 3,20641 \times 10^{-9} \ = \ 1.28256 \times 10^{-8}$

Second, we calculate the probability of having 5 cards of the same suit. The suits are again exclusive to each other, and therefore you have to add 4 times (multiply by 4) the probability of obtaining the 5 cards of a particular suit. Since there are 13 cards of each suit, this will be

$P(a\ suit) = 13/52 \times 12/51 \times 11/50 \times 10/49 \times 9/48 \ = \ 3.73452548 \times 10^{-9}$

$P(same\ suit) = 4 \times 3.73452548 \times 10^{-9} \ = \ 1.49381?`10^{-8}$

The probability of a run (10, jack, queen, king and ace of different suits), since these figures add up to 20 of the 52 cards, will be

$P(straight) = 20/52 times 16/51 times 12/50 times 8/49 times 4/48 = \ 7.5769763 \times 10^{-7}$

Therefore, property 3 derives that the probability of having the 5 cards of the same suit or having a run of 10, jack, queen, king and ace of different suits is

$$\begin{aligned} P(straight\ or\ same\ suit) &= 7,5769763 \times 10^{-7} \ + \ 1.49381?`10^{-8} \ - \ 1.28256 \times 10^{-8} \\ &= 7,5981 \times 10^{-7} \end{aligned}$$

2.1.2 Random variables: probability distribution and density

A random or stochastic variable is a variable whose values are obtained from measurements in some type of random experiment. Formally, a random variable is a function that assigns events (e.g., the possible results of throwing a dice twice: (1, 1), (1, 2), etc.) to real numbers (e.g., its sum).

A random variable is discrete if its path is a discrete set, that is, if the possible set of values it can take is a finite or infinite countable set (representing a set of points on the real line). On the contrary, a random variable is continuous if its path is not a countable set. This means that the set of possible values of the variable covers an entire interval of the real line. For example, the variable that assigns the stature to a person drawn from a certain population is a continuous variable, since theoretically every value between, say, 0 and 2.50 m, is possible.

The possible values of a random variable can represent the possible results of an experiment not yet performed, or the possible values of a quantity whose current value is uncertain (e.g., as a result of incomplete or inaccurate measurement). Intuitively, a random variable can be taken as an amount whose value is not fixed but can be various values. A probability distribution is used to describe the probability of different values being given. Normally, they are denoted by capital letters.

The **probability distribution**, also called **cumulative probability distribution function** or **distribution function** of a random variable X, also called the distribution function of X, is the function

$$F_X(x) = P(X \leq x)$$

Thus, the distribution function of X evaluated at the point x is the probability that a set of experimental results will be mapped in the interval $(-\infty, x)$. From the axioms of probability, it can be shown that:

1. $F(-\infty) = 0$.
2. $F(\infty) = 1$.
3. $F_X(x_1) \leq F_X(x_2)$ for $x_1 \leq x_2$.

From the probability distribution function of a random variable you can calculate the probability that an event falls within a range:

$$P(x_1 < X \leq x_2) \;=\; F_X(x_1) - F_X(x_2)$$

The **probability function** (also called **probability mass function**) is a function that associates to each point of its sample space X as the probability that this will occur. Specifically, if the sample space E of the random variable X consists of the points $x_1, x_2, ..., x_k$, the probability function P associated with X is

$$P(x_i) = p_i$$

where p_i is the probability of the event $X = x_i$. By definition of probability, $\sum_{i=1}^{k} P(x_i) = 1$.

It should be noted that the concept of probability function only makes sense for random variables that take a discrete set of values. For continuous random variables, the analogous concept is that of the **probability density function (FDP)** or, simply, **density function**, commonly represented as $f(x)$.

$$P(a \leq X \leq b) = \int_a^b f_X(x)\,dx$$

The PDF is the derivative (ordinary, or in the direction of the distributions) of the probability distribution function $F(x)$.

$$f_X(x) = \frac{dF_X(x)}{dx}$$

or conversely, the distribution function is the integral of the density function:

$$F_X(x) = \int_{-\infty}^{x} f_X(\tau)\,d\tau$$

(τ denotes the change of variable to the Lebesgue measure[1])

Intuitively, you can think that $f(x)\,dx$ is the probability that X takes values in the infinitesimal range $[x, x + dx]$. The density function of a random variable determines the concentration of probability around the values of a continuous random variable.

Parameters of a random variable

The density function or the probability distribution of a random variable contains all the information about the variable. However, it is convenient to summarise its main characteristics with a few numerical values. These are, fundamentally, hope and variance.

The **mathematical expectation** (or simply expectation), or first order central moment, or expected value of a r.v., is the sum of the product of the probability of each event by the value of such event. If all events are equally likely, the expectation is the arithmetic mean.

For a discrete random variable with possible values $x_1, x_2 \ldots . x_n$ and with its probabilities represented by the probability function $p(x_i)$, expectation is calculated as:

$$E[X] \;=\; \sum_{i=0}^{n} x_i \cdot p(x_i) \tag{2.1}$$

For a continuous random variable, the expectation (which is also denoted by the letter μ), is calculated by the integral of all the values and the density function f(x):

$$\mu \;=\; E[x] \;=\; \int_{-\infty}^{+\infty} x\, f(x)\, dx$$

The concept of expectation is commonly associated in gambling with that of medium profit or long-term expected profit.

The variance is a measure of dispersion of a random variable X with respect to its expectation $E[X]$. It is defined as the expectation of the transformation$(X - E[X])^2$. That

[1]The Lebesgue measure is the standard way of assigning a length, area, or volume to the subsets of the Euclidean space (in this case, length). A is a closed interval [a, b], its Lebesgue measure is the length ba, the open interval [a, b] has the same measure, since the difference between the two sets has measure zero. If A is the Cartesian product of two intervals [a, b] and [c, d], it is a rectangle whose Lebesgue measure is the area (b-a) x (d-c).

is, the second order variance or central moment (and denoted by σ^2) is defined as the mean of the square of the distances of each value to the mean:

$$\begin{aligned}\sigma^2(x) &= E[(x-E[x])^2]; = \; E(X^2) - E(X)^2 \\ &= \sum_{i=1}^{n} p_i(x_i-\mu)^2 \;=\; \int_{-\infty}^{+\infty} (x_i-\mu)^2\, f(x)\, dx\end{aligned} \tag{2.2}$$

The **standard deviation** σ is defined as the square root of the variance, and indicates the dispersion that has occurred to the values of the random variable with respect to the mean.

The **coefficient of variation** is defined as the ratio between the standard deviation and the mean

$$C.V. \;=\; \frac{\sigma}{\mu}$$

The **α-percentile or α-quantile** is defined as that value a of the random variable X such that the probability that X takes a value less than or equal to a is α

$$a \;/\; P(X \leq a) = \alpha$$

To clarify this concept, let's look at an example. Let the random variable x have the following observation path (values that the variable takes in different observations): [2]

1 2 3 3 3 4 5 5 6 6

We have that there are 10 samples or observations. In this case, the 80-percentile of the random variable x will be that value a of the observations path such that at least 80 % of the samples are less than or equal to a. Since there are 10 samples, 80 % of 10 samples is 8. Therefore, as the eighth sample (in ascending order) is 5, the 80th percentile of x is 5. Indeed, if we count, we will see that at least 8 of the 10 samples are less than or equal to 5.

[2]WARNING: The observation path contains the ordered samples from the lowest to the greatest value, not chronologically ordered. That is, it is not necessary for the samples to be taken in that order.

The **median** is defined as the 50-percentile of a random variable x. If the number of samples n of the random variable is odd, then the median is the central value, $x_{\frac{n}{2}}$. If n is even, then the median is the mean of the central values, $(x_{\frac{n}{2}} + x_{\frac{n}{2}+1})/2$.

Finally, the **mode** is the most probable value of the random variable, that value x_i such that p_i is maximum.

The most important discrete variable distributions in telecommunications are the following:

- Binomial distribution
- Poisson distribution
- Geometric distribution
- Bernoulli distribution
- Discrete uniform distribution, where all elements of a finite set are equiprobable.

The most important continuous variable distributions in telecommunications are the following:

- Exponential distribution
- Normal distribution
- Uniform distribution (continuous)

Let's look at some of these distributions:

Binomial distribution

Binomial distribution arises naturally from a sequence of independent experiments called *Bernoulli trials*. Suppose that an experiment can have only two possible outcomes, which we will call *success* or *failure*, respectively. Let P be the probability of success. Therefore, the probability of failure will be $1 - P$. Suppose that the experiment is repeated n times with the condition that they are performed independently, that is, that the result of a test does not depend on the result of any other test.

We want to calculate the probability of k success, $0 \leq k \leq n$. But there are many ways to make this happen. For example, we might have the particular case of obtaining the first k successful trials and the last $n-k$ failed, or any other sequence of k successes and $n-k$ failures. The probability of an event consisting of a particular sequence of k successes is $P^k(1-P)^{n-k}$, $k = 0.1, \cdots, n$. A more complex event, consisting of a number of particular cases, is the occurrence of k successes in any order. But there are $\left(\begin{array}{c} n \\ k \end{array} \right)$ ways that there are k successes in n trials, where each pattern is a mutual event exclusive, and therefore the probability of the most complex event is the sum of the probabilities of the particular cases.

$$P(k\,success\,en\,n\,tests) \;=\; B(k;n,P) \;=\; \left(\begin{array}{c} n \\ k \end{array} \right) P^k(1-P)^{nk} \; k = 0,1,2,\cdots,n \quad (2.3)$$

This is the binomial distribution. In this distribution, the first and second order moments (mean and variance) are calculated directly by substituting the equation 2.3 in the equations 2.1 and 2.2:

$$E(B) \;=\; \sum_{k=0}^{n} k \left(\begin{array}{c} n \\ k \end{array} \right) P^k(1-P)^{nk} \;=\; nP\sum_{k=0}^{n-1} \left(\begin{array}{c} n-1 \\ k \end{array} \right) P^k(1-P)^{nk-1} \;=\; nP \quad (2.4)$$

$$\begin{aligned} Var(B) \;&=\; \sum_{k=0}^{n} k^2 \left(\begin{array}{c} n \\ k \end{array} \right) P^k(1-P)^{nk} - (nP)^2 \\ &=\; nP\sum_{k=0}^{n-1}(k-1) \left(\begin{array}{c} n-1 \\ k \end{array} \right) P^k(1-P)^{nk-1} - (nP)^2 \\ &=\; nP \cdot P(n-1) + nP - (nP)^2 \;=\; nP(1-P) \end{aligned}$$

Poisson distribution

In particular, for the theory of queues we are going to use the **Poisson distribution**. This is a discrete probability distribution that expresses, from a frequency of average occurrences, the probability that a certain number of events will occur during a certain period of time.

This distribution is applied to events with a very low probability of occurrence, obtained as the limit distribution of a sequence of binomial variables, $\mathbf{B}\left(n,p\right)$, where $n{\cdot}p = \lambda$, and $n \rightarrow \infty$ (therefore $p \rightarrow 0^{+}$). We will generally use the Poisson distribution as an approximation of binomial experiments where the number of tests is very high, but the probability of success is very low.

The mass function of the Poisson distribution is

$$f(k;\lambda) = \frac{e^{-\lambda}\lambda^{k}}{k!},$$

where

- k is the number of occurrences of the event or phenomenon (the function gives us the probability that the event will happen exactly k times).
- λ is a positive parameter that represents the average success rate, or the number of times the phenomenon is expected to occur during a given interval. For example, if the event studied takes place on average four times per minute and we are interested in the probability that it will occur k times within a ten-minute interval, we will use a Poisson distribution model with λ = 10x4 = 40
- e is the base of the natural logarithms (e = 2,71828 ...)

Both the expected value (mean) and the variance of a random variable with a Poisson distribution are equal to λ. Indeed, from the equation 2.4 we have that if $n \cdot p = \lambda$ then $E(P) = nP = \lambda$. If we also have that $p \rightarrow 0^{+}$, then from the equation 2.5 it also follows that $Var(P) = nP(1-p) = \lambda(1-0) = \lambda$.

The mode of a random Poisson distribution variable with a non-integer λ is equal to the largest of the integers less than λ. When λ is a positive integer, the fashions are λ and $\lambda - 1$. Poisson random variables have the property of being infinitely divisible.

Example: *Some disease has a very low probability of occurring, p = 1 /100,000. Calculate the probability that in a city with 500,000 inhabitants there are more than 3 people with this disease. Calculate the expected number of inhabitants who suffer the disease.*

If we consider the r.v. X that counts the number of people suffering from the disease, it is clear that it follows a binomial model, and that it can be very well approximated by a Poisson model, so that

$$X = B\left(n = 500,000, p = \frac{1}{100,000}\right) ......\approx bfPoi\left(\lambda = 5\right)$$

So the expected number of people suffering from the disease is $\mathbf{E}\left[X\right] = 5$. Since $\mathbf{Var}\left[X\right] = 5$, there is a large dispersion, and it would not be strange to find that there are actually many more or fewer people who have the disease. The probability that there are more than three sick people is:

$$\begin{aligned} P[X > 3] &= 1 - P[X \leq 3] \\ &= 1 - (P[X = 0] + P[X = 1] + P[X = 2] + P[X = 3]) \\ &= 1 - \frac{e^{-5} \cdot 5^0}{0!} - \frac{e^{-5} \cdot 5^1}{1!} - \frac{e^{-5} \cdot 5^2}{2!} - \frac{e^{-5} \cdot 5^3}{3!} \\ &= 0.735 \end{aligned}$$

Exponential distribution

The **exponential distribution** will also be very important. Despite the analytical simplicity of its definition functions, exponential distribution has a great practical utility, since we can consider it as an adequate model for the probability distribution of the waiting time between two events that follow a Poisson process. In fact, the exponential distribution can be derived from an experimental Poisson process, with the same characteristics as the Poisson distribution, but taking as a random variable (in this case, the time it takes to produce a fact instead of the number of its occurrences in a interval).

Obviously, then, the random variable will be continuous, and it has a great utility in the following cases:

- Distribution of the waiting time between events of a Poisson process
- Distribution of the time that passes until a failure occurs, if the condition is met that the probability of a failure occurring in an instant does not depend on the elapsed time

We are going to define the exponential distribution from the specification of its density function. Given a random variable X that takes non-negative real values, we will say that it has an exponential distribution of parameter μ with $\mu \geq 0$, if and only if its density function has the expression:

$$f(x) = \mu \cdot e^{-\mu x}, \;\; x \geq 0$$

where μ = 1 /average time of the interval. Consequently, the distribution function will be

$$F(X) \; = \; \int_0^{X_0} \; \mu \cdot e^{-\mu} x \, dx \; = \; \left[-e^{-\mu x}\right]_0^X \; = \; 1 - e^{-\mu X}$$

In the main application of this distribution, which is the theory of reliability, it is more interesting than the distribution function known as the **survival function or reliability function**. This is defined as the probability that the random variable takes values higher than the given value X:

$$S(x) \; = \; P(X > x) \; = 1 - F(X) \; = \; 1 - (1 - e^{-\mu X}) \; = \; e^{-\mu X}$$

If the meaning of the random variable is ‘the time that passes until the failure occurs’: the distribution function will be the probability that the failure will occur before or at the time X_0, and consequently, the survival function will be the probability that the failure will occur after time X has elapsed; therefore, it will be the probability that the element, or piece being considered ‘survives’ at time X; hence the name.

The moments of the exponential distribution (mean and variance) are:

$$E[X] \; = \; \int_0^{\infty} \; x\mu \cdot e^{-\mu t} \, dx \; = \; \frac{1}{\mu} \tag{2.5}$$

$$Var[X] \; = \; \int_0^{\infty} \; x^2\mu \cdot e^{-\mu t} \, dx \; - \; \frac{1}{\mu^2} = \; \frac{1}{\mu^2} \tag{2.6}$$

Let’s consider two examples of probabilistic distributions:

Example 1 (continuous distribution): mean time between failures (MTBF) of an ATM switch card. Many fault phenomena have an exponential time distribution. Suppose that the MTBF has the function of exponential distribution with parameter $h = 0.001$, and therefore the distribution of probabilities is of the form $0.001e^-0.001h$. The probability of failure in h hours will be $1 - e^{-0.001h}$ (the change in the formula is due to the integration process required for continuous distributions, this expression is the cumulative distribution function). Therefore, it follows that

Probability that the card fails in 100 hours = $1 - e^{-0,1}$ = 0,1;

Probability that the card fails in 1000 hours = $1 - e^{-1}$ = 0.63;

Probability that the card fails in 10000 hours = $1 - e^{-10}$ = 0.99;

etc.

And the mean time between failures is 1 /0.001 = 1000 hours.

Example2: Exponential distribution has **NO** *memory:* it has been proven that the lifetime of a certain type of pacemaker follows an exponential distribution with an average of 16 years. What is the probability that someone who has been implanted with this pacemaker should be re-implanted in less than 20 years? If the pacemaker has been functioning correctly for 5 years in a patient, what is the probability that it has to be changed before 25 years?

Solution: Let T be the random variable that measures the duration of a pacemaker in a person. We have that

$$\begin{aligned} T \rightsquigarrow Exp\left(\mu = \frac{1}{16}\right) \quad &\Longleftrightarrow \quad f(t) = \mu e^{-\mu t} \; \forall t \geq 0 \\ &\Longleftrightarrow \quad F(t) = 1 - e^{-\mu t} \end{aligned} \tag{2.7}$$

Then,

$$P[T \leq 20] = \int_0^{20} f(t)\,dt = F(20) = 1 - e^{-\frac{20}{16}} = 0.7135$$

And the answer to the second question is computed in this way:

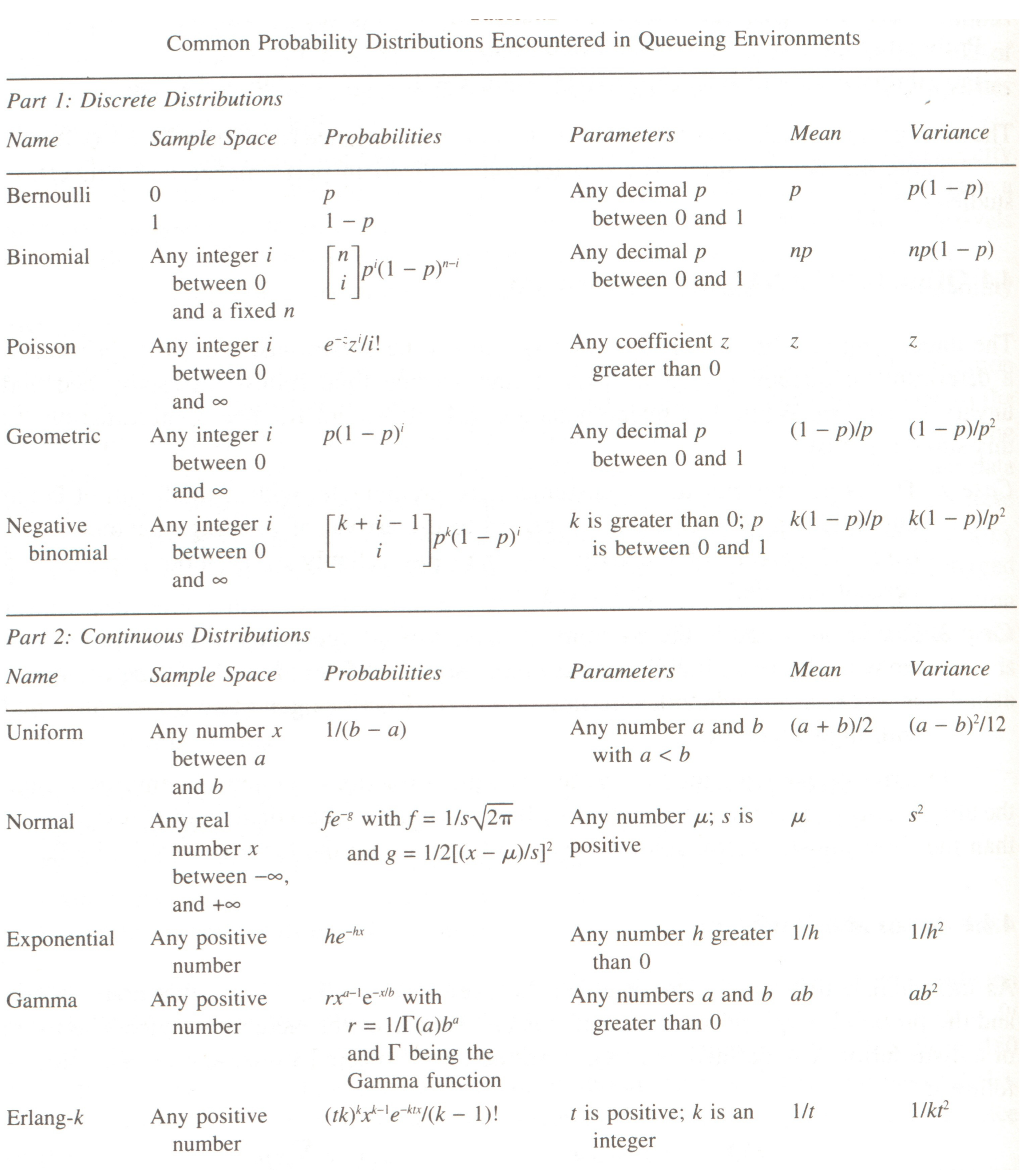
$$\begin{aligned}
P[T \leq 25_{|T \geq 5}] &= \frac{P[5 \leq T \leq 25]}{P[T \geq 5]} = \frac{0,522}{0,7326} = 0,7135 \\
P[5 \leq T \leq 25] &= \int_5^{25} f(t)\,dt = F(25) - F(5) = 1 - e^{-\frac{25}{16}} - 1 + e^{-\frac{5}{16}} = 0,522 \\
P[T \geq 5] &= \int_5^{+\infty} f(t)\,dt = F(+\infty) - F(5) = 1 - 1 + e^{-\frac{5}{16}} = 0,7316
\end{aligned}$$

So, as it could be expected as being inherent to an exponential distribution

$$P[T \leq 25_{|T \geq 5}] = P[T \leq 20]$$

that is, the time that the object has been working currently has no effects on the duration expected for the object from now on. This is why it is said that 'exponential distribution is memoryless'.

Finally, Figure 2.1 shows a table with some of the distributions commonly used in network performance studies.



Common Probability Distributions Encountered in Queueing Environments

Part 1: Discrete Distributions

Name	*Sample Space*	*Probabilities*	*Parameters*	*Mean*	*Variance*
Bernoulli	0 1	p $1 - p$	Any decimal p between 0 and 1	p	$p(1 - p)$
Binomial	Any integer i between 0 and a fixed n	$\begin{bmatrix} n \\ i \end{bmatrix} p^i(1 - p)^{n-i}$	Any decimal p between 0 and 1	np	$np(1 - p)$
Poisson	Any integer i between 0 and ∞	$e^{-z}z^i/i!$	Any coefficient z greater than 0	z	z
Geometric	Any integer i between 0 and ∞	$p(1 - p)^i$	Any decimal p between 0 and 1	$(1 - p)/p$	$(1 - p)/p^2$
Negative binomial	Any integer i between 0 and ∞	$\begin{bmatrix} k + i - 1 \\ i \end{bmatrix} p^k(1 - p)^i$	k is greater than 0; p is between 0 and 1	$k(1 - p)/p$	$k(1 - p)/p^2$

Part 2: Continuous Distributions

Name	*Sample Space*	*Probabilities*	*Parameters*	*Mean*	*Variance*
Uniform	Any number x between a and b	$1/(b - a)$	Any number a and b with $a < b$	$(a + b)/2$	$(a - b)^2/12$
Normal	Any real number x between $-\infty$, and $+\infty$	fe^{-g} with $f = 1/s\sqrt{2\pi}$ and $g = 1/2[(x - \mu)/s]^2$	Any number μ; s is positive	μ	s^2
Exponential	Any positive number	he^{-hx}	Any number h greater than 0	$1/h$	$1/h^2$
Gamma	Any positive number	$rx^{a-1}e^{-x/b}$ with $r = 1/\Gamma(a)b^a$ and Γ being the Gamma function	Any numbers a and b greater than 0	ab	ab^2
Erlang-k	Any positive number	$(tk)^k x^{k-1} e^{-ktx}/(k - 1)!$	t is positive; k is an integer	$1/t$	$1/kt^2$

Figure 2.1: Probability distributions most commonly used in queuing theory

3 BASIC CONCEPTS OF QUEUING THEORY

Now that we have studied the basics of probability, we are going to study queuing theory as a tool for computing the potential performance of networks.

3.1 Kendall notation

The most common way to specify queue systems, usually to identify models that share the same analysis, is to use the *Kendall notation*. This notation consists of the following fields:

$$A/B/m/K/N/Z \tag{3.1}$$

where each refers to:

A: Distribution of the random variable *inter-arrival time* of clients to the queue.

B: Distribution of the random variable *service time* of the queue.

m: Number of servers available to service the queue. Implicitly all the servers of equal operation are assumed.

K: Maximum size of the queue measured in the total number of clients (queue + servers) that it can host.

N: Size of the population that generates the traffic.

Z: Discipline of queue management. This discipline establishes how the tasks that are waiting in the queue access the system servers. The most used disciplines are the following:

FIFO (first in first out). In this service discipline, the tasks are served in order of arrival (as in the hairdressing salon). It is not always the most convenient from the point of view of the system, since a very long task or client will monopolise use of the resource that it occupies.

SIFO (shortest in first out). It serves first those tasks or clients whose service demand is lower.

LIFO (last in first out). The last task to arrive is the first to be served.

RR (round robin). The time of the resource is divided equally among all the tasks in the queue.

In Kendall notation, when the fourth field, K, is omitted, an unlimited capacity of the system is assumed (unlimited size of the waiting queue). If the fifth field, N, is omitted, or its value is infinite, then an arrival process is independent of the state of the system. Finally, if the sixth field, Z, is omitted, FIFO service discipline is understood.

Thus, for example, the queue $D/D/3/3$ designates a system where the distribution of the variable time between arrivals is deterministic, the variable service time is also deterministic, there are three servers in the system (servers + queue) that fit three tasks, the population that generates traffic is infinite and the service discipline of the queue is FIFO.

A queue denoted as $M/U/2/40/\infty/RR$ indicates that the distribution of the variable time between arrivals is exponential, the variable service time has a uniform distribution, it is a queue with two servers, it accommodates 40 clients or tasks in the system (that is, there are two servers and 38 queue positions), the population that generates traffic is infinite, and the service discipline of the queue is round-robin.

Typically, queue systems are represented by schematic figures such as those shown in Figure 3.1

This schematic notation may vary according to the authors. For example, in Figure 3.2 the circles that designate the number of clients accepted by the queue become vertical stripes, and the resource stops being a closed rectangle to become an open rectangle on the left (where the customers arrive). This figure shows at the top a queue with a single server

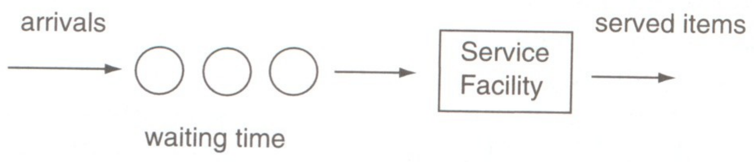


Figure 3.1: Components of a queuing system

and a single class of clients. In this case, we will have a distribution of arrivals, a service distribution, and an exit distribution. In the case below, where we have three classes of clients, we will have three arrival distributions (which may be different).

Figure 3.3 shows the schematic representation of other examples of queue systems. In this case, the upper part of the figure shows a queue with a single class of clients and with two servers. The lower part of the figure shows what is called a multi-queued network.

In telecommunications, the most common distributions found in the first two fields (distribution of arrivals and distribution of service) are the distributions without memory or *exponential distribution*, denoted by $\mathbf{M}$ (the letter M is due to its intimate association with Markov processes); the deterministic distribution (where the random variable is a constant), denoted by the letter $\mathbf{D}$; the uniform distribution, denoted by the letter $\mathbf{U}$; the hyperexponential distribution of k levels, denoted by the letter $\mathbf{H_k}$; the Erlangian distribution of r stages, denoted by the letter $\mathbf{E_r}$, and the arbitrary distribution, denoted by the letter $\mathbf{G}$.

For example, the system $M/D/3/20/1000$ would be a queue (a hairdresser, according to our example) with a time between arrivals that would follow the exponential distribution, a fixed service time (all clients would cost the same time to cut their hair), 3 servers (hairdressers), a queue of 20 clients (waiting room for 17 customers), and a population (a potential clientele) of 1000 possible customers. Since service discipline does not appear, it would be assumed to be FIFO.

Some of the typical queuing models of relevance in telecommunications are the following:

- $M/M/1$: Exponential distribution between arrivals, exponential service distribution, and a single server. This would be the model of an email server connected to a LAN, or a voice storage system connected to a PBX.

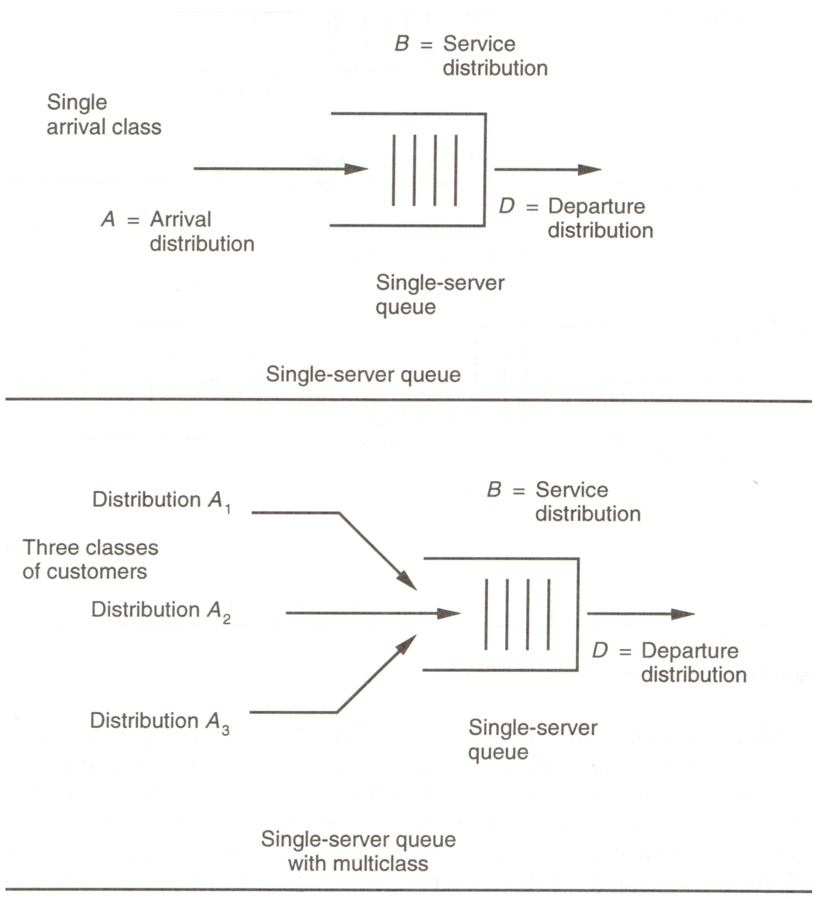


Figure 3.2: Queuing models of a single server and one or several kinds of clients

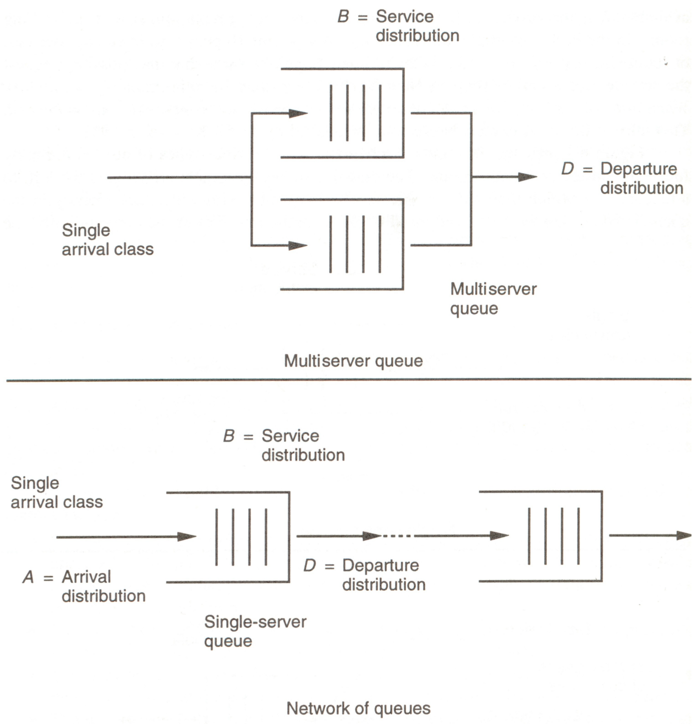


Figure 3.3: Example of multi-server queue and queue network

- $M/M/C$: Exponential distribution between arrivals, exponential service distribution and c servers. This would be the model for a trunk link between switches, with c trunks in the beam, or c modem ports in a pool of modems.

- $M/D/1$: Exponential distribution between arrivals, deterministic (fixed) service distribution and a single server. This would represent a trunk serving a packet switching network.

3.2 Stochastic processes

In the analytical modelling of the queuing theory, not only random variables such as those we have seen are used, but also different sequences or families of random variables that are a function of time. A stochastic process is a mathematical concept that enables characterising a succession of random (stochastic) variables that evolve based on another variable, usually time. Each of the random variables of the process has its own probability distribution function and, among them, they can be correlated or not.

Each variable or set of variables subjected to random influences or impacts constitutes a stochastic process. For example, suppose that $n(t)$ denotes the number of jobs in a CPU in a computer. If we take several suitable computers and observe the number of jobs in the CPU as a function of time, we would find that $n(t)$ is a random variable. To specify its behaviour we would need to specify the probability distribution functions for $n(t)$ at every possible instant t. Similarly, the waiting time in a queue $w(t)$ is a random function of time. These functions or random sequences are called **stochastic processes**. These processes are very useful to represent the state of queuing systems. There are several types of stochastic processes:

1. *Discrete processes or continuous processes:* A stochastic process is called discrete if the number of possible values that the variable can take is discrete (finite). For example, the number of jobs in a CPU $n(t)$ can only take integer values. However, the waiting time in a queue $w(t)$ is a continuous process, because it can take any value from the real line.

2. *Markov processes:* If the future states of a process are independent of the past and only depend on the present, then the process is called a **Markov process**. A stochastic discrete Markov process is called a **Markov chain**.

3. *Birth and death processes:* Markov chains where transitions occur only between adjacent states are called **birth and death processes**. In them the states can be represented by integers in such a way that the state n can only change to the state $n + 1$ or to the state $n - 1$. For example, the number of jobs in a queue and with individual arrivals can be represented by a process of birth and death. An arrival to the queue (birth) causes the step to the current state + 1, while the departure of a job after being served (death) causes the change to the current state - 1.

4. *Poisson processes:* If the intervals between arrivals are independent of each other and are distributed exponentially, then the number of arrivals n in a range $(t, t + x)$ has a distribution of Poisson (see Section 2.1), and the arrival process is called a **Poisson process**.

   Poisson processes have the following properties:

   (a) By mixing k Poisson processes of average arrival rates λ_i result in a Poisson process whose average arrival rate λ is

$$\lambda = \sum_{i=1}^{k} \lambda_{i-1}$$

(b) If a Poisson process is split into k threads such that the probability that a job goes to the i subprocess is p_i, then each subprocess is also a Poisson process with an average rate of $p_i \lambda_i$

(c) If the arrivals at a single server with exponential service time are a Poisson process with average rate λ, the server outputs are also a Poisson process with the same average rate λ, provided that the arrival rate λ is less than the server service fee μ.

3.2.1 Birth-death processes

Birth and death processes analysis methods are useful for the simplified analysis of communication networks. These processes are a special case of Markov processes or chains (systems meeting the condition that the evolution of the system to another state depends only on the current state, and not on the states in the previous instants) where **only transitions are made between adjacent states**. Figure 3.4 illustrates this behavior. Each state of the system is composed of the number of clients or tasks in the system. This number is a discrete random variable N(t).

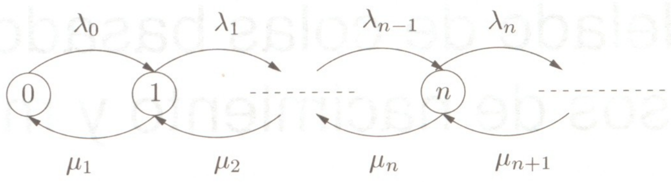


Figure 3.4: Transitions among states in a birth-death process

λ_m parameters indicate the *birth rate in the m state*, that is, how many new clients or tasks per unit of time will arrive to the system when it already contains m clients. The parameters μ_m indicate the *death rate in the m state*, that is, the number of clients or tasks that leave the system (are served by the system) per unit of time when there are m clients in the system. Obviously, $\mu_0 = 0$.

The processes of birth and death can change to a single adjacent state in a sufficiently small interval Δt. In other words, the probability of occurrence of two simultaneous events (births and /or deaths) is null.

It can be shown (see [5], page 520) that the probability that the queue system is in the n state (that is, that the system contains n tasks) is

$$p_n = p_0 \cdot \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i} \tag{3.2}$$

When the system is stable then p_n is a probability distribution, and it meets that

$$\sum_{n=0}^{\infty} p_n = 1 \tag{3.3}$$

With these two equations we obtain that

$$p_0 = \frac{1}{1 + \sum_{n=1}^{\infty} \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i}} \tag{3.4}$$

4 PERFORMANCE MEASURES OF A QUEUING SYSTEM

The performance measures of a waiting system or queue can be classified into system-oriented performance measures and user-oriented performance measures. The first serve to establish in a simple way what is the demand for resources made by users and the use of these resources. The second measures are interesting for the user because they measure the degree of quality of service perceived by the user.

4.1 Queuing system parameters. Little's law

In this section, we are going to examine the key parameters in the analysis of simple queuing systems, as well as the relationships among them. These parameters are the following:

τ = Time interval elapsed between two successive arrivals. It can be different for each arrival.

λ = Average arrival rate = 1/E[τ].

s = Service time for a customer, task or client. It can be different for each customer.

μ = Average service rate for each server = 1/E[s]. The total service rate in a queue with m servers is $m\mu$.

n = Number of tasks in the system. This is also denoted as **queue length**. **It includes both the task being served in the server(s) and the task waiting in the queue to be served.**

n_q = Number of tasks waiting in the queue. This will be always less than n

n_s = Number of tasks being served.

r = System response time. This includes two terms, the waiting time in the queue and the service time in the server.

w = Waiting time in the queue, that is the time interval between arrival and the instant when it starts being served.

Figure 4.1 shows the meaning of each of these parameters.

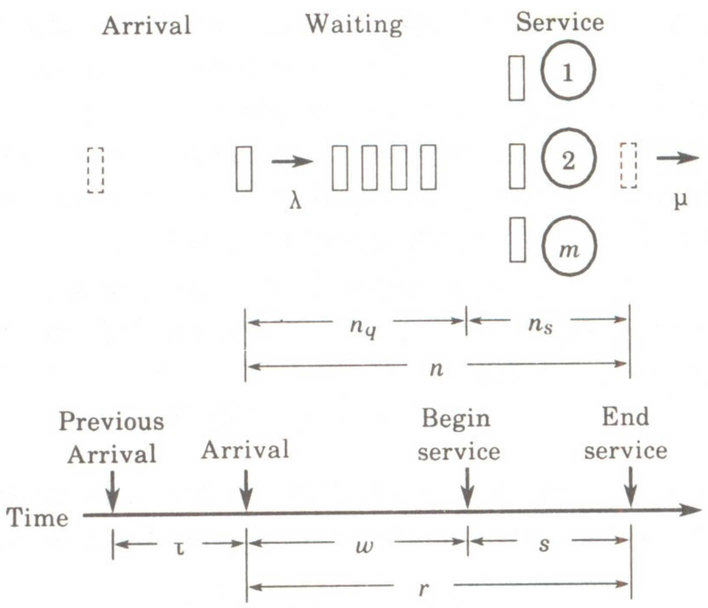


Figure 4.1: Usual parameters in a queuing system

All the parameters described, except λ and μ, are random variables. From these parameters, different performance measurements from queuing systems can be obtained, both system-oriented and user-oriented.

4.1.1 Performance measures for evaluating a queue system

The ultimate goal of queuing theory is to answer administrative questions relative to the design and operation of a queue system. On a search server, the administrator may want to decide if three or four front-end servers are designed.

Any queuing system goes through two basic phases. For example, when the bank opens in the morning, there is no one in the system, so the first customer is serviced immediately. As more customers arrive, the queue slowly forms and the amount of time they have to wait begins to increase. As the day progresses, the system reaches a condition in which the effect of the initial lack of customers has been eliminated and the waiting time for each client has reached fairly stable levels.

There are many different performance measures that are used to evaluate a queuing system in stable state. To design and operate a queue system, administrators usually care about the level of service a customer receives, as well as the proper use of the company's service facilities. Some of the measures that are used to evaluate performance arise from asking the following questions:

Questions related to time, focused on the client, such as:

- What is the average time an arriving client has to wait in line before being served? The associated performance metric is the average wait time.
- What is the time that a client spends in the whole system, including waiting time and service time? The associated performance metric is the average time in the system.

Quantitative questions related to sizing and use of the system, such as:

- On average, how many customers are waiting in line to be served? The associated measure of performance is the average length of the queue.
- What is the average number of clients in the system? The associated performance measure is the average number in the system.

Probabilistic questions that involve both clients and servers include:

- What is the probability that a client has to wait to be served? The associated performance measure is the blocking probability, which is represented by p_w and is calculated as the probability that all the servers in the system are occupied, that is, that there are as many clients in the system as servers.
- At any particular time, what is the probability that a server is busy? The associated performance measure is the utilization factor, denoted by U. This measure also indicates the fraction of time a server is busy.
- What is the probability that there are n customers in the system? The associated performance measure is obtained by calculating the probability P_n of having n clients in the system.
- If the waiting space is finite, what is the probability that the queue is full and that an arriving client is not served? The associated benefit measure is the probability of service denial, represented by p_d.

4.1.2 Little's law

One of the most commonly used theorems in queuing theory is **Little's law**, which enables you to relate the average number of tasks in any system or subsystem to the average response time of the system. The starting assumptions of this law are that the number of jobs or tasks that enter the system is equal to the number of tasks that come out of it (that are served). That is, they apply to a system where tasks are not lost or discarded.

Specifically, Little's law says that

The average number of tasks in the system = arrival rate x average response time

It should be noted that since Little's law can be applied to both systems and subsystems, in our case it can be applied to both the waiting queue and the system servers.

5 BASIC QUEUING MODELS

In this chapter we will study some of the most widespread basic models of queuing theory. The assumptions that these methods assume (and that the designer of the network must take into account, as explained in the next chapter) are the following:

- Users have no memory (the arrival of a user does not depend on the moment in which the previous user arrived)
- Servers have no memory (the service time for a task or user does not depend on how long it took for the previous user /task to be served).
- Infinite population
- Time invariance
- Stable system status
- Independence of users
- Traffic of a single class

5.1 M/M/1 model

The M/M/1 model is the most commonly used type of queue, since it can model single-server systems such as routers or any monoprocessor system. In this model it is assumed that the times between the arrival of tasks and the service times of the different tasks follow an exponential distribution, and that there is only one server. There is no limit on the size of the queue or the population that generates the requests (tasks), and the service discipline is FIFO. It is assumed that the average rate of arrivals is λ tasks /period and that the average service rate is μ tasks/period.

The state of this queue model is given by the number of tasks in the system. The state diagram of this model would as shown in Figure 5.1

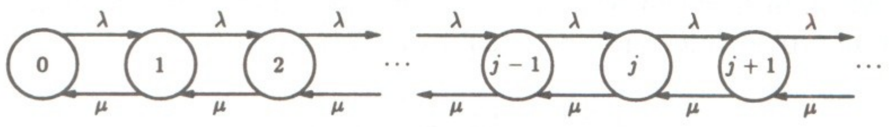


Figure 5.1: State diagram of an M /M /1 model

In this case, given that the distribution is exponential or without memory, λ and μ in each state does not depend on the time elapsed since the previous birth and /or death. Therefore, we have the following correspondences with the diagram transitions of states of a birth and death process:

$$\begin{aligned} \lambda_n &= \lambda \qquad n = 0, 1, 2, \cdots, \infty \\ \mu_n &= \mu \qquad n = 0, 1, 2, \cdots, \infty \end{aligned}$$

As seen in the processes of birth and death, with these correspondences we have

$$p_n = p_0 \cdot \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i} = \left(\frac{\lambda}{\mu}\right)^n \cdot p_0 \qquad n = 0, 1, 2, \cdots, \infty \tag{5.1}$$

The term λ/μ is known as **traffic intensity**, and is denoted as ρ. We define the **server utilisation factor** U or **flow rate** that the server will have as the traffic intensity divided by the number of servers in the system. If there are n servers, the utilisation factor of each server will be

$$U = \frac{\lambda}{n \cdot \mu} = \frac{\rho}{n} \tag{5.2}$$

where n is the number of servers in the system. In the case of the model $M/M/1$ we have $n = 1$ and therefore the server utilisation factor is equal to the traffic intensity.

So, in terms of the traffic intensity ρ the equation 5.1 remains

$$p_n = \rho^n \cdot p_0 \tag{5.3}$$

Equation 3.3 tells us that the sum of all probabilities must add 1. Therefore, we can clear p_0:

$$p_0 = \frac{1}{1+\rho+\rho^2+\cdots+\rho^\infty} = 1-\rho \tag{5.4}$$

And replacing this result in equation 5.3 means

$$p_n = (1-\rho)\cdot\rho^n, \qquad n = 0,1,2,\cdots,\infty \tag{5.5}$$

The utilisation of the system will be given by the probability of having one or more tasks in the system:

$$U = 1 - p_0 = \rho \tag{5.6}$$

The average number of tasks in the system is given by

$$E[n] = \sum_{n=1}^{\infty} n\cdot p_n = \sum_{n=1}^{\infty} n\cdot(1-\rho)\cdot\rho^n = \frac{\rho}{1-\rho} \tag{5.7}$$

The variance of the number of tasks in the system is

$$Var[n] = E[n^2]-(E[n])^2 = \left(\sum_{n=1}^{\infty} n^2\cdot(1-\rho)\cdot\rho^n\right) - (E[n])^2 = \frac{\rho}{(1-\rho)^2} \tag{5.8}$$

The probability that there are n or more tasks in the system is

$$P(\geq n) = \sum_{j=n}^{\infty} p_j = \sum_{j=n}^{\infty}(1-\rho)\cdot\rho^j = \rho^n \tag{5.9}$$

To find the average response time, we can use Little's law:

Average number of tasks in the system = arrival rate x average response time

Therefore, $E[n] = \lambda \cdot E[r]$. Clearing, we have to

$$E[r] = \frac{E[n]}{\lambda} = \left(\frac{\rho}{1-\rho}\right)\frac{1}{\lambda} = \frac{1/\mu}{1-\rho} \tag{5.10}$$

The average number of tasks in the queue will be given by the expression

$$E[n_q] = \sum_{n=1}^{\infty}(n-1)\cdot p_n = \sum_{n=1}^{\infty}(n-1)\cdot(1-\rho)\cdot\rho^n = \frac{\rho^2}{1-\rho} \tag{5.11}$$

When there are no tasks in the system, it is said that the system is free or unoccupied. The interval of time between two successive unused intervals is called the **busy period**.

The other formulas and measures of benefits of the model $M/M/1$ are shown in the tables that appear in Figures 5.4 and 5.5.

5.2 M/M/m model

The M/M/m queue can be used to model multiprocessor systems or devices where there are several identical servers, and where all the tasks or clients waiting to be served on these servers wait in a single queue.

This model assumes that there are m servers, each with a service rate of μ tasks or clients per unit of time. The arrival rate is λ tasks per unit of time. When a new task arrives, if any of the servers is unoccupied then the task is served immediately, without any waiting time. If the m servers are busy at that moment, then the incoming tasks wait in a single queue with FIFO service discipline. The state of the system is represented by the number of tasks n in the system. The state transitions diagram corresponding to this model is shown in Figure 5.2.

Since in this case the death rate is proportional to the number of tasks in the system n until $n = m$, we have that

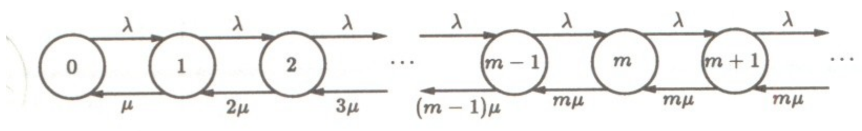


Figure 5.2: Diagram of M/M/m model state transitions

$$\lambda_n = \lambda, \qquad n = 0, 1, 2, \cdots, \infty$$

$$\mu_n = \begin{cases} n\mu, & n = 0, 1, 2, \cdots, m-1 \\ m\mu, & n = m, m+1, \cdots, \infty \end{cases}$$

With these equivalences, when applying the formula 3.2 to these processes of birth and death, we have that

$$p_n = \begin{cases} \frac{\lambda^n}{n!\mu^n} p_0, & n = 0, 1, 2, \cdots, m-1 \\ \\ \frac{\lambda^n}{m!m^{n-m}\mu^n} p_0, & n = m, m+1, \cdots, \infty \end{cases} \tag{5.12}$$

In terms of the traffic intensity $\rho = \frac{\lambda}{m\mu}$ the equation 5.12 remains

$$p_n = \begin{cases} \frac{(m\rho)^n}{n!} p_0, & n = 0, 1, 2, \cdots, m-1 \\ \\ \frac{\rho^n m^m}{m!} p_0, & n = m, m+1, \cdots, \infty \end{cases} \tag{5.13}$$

The probability that there are zero tasks in the system is calculated taking into account that the state transitions diagram (and therefore the model $M/M/m$) is a probability distribution, and therefore the sum of all must give 1 (formula 3.3). If we develop this summation, we have that

$$p_0 + p_0 \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!} + p_0 \frac{(m\rho)^m}{m!} \sum_{n=m}^{\infty} \rho^{n-m} \; = \; 1 \tag{5.14}$$

and solving p_0 from here we have

$$p_0 \; = \; \left[1 + \frac{(m\rho)^m}{m!(1-\rho)} + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!} \right]^{-1} \tag{5.15}$$

The probability that when a task arrives it will have to wait in the queue is called **probability of waiting** and it is denoted by ϱ. The probability of waiting is given by

$$\begin{aligned} \varrho \; = \; P(\geq m) &= p_m + p_{m+1} + p_{m+2} + \cdots = \\ &= p_0 \frac{(m\rho)^m}{m!} \sum_{n=m}^{\infty} \rho^{n-m} = \\ &= p_0 \frac{(m\rho)^m}{m!(1-\rho)} \end{aligned} \tag{5.16}$$

This formula is known as the **formula of Erlang-C**. This probability is a very important performance parameter for the $M/M/m$ models, since it is used to calculate almost all the performance measures in this model. Tables 5.6 and 5.7 show all the performance measure of this model.

It is noteworthy that for the case of a single server ($m = 1$) we have $\varrho = \rho$.

5.3 M/M/m/B model

The $M/m/m/B$ model is similar to the $M/M/m$ model, except that in the model $M/M/m/B$ the number of buffers is finite (limited). When the B buffers are filled, all arrivals are lost and they do not enter the system. We will always assume that B is greater than or equal to m and that B **includes the** m **servers**.

The state transitions diagram is shown in Figure 5.3

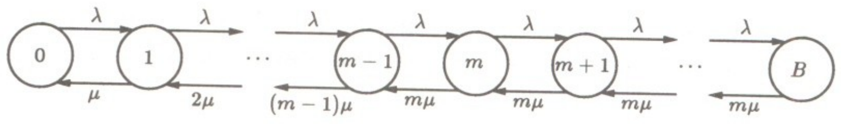


Figure 5.3: State transition diagram for Model M/M/m/B

This system can be modeled as a birth and death process with the following arrival and service rates:

$$\lambda_n = \lambda, \qquad n = 0, 1, 2, \cdots, B-1$$

$$\mu_n = \begin{cases} n\mu, & n = 0, 1, 2, \cdots, m-1 \\ m\mu, & n = m, m+1, \cdots, B \end{cases}$$

As in the previous model, the probabilities of the birth and death process give us the following probability that there are n tasks in the system:

$$p_n = \begin{cases} \frac{\lambda^n}{n!\mu^n} p_0, & n = 0, 1, 2, \cdots, m-1 \\ \\ \frac{\lambda^n}{m! m^{n-m} \mu^n} p_0, & n = m, m+1, \cdots, B \end{cases} \tag{5.17}$$

In terms of the traffic intensity $\rho = \frac{\lambda}{m\mu}$ equation 5.17 is

$$p_n = \begin{cases} \frac{(m\rho)^n}{n!} p_0, & n = 0, 1, 2, \cdots, m-1 \\ \\ \frac{\rho^n m^m}{m!} p_0, & n = m, m+1, \cdots, B \end{cases} \tag{5.18}$$

The probability that there are zero tasks in the system is calculated taking into account that the state transitions diagram is finite, with only B tasks. So,

$$\sum_{n=0}^{B} p_n = 1 \tag{5.19}$$

and therefore

$$p_0 + p_0 \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!} + p_0 \frac{(m\rho)^m}{m!} \sum_{n=m}^{B} \rho^{n-m} = 1 \tag{5.20}$$

And when we solve p_0 from here, we have

$$p_0 = \left[1 + \frac{(1-\rho^{B-m+1})(m\rho)^m}{m!(1-\rho)} + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!} \right]^{-1} \tag{5.21}$$

Using the expression to find p_n, we calculate the average number of tasks in the system $E[n]$ and in the queue $E[n_q]$:

$$E[n] = \sum_{n=1}^{B} n \cdot p_n \tag{5.22}$$

$$E[n_q] = \sum_{n=m+1}^{B} (n-m) \cdot p_n \tag{5.23}$$

It is important to note that in this model all the tasks that arrive when the system is in the state $n = B$ are lost. The rate of tasks that actually enter the system is called **effective arrival rate**, and is denoted as λ'. This rate is defined as

$$\lambda' = \sum_{n=0}^{B-1} \lambda p_n = \lambda \sum_{n=0}^{B-1} p_n = \lambda(1 - p_B) \tag{5.24}$$

The difference between λ and λ' represents the **package loss rate**.

Since tasks are not lost once they enter the system, Little's law can be used to determine the average response time of the system:

$$E[r] = \frac{E[n]}{\lambda'} = \frac{E[n]}{\lambda(1-p_B)} \tag{5.25}$$

And the waiting time in queue:

$$E[w] = \frac{E[n_q]}{\lambda'} = \frac{E[n_q]}{\lambda(1-p_B)} \tag{5.26}$$

If we observe the system for a time T, the total number of tasks that arrive at the system and obtain service (enter it) will be $\lambda' T$. The total time occupied by the m system servers will be $\lambda' T/\mu$, and the use **of each server** will be

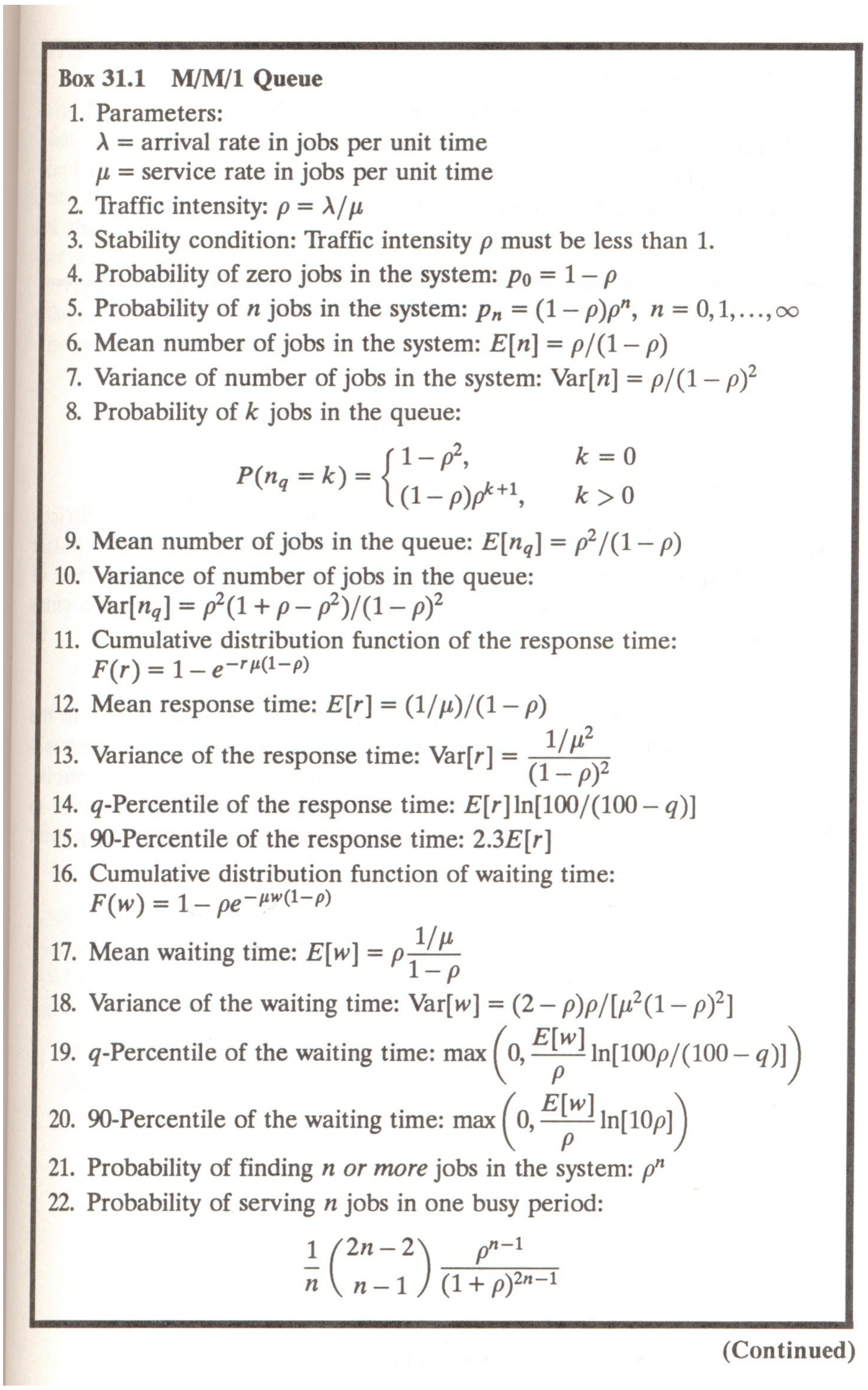
$$U = \frac{Tiempo\,ocupado}{Tiempo\,total} = \frac{\lambda' T/\mu/m}{T} = \frac{\lambda'}{m\mu} = \rho(1-p_B) \tag{5.27}$$

The probability that the system is full is given by p_B. In the case of a system $M/M/m/m$ the number of buffers is equal to the number of servers, and therefore there is no waiting queue. In this case, the probability of loss is

$$p_m = \frac{(m\rho)^m}{m!} = \frac{\frac{(m\rho)^m}{m!}}{\sum_{j=0}^{m} \frac{(m\rho)^j}{j!}} \tag{5.28}$$

This formula is called **Erlang loss formula** and was established by Erlang to compute the probability of losing a call in a telephony centre. Effectively, there are no waiting spaces in a switch, there are only as many registrars as switch inputs. If there are more simultaneous calls than switch inputs, the excess calls are dropped.

The results for the $M/M/m/B$ system are shown in Figure 5.8. Also, Figure 5.9 shows the formulae for the particular case of model $M/M/1/B$



Box 31.1 M/M/1 Queue

1. Parameters:
   λ = arrival rate in jobs per unit time
   μ = service rate in jobs per unit time
2. Traffic intensity: $\rho = \lambda/\mu$
3. Stability condition: Traffic intensity ρ must be less than 1.
4. Probability of zero jobs in the system: $p_0 = 1 - \rho$
5. Probability of n jobs in the system: $p_n = (1-\rho)\rho^n$, $n = 0, 1, \ldots, \infty$
6. Mean number of jobs in the system: $E[n] = \rho/(1-\rho)$
7. Variance of number of jobs in the system: $\text{Var}[n] = \rho/(1-\rho)^2$
8. Probability of k jobs in the queue:

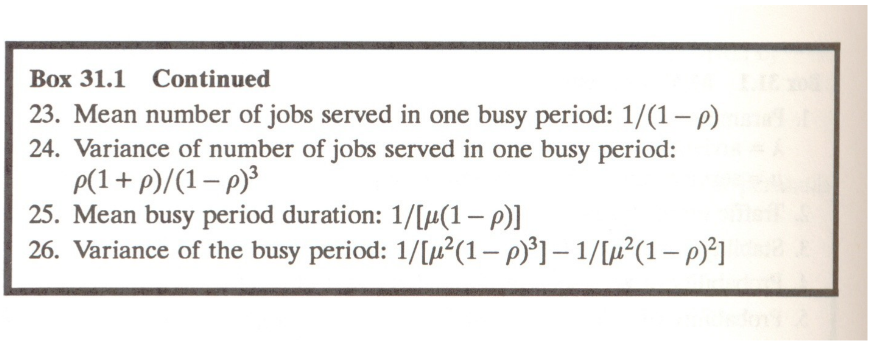
$$P(n_q = k) = \begin{cases} 1-\rho^2, & k = 0 \\ (1-\rho)\rho^{k+1}, & k > 0 \end{cases}$$

9. Mean number of jobs in the queue: $E[n_q] = \rho^2/(1-\rho)$
10. Variance of number of jobs in the queue:
    $\text{Var}[n_q] = \rho^2(1+\rho-\rho^2)/(1-\rho)^2$
11. Cumulative distribution function of the response time:
    $F(r) = 1 - e^{-r\mu(1-\rho)}$
12. Mean response time: $E[r] = (1/\mu)/(1-\rho)$
13. Variance of the response time: $\text{Var}[r] = \dfrac{1/\mu^2}{(1-\rho)^2}$
14. q-Percentile of the response time: $E[r]\ln[100/(100-q)]$
15. 90-Percentile of the response time: $2.3E[r]$
16. Cumulative distribution function of waiting time:
    $F(w) = 1 - \rho e^{-\mu w(1-\rho)}$
17. Mean waiting time: $E[w] = \rho\dfrac{1/\mu}{1-\rho}$
18. Variance of the waiting time: $\text{Var}[w] = (2-\rho)\rho/[\mu^2(1-\rho)^2]$
19. q-Percentile of the waiting time: $\max\left(0, \dfrac{E[w]}{\rho}\ln[100\rho/(100-q)]\right)$
20. 90-Percentile of the waiting time: $\max\left(0, \dfrac{E[w]}{\rho}\ln[10\rho]\right)$
21. Probability of finding n *or more* jobs in the system: ρ^n
22. Probability of serving n jobs in one busy period:

$$\frac{1}{n}\binom{2n-2}{n-1}\frac{\rho^{n-1}}{(1+\rho)^{2n-1}}$$

(Continued)

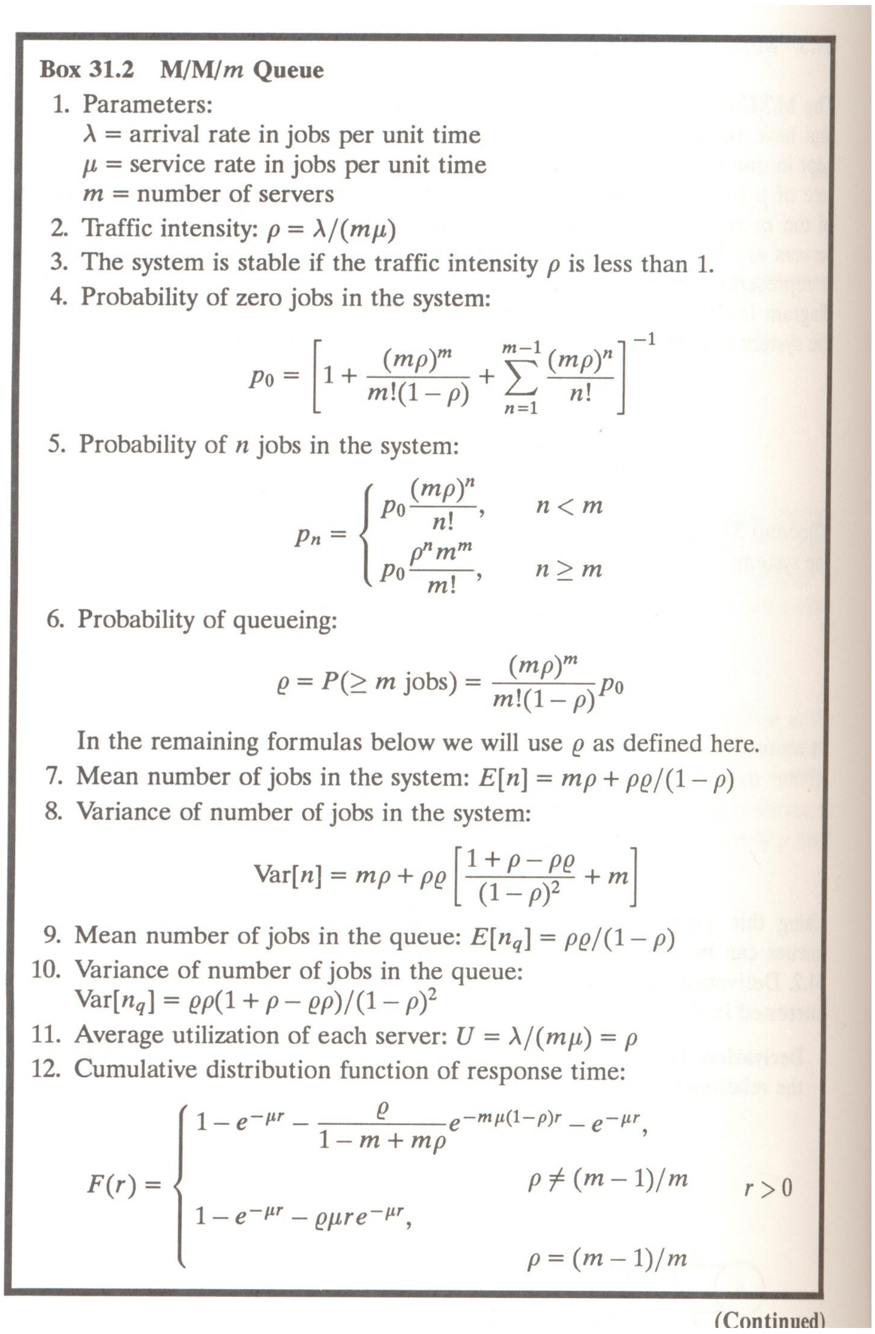
Figure 5.4: Performance measurements for M/M/1 model



Box 31.1 Continued

23. Mean number of jobs served in one busy period: $1/(1-\rho)$
24. Variance of number of jobs served in one busy period: $\rho(1+\rho)/(1-\rho)^3$
25. Mean busy period duration: $1/[\mu(1-\rho)]$
26. Variance of the busy period: $1/[\mu^2(1-\rho)^3] - 1/[\mu^2(1-\rho)^2]$

Figure 5.5: Performance measurements for M/M/1 model (cont.)



Box 31.2 M/M/m Queue

1. Parameters:
   λ = arrival rate in jobs per unit time
   μ = service rate in jobs per unit time
   m = number of servers
2. Traffic intensity: $\rho = \lambda/(m\mu)$
3. The system is stable if the traffic intensity ρ is less than 1.
4. Probability of zero jobs in the system:

$$p_0 = \left[1 + \frac{(m\rho)^m}{m!(1-\rho)} + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!}\right]^{-1}$$

5. Probability of n jobs in the system:

$$p_n = \begin{cases} p_0 \dfrac{(m\rho)^n}{n!}, & n < m \\ p_0 \dfrac{\rho^n m^m}{m!}, & n \geq m \end{cases}$$

6. Probability of queueing:

$$\varrho = P(\geq m \text{ jobs}) = \frac{(m\rho)^m}{m!(1-\rho)} p_0$$

   In the remaining formulas below we will use ϱ as defined here.
7. Mean number of jobs in the system: $E[n] = m\rho + \rho\varrho/(1-\rho)$
8. Variance of number of jobs in the system:

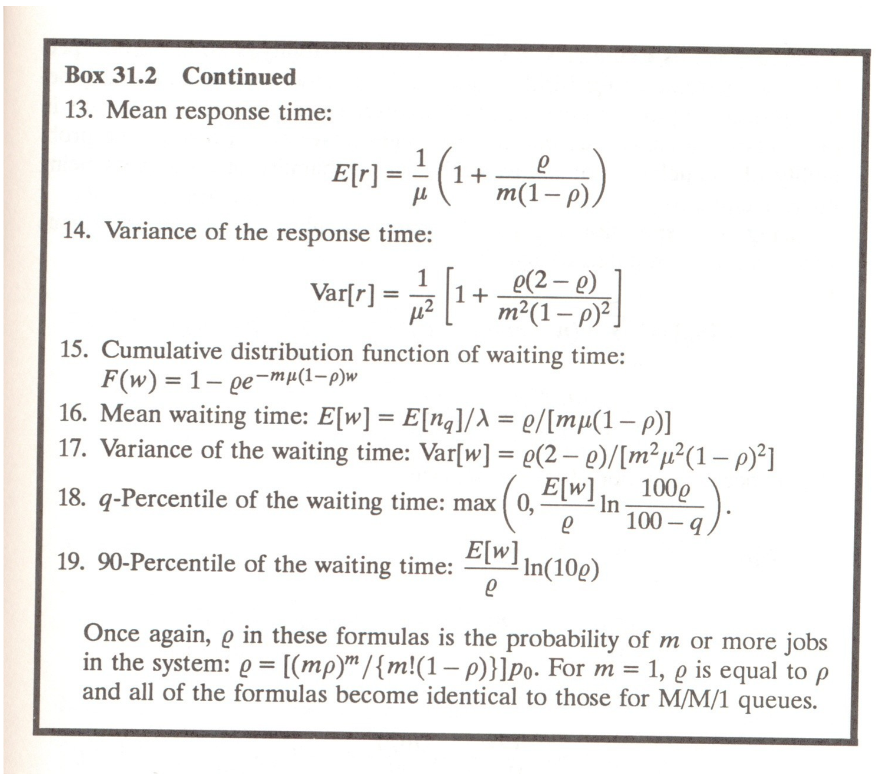
$$\text{Var}[n] = m\rho + \rho\varrho\left[\frac{1+\rho-\rho\varrho}{(1-\rho)^2} + m\right]$$

9. Mean number of jobs in the queue: $E[n_q] = \rho\varrho/(1-\rho)$
10. Variance of number of jobs in the queue:
    $\text{Var}[n_q] = \varrho\rho(1+\rho-\varrho\rho)/(1-\rho)^2$
11. Average utilization of each server: $U = \lambda/(m\mu) = \rho$
12. Cumulative distribution function of response time:

$$F(r) = \begin{cases} 1 - e^{-\mu r} - \dfrac{\varrho}{1-m+m\rho} e^{-m\mu(1-\rho)r} - e^{-\mu r}, & \rho \neq (m-1)/m \\ 1 - e^{-\mu r} - \varrho\mu r e^{-\mu r}, & \rho = (m-1)/m \end{cases} \quad r > 0$$

(Continued)

Figure 5.6: Performance measurements for M/M/m model



Box 31.2 Continued

13. Mean response time:

$$E[r] = \frac{1}{\mu}\left(1 + \frac{\varrho}{m(1-\rho)}\right)$$

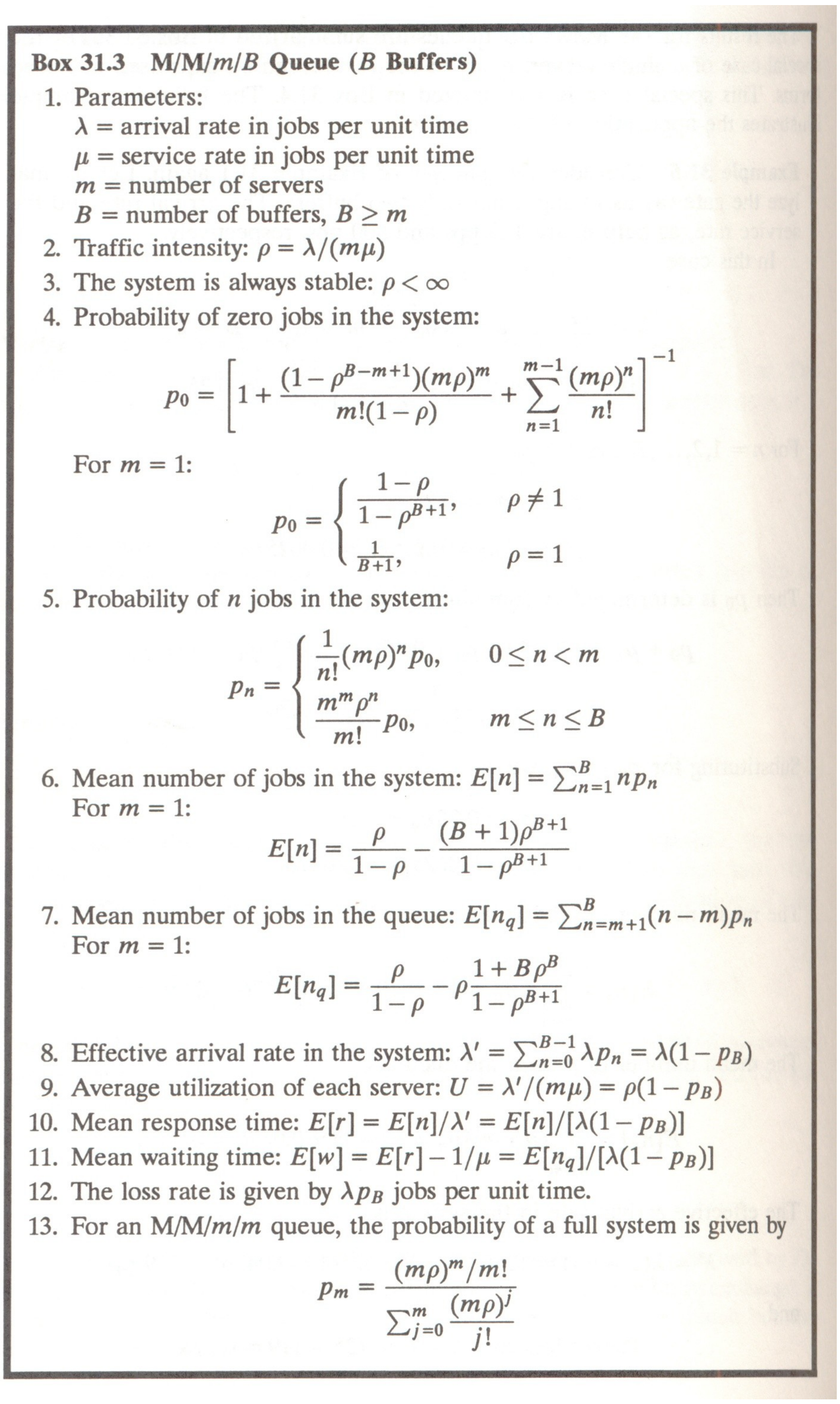
14. Variance of the response time:

$$\mathrm{Var}[r] = \frac{1}{\mu^2}\left[1 + \frac{\varrho(2-\varrho)}{m^2(1-\rho)^2}\right]$$

15. Cumulative distribution function of waiting time: $F(w) = 1 - \varrho e^{-m\mu(1-\rho)w}$
16. Mean waiting time: $E[w] = E[n_q]/\lambda = \varrho/[m\mu(1-\rho)]$
17. Variance of the waiting time: $\mathrm{Var}[w] = \varrho(2-\varrho)/[m^2\mu^2(1-\rho)^2]$
18. q-Percentile of the waiting time: $\max\left(0, \frac{E[w]}{\varrho}\ln\frac{100\varrho}{100-q}\right)$.
19. 90-Percentile of the waiting time: $\frac{E[w]}{\varrho}\ln(10\varrho)$

Once again, ϱ in these formulas is the probability of m or more jobs in the system: $\varrho = [(m\rho)^m/\{m!(1-\rho)\}]p_0$. For $m = 1$, ϱ is equal to ρ and all of the formulas become identical to those for M/M/1 queues.

Figure 5.7: Performance measurements for M/M/m model (Cont.)



Box 31.3 M/M/m/B Queue (B Buffers)

1. Parameters:
   λ = arrival rate in jobs per unit time
   μ = service rate in jobs per unit time
   m = number of servers
   B = number of buffers, $B \geq m$
2. Traffic intensity: $\rho = \lambda/(m\mu)$
3. The system is always stable: $\rho < \infty$
4. Probability of zero jobs in the system:

$$p_0 = \left[1 + \frac{(1-\rho^{B-m+1})(m\rho)^m}{m!(1-\rho)} + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!}\right]^{-1}$$

   For $m = 1$:

$$p_0 = \begin{cases} \frac{1-\rho}{1-\rho^{B+1}}, & \rho \neq 1 \\ \frac{1}{B+1}, & \rho = 1 \end{cases}$$

5. Probability of n jobs in the system:

$$p_n = \begin{cases} \frac{1}{n!}(m\rho)^n p_0, & 0 \leq n < m \\ \frac{m^m \rho^n}{m!} p_0, & m \leq n \leq B \end{cases}$$

6. Mean number of jobs in the system: $E[n] = \sum_{n=1}^{B} n p_n$
   For $m = 1$:

$$E[n] = \frac{\rho}{1-\rho} - \frac{(B+1)\rho^{B+1}}{1-\rho^{B+1}}$$

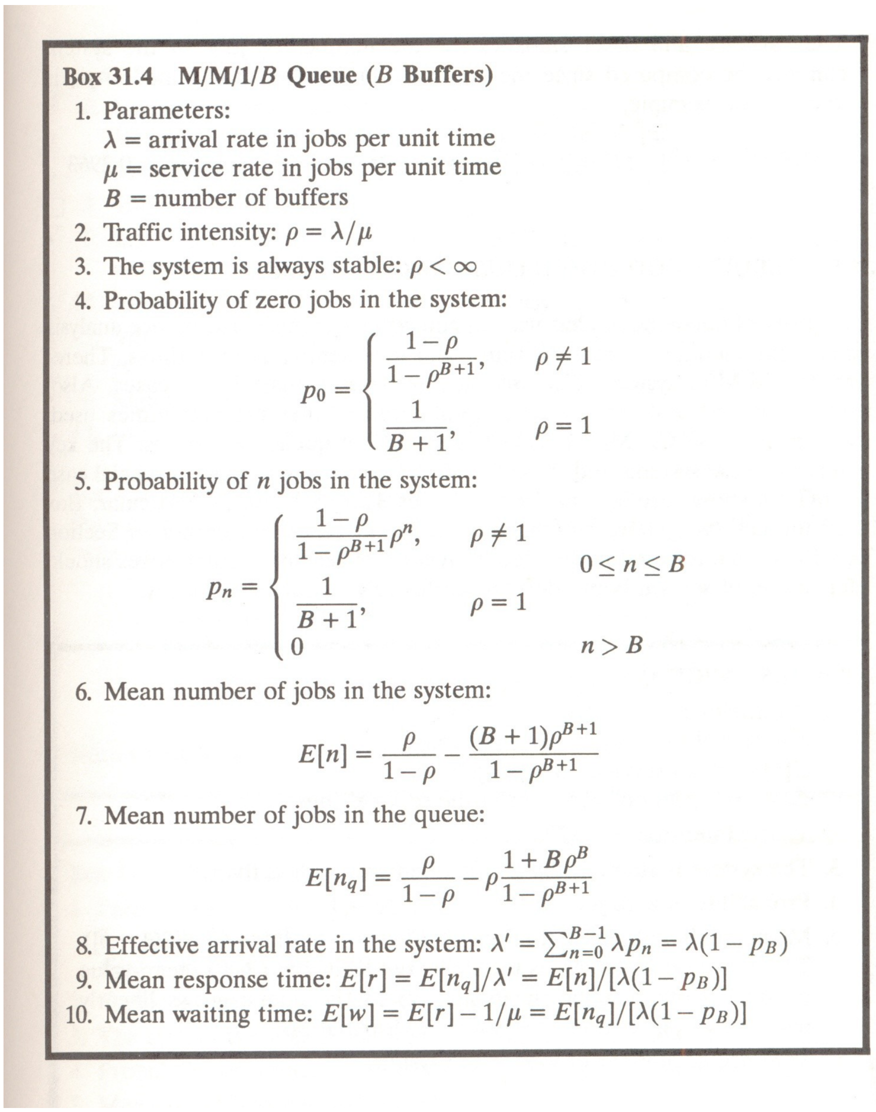
7. Mean number of jobs in the queue: $E[n_q] = \sum_{n=m+1}^{B} (n-m) p_n$
   For $m = 1$:

$$E[n_q] = \frac{\rho}{1-\rho} - \rho \frac{1+B\rho^B}{1-\rho^{B+1}}$$

8. Effective arrival rate in the system: $\lambda' = \sum_{n=0}^{B-1} \lambda p_n = \lambda(1-p_B)$
9. Average utilization of each server: $U = \lambda'/(m\mu) = \rho(1-p_B)$
10. Mean response time: $E[r] = E[n]/\lambda' = E[n]/[\lambda(1-p_B)]$
11. Mean waiting time: $E[w] = E[r] - 1/\mu = E[n_q]/[\lambda(1-p_B)]$
12. The loss rate is given by λp_B jobs per unit time.
13. For an M/M/m/m queue, the probability of a full system is given by

$$p_m = \frac{(m\rho)^m/m!}{\sum_{j=0}^{m} \frac{(m\rho)^j}{j!}}$$

Figure 5.8: Performance measurements for $M/M/m/K$ model



Box 31.4 M/M/1/B Queue (B Buffers)

1. Parameters:
   λ = arrival rate in jobs per unit time
   μ = service rate in jobs per unit time
   B = number of buffers
2. Traffic intensity: $\rho = \lambda/\mu$
3. The system is always stable: $\rho < \infty$
4. Probability of zero jobs in the system:

$$p_0 = \begin{cases} \dfrac{1-\rho}{1-\rho^{B+1}}, & \rho \neq 1 \\ \dfrac{1}{B+1}, & \rho = 1 \end{cases}$$

5. Probability of n jobs in the system:

$$p_n = \begin{cases} \dfrac{1-\rho}{1-\rho^{B+1}}\rho^n, & \rho \neq 1 \quad 0 \leq n \leq B \\ \dfrac{1}{B+1}, & \rho = 1 \\ 0 & n > B \end{cases}$$

6. Mean number of jobs in the system:

$$E[n] = \frac{\rho}{1-\rho} - \frac{(B+1)\rho^{B+1}}{1-\rho^{B+1}}$$

7. Mean number of jobs in the queue:

$$E[n_q] = \frac{\rho}{1-\rho} - \rho\frac{1+B\rho^B}{1-\rho^{B+1}}$$

8. Effective arrival rate in the system: $\lambda' = \sum_{n=0}^{B-1} \lambda p_n = \lambda(1-p_B)$
9. Mean response time: $E[r] = E[n_q]/\lambda' = E[n]/[\lambda(1-p_B)]$
10. Mean waiting time: $E[w] = E[r] - 1/\mu = E[n_q]/[\lambda(1-p_B)]$

Figure 5.9: Performance measurements for $M/M/1/K$ model

6 LIMITATIONS OF QUEUING MODELS

In this chapter we are going to review the limitations that the queuing models have in practice, and show how these models must be used with common sense.

As an illustrative example, suppose there are five people working in a group and all are connected through a LAN. Each one produces eight letters in electronic format on their PCs, which are sent to the common network printer. Now suppose that five more people are temporarily incorporated into that workgroup and connected to the network. Would the original five people now send only four letters a day to the printer? Suppose now that ten more people are connected to the network. Would that mean that the original five people and the other five who joined then now send only two letters? Well, that's exactly what the Poisson model predicts. However, common sense and experience tells us quite the opposite. Each new person incorporated will tend to send a similar number of documents to the printer as did each original member of the group. Perhaps there is more competition now, and each person tries to outdo the others by making and sending ten letters per day. You can also expect a large increase in email, since the more people in a group, the more emails. All this runs contrary to the Poisson model.

In short, before effectively using all the arsenal of tools offered by the queue models, you have to become familiar with the intrinsic limitations of each model. The indiscriminate use of these tools as 'cooking recipes' without taking into account the underlying assumptions can result in a network that does not work as expected and works badly.

6.1 Types of models

The models are *simplifications* of the real world that enable a parametric analysis of a problem, particularly when a large number of variables are involved. In this process, many variables and factors can be ignored, discounted, or simplified.

The purpose of a model is to interrelate the variables functionally, so that one is able to predict a quantity when some or all of the other variables change their values. Models are useful or desirable when you need to obtain an equation to predict behaviour. For example, you can have a model of the behaviour of certain substances and use it to predict how a mixture of this substance will behave with another substance. This in turn helps to understand how the product resulting from the chemical reaction works.

A model can be deterministic, probabilistic, or statistical.

A **deterministic** model provides a functional relationship (such as an equation) where the behaviour of the desired variable is fixed exactly by the knowledge of other variables. For example, if D represents the money (in euros) in a person's pocket, then the value of M can be fixed by the formula

$$D = 0.01p + 0.05c + 0.1d + 0.2v + 0.5m + 1e + 5u \tag{6.1}$$

where p would be the number of 1-euro coins that the person has in his/her pocket, c the number of 5-euro coins, v the number of coins of 20 coins, etc.

A **probabilistic** model is a model where the behaviour of the variable in which we are interested can be related to other variables only by a range (technically a probability distribution) of values or possibilities. The number of possible results becomes geometrically larger. Furthermore, given an observation, one cannot predict which of the possible outcomes will materialise. The useful aspect of probabilistic models is that if enough repetitions of the same experiment are performed, then it can be assured that certain aspects of those experiments are predictable. Communications in general, and the design of WAN communications in particular, are treated as probabilistic problems.

As an example, let's suppose that two people have two coins in their pocket each. What is the total amount of money they have together? Clearly, the problem is not deterministic, since it depends on the currencies. We can only predict a possible range of values for this response. Each person can have one of these 15 possible combinations:

$$\begin{aligned}
&2p\\
&2d\\
&2d\\
&2v\\
&2m\\
&1p, 1c\\
&1p, 1d\\
&1p, 1v\\
&1p, 1m\\
&1c, 1d\\
&1c, 1v\\
&1c, 1m\\
&1d, 1v\\
&1d, 1m\\
&1v, 1m
\end{aligned}$$

and therefore can have from 0.02 euros to 1 euro, along with 13 other possibilities. As the question refers to the total amount of money of both people, the multiplicative effect of the range of 15 possible combinations results in that if there are two people there are 15^2 possible combinations, with a total value from 0.04 up to two euros (although the total value of some combinations could match). As the number of variables grows (the number of people), the number of possible results increases geometrically (multiplicative).

Given a probabilistic situation, you cannot predict an exact amount, but only a range of values. In the same way, it cannot be predicted if exactly six SONET trunks are needed between two ATM switches; it can only be predicted that, for example, between four and nine trunks would be needed, or perhaps seven on average.

An intrinsic consequence of the probabilistic mechanism is that the observer cannot predict the exact result of a single situation. If enough repetitions of the same experiments are performed, then in general, it is certain that certain aspects of those experiments will be predictable. For example, you cannot say ‘when picking up the phone the delay in getting a PBX line will be 1.3076648 ms’, but something like ‘If you raise 20 times the handset the system will give you a dial tone in less than one second at least 15 times'. In the same way, you cannot predict the result we will obtain by throwing one dice. But we can say that by rolling the dice 600 times, 100 of those rolls will produce a '1'.

Finally, **statistical** models only provide certain empirical relationships that are corroborated, or simply documented by observation, without implying a cause-effect relationship.

6.2 Limitations of classic models

Understanding the limitations of the models used is crucial for the design task.

6.2.1 Assumptions that are rarely met

Below is a description of several assumptions that are intrinsic to many models and that tend to diverge from the real conditions found in networks.

Number of system users

Many models assume, for mathematical reasons, that the number of users who need access to a resource is infinite. Even assuming that all other assumptions made by the most common network models (such as Erlang B) are true, just this assumption can lead to drastically different results compared to the responses received when considering the size of the real population.

For example, with a PBX in Valencia that connects with dedicated lines to a PBX in Corunna the number of company users who call from Valencia to Corunna in a day is not only not infinite and not equal to the total workforce of the company, but it can be as few as a dozen employees or even fewer. Any calculation made with a current model that assumes an infinite population will sometimes lead to incorrect results. It turns out that some of these models have been used by large telephone companies, and for that reason they seem to be valid models also for small or medium voice networks. However, a model that may be appropriate for designing a network for a telephone exchange serving 20,000 users or a transatlantic cable serving 40,000 telephone conversations may be inappropriate for the 10, 20, or 30 users who use the resources on a daily basis.

Other more sophisticated models (such as the Engset model) take into account the finite nature of the user population, and lead to more realistic estimate of the resources needed.

Lost calls/users

Many network models assume that unattended service requests are lost, and that the user does not retry to obtain the service, when in reality these requests are rarely lost and can even be re-attempted in a few seconds. Examples of this include the traffic generated by computers and some terminals or switches. In other words, the model will suggest the use of less communications equipment that may actually be needed because of the number and frequency of service request retries. A user who needs to make a call will keep trying until he gets through. Therefore, a saturated network will cause repeated attempts.

Invariance in time

Another assumption of classical models is that traffic patterns are consistent over time, when in fact they vary intrinsically with the day of the week or with the time of day and with the source of the traffic. There is no guarantee that a given population will behave in an intrinsically stationary way. For example, on Monday and Friday mornings, traffic can skyrocket, as everyone tries to use resources.

The clearest example that the Poisson M/M/1 model and similar models fail is the arrival rate at a fast food restaurant near an office complex: the distribution of arrivals will be practically zero between before lunch. However, at around 13:30 or when the companies in the complex let their employees go for lunch, the arrival rate will shoot up.

Some companies or industries are affected by seasonal factors, particularly in the tourism sector. If the seasonal factor is substantial and predictable it can be incorporated into the design. For example, additional lines can be designed in June-September in a travel agency for the summer season.

The traffic generated by computers can be almost deterministic with high load, and can be more bursty with lower loads. This limitation can be overcome by using time-dependent queuing models, although in practice this is not done.

Mix of different types of traffic

In most situations, traffic can come from radically different sources, such as voice and data inputs to a PBX, or as local telephone traffic, and traffic from a national trunk and an international trunk. Classic models do not take into account this mixture of traffic. The waiting times can be a function of the time of day, content of the message, or other factors. Traffic patterns in business centres differ from traffic patterns in semi-rural areas. For example, the sizing of public and private ATM switches varies as they have to support traffic from sources as diverse as terminals, workstations, servers, LAN traffic, voice storage, file transfers from PCs, and video. The assumption of a single type of traffic clearly needs to be reviewed.

Independence of users

Another very common assumption in many models is that users are independent of each other, when the opposite is true. For example, consider the problem of determining the number of lines needed between the central node of a customer service centre and a terminal node in a communication network with a star configuration. All the customer data goes to a hub in a remote city. A failure in this hub or in the data communication line would cause all customers in that city to call the customer service centre. These calls would not be independent of each other. They would all share the same cause.

The vast majority of models assume that the members of the population under study are totally independent of each other. There are two reasons for such assumption:

1. Mathematical simplicity of the model solution
2. The independence condition provides uniqueness. Dependence cannot be expressed in a single formula, and there may be many possibilities.

Many systems that could reasonably be classified as independent, such as trunk traffic arriving at a switchboard, lose their independence under certain circumstances. For example, on Christmas Eve or on Mother's Day arrivals on the backbone are not independent, because the population is directed by a social mechanism that dictates certain behaviors.

Dependency is the reason why it is sometimes impossible to telephone over a public network, or the call tone takes a long time during a localised disaster, or at midnight on the

1st of January. When a locality suffers a disaster, many people in distant places try to call the locality in question.

A particular type of dependency is autocorrelation, where the traffic is related to itself on the time axis (this condition is related to the condition of ‘without memory’). Most events are dependent on something that happened earlier in time, instead of existing in a vacuum, as the models suggest. For example, it is reasonable to think that the outgoing traffic of a telephone exchange is autocorrelated with the incoming traffic.

Any model ‘without memory’ assumes that what happens in the following time intervals is totally independent of what happened in the past intervals.

Stable state

The most commonly used models assume that networks operate in a stable state. However, most communication systems work in a transient mode. The most illustrative analogy would be a model of the surface of a pond when a stone is thrown. After a transitory period (a few minutes), the surface will be flat. The network model models that stable state, but in reality, it is that users throw stones every few seconds, and some very flat stones.

6.2.2 Conclusions on models

Everything presented above does not mean that analytical models are not useful. It is only to emphasise that they have certain limitations. It is the common sense of the engineer that should be used to apply the appropriate model to each case. When evaluating network models it is important to take into account the benefits sought. For example, it would be absurd to optimise a network by trying to use a more accurate model when the input traffic component may have an error of 20 %. Some try to tune their network design to increase the accuracy of their model by 5 % while other components may have an error of up to 20 %.

Another very important factor is the validity of the available data. For example, having three different sources of data does not guarantee the independence of the sources, which could be three copies of the same data. Therefore, do not rely on the number of different sources.

In summary, to design networks using models, the following factors must be taken into account:

- The most widespread models are very approximate. Any design based on these models should be taken as a first approximation that should be refined later.
- If possible, a more precise model should be used or derived. This may involve additional data and computer programs to solve numerical problems.
- You must ensure that all variables used in a model have the same level of accuracy. If 99 items have an accuracy of 95 %, but one piece has a precision of 20 %, it is very likely that the final answer has only a precision of 20 %.
- Routers, PBX, switches, network monitors, and other equipment provide excellent measurements. It may be interesting to use measures of real scenarios rather than a hypothetical environment derived from an abstract model.

7 TELETRAFFIC

In this chapter, we will introduce other queue models that are telecommunications systems, and hence their name. First of all we need to define some basic concepts:

The amount of **traffic offered** to a trunk or other telecommunication systems may not equal the amount of *traffic delivered* by that system. Some of that traffic may be blocked or delayed. If T_o is the offered traffic, T_c is the traffic delivered (the c comes from *carried*), and b is the blocked traffic, then the following relations are kept:

$$\begin{aligned} T_c &= (1-b) \times T_o \\ b &= (T_o - T_c)/T_o \end{aligned}$$

Blocked traffic (or overflow or overflow traffic) must be managed by a secondary system.

7.1 Erlang-B model

The Erlang-B model is the model most commonly used in teletraffic, since it corresponds to the paradigm of communication where blocked calls are lost to the system (in the telephone system, when someone calls and the system does not manage that call, it is lost), there is no ‘call buffer’. To avoid those traffic and benefit losses, voicemail was invented). The Erlang-B model corresponds to the $M/M/m/m$ model, that is, a particular case of model $M/M/m/K$ where $K = c$. That is, the number of spaces in the queue matches the number of servers. There is, therefore, no space in the waiting queue, and a user that requires service either finds a free server or leaves the system. Therefore, there are no delays in the system, only blockages, since there cannot be more users than servers. The Erlang-B model has been used in the design of telephone exchanges since 1917. Each server is a trunk of a set of c trunk. When a user requires a trunk, either one of the c trunks will be used or the user will be rejected. According to this model, the probability that c servers (channels) are busy is shown in equation 5.28, and this is easily programmable in a PC. Given its importance, this has been tabulated. A small portion is shown in Table 7.1. There are three variables in this table: the number of channels c; the probability p_c that all of these channels are in use; and the amount of traffic offered to the entire queue system (each value in the table). Knowing the value of two of these three variables, we can derive the value of the third variable with this table.

Table 7.1: Part of Erlang B table

c	p_c=0.001	p_c=0.01	p_c=0.05	p_c=0.1
5	0.76	1.36	2.22	2.88
10	3.09	4.46	6.22	7.51
15	6.08	8.11	10.6	12.5
20	9.41	12.0	15.2	17.6

For example, if the traffic arriving to the system and the number of channels is known, we can obtain the probability that an incoming call finds all the channels occupied. Thus, if 12.5 units of traffic reach a PBX with 15 channels, the table tells us that the percentage of calls that cannot be served by the PBX will be 10 % (we search in the row of c = 15 for the value closest to 12.5, which turns out to be the column of p_c = 0.1, that is, a probability of 10 %).

If the incoming traffic is 4.46 units and the designer wants only 1 % of the calls to be lost, the number of necessary channels will be 10 (we look for the first row in the column of p_c = 0.01) that is equal to or greater than 0.46, which is the number of channels c = 10).

Finally, if the number of channels and the probability of losses are fixed, the amount of traffic that the system can handle can be calculated. For example, one trunk accommodates 20 channels and the percentage of losses is set at 5 %. How much traffic can this trunk handle? The table tells us that it can handle 15.2 units (erlangs).

8 QUEUE NETWORKS

8.1 Introduction

Most computer systems, including computer networks, have several congestion points caused by sharing various resources. In these cases, it is difficult and sometimes too restrictive to represent the behaviour of the complete system by means of a single station or queue model such as those we studied in the previous chapter. It seems appropriate to explicitly model the different congestion points of the system. The resulting model is a network of queues, that is, a set of interconnected service stations, through which jobs circulate following a deterministic or probabilistic pattern.

Formally, we can define a network of queues as a directed graph whose nodes are the service stations. The edges between these nodes indicate the possible transitions between service stations. The jobs that circulate through the network can be of different classes. The jobs of different classes can follow different routes through the network.

In general, the study of this complex probabilistic model is much more difficult than that of an isolated service station. There are, however, many special cases, characterised by certain interconnection structures of the stations, and by various distributions of the processes of arrival and service, which have been analysed by different authors and where analytical solutions been found through relatively simple mathematical techniques. In the next sections we will study some of these techniques.

8.2 Network types

Networks can be classified according to the types of jobs that circulate through their stations.

1. If in all stations of the network the works have the same (random) behavior, both in terms of service times and in terms of the path they follow when leaving the station, we say that the network has *unique kinds of jobs*. All works of the same class are statistically indistinguishable.
2. In a network with *multiple kinds of jobs* works of different kind can have various service time characteristics and follow different routes through the network.

Networks can also be classified according to the topology of the underlying graph:

1. Open networks. They are characterised by (Fig. 8.1):
   - The existence of at least one source of work and one or more sinks that absorb the jobs that leave the system.
   - The possibility of finding a path that, from each node, leads (eventually) outside the network.

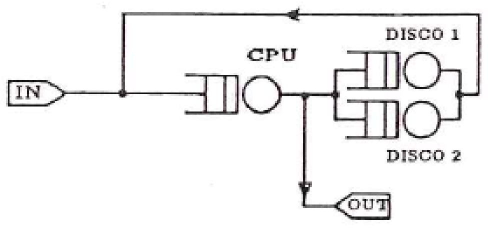


Figure 8.1: Example of an open network

In this type of network, the number of jobs in the system varies over time. The productivity (throughput) of an open network is usually a known parameter, since it is equal to the rate of entry to the system.

2. Closed networks (Fig. 8.2). In these networks, jobs neither enter nor leave the system, and the number of jobs within remains constant. In some cases it is interesting to contemplate a closed system as a system in which the output is linked to the input, so that the jobs that 'come out' of the system immediately 'return' to it. With this vision of the system, the workflow through the link 'exit-entry' defines the productivity of the closed network.

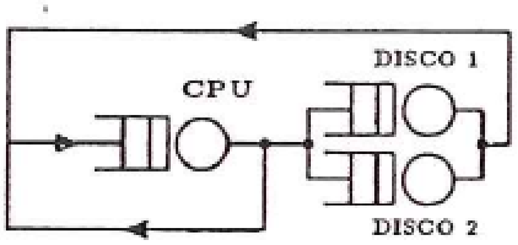


Figure 8.2: Example of closed network

In summary, the behaviour of the job in the system or the network is characterised by the *transition probabilities* between service stations and the distribution of times in each station. For each station, the number of servers, the service discipline and the capacity of the station must be specified. The system or network contains K stations or devices. The outside or exit of the network, that is, the own network of queues, is indicated as device 0 (zero).

8.3 Operational laws

A large number of problems related to performance analysis in computer networks can be solved by several simple relationships that do not require any hypotheses about distributions of service times or arrival times. These relationships, known as *operational laws*, were originally obtained by Buzen and Denning (1976-78). In this context, the word 'operational' means directly measurable.

Thus, an *operationally testable hypothesis* is a hypothesis that can be checked by taking measurements of the system. For example, in a system it is easy to verify the hypothesis that the number of arrivals at the system is equal to the number of departures from the system in a time interval. Therefore, this hypothesis (called a balanced workflow) is operationally verifiable. However, it cannot be established by measurement if a set of times for observed services form or not a sequence of independent random variables. Therefore, the independence hypothesis is not operationally verifiable.

The *operational laws* are therefore simple laws which:

1. are not based on considerations of time-of-service or arrival distributions.
2. result from applying elementary algebra (type f = ma).
3. are called ‘laws’ because they are met without consideration.

The *operational amounts* are directly measurable amounts during a period of finite observation. Consider a device or station i of a teleinformatic system as a queue similar to that shown in Figure 8.3. This device can be seen as a black box, from which we can take several measures:

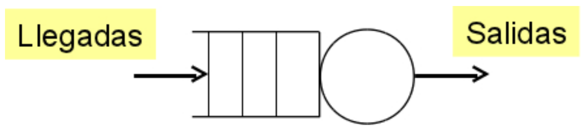


Figure 8.3: Device in a queue network

- Let the temporary variable T be the duration of the observation interval or measurement period of the system (time).
- Let the variable A_i be the number of jobs or requests that arrive (arrivals) to the device i.
- Let the variable C_i be the number of completed requests (completions) during the interval T on the device i.
- Let the variable B_i be the time that the device i has been busy (busy time) during the interval T.

From these operational magnitudes, we can derive the following magnitudes for each device i:

- Arrival rate λ_i = arrivals number / observed time = $\frac{A_i}{T}$.
- Throughput X_i = Number of completed jobs / observed time = $\frac{C_i}{T}$.
- Utilisation U_i = time spent / observed time = $\frac{B_i}{T}$
- Average service time S_i = time occupied / num. jobs. complet. = $\frac{B_i}{C_i}$
- Reason for visit V_i = Number of completed works in i / num jobs. completed by the system = $\frac{C_i}{C_0}$
- Demand time D_i = $V_i \times S_i$

Analogously, we can enunciate and derive the same variables for the entire system or network of queues:

- A_0 = Number of arrivals (arrivals) to the system in the interval T.
- C_0 = Number of completed jobs (completions) or that leave the system in the range T.
- λ_0 = Rate of arrivals (arrival rate) of the system = $\frac{A_0}{T}$.
- Productivity (throughput) X_0 = number of jobs completed by the system / observed time = $\frac{C_0}{T}$.

It is important to list some important details of these magnitudes or operational quantities:

1. The deduced variables are average values.
2. The utilisation of a device (and the entire system) is between 0 and 1.
3. The service time is the time a job spends on the device server.
4. The visit ratio V_i indicates how many times a job visits a particular device.
5. The service demand D_i does not take into account the possible wait in queue and represents the load that a job causes in the station i

With these operational quantities, we can state the *operational laws* or relationships that are maintained for each observation period.

8.3.1 Balanced workflow hypothesis

If the observation period T is such that $\forall\, i$ is satisfied that

$$A_i \;=\; C_i \tag{8.1}$$

then we say that each device i satisfies the hypothesis of balanced work flow. Taking a sufficiently large observation period T, it is possible that $A_i - C_i$ are negligible in relation to C_i, and therefore the hypothesis is approximately correct.

Note that $A_i \;=\; C_i$ implies that $\lambda_i \;= X_i$

8.3.2 Law of utilisation

The law of utilisation says that

$$U_i \;=\; \frac{B_i}{T} \;=\; \frac{B_i}{C_i}\cdot\frac{C_i}{T} \;=\; S_i X_i \tag{8.2}$$

And if the hypothesis of balanced work flow is also fulfilled, then

$$U_i \;=\; S_i \lambda_i \tag{8.3}$$

8.3.3 Law of forced flow

The law of forced flow relates the throughput of the system to the throughput of an individual device. In an open system, throughput is defined by the number of jobs that the system leaves per unit of time. However, in a closed system no work leaves the system. However, the jobs that cross the link that connects the exit of the system with the entrance are as if they left the system and immediately came back to it again. Therefore, the throughput of the system in closed systems is measured as the number of jobs that cross this link per unit of time.

The total throughput of the system during the observation period is

$$X_0 = \frac{C_0}{T} \tag{8.4}$$

and the throughput of the i-th device is

$$X_i = \frac{C_i}{T} = \frac{C_i}{C_0} \cdot \frac{C_0}{T} = X_0 V_i \tag{8.5}$$

Therefore, the law of forced flow says that

$$X_i = X_0 V_i \tag{8.6}$$

That is, it establishes that the workflow through a determined device in the network determines the flow in any other device. This law is valid if is the hypothesis of balanced work flow.

Combining the law of forced flow (equation 8.6) and the law of utilisation (equation 8.2) we have that

$$U_i = S_i X_i = X_0 V_i S_i = X_0 D_i \tag{8.7}$$

where $D_i = V_i S_i$ is the *total demand for service* on the device i in all visits that a worker makes to that device. Equation 8.7 states that the use of each device in the system is proportional to its service demand. Therefore, the device with the highest demand D_i will have the highest utilisation and will be the device **bottleneck** of the system.

Visit ratios are a way to specify the routing of jobs through the network. Another alternative form is the **transition probabilities**, p_{ij}, which specify the probability that a job will pass to station j after finishing at station i. Visit ratios and transition probabilities are equivalent in the sense that the latter can be derived from the former and viceversa.

In fact, in a system with M devices and with a balanced flow, it is fulfilled that

$$C_j = \sum_{i=0}^{M} C_i p_{ij} \tag{8.8}$$

where the subscript 0 represents the world outside the system and where p_{i0} is the probability that after receiving service in the station i a job leaves the network. Dividing both terms of the above equation by C_0 we obtain that

$$V_j = \sum_{i=0}^{M} V_i p_{ij} \tag{8.9}$$

which are the equations of the visit ratios. Since each visit to the outside world corresponds to the completion of a job, we have that the visit ratio to the network of queues for any job is

$$V_0 = 1 \tag{8.10}$$

8.3.4 Little's law

Little's law assumes compliance with the hypothesis of a balanced flow of jobs, and relates the number of jobs in the system with the time spent and their productivity or arrival rate. This law can be applied to different levels of the system.

If we denote as N_i the number of jobs from the station i and R_i the response time of station i, then Little's law says that

$$N_i = \lambda_i R_i \tag{8.11}$$

And since we have a balanced work flow, we have

$$N_i = X_i R_i \tag{8.12}$$

8.3.5 General response time law

This law is independent of the type of system (open or closed), and it only considers the visit ratios and the response times of each station.

Let N be the number of jobs in a queuing network consisting of M stations (queues). N can be calculated as

$$N = N_i + N_2 + \cdots + N_M \tag{8.13}$$

And applying Little's laws

$$X_0 R = X_1 R_1 + X_2 R_2 + \cdots + X_M R_M = \sum_{i=1}^{M} X_i R_i \tag{8.14}$$

Applying the forced flow law, we have that

$$X_0 R = X_0 V_1 R_1 + X_0 V_2 R_2 + \cdots + X_0 V_M R_M = \sum_{i=1}^{M} X_0 V_i R_i \tag{8.15}$$

and dividing both sides of this equation by X_0 we obtain

$$R = V_1 R_1 + V_2 R_2 + \cdots + V_M R_M = \sum_{i=1}^{M} V_i R_i \tag{8.16}$$

which is the general response time law. This law shows that the permanence time of a job in a system depends on the number of visits made to each device and the response time experienced in each of the visits.

8.3.6 Interactive response time law

Any interactive system can be split into two sub-systems, as shown in Figure 8.4. In these systems, we have N terminals (where we assume one terminal per user) and the central subsystem containing the rest of the devices.

The operation of the system is as follows: users generate requests (one per terminal) that are served in the central subsystem, and subsequently return to the terminals. After a *thinking time* Z, the users generate the following request.

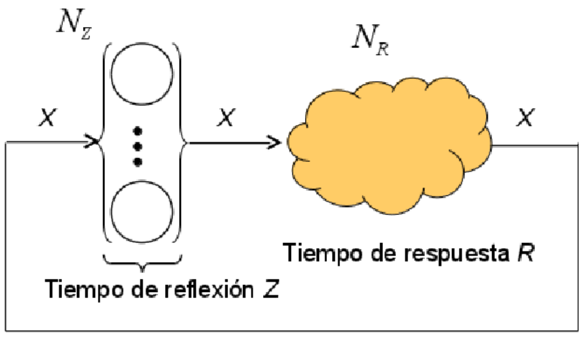


Figure 8.4: Example of interactive system

Thus, the law of interactive response time is obtained by applying Little's law to a computer system when the load is interactive ($Z > 0$) or batch ($Z = 0$). The number of jobs in the system is $N = N_Z + N_R$, one per terminal. We can apply Little's law to the central sub-system and to the set of terminals. The average time a work remains in the set is equal to the time it remains in the terminals, Z, plus the time that it remains in the central subsystem, R. Therefore, according to Little's law,

$$N = (Z + R)X_0 \tag{8.17}$$

and so

$$R = \frac{N}{X_0} - Z \tag{8.18}$$

8.3.7 Bottleneck analysis

A consequence of the law of forced flow is that the uses of the devices are proportional to their service demands. As we have seen, the device with the highest service demand will have the highest utilisation, and is called the **bottleneck** device, and is the device that limits the productivity (throughput) of the system. Improving the performance of the bottleneck device will offer greater benefits to the system than improving the performance of any other device, since it will enable the entire system to reach a higher throughput.

We will focus on an interactive system. We will denote the bottleneck device by the suffix b. Thus,

$$\begin{aligned} D_b &= max\,\{D_1, D_2, \cdots, D_M\} \qquad (8.19) \\ U_b &= X_0 D_b \qquad (8.20) \end{aligned}$$

Since U_b nor any other utilisation can be greater than one,

$$X_0 D_b \leq 1 \;\Rightarrow\; X_0 \leq \frac{1}{D_b} \qquad (8.21)$$

However, due to the interactive response time law

$$X_0(N) \;=\; \frac{N}{R(N)+Z} \qquad (8.22)$$

Where $R(N)$ is the response time of the central subsystem with N jobs and $X_0(N)$ is the system throughput with N jobs. The best response time of the central subsystem will be obtained when there is only 1 job in it, since such work will not have to wait in the queue of any device. Then,

$$R(1) \;=\; D_1 + D_2 + \cdots + D_M = D \qquad (8.23)$$

Where D is the sum of all the service demands. If there is more than one customer, queues may form and the response time will be longer, that is, $R(N) > D$. Therefore,

$$X_0(N) \;=\; \frac{N}{R(N)+Z} \leq \frac{N}{D+Z} \qquad (8.24)$$

And we have that the system throughput limits are given by two straight lines or asymptotes:

$$X_0(N) \;\leq\; min\left\{\frac{1}{D_b}, \frac{N}{D+Z}\right\} \qquad (8.25)$$

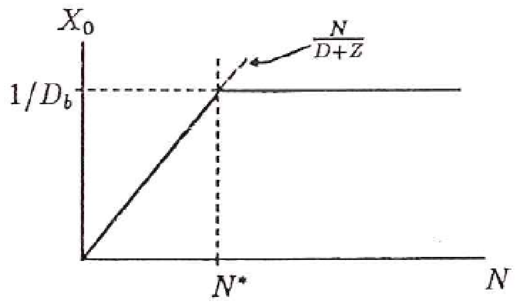


Figure 8.5: Asymptotic throughput bounds

Graphically, these lines are those that appear in Figure 8.5. We will denote the intersection point as N^* or knee, and it denotes the minimum number of tasks with which the maximum system throughput is reached.

In the same way, the limit for the response time can be obtained:

$$R(N) = \frac{N}{X(N)} - Z \geq ND_b - Z \tag{8.26}$$

and then

$$R(N) \geq max\{D, ND_b - Z\} \tag{8.27}$$

Figure 8.6 shows these asymptotes. In this case, the knee N^* denotes the maximum number of jobs with which we can obtain the shortest response time.

To calculate the point of intersection of the lines in both graphs, we only have to match the lines that intersect: from equation 8.25 we can derive that

$$\frac{1}{D_b}, = \frac{N^*}{D+Z} \Rightarrow N^* = \frac{D+Z}{D_b} \tag{8.28}$$

and from equation 8.27 we can also derive that

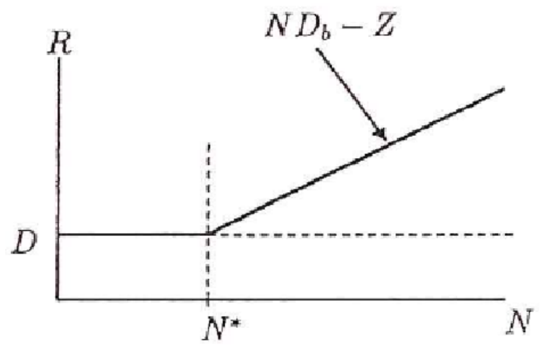


Figure 8.6: Response time asymptotic bounds

$$ND_b - Z = D \Rightarrow N^* = \frac{D + Z}{D_b} \tag{8.29}$$

8.4 Operational analysis of open networks

The open networks on which this analysis can be applied are composed of two types of stations:

1. Stations with a single server and exponential time distribution: **queue** type stations.
2. Stations with infinite servers and exponential time distribution: **delay**type stations.

The parameters that we will assume as known values are V_i and S_i for $i = 1, 2, \dots, M$. We will suppose that the arrival process is Poisson with an average arrival rate $\lambda < \lambda_{saturation} = 1/D_b$, and that the workflow is balanced.

With this input data, we want to solve the network and obtain the following output values:

- For each station: X_i, N_i, R_i, U_i.
- For the network: R, N

Before studying the algorithm for this analysis, it should be noted that for all stations with fixed service capacity of a queue network, the response time is given by the equation

$$R_i = S_i(1 + Q_i) \tag{8.30}$$

This equation is understood when considering what a job finds when arriving at station i. It will see that there are Q_i jobs in front (including the one(s) being served at that moment), and therefore it will have to wait Q_iS_i seconds to be served. And of course, its service will require S_i seconds more. At this point it should be noted that this reasoning assumes that the service time is exponentially distributed, that is, it lacks memory. Therefore, it is not necessary to take into account the service time already received by the work that is on the server when the arrival occurs. But since this property cannot be checked operationally, equation 8.30 is **not** an operational law.

So, using this equation the algorithm for solving open networks is the following:

1. From the balanced flow hypothesis, $X_0 = \lambda$.
2. From the law of forced flow, $X_i = X_0V_i$.
3. By the law of utilisation, $U_i = X_iS_i$.
4. By Little's law, $N_i = X_iR_i = \begin{cases} X_iS_i(1+N_i) = U_i(1+N_i) \Rightarrow N_i = \frac{U_i}{1-U_i} & queue\ station \\ X_iS_i = U_i\ delay\ station \end{cases}$
5. We calculate the response time in each device as
$R_i = \begin{cases} S_i(1+N_i) = S_i(1+\frac{U_i}{1-U_i}) = \frac{S_i}{1-U_i} & queue\ station \\ S_i\ delay\ station \end{cases}$
6. By the law of the general response time, $R = \sum_{i=1}^{M} R_iV_i$
7. Finally, $N = \sum_{i=1}^{M} N_i = \lambda R$.

8.5 Mean value analysis

The method of the mean value analysis is applied for the resolution of closed networks. Like the open network resolution algorithm, **it is only used to calculate the average values**

of network performance indices. The algorithm cannot be used to calculate distributions of the number of jobs or dwell times.

This algorithm assumes, like the previous one, that networks are formed of two types of stations: queue type and delay type. The parameters that we will assume as known values are V_i and S_i for $i = 1, 2, \ldots, M$ and N. We will assume, in addition, that the work flow is balanced.

With this input data, we want to solve the network and obtain the following output values:

- For each station:X_i,N_i,R_i,U_i.
- For the whole network:X_0,R

Before studying the algorithm that allows this analysis, we have to introduce a new relationship for the queue stations:

$$R_i(N) \;=\; S_i[1 + N_i(N-1)] \tag{8.31}$$

where $N_i(N-1)$ is the number of jobs in station i when there are $N-1$ jobs in the closed network. This relationship expresses that the state of the network as seen by a work that is in transit from one station to another has the same distribution as the state that a random observer would see if the total number of jobs in the network was $N-1$. Although this explanation is purely intuitive, we will omit the formal proof of equation 8.31, since it is quite complex.

It is important to note that equation 8.31 relates two performance indices, one for the state where there are N jobs in the network, and another for the case when there are $N-1$ jobs in the network. Therefore, this relationship gives rise to a recursive algorithm. Since for $N = 0$ it is satisfied that $N_i = 0$, we have that $R_i(1) = S_i$, $i = 1, \ldots, M$. For the delay type stations, it is also true that $R_i(N) = S_i, \forall N$.

With these initial values, the algorithm is as follows:

1. The total response time of the system is calculated using the law of the general response time:

$$R_0(N) = \sum_{i=1}^{M} V_i R_i(N) \tag{8.32}$$

2. By using Little's law, the system throughput is calculated:

$$X_0(N) = \frac{N}{R_0(N)} \tag{8.33}$$

3. The forced flow law is used to compute the throughput for each device:

$$X_i(N) = X_0(N)V_i \tag{8.34}$$

4. The utilisation law is then used to calculate the utilisation of each device:

$$U_i(N) = X_i(N)S_i \tag{8.35}$$

5. By using Little's law, we calculate the average number of jobs in each station:

$$N_i(N) = X_i(N)R_i(N) = X_0(N)V_iR_i(N) \tag{8.36}$$

6. With these $N_i(N)$ values, we compute $R_i(N+1)$ through equation 8.31 and closing the recursive algorithm in this way: $R_i(N+1) = \begin{cases} S_i[1+N_i(N)] \; queue \; station \\ S_i \; delay \; station \end{cases}$

7. Using these $R_i(N+1)$ values, go to step 1.

Bibliography

[1] Jeremiah F. Hayes, Thimma Ganesh Babu, *'Modeling and Analysis of Telecommunication Networks'*, Ed. Wiley Interscience, 2004.

[2] Gilbert Held, *'Enhacing LAN Performance'*, Ed. Wiley & Sons, 3¿¿ ed., 2000.

[3] Raj Jain, *'The Art of Computer Systems Performance Analysis'*, Ed. Wiley & Sons, 1991.

[4] D. Minoli, *'Broadband Network Analysis and Design'*, Ed. Artech House, 1993.

[5] J. J. Pazos, A. Su¿rez, R. D¿az, *'Teor¿a de Colas y Simulaci¿n de Eventos Discretos'*, Ed. Prentice-Hall, 2003.

DEPARTAMENTO DE INFORMÁTICA
E.T.S.E.

Grado de Ingeniería Telemática
Planificación de Redes

Lesson 4

Network Design

Prof. Juan Manuel Orduña Huertas

Contents

1 ANALYSIS AND DESIGN OF DATA NETWORKS

The design, installation, and start-up of a data network (of computers) comprises various stages that must be studied in detail to provide a network that complies throughout its life time with the services that the client demands. These stages are the following:

1. Requirements specification and selection of technology.
2. Selection of service provider (if applicable).
3. Design of backbone network and access network.
4. Design of network security aspects.
5. Management and documentation.

In this chapter, we will study in detail each of these aspects or phases.

1.1 Requirements specification

The correct analysis of the requirements is the first and most important step in the design of a network. However, before being able to determine the requirements of the network, it is necessary to understand the requirements and challenges of the company or institution where the network is going to be installed. In fact, in many cases the network itself is an extension of the business or company. Therefore:

- There should be a direct correlation between the objectives of the company and those of the network and people who maintain the network. The network is built for end users, and therefore there is no design aspect that is more important than fully understanding the needs of the users.

- The two main points of view of the requirements are from the user and from the designer. The user looks at the network from outside, and the designer looks at the network from the service provider (backbone network).

- It is important to understand that customer satisfaction depends mainly on how the designer works with the user. Even the best designs will always have some problems; but conflicts can be reduced to a minimum if, from the beginning, clear expectations of what is expected from the network are properly detailed, and the designer establishes a good working relationship with the user from the first day.

1.1.1 Business requirements and specifications

Before the network design can begin, the network designer must understand the technical and business challenges. The key business requirements, as well as the typical business strategies to achieve these requirements, are listed in Table 1.1

Requirement	**Strategy**
Business expansion	Mergers and acquisitions Globalisation
Financial control	Expenditure control Return on investment in 18 months
Competitiveness	Adaptation to technological change Use of e-commerce initiatives Improvement and advanced functionality of website
Business coverage	Support and connection of corporate users Access to remote / mobile workers
Security	Network security policies
Quality in customer support	Speed and instant gratification Customisation / customer adaptation

Table 1.1: Business requirements and strategies

Taking into account these business requirements, each option and/or network technology must be evaluated based on its cost and impact on the business. This impact on the business is commonly known as *return on investment or ROI*).

1.1.2 Requirements and technical specifications

The network designer must also understand the technical requirements of the company. Table 1.2 shows some examples of what could be the technical requirements of a company.

Requirement	**Example / comment**
Scalability	Network can grow to support mergers/growth of company
Flexibility	Network can incorporate other systems with different design
Accessibility	Geographical access limitations. Access methods
Reliability	Network very rarely (or ideally never) must be out of service
Performance and network productivity (throughput)	
Security	Ensure that network is protected against unwanted accesses
Legacy systems	Transparent access to old systems / networks
Storage	Storage plan and data restoration
Quality of service	Users must always experience the same quality of service

Table 1.2: Examples of technical requirements

It is important to note that the goal of the network designer is to determine how the network can provide significant results to the business or company. Therefore, when the network designer communicates with the management of the company, he must ensure that these results are transferred to management through specific and measurable results. Table 1.3 shows some examples of significant results that companies can achieve and how those objectives are specified.

1.1.3 User and application requirements

Before starting the process of network design, there are two initial steps that the network designer must perform: build and document an inventory of the current network (if it exists), and discover and document the expectations of the client about the new network. In this section we will substitute the word ‘client’ for the term ‘end user’.

Network hardware manufacturer Cisco Systems recommends making an inventory of 12 items from the customer’s current network. These 12 items are listed in Table 1.4. The client should provide most of this information, but the harsh reality is that clients rarely

Result	**Specification**
Shorten in days / weeks the process of production of goods or services	Enable launching products in 5-6 months
Improve cash management	Cut inventory time x % Process and send invoices 10 times faster Reduce searches from 5 minutes to 15 seconds
Enable business scalability	Enable the integration of new devices and applications Enable integration of extranets of new partners and clients
Improve communications	Enable email Enable online sales Enable instant messages Electronic sales improve profits by 20 %

Table 1.3: Examples of significant results that networks can bring

have documentation with this level of information. For this reason, the designer must often collect this information on his own, using a protocol and application analyser.

Once the current network has been inventoried and documented, the next step is to understand the client's expectations. There are five categories in which to collect information from the client:

- Business restrictions include items such as approval of projects, budgets, training, personnel or temporary restrictions. The client must express what he expects with respect to the time that the network will be inoperable during its installation and what he expects from the implementation of the network. What is the cost of network downtime?
- Security requirements are critical for most companies. Who can access each resource on the network? What should be accessible and when? The level of security of each resource must be determined.
- The management requirements can vary from documenting through a PING that every device in the network is alive, to complete network management systems with intelligent management engines that relate alarms to certain network events and recommend actions to be taken.
- Future application requirements that will result in protocol requirements and future benefits.

Inventory item	**Description**
Existing applications	Applications running on the network
Existing network protocols	Protocols running
Network topology and addressing	Topology map (*all* network devices)
Potential bottlenecks	Monitor bandwidth and traffic patterns
Business constraints	Political factors that could affect the design of the network
Current availability of the network	MTBF mean network times not available
Current network performance	Performance measured between multiple servers and hosts latency versus bandwidth
Current reliability of the network	Number of CRC errors, collisions, etc.
Current use of the network	Use of segments, links, and protocols
Status of existing routers	CPU usage, memory, and packet loss Determine processing limitations under current load
Existing management system	Management tools and on which platform they run
General network health	Ethernet segments do not exceed 40 % utilisation WAN links do not exceed 70 % utilisation Broadcast / multicast traffic less than 20 % of total traffic There is more than one error CRC (Ethernet) per million bytes CPU utilisation of routers does not exceed 75 %

Table 1.4: Inventory of the current network

- Performance requirements and quality of service.

As a task of continuous realisation, it must be borne in mind that the expectations of the end user must be adequately managed, so as to ensure complete satisfaction when the network is launched, and during the continued use of the network. Therefore, the best way to achieve this satisfaction is to jointly establish these requirements with the user, **document the requirements**, and design the network so that it reaches and exceeds these initial requirements of customer satisfaction. If the network is new, it is necessary to establish gradual levels of introduction of network services:

Technology test: Test environment where both the designer and the end customer are learning from each other. It is expected that the network reconfiguration time, or the time with dropped network is high. Network services are limited in scope and functionality. The target group in this state should be limited to experienced users or users not affected by the network crash.

Alpha test: Pre-release version of the network where the customer is still learning from the network provider, although the latter already has the design completed and the hardware is fully operational. The time with a dropped network is minimal, and there are very few users in the network. This small-scale version of the real network, as with a test run, is expected to have errors.

Beta test: Final test phase before offering a commercial service or public availability. Most of the errors have already been eliminated, and users experience few or very few network crashes. Most new network technologies will be tested in parallel with legacy networks at least during the beta test phase.

Production: The network starts up officially, with full functionality and without errors. There are no unscheduled drops in the network.

The final version of the network in production must be transparent to the average user. Each user must perceive that they have total connectivity when required, and with the required bandwidth. Above all, we must remember that the end user is the customer and the designer or service provider is the provider. Therefore, **the provider must show a professional and friendly attitude towards the user**. For the provider, user perception *is* reality.

1.1.4 Protocol requirements

You must first define the complete set of user protocols that will be used by both existing applications and applications that are planned for the future. This includes all the protocols of the ISO / OSI reference model. Often there are multiple protocols per layer, especially in the upper layers.

We must next define the protocols that will be used from the side of the network, and how users will interconnect these protocols. The bridging, routing, or switching that exists in the network will depend mostly on the protocols. For example, will the user need encapsulation, conversion and / or translation of protocols, or just bridging and switching?

Given the boom in the use of the TCP / IP protocol, the management plan can be critical. Therefore, the entire addressing system must be clearly defined:

- Define what is the addressing scheme currently used.
- Is a flat or hierarchical scheme followed?

- Are they permanent addresses? If not, can changes be made easily? Can they fit into a global addressing system?
- Are the addresses public or private?
- Is there a corporate addressing plan, or does each facility choose its addresses?

Apart from addressing, the protocols can also influence network delays and latencies. Therefore, one of the most important aspects of data communication are time and delay considerations, and how applications and protocols manage delays. Remember that the response time of the network is the *round-trip time (rtt)* of the source to the destination. This global delay can be decomposed into the following delays:

Access delay to the medium: The time needed to gain access to the medium in a CSMA / CD network. Although this delay will normally be short, during congestion it can be significant.

Serialisation delay: This is the inverse of the transmission speed, and it is also defined as the time it takes to put a packet in a channel or serial link. It is significant in low speed transmission links.

Network delay: The time it takes for a packet / frame / cell to cross the network of the service provider. This delay includes the components of propagation delay through links, switching delays and routing, and delays by hardware / software processing.

The last component may not be directly controllable by the designer of the network, because in some cases (such as ADSL) the length of the subscriber loop depends on the distance of the client to the switch of the service provider. Basically, all components of the network delay minus the propagation delay will vary depending on the offer or contract with the service provider. Since the delay can be important with respect to the productivity of the application, how the application is written and the protocols that the application uses will significantly affect the productivity of the application.

1.1.5 Connectivity requirements

The user's connectivity requirements must be defined. First, we will differentiate between user-network connectivity and network-network connectivity. The user-network connectivity is framed by these items:

- What is the minimum speed required for connectivity?
- Which devices are going to connect and at what speeds do they transmit?
- What is the distance between the user's equipment and the communications equipment?
- What is the hardware / specific software of a vendor?
- What type of addressing is needed for the access network and backbone network?
- What protocols are needed for basic connectivity?

Once the user-network connectivity has been defined (the specified connectivity between the user and service provider), the interconnection requirements must be defined between any network-network interface that exists to interconnect the new and existing networks to the network. It is necessary to define the extent to which ubiquitous connectivity is required.

Geographic connectivity requirements play a very important role in traffic patterns. We must determine:

- Where do the applications reside?
- Is the planned network local, regional, national, or global?
- Are the remote locations major cities or locations with low bandwidth?
- Are there geographical restrictions based on the topology or availability of transport networks?

The local geographic layout must also be taken into account in the design. Is it a hierarchical network by nature? In theory, nowadays every network should be. This will make the reliability and maintenance requirements of the network more easily and conveniently met. In this sense, *maintainability* includes a definition of how the designer will expand the network when the requirements increase. It is also necessary to detail the flexibility and ease of hardware expansion, as well as the ease of updating the software and the tools installed for the management, monitoring, and detection of network problems.

1.2 Network service scheduling

The scheduling of network services refers to the choice of the technology, as well as the provider of both the access and the trunk networks to be installed.

1.2.1 Taxonomy of data communication methods

Let's review the three switching methods that are used in network technologies: circuit switching; message switching; and packet switching. All categories of data communication can be classified according to this criterion. Figure 1.1 shows some examples of technologies that have emerged at different times and how all have used some of these types of switching.

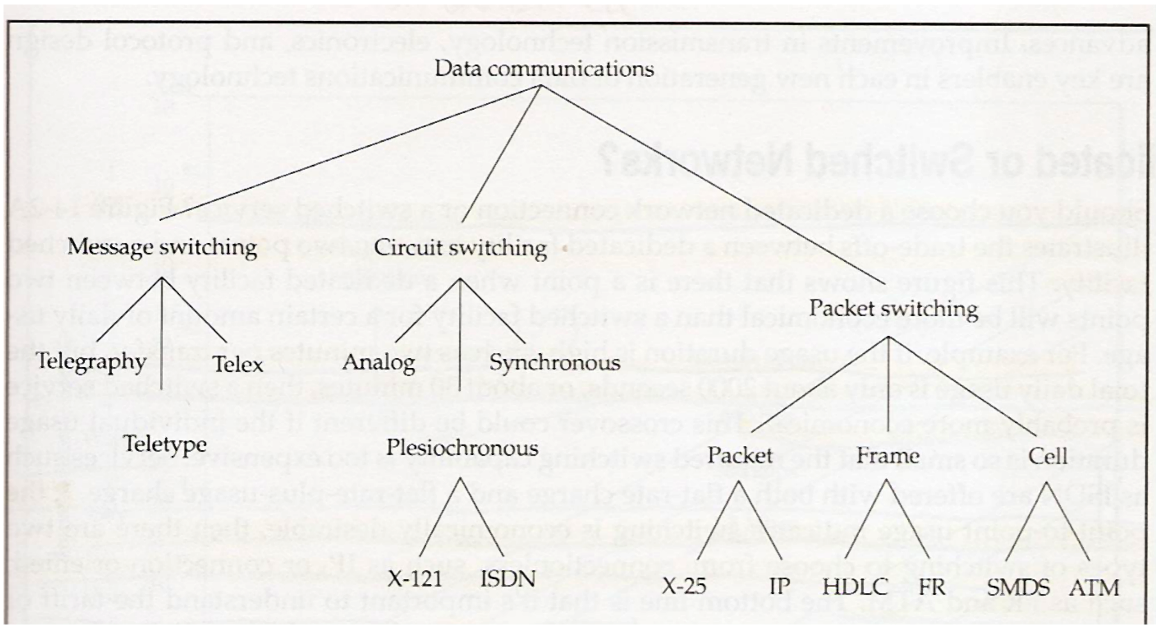


Figure 1.1: Taxonomy of historical data communication technologies

One of the key decisions in the design of the network is the dilemma between a dedicated WAN line or a switched service. To be able to choose properly, three parameters must be taken into account: the daily use of the line (measured in seconds, minutes, etc.); the duration of each line use (i.e., how long each transfer lasts); and the line setup time in seconds. Figure 1.2 illustrates the trade-off compromises among these parameters.

Figure1.7 A) shows comparative daily use versus the duration of use. So, for example, if the duration of use is high, such as ten minutes per transfer, but the total daily use is only 2000 seconds, then a switched service will probably be cheaper. Figure 1.7 B shows the relation between the circuit setup time and the duration of use. As a general rule, the average duration of the transaction should be more than an order of magnitude greater than the circuit setup time in order for connection-oriented network technology to be the better choice (Frame-Relay, ATM). Otherwise, it is better to use a non-connection-oriented technology.

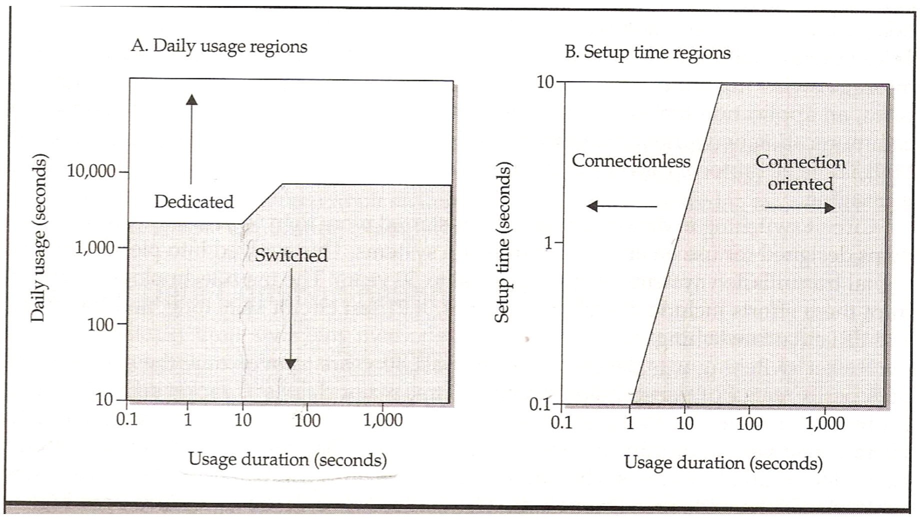


Figure 1.2: Switching application ranges / dedicated lines

An important aspect of switching is whether the arrival of the packets at their destination is guaranteed or not. X.25 and TCP standards ensure that packets arrive reliably link by link. However, technologies or standards such as Frame-Relay and SMDS do not ensure the correct arrival of packets. ATM can work in both modes of service, depending on the chosen traffic class.

1.2.2 Public network selection

Another important decision for the network designer is whether a shared public network is better for the set of protocols and applications that are going to run on the network, or whether a private network, or a combination of both, is better. Of course, a key element is the total cost, including the planning, design, implementation, support, service, maintenance and management of the network, and the subsequent updates and improvements of the technology. All this must be taken into account when evaluating a public network service. If in the end, the option chosen is a public network, then the next step is to choose the correct service or correct combination of services.

A private line or dedicated circuit can be convenient when:

- Point-to-point connectivity between two locations is required.
- The average traffic volume between two points is very high (more than 30 % of the average busy during a 'busy hour' of a 'busy day').
- Build a star topology with a single location for the central hub.
- The characteristic traffic is CBR (constant bit rate) type, such as video conferencing.
- A consistent quality of service that can only be offered by a dedicated circuit is required.

Frame-Relay can be convenient when:

- The consolidation of multiple protocols on a single access circuit is needed.
- A large amount of traffic must be transported from one LAN to another.
- Traffic is variable bit rate (VBR).
- When more than three locations require interconnectivity between them.
- When online management capacity is required, as in SNMP.
- When flexibility or fast reconfiguration of the network is needed.
- When international alternatives are required.

ATM can be effective when

- High-speed interconnectivity is required (DS1 to DS3).
- When native LAN interconnection speeds of 100 Mbps or greater are required.

- When multiple levels of service quality are needed in a single circuit or route.
- When interconnection of multiple network services (FR to ATM, etc.) is required.
- Access or integrated transport of multiple services.

1.2.3 Selection of service provider

The correct choice of a service provider should be made through requests for information (RFI) and then through request for proposals (RFP) after providing a list of requirements through which to build relationships that must last all the network life, or to the satisfaction of both parties.

The service provider responds (or should respond) to the RFI / RFP with a list of what it can provide, when it can be provided, and at what cost or with what expectations. The capabilities of each provider must be analysed carefully. A service contract must be signed, forming a partnership that should last the entire life of the network. The key is never losing communication with the service provider, since the success of the network can affect the users who use it, and therefore the company itself. Therefore, the choice of the service provider is extremely important.

Request for proposals / projects

RFIs are designed to obtain information about potential suppliers to enable the user to eliminate those suppliers that do not meet minimum requirements. The RFI simply offers potential suppliers a version of the desired final result. This is a stage in discovering who can offer something.

In an RFI the user typically provides:

- Business objectives and network performance.
- Existing applications and technologies to be integrated or replaced.
- Volume and characteristics of the traffic.
- Requirements and future plans.

- Evaluation criteria.

and it also demands from potential suppliers:

- A specific response to each item of the RFI.
- Specific capabilities of each supplier company to meet each requirement on time.
- Dates of delivery of products or services.
- Complete documentary support.
- General rates.

After receiving the RFI responses, the user must select a limited number (3-5) of providers and send the RFP to them. The structure of an RFP includes:

- Synopsis with business objectives.
- General contractual conditions.
- Scope of work or service.
- Temporary planning with milestones and deliverables.
- References of the bidder.
- Systems, technologies, architectures, protocols, services and current applications.
- Software or hardware technical specifications or proposed configurations with accepted substitutions.
- Environmental requirements.
- Engineering and facilities required by the bidder.
- Implementation plan
- Service level agreements.
- Guarantees or availability requirements.

- Payment options.
- Civil liability (insurance) of the bidder.
- Assumptions assumed in the proposal.
- Training requirements.
- References of the bidder.
- Legal or contractual considerations.

Selection of proposals / projects

Each RFP bidder must respond paragraph by paragraph. You have to be attentive and avoid proposals that contain answers such as ‘substantial compliance’ or ‘will be contained in future versions’. We must select those proposals that contain terms such as ‘full compliance’, and if possible with an explanation of how or to what extent they will comply. Vendors must also clearly specify what they will support. The responses to the RFP must contain:

- Price of the product.
- Maintenance requirements.
- Learning and training curve.
- Regular expenses.
- Tax filing (VAT).
- Guarantees.
- Documentation.
- Any other additional cost.

However, it must be noted that although a RFP helps greatly in selecting suppliers, there are many other reasons to select one vendor or supplier over another. Factors such as politics, business financing, operational needs, future position or potential associations, and trade balance, may go beyond purely technical reasons. This happens especially in large corporations.

The relationship established with the vendor or supplier is practical for business needs. It is *fundamental* to study the supplier's past not only for the product or service that we are going to contract, but also their financial history, payment history, accuracy in billing, delivery, and receipt of the products/services, past and current clients, and industrial experience. References should be sought (beyond the provider's website), inquiring both in internet forums and directly asking the provider's customers. Some evaluation criteria for service providers and hardware include:

- Seller history.
- Versatility of the product / service and ability to provide a total solution.
- Processor and data processing speeds.
- Support for interfaces and protocols.
- Capacity for local or remote support.
- Characteristic features.
- Network management.
- Security
- Price / performance ratio.

Finally, we must emphasise that price is not often the most important evaluation factor, because the work and resources committed to making the network operational often exceed hardware costs.

It is quite effective to build a weighted matrix of requirements assessment, where each requirement appears with a weight that weighs its importance. Table **??** shows an example:

1.3 Design phase

As an introduction to the design phase, the first step is to review the four primary levels of network design, according to the standard terminology accepted by the industry:

Requirement	**Weight**	**Scoring**	**Total**
Total cost of the solution	10	7	70
Prices and billing options	5	8	40
Interfaces and protocols	10	10	100
Reputation and philosophy of seller and product	5	4	20
Architecture and migration flexibility	5	6	30
Performance and capacity of the system	15	8	120
Reliability and recovery from catastrophes	15	4	60
Network management	25	8	200
Customer hardware support	5	8	40
Support and customer service	5	4	20
TOTAL	100	-	700

Table 1.5: Example of a weighted matrix of requirements assessment

Application level: These are the programs and protocols that are executed in the workstations of the customer's network, as well as the terminal equipment that communicates through the network. The level of applications include operating systems.

Client equipment architecture: This is the software, hardware, and local area networks existing in the customer's premises, and is usually denoted as *customer premises equipment (CPE)* or equipment on the customer premises.

Access network: All elements and environment that provide the interface between the CPE or business network and the WAN environment of the service provider. According to the definition of *access* from Cisco Systems, access includes all the devices that interface between multiple LANs or workstations, thus blurring the distinction between customer equipment and access network.

Backbone network: This is the environment and the elements that provide transport, switching, and routing between the access elements and trunk devices. The backbone network can be a cloud network of the service provider, such as FR or ATM, or simply a dedicated line between two different locations.

The historical trend has been that Ethernet and IP have clearly become the dominant LAN protocols, while the falling price of Ethernet, Fast Ethernet, and gigabit Ethernet switches has blurred the distinction between local and metro access.

1.3.1 Access network design

The design of the access network involves selecting or modifying the LAN, MAN, RTC, or dedicated access lines. Before starting with traffic analysis, network modeling and network infrastructure design, you first need to analyse the information collected in five areas:

Physical connectivity: If a current network already exists, what are the physical and logical configurations, and why? If there is no such network, what is the design that makes the most sense given the current and future requirements?

Protocols: Are there several protocols in layers 2 and 3, or are they all Ethernet and IP? In the first case, we must consider building the access network to accommodate the inherited protocols.

Switches versus routers: You have to know when it makes sense to use routers instead of switches.

Quality of service (QoS): You should analyse which applications need quality of service and what level of service quality they need. It must be analysed if the network needs to support voice, data, video and internet access separately or in integrated manner. In this last case, it is necessary to determine how the voice and video requirements will be guaranteed in terms of delay, jitter, and packet loss.

Fixed / mobile network: It is necessary to analyse how many users will be in the corporate LAN and how many need wifi network mobility.

In terms of physical connectivity, in principle there is a wide variety of connectivity requirements in terms of transmission media (copper, fibre optic, wifi, satellite, etc.). In terms of copper and fibre optic cabling, the structured cabling standards should be followed, as far as possible: EIA / TIA 768A or B. In addition, the network should be designed avoiding a 'flat' network topology. Figure 1.3 shows a schematic example of a network with a flat topology.

In general terms, we can say that a flat topology is one that enables all users to transmit and receive data from the network through a shared medium or device. The main problem with a network with a flat topology is that the problems of performance degradation are not easily diagnosed, and the network can become very difficult to manage as it grows in size. Such a network also offers a point of total failure, which can lead to the

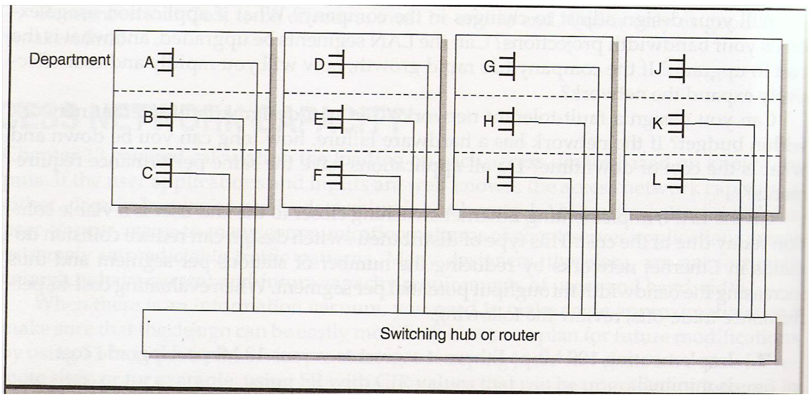


Figure 1.3: Example of a network with flat topology

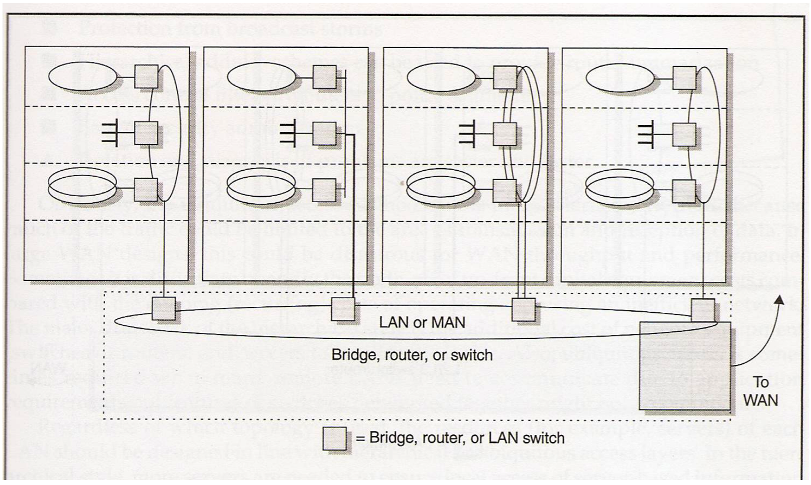


Figure 1.4: Example of a network with hierarchical topology

total collapse of the network at any given time. Therefore, it is better to design networks with access or hierarchical topology. Figure 1.4 shows an example of this type of topology.

Hierarchical networks offer an access hierarchy where the traffic destined for the local, metropolitan, and wide-area zones remains in its geographic area, instead of accessing all the switching and trunk routing devices. The hierarchical approach offers many advantages over flat networks:

- Enhanced performance due to the limited size of the segments.
- Protection against broadcast traffic.
- Hierarchical addressing schemes can be used.
- Enable use of access control filters in the routers for different segments.
- Ease of security administration.
- Easy and faster isolation and fault diagnosis.

As an example, let's see what happens when a legacy flat network grows to congest LAN segments. Figure 1.5 shows an example of such a network.

It is noteworthy that the collision domains of the network and the broadcast domains coincide and cover the entirety of the network. This makes the network easily saturated. The solution that appears in Figure 1.6 could solve this problem.

The proposed solution generates two different broadcast domains, one for the servers and the other for the rest of the stations. This limits the congestion produced by broadcast traffic. The scope of the collision domains is also significantly reduced, since the LAN switch has four domains instead of a single domain for the workstations (of course, the servers form another separate domain). This configuration would significantly reduce the congestion of the network for the same traffic load.

Regarding the issue of routers versus switches, it is obvious that switches establish different collision domains, while routers establish different broadcasting domains.

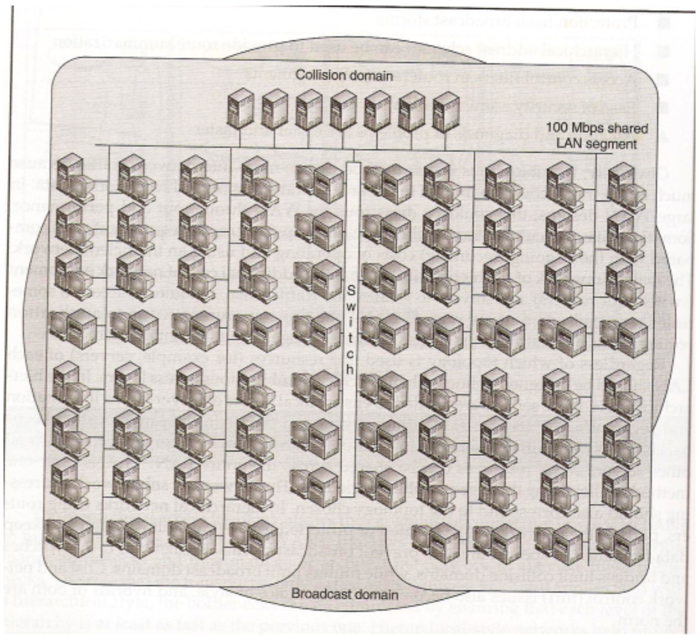


Figure 1.5: Example of a congested legacy network.

1.3.2 Backbone network design

Trunk networks usually consist of virtual public network services such as IP, Frame-Relay, ATM, virtual private networks (VPNs), private fibre optic networks, and dedicated lines. Public WANs are constituted by private lines, FR, ATM, and IP services of telephone companies.

Some efficient compromises when selecting the technology or service for the trunk network are the following:

- Traffic consolidation and convergence: will voice / data / video be shared over the same network / access / transport?
- Reliability: re-routing capacity and redundancy.
- Scaling economics.
- Equipment sharing between several locations.
- Dynamic allocation of bandwidth.

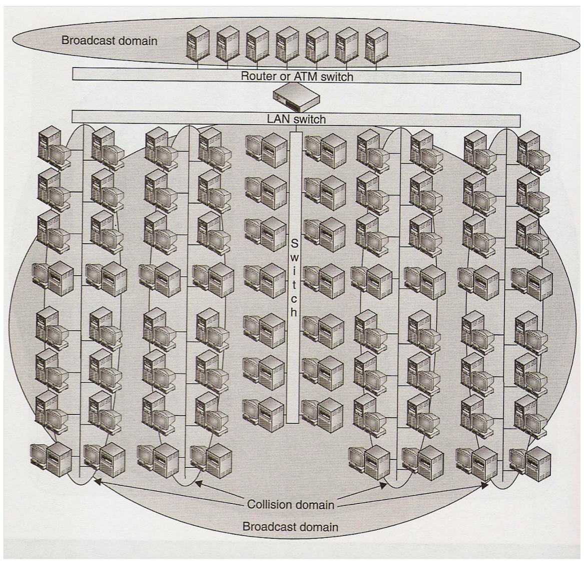


Figure 1.6: Example of a solution with a router and a switch

- Distributed or centralised management of the network.
- Flexibility and bandwidth of the CPE (client equipment).

As an example, Figure 1.7 shows a WAN network with private lines that form a topology of total interconnection between six nodes or access concentrators.

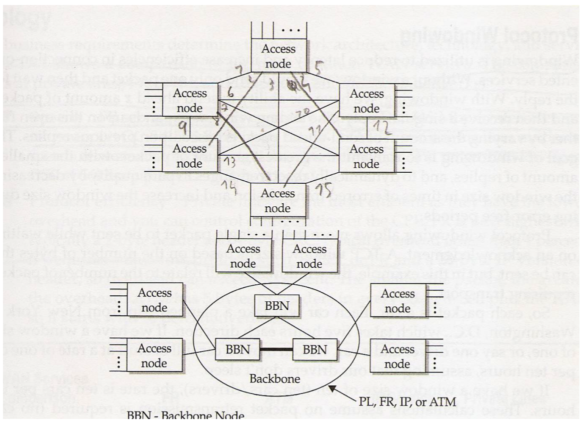


Figure 1.7: Fully connected backbone network

As the number of dedicated point-to-point circuits (or trunks) among the access nodes increases, it becomes necessary to design a trunk network hierarchy that can save circuit costs and improve the scalability of the network. In the example in Figure 1.7, we have opted for a backbone network (BBN) of three nodes, grouping access nodes by geographic regions. The backbone cloud could be a public service such as ATM, FR or ATM.

As Figure 1.7 shows, the number of circuits needed to support a total interconnection topology of N access nodes is $N(N-1)/2$, which for six access nodes means 15 circuits. As each private line is actually two circuits (to obtain full duplex transmission), the total number of circuits is 30. If, for example, the user uses FR, the total number of circuits is reduced to six with the BBn solution. The total number of ports on each router of the access node is $N-1$ (if there are N access nodes, each node has to connect with $N-1$ nodes), whereas with the BBN solution, each access node needs only one port of access to the trunk node.

Regarding the required capacity for the backbone network, it is necessary to look at the required bandwidth. Trunk capacity refers to the total amount of bandwidth required to carry incoming traffic to a remote destination node. The formula

$$(T)(S)(\%UT) = \sum(p_i)(s)(\%Up_i) \tag{1.1}$$

where

- T = number of required trunks.
- p_i = number of type i ports.
- S = trunk speed.
- s = port speed.
- $\%UT$ = percentage of utilisation of the trunks.
- $\%Up_i$ = percentage of utilisation of type i ports.

can be used to calculate the number of necessary trunks – or the use of trunks according to the number of access ports, their speeds, and uses.

Example: Let us suppose a backbone network that must link 27 remote sites. The speeds and uses of these remote sites are as follows: 5 T1 at 37 % utilisation, 10 FT1 (256Kbps) at 57 % utilisation, and 12 DS0 (64 kbps) at 75 % utilisation. The objective is to obtain trunk ports for an average use of 50 %. How many T1 ports are needed?

(T) (1536 kbps) (0.5) = (5) (1536 Kbps) (0.37) + (10) (256 kbps) (0.57) + (12) (64 kbps) (0.75))

T = 6.35, that is, 7 trunks.

It is important to note that the network should not be designed for the bandwidth required during normal network operation, but for the peak bandwidth during the busy period. You also have to take into account the percentage of local traffic, that is, entering and exiting through the same trunk router. Figure 1.8 offers an example.

In this figure, a network of 12 nodes is shown with simple trunk paths to the access nodes. Some 50 % of traffic remains local (exits and enters through the same trunk access node).

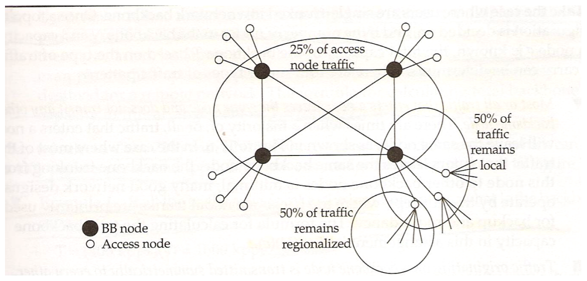

25% of access node traffic
50% of traffic remains local
50% of traffic remains regionalized
BB node
Access node

Figure 1.8: Example of local traffic in backbone network with simple links

Given the trunk traffic patterns, the backbone capacity can be calculated in two ways:

- Determine the capacity per node based on the chosen number of nodes.
- Determine the number of nodes based on capacity limitations per node.

Assuming that users only have one link to the trunk network, then once the topology is decided, and the number of nodes in the backbone N, and the capacity of each node is known, the total capacity T of the trunk network can be determined, based on the type of traffic it will transport. There are four main types of traffic patterns:

- Most or all of the traffic that enters a node exits the node and does not circulate through any other trunk node. In this case the trunk nodes are used mainly as redundancy. In this case, the formula for calculating the capacity of the trunk network is $T = (N)(c)$.
- Traffic originating from a trunk node is transmitted *symmetrically* to each of the other trunk nodes. this is the case of a broadcast backbone network. In this case, the formula for calculating the trunk capacity is $T = (N + 1)(c)/2$.
- All traffic patterns are asymmetric and are divided into user classes. In this case, the use of links varies. The formula for calculating the trunk capacity is $T = (N^2)(c)/(2N - 1)$.
- Users never transmit to nodes on the same trunk, and all traffic is directed to other trunk nodes. The formula for calculating the trunk capacity in this case is $T = (N)(c)/2$.

In summary, as the traffic pattern becomes more distributed, the capacity of the network to support it decreases. This is because there are more options for traffic to use resources with limited bandwidth.

Regarding the most suitable topologies for the trunk network, trunk topologies seem to be divided into two tendencies: those networks that have been planned, and those that simply grow according to needs. Private networks, especially the LANs that grow to become first MANs and then WANs, tend to be very asymmetric, with very defined communities of interest. In any case, the most common topologies in the backbone network are the following:

Star: Topography where all communications go through a central node. Therefore, $N-1$ links are needed to connect N nodes. They are often used in ATM or switched LAN networks where there is a central switch. Usually the central node is a multiport device (scalable as much as possible). Obviously, this topology is susceptible to the total failure of the network when the central node fails.

Ring: Circular topology where N links are needed to join N nodes. This topology is used mainly for distributed networks where the majority of network traffic goes to adjacent neighbours or to relatively short distances (in hops).

Meshed and semi-meshed network: A mesh corresponds to a network with total interconnection between its nodes, that is, each node has a direct link with each of the other nodes. When there is an exception to this total interconnection, then this topology is known as a semi-meshed network. Figure 1.9 shows an example of each of these types of topology. The degree to which a mesh network is built depends on the hardware and software cost of the ports. The number of channels or links required for a mesh is $N(N-1)/2$. Therefore, the number of channels increases drastically with the number of nodes.

Daisy chained (chained): All network access devices are connected to two high-capacity trunk switches. This provides high availability, although it is wasting bandwidth if the applications are regional or their process is distributed. Figure 1.10 shows at the top (A) an example of a typical daisy-chain network, while the lower part shows an alternative where each access device also acts as a switch through the chain of devices.

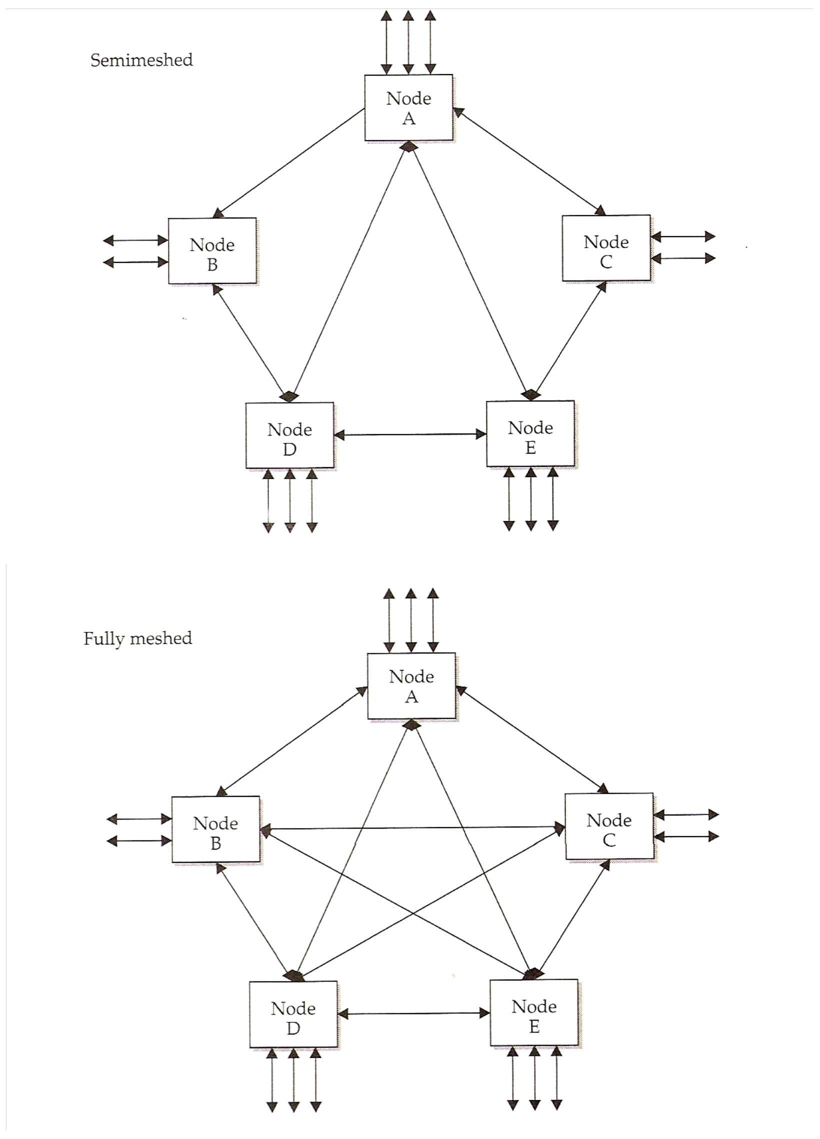


Figure 1.9: Example of semi-mesh and mesh topologies

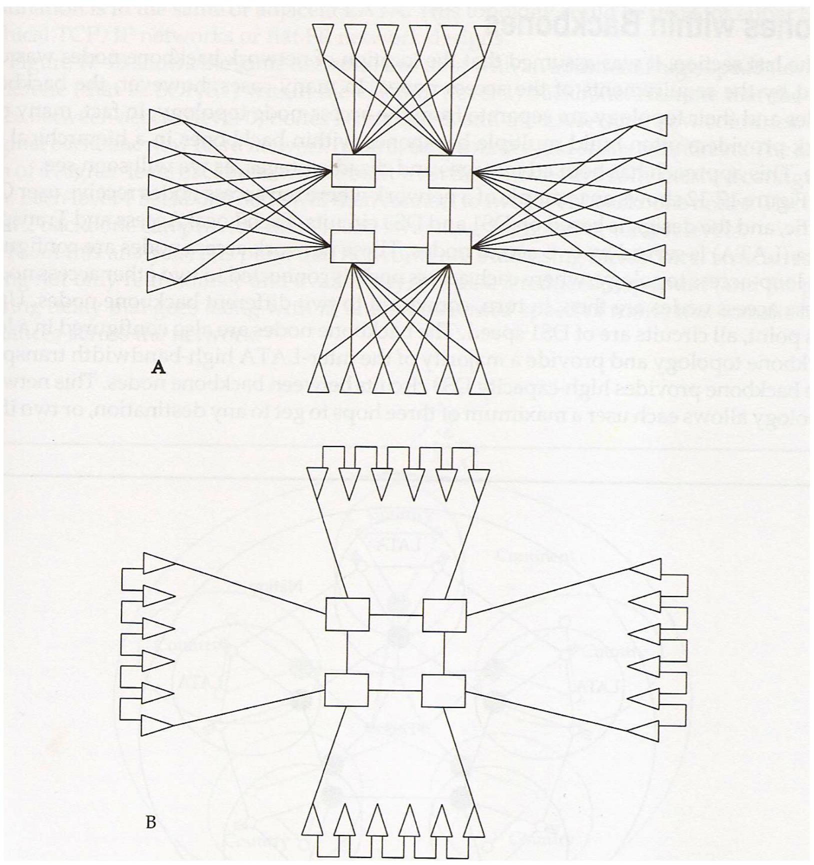


Figure 1.10: Example of daisy-chain topology

1.3.3 Security

The history of network security begins with the building of the first computer. In the 1960s, MIT borrowed the term 'hacker' from a term describing the members of a model train group who modified electric trains, tracks, and switches to make trains run faster and in a different way.

The 1970s was the decade of the 'phreaks', who were no more than telephone 'hackers'. One of these hackers, known as 'Captain Crunch', realised (no one knows how) that a toy whistle stuck inside a cereal box of the brand 'Captain Crunch' generated a signal of 2600 Hz, which was the same tone that allowed access to the long distance network of the company AT&T. The 'phreakers' created a 'blue box' that, together with the whistle, allowed free calls to be made. In fact, two students, named Steve Wozniak and Steve Jobs, launched a home business and sold these 'blue boxes' before founding Apple Computer.

'Bulletin board systems' (BBSs), precursors of news groups and electronic mail, appeared in 1980 and enabled phreakers and hackers to freely exchange experiences and information. In 1988, Robert Morris, a recent graduate of the University of Cornell, launched a self-replicating program or virus on the ARPANET network of the US Government, affecting 6000 computers. It was just a warning of what was to come.

Despite the fact that in the 1990s the US Secret Service conducted several raids and arrested many hackers, they were unable to reduce the number of computer attacks. The focus of security was moved to antivirus in 1999, but there were still numerous denial-of-service attacks against famous websites.

Computer attacks and network security have become a national security issue, and China and the US even maintain tense diplomatic relations over this type of attack. The reason is that with the generalisation of computer science and computer networks for all types of facilities (including strategic industries such as energy and water distribution, transportation, etc.), network security has become vital for the survival of any company or institution.

The security threats can be classified into three major groups:

Unauthorised access or intrusions: consists in the access of an unauthorised person or system into a system or network. An example may be the use of an unprotected wifi network that is within the range of a router.

Privacy violation or impersonation: data is inspected in transit or by any other means, in such a way that this data enables someone to present authorised user credentials when it is really someone else or a machine. A widespread example are sniffers that read credit card data and then make fraudulent purchases.

Denial of service (DOS): Consists of an attack that floods the access network with multiple requests in a short period of time, saturating the access system and thereby preventing access for normal users.

Unauthorised access occurs when someone gains access to a service or system such as a computer, and the unauthorised individual makes unauthorised changes to the system. Access can be achieved by a privacy violation or by exploiting some weakness of the system (software, operating system, or hardware).

The violation of privacy can occur through the tracing of network packets (sniffers) or simply because a user has all their passwords on a label stuck on the wall or on the computer screen. A sniffer can capture all the traffic that circulates in the collision domain. Although in recent times the trend to switched networks has decreased the size of these domains and therefore the sniffers have a limited scope, they can still be very effective. To protect against privacy violations, pay attention to both the physical security of the network and the information encryption systems and firewalls.

In the face of these attacks, security precautions must be taken in the design and implementation of the network. Obviously, this will not rid the network of possible threats, since it will be the daily behaviour of users that will determine the actual protection. It is important to highlight this fact to the end user: without their collaboration there is no possible safeguard against threats. But with regard to network design, all possible precautions should be taken. The first one is that the network equipment must be located in a closet or locked room with access limited to authorised personnel. Employees (users of the network) are the individuals that cause the most security incidents, which are almost never intentional. Therefore, the possibility of its occurrence must be prevented.

The following is a list of questions that help determine the vulnerability of the network to be installed:

- Who has keys to which doors?
- Is the company that owns the premises trusted?

- What is on top of the (false) ceiling? Could someone enter there?
- Does the route of the network or power grid cables run through exposed areas or vulnerable areas of the building?
- Would a motion detector be useful?
- Are visits controlled?
- Who has physical and logical access to computers?
- Do employees bring their family members on weekends or after work?
- Who has access to administrator passwords, and how often are they changed?
- Are locks and passwords changed every time an employee leaves the company?
- Are the cleaning staff trusted?
- Do you immediately inform salespeople and employees that someone has left the company?
- Is there a defined policy of destroying documents before throwing them away?

You also have to pay attention to the password management policy. Passwords are needed to access almost any network resource, but weak passwords can be easily guessed, and there is free software available on the internet to break weak passwords. Table 1.6 shows a list of what should be done and should not be done when selecting passwords.

NOT due	**Must**
Use the username under any form	Use passwords with may. and minus.
Use family names or surnames	Use numbers or punctuation
Use easily obtainable personal information	Use easy to remember password
(registration, telephone, ID, etc.)	So you do not have to write it)
Use password of only numbers	Use a password breaker periodically
	to detect fragile passwords and change them
Use words in dictionaries or lists	
Use passwords less than 6 characters	

Table 1.6: Passwords policies

We must implement an antivirus management policy for the entire network or for individual computers, and keep the antivirus updated. You also have to manage a firewall policy from the beginning, which is nothing more than a mechanism to protect the system (either the network or each computer) from the outside world. If there are mobile users, or simply to improve security from the outside, it is worthwhile implementing virtual private networks (VPN). A VPN is a private connection between two or more network elements over a shared structure (typically a public network).

1.4 Documentation and network management

Documentation is one of the most important steps in the network design process. Documentation enables people to understand a design created at some point in the past, and enables them to understand the assumptions of departure made during the design, as well as keep a record of updates and modifications to the original design. The documentation has to be a continuous process, never a late event, or a task to be done a posteriori. Both the designer and the user will suffer the consequences of poor documentation, and therefore the documentation should be the result of a consensus between both.

The four main documents that are needed to correctly document the design of a network are:

1. Contract for the design / installation of the network.
2. Document of user specifications and acceptance of them.
3. Project and engineering plan.
4. Operation and maintenance document.

Of these four documents, the first two must be completed before the network design begins. The contract document must indicate the mission of the business, the vision thereof, the goals, objectives, and specific requirements of the network to be designed. The justification for the design cost of the current and future network must then be defined. The document of user specifications and acceptance of the same will define the requirements of applications, protocols, and all elements of the network that will inject traffic to it. The network is the highway that will contain all the traffic that circulates through it, and this document should define the ‘vehicles’ that will use the highway. The project and engineering plan is developed during the design, and is distributed when the design has been finished. The operation and maintenance document will be distributed when the network has already been installed, but requires some specific data from the engineering plan.

The project and engineering plan explains the history of the current connectivity, the reasons for the design of the network, what the network will support, and why this support is needed, as well as the complete design of the network, from the applications to the connectivity trunk. The modeling of the network should also be included in this plan, or it can be a separate document. The specific information in the engineering plan should include:

- Requirements matrix.
- Restrictions and design hypotheses.
- Configuration of nodes.
- Circuit diagrams.
- Physical and logical configurations.
- Location-specific information.
- Addressing plan for all elements of the network.
- Serial numbers and models.
- Current firmware and software versions.
- Technology details (FR, ATM, IP, MPLS, etc.)

For its part, the operation and maintenance manual must present all the information that is needed to install and maintain the network. Some of the sections in this manual will be drawn directly from the engineering plan, and others will represent standard procedures for installing and maintaining network equipment. Some of the topics that the operation and maintenance document of the network should include are:

- Description of the network.
- Physical and topological description.

- Logical and functional description.
- Design of the network.
- Impacts on the budget.
- Network administration.
- Network management procedures.
- Network installation and testing data.
- Billing procedures.
- Procedures for updating and scaling the network.
- Maintenance and repair of hardware and software.
- Training.
- Specific documentation from manufacturers.
- Glossary, acronyms, and references.

With regard to the location of the network, the documentation should include:

- Electrical installation and land.
- Implementation technology.
- Nomenclature and addressing conventions.
- Device-specific wiring.
- Device-specific tests.
- Construction plans.
- LAN or MAN / WAN plan or scheme.
- Topology.
- Location of the devices.
- Location of shared resources.

- Location of management and administration positions.
- Maintenance planning.

2 CHARGING MODELS

The economic aspect of the network and the financing models are key facets of the network design process, but nevertheless these facets only pay attention to telephone companies and service providers. In practice, there is an obvious disproportion between the time spent planning the topology, capacity, etc. and the cost. This is probably due to the difficulty in finding and analysing relevant rates and / or data of the real cost of the network. In terms of cost, the designers of the network usually look only at the initial cost. However, a complete financial analysis can result in an effective short-term solution that could be much less attractive in the long term for many reasons:

- In the majority of connections between networks, the costs of WAN lines represent a substantial part of the financial cost.
- High maintenance costs, training, and support can seriously undermine annual budgets.
- Poorly chosen equipment can lead to early obsolescence, affecting the longevity of the network.
- A design that works today but is not specified for the future can be very expensive to update.

As several of these hidden costs are recurrent, the penalty for a bad decision will be repeated year after year. To avoid this, we must take a global approach that includes an analysis of the total cost of the network throughout its life time, and the designer must consider all the financial components of the design.

2.1 Economic aspects of the network

2.1.1 Basic economic terminology

The network designer must be familiar with the basic economic terminology. A list of some basic terms is found below:

Amortisation: In economics, this term may refer to the gradual repayment of a loan or debt (repayment of a loan). It can also refer to the recovery of funds invested in a company (amortisation of an investment). But in this environment, we will understand it in its third meaning: the periodic devaluation of assets and possessions whose value decreases with time or with use.

Balance point: The level of cash flow in which revenues and expenses are equalised, and therefore there is no loss or profit for the company.

Budget line: The line that links all the possible combinations of expenses with a single income.

Capital assets (or simply assets or assets): All economic resources (including money available, debts contracted, buildings, factories, and machinery, as well as the entire set of goods) of a company.

Budget: Declaration of the planned investments (normally based on the availability of capital).

Gain / loss of capital: The difference between the original cost of a good and its final sale price. This gain or loss can only exist when the good (or asset) is sold.

Cash flow: The difference between the income generated and the expenses paid. This term is inherently dynamic, and represents a fixed picture of the financial status of the company that will vary over time.

Interest or cost of capital: The interest rate that the lender charges the debtor to secure a loan to finance a project or investment. The rate of return of an investment must be at least equal to this percentage. Otherwise instead of an investment it becomes a fixed loss.

Depreciation: Money counteracted or deducted from the company's income, an initial percentage of the cost of a good subtracted during the entire useful life of said good.

Economics of scale: The reduction in production costs per unit gained by increasing the size of the factory, plant, organisation, or industry.

Future value (or simply future): Projected value of money at some point in the future. If the purchasing power of the money invested has to remain the same as now, then the interest rate needs at least to be the same as inflation throughout the investment period.

Inflation: The rise in the general price level of a country.

Lease: A contract between the leasor and a tenant, where the tenant pays a regular (typically monthly) rate to the landlord for the loan and use of a property owned by the latter.

Net lease: A lease where the lessee pays all maintenance costs associated with the asset.

Amortisation period: Period of time during which the asset generates sufficient cash flow to cover the cost of acquisition.

Present value: The value of money in the present moment. This value will always be greater than the value of the same money in the future, due to inflation, the possibility of investing this money and obtaining interest, and the uncertainty of the future.

Prima facie: Common term in economics that means first impression.

Principal capital: The sum of money on which interest is paid when borrowed. The principal capital decreases according to the amortisation plan.

Residual value: The value of a good (the amount for which the company expects to sell that good) after it has reached the end of its useful life.

Simple interest: Interest rates that are a linear function of the principal capital.

Tender: Offer to provide some type of good or service or perform a job at a stipulated price.

Utility: Satisfaction derived from goods or services.

One of the fundamental concepts in economics is the difference between the present value and the future value of money, and this concept is critical for the long-term financial analysis of the network design project. Money today is worth more than the same amount of money in the future. To calculate the future value of a quantity of money can be calculated as:

$$F = P \times (1/(1+i)^n) \tag{2.1}$$

where

- P = present value of money.
- F = future value.
- i = interest rate (eg, inflation).
- n = number of years.

So, for example, if we have 1000 euros today and an inflation of 10 %, in 10 years we will have lost two thirds of their value:

$$F = 1000 \times (1/(1+0,10)^{10}) = 386\, euros$$

2.1.2 Decision models

In the design projects of the network there are some relatively direct financial decisions that must be taken according to the costs of a certain number of technologies. These decisions involve a series of calculations that can be made using a simple spreadsheet, to be reused when needed. There are a number of decision methods widely used to build comparative financial models for network engineering projects. These methods are the following:

Net present value method (NPV): consists of calculating the net cash flow created by the project, that is, project profits minus expenses (apart from depreciation), financial expenses, and taxes. A positive NPV implies that the project produces a rate of return that exceeds the cost of capital. When comparing projects, we should choose the one with the highest positive NPV.

Internal rate of return method (IRR): This method provides the return rate of a project. If the rate of return exceeds some threshold value (e.g. the cost of capital) then the project is selected.

Depreciation period method: This method is used to compare projects and determine which has the shortest amortisation period, that is, which requires less time to recover the initial investment.

Return rate method (ROI or *return on investment:*) Annual profit after taxes, divided by the initial expense.

The NPV method is generally considered the most appropriate approach. The IRR method is worse than the NPV, since it is only useful when comparing investment projects or similar outlays, since it compares percentages. The amortisation period method is also considered to be less useful than NPV, since its scope is limited to the amortisation period. Finally, the ROI method is also considered lower, because it does not analyse cash flows, nor does it take into account the cost of money.

To illustrate the NPV method, let's look at an example. Suppose we have the opportunity to either buy a router or rent it. We know that we can amortise this equipment in three years, at a cost of capital (interest) of 10 %, and that the expected annual profits for having the router will be 18,000 euros. There are two options:

- Purchase option: The router costs 20,000 euros, but the seller offers a discount of 7 %. After three years the value of the router is only 1,200 euros if we resell it.
- Leasing option: The router costs 850 euros per month, but after three years we own nothing.

Without a detailed analysis it is not obvious which solution is best, so we need to apply the NPV method, taking into account the value of the money over a period of three years. The main calculations are the following:

1. NPV is the sum of all the present values (purchase price - discount).
2. The present value is calculated as: $Net \times (1/(1 + interest)^{year})$.
3. Depreciation is calculated as (purchase price - residual value) / years

Table 2.1 shows the calculations to be made. The upper part shows the starting data to apply the model. The central part shows the calculations necessary to obtain the net cash flow for one year. Finally, the bottom part shows the NPV calculations of both

Item	**Rent**	**Buy**
Interest	10 %	10 %
Purchase price	0	20,000
Monthly rental	850	0
Discount	0	7 %
Taxes	40 %	40 %
Amortisation (years)	3	3
Residual value	0	1,200
Annual income	18,000	18,000
Depreciation	0	$\frac{20000-1200}{3} = 6.267$
Expenses	850 $\times$ 12 = 10,200	0
Income	18,000	18,000
Expenses (other than depreciation)	10,200	0
Taxes		
Income minus expenses	7,800	18,000
Depreciation	0	6,267
Taxable base	7800	11,733
Taxes (40 %)	3.120	4693
Annual cash flow	7,800-3,120 = 4,680	18,000-4,693 = 13,307
Cash Flow	Net Present Value (NPV)	Net Present Value (NPV)
Year 1	$4,680 \times \frac{1}{(1+0,1)^1}$ = 4,255	$13,307 \times \frac{1}{(1+0,1)^1} = 12,097$
Year 2	$4,680 \times \frac{1}{(1+0,1)^2} = 3,868$	$13,307 \times \frac{1}{(1+0,1)^2} = 10,997$
Year 3	$4,680 \times \frac{1}{(1+0,1)^3} = 3,516$	$14,507 \times \frac{1}{(1+0,1)^3} = 10,899$
NPV	11,638	33,993

Table 2.1: Example of NPV model for rental and purchase options

options. Note that the cash flow of the last year incorporates, in the case of purchase, the residual value obtained (1,200 euros).

Table 2.1 shows that the purchase option is clearly better, since the net present value that we obtain when buying is about three times higher than if we rent. Of course, with a simple spreadsheet we can perform simulations for different situations (model a lower rent, different types of interest, increase / decrease the purchase price, etc.). For example, how much should a router be worth to be worth renting, if the other conditions are maintained?

However, the biggest drawback of all decision models is that they are static, in the sense that they assume that all the necessary information is available when performing the calculation, and do not consider the uncertainty of the future. For example, interest rates, inflation and currency exchange rates fluctuate over time. Additionally, the other aspect that models do not contemplate is risk. For example, if we decide to buy the router and it turns out to be broken, what is the probability of failure and the financial impact of replacing it? These topics are the subject of risk analysis and probability theory, which are studied in more sophisticated models that manage variability.

2.2 General bandwidth charging model

To charge for network services, it is important to distinguish between each of the existing types of fees:

- Fixed fee or fee for connection or access.
- Usage fee.
- Fee for congestion.
- Fee for quality of service.

The fixed fee is a monthly fee charged for access to the network, where this access can be unlimited or limited to certain hours a day. This fee is independent of the number of accesses made or the amount of data sent or received. Ideally this quota should equal the cost of the network and the benefit of the client, strictly covering the costs of providing access to the network. Thus, for example, the fixed fee of the telephone should cover the cost of the subscriber loop to the telephone exchange.

The usage quota is based on each connection made, the amount of data sent, or the duration of the call, or circuit established. It can also be a function of the destination or distance covered by the established circuit. The duration of the circuit is usually rounded, and the next integer is charged. Therefore, the smaller the time of collection unit is, the better for the client. The usage fee must also match cost with benefit. Ideally, the usage fee should equal the cost of the additional resources that a call or connection needs with respect to fixed access costs.

The congestion fee depends on the network load at the time of user connection. In Europe, this type of quota is not usually applied, and the other quotas are increased instead. Thus, the more loaded the network, the greater the charge, and there will be no additional charges if the network is not congested. The reason for this is that the quality of service degrades rapidly as congestion increases. The congestion fee is supposed to allow users to decide whether or not to make a call or connection during periods of maximum occupancy or to postpone it.

The service quality quota reflects the fact that different users may choose to use more network resources as part of a service level agreement made with the service provider. This fee is charged on the types of networks that support different levels of quality of service, but it is charged as a higher contract price, not as a variable fee according to the generated traffic.

2.2.1 Charging models

We can distinguish three different charging models in the network services offered by the service providers:

Rented (or dedicated) line: The charging model of a dedicated line consists of charging an access fee to each of the two access points of the line, plus a fixed rental fee for the line, depending on its bandwidth.

ISDN line or service: In this case, an access fee is also charged for each of the two access points to the ISDN line and a fixed rental fee according to the bandwidth of the line, and a user fee is also charged.

ATM or frame-relay service or line: In this case we have two final access points (the access points of the two client sites to be interconnected) and each final access point is connected to a *point of presence or PoP*, which is nothing more than a switch, router, or gateway that is part of the backbone infrastructure of the service provider. The physical location of these PoPs is almost never public knowledge, due to competition or security reasons. The two PoPs are connected to each other through a circuit established between the infrastructure available from the service provider. The connections that go from the final access point to the PoP are charged each with an access fee and a line rental according to the contracted bandwidth, while the line or connection between the PoP is charged only the rental fee according to the contracted bandwidth.

These are the most commonly used generic schemes, but of course there may be variants in these charging structures. In general, to calculate a useful rate, it is necessary to know the following network information:

- Exact location of access points in longitude and latitude coordinates.
- Distance between the sites to be interconnected and between the PoPs (that is, the nearest switch or router).
- The telephone code and postal code of the sites to be interconnected. Some services such as voice and ISDN in some countries locate the PoP by postal code instead of distance.
- The bandwidth required by the interfaces and the required number of interfaces.
- The expected duration of the contracts. In general, the longer the contract duration, the better the discounts.

For an international network, the latitude and longitude coordinate system is usually used. To calculate the distances in kilometres from the latitude and longitude of two access points X and Y you can use the C code that is shown in Figure 2.1.

```
%

double GetDistanceByLatLong double Xlat, double Xlong, double YLat, double Ylong)

   static double circunf = 40077.0;   //Earth circunference
   static double pi = 3.1415926535
   static double radianCF = 3.1415926535/180 //Degree -> radian conversion
   double distanceKm      // Result is stores here
   double w0, w1, w2, w3, w4  // Variables aux.

   w0 = fabs(Xlong - Ylong);
    if ((360.0 - w0) < w0) then   // Ensures it is lower than  360 degrees
        w0 =  360.0 - w0;
   w0 = w0 * radianCF;           // Converted to radians
   w1 = (90.0 - Xlat) * radianCF  // distance
   w2 = (90.0 - Ylat) * radianCF  // distance
   w3 = cos(w2) * cos(w1) + sin(w2) * sin(w1) * cos(w0);
   w4 = pi * 0.5 - asin(w3);          // Spheric geometry
   distanceKm = circunf * w4/(2 * pi);

   return(distanceKm);
```

Figure 2.1: C Code for finding the distance between two points starting from their coordinates

Many services charge fees based on the distance between the access points, and optionally, depending on the distance to the operator's headquarters. It is rare to find rates based on distance that only use a linear function. For example, an access fee and one of the linear multipliers $m1$ a $m4$ per kilometre is used to calculate the monthly rate:

- Multiplier m1 for distances less than 15 km.
- Multiplier m2 for distances between 15 and 55 km.
- Multiplier m3 for distances between 55 and 100 km.
- Multiplier m4 for distances greater than 100 km.

However, postal codes or telephone prefixes may be important, because certain areas may have better infrastructures and therefore some services may depend more on which codes are located than on the distance. Greater distance does not always represent greater cost. Therefore, if you have to connect multiple short or medium distance circuits, the savings can be substantial.

Additional to these problems, there is the fact that in some countries tariffs are governed not by distances but by regions. Luxembourg, for example, is divided into 7 regions and each sub-region is subdivided into different areas. There is a structure of different prices depending on where each circuit ends at both ends. Sweden, Italy and Belgium have similar schemes.

Therefore, when establishing the best strategy to design the best WAN circuits adapted to the desired network, we must study the geographical issues of each country. There are also other anomalies with international circuits. For example, intuitively it would seem better to concentrate all the regional traffic of a country into a single central site and from there use a single international link. As it turns out, in some cases it is cheaper to use multiple international links from several sites. For example, between Germany and London it could be cheaper to use several circuits of 256 Kbps than to join them all in a single international circuit of greater bandwidth, with the additional benefit of reducing the number of hops.

2.2.2 Peak load rates

An observation of the dynamics of traffic shows that this has a clearly cyclical behaviour throughout the day and usually also during the week. In the United States, for example, it was observed that traffic peaks usually occur between 9:00 a.m. and 5:00 p.m. during a working day. There are also periods of low traffic levels during lunch and dinner, and interestingly, traffic increases again after 11.00, when there are cheaper telephone rates. A similar pattern exists in Europe, with small variations due to Mediterranean lifestyles.

The fact that there are cheaper rates during the most untimely hours is no coincidence, but the result of the operators applying *peak load pricing:* a traffic management technique that tries to move part of the demand outside the periods of maximum load. Without this technique, the operator would be forced to over-provision its capacity, which would be used only for a small portion of the day, and therefore would not be profitable. Alternatively, users would have delays and congestion during certain periods of time. Freight charge rates can also be applied to the weekly period. In this case, there may be more attractive rates for the weekend in some services. To implement peak load rates, operators define several bands during which rates differ: peak, off-peak, and holiday. Countries and companies may have even more slots.

In general, users of off-peak time slots should only be charged for operating costs plus the profit margin. Users of the peak load must also be charged for the depreciation of the capital and for the expansion of the capacity required for the peak load.

2.3 Internet charging models

Internet differs from private networks in that it is essentially a shared medium, providing access to companies and individuals competing for the same resource. The most widely used pricing model is the flat rate, where a fixed rate is charged regardless of usage. This mechanism is clearly in dissonance with the different classes of users that access the network, the patterns of use throughout the day, etc. In the flat rate, there are two key variables to set this rate: the profit per subscriber and the capacity consumed by the subscriber. In most infrastructures, operators assume that subscribers only use a small fraction of their connection, and the connection fee is calculated as the cost of providing that capacity plus the profit. Of course, this rate is subject to much fluctuation depending on the competition that exists. And what's worse, fluctuations in the pattern of network usage by the subscriber often break the static design of the network capacity, producing poor performance for all users, especially during load spikes.

As well as the flat rate there is also the possibility of multilevel rates, in vogue a few years ago in Spain, but now in disuse. This system marked the traffic according to the characteristics contracted by addressing, or by type of application. When there is congestion in the network, traffic is prioritised according to its level.

2.4 Collection models of private networks

Private network rates are the published rates for the different telecommunications services offered by providers. Large providers (such as Telefonica) publish different rates for the different services they offer, and there may be differences according to the position of the provider in the market. Tariffs can be very sensitive both to the location of access to the service and to distance. However, even with the published rates there are times when it is very difficult to calculate the actual rate, due to the complex structures of discounts, variations in the time slots, and terms of variation depending on the duration of the contract.

Dedicated lines (rented) and satellite connections are the most easily identifiable by their charging structure. The main feature of these services is that they provide a fixed point-to-point bandwidth that is always available. Typically, they have a single interface to connect a device to the end of the link. The fee usually consists of a connection fee, scaled according to the speed of the interface, and a rental fee also scaled with the speed of the link. For national circuits, the cost usually applies only to one end of the circuit, while for international circuits it is charged at both ends, half of the connection and rental rates being applied, each with a different provider.

Frame relay is a packet switching network with permanent or switched virtual circuits. In the price list of a frame-relay service provider there are several important methods that affect the level of service offered:

- Committed information rate (CIR): Measured in bits per second over a time interval T (the key parameter). This is the maximum level of traffic allowed (and guaranteed under normal operating conditions) that the provider will offer at a given input interface. It can reach the capacity of the physical line. Any traffic beyond the CIR is considered best effort traffic (without guarantees).

- Committed burst size (Bc): The number of bits that the frame relay network commits to accept and transmit in CIR during a time interval. This represents the traffic that will probably be delivered.

- Excess burst size (Be), sometimes called excess information rate (EIR): Absolute bit limit for an interface or DLCI (data link connection identifier, data link connexion identifier). This is the number of bits that the frame relay network will try to transmit, after accommodating Bc, during time interval T. The frame relay peak transmission rate R_MAX, is determined according to the formula:

$$R_{MAX} = (Bc + Be)/Bc \times CIR$$

Frame relay is one of the most complex services to charge. Typically there is a connection fee, scaled according to the speed of the interface; a rental rate for the line, also scaled by the speed of the line; and dynamic charges based on the use of private or switched virtual circuits, scaled by the speed attributes. The connection and rental charges go through each physical interface (port). Virtual circuits may also have rental charges dependent on bandwidth and possibly also usage charges. The connection charge may vary depending on the distance to the nearest gateway or point of presence (PoP).

In some countries there is also an SMDS service (switched multimegabit data service), which can be used as an alternative to dedicated lines, and typically offers speeds from 0.5 Mbps to 45 Mbps, and is usually cheaper than frame relay. In this case there is a single physical interface where different logical connections can be established to different destinations.

ATM is one of the services with complex tariffs. Typically, the user pays for a considerable physical bandwidth (usually available from DS1, DS3, and OC-3c) on several private or switched virtual circuits to multiple destinations. The services that can be offered can be constant bit rate (CBR), non-real-time variable bit rate (nrt-VBR), or unspecified bit rate (UBR). Charges are made for bandwidth (individual or aggregate) used in peak cell rate (PCR), and optionally in sustainable cell rate (SCR) and maximum burst size (MBS). The connection and rental charges apply to each physical interface (port). ATM circuits are charged according to the configured values and parameters of PCR, SCR, and MBS of each circuit, and there may be incremental costs for the total length of the circuit, the distance from switch to switch, or both.

Bibliography

[1] Kenyon, T.:*‘High Performance Data Network Design’*, Ed. Digital Press, 2002. ISBN: 1-55558-207-9

[2] Darren L. Spohn, *‘Data Network Design”*, Ed. Osborne, 3rd. Ed., 2003.