Real-Time Communication

Real-time applications are increasingly being implemented on distributed platforms. There are many reasons why distributed implementations are finding favour. One important reason is that it is often cost-effective to have a distributed solution using many pieces of cheap hardware, rather than having a centralized, sophisticated, and costly machine. Further, many real-time applications are inherently distributed with different sensors and actuators of a system placed at geographically separate locations. Another point going in favour of distributed implementations is fault-tolerance. Fault-tolerance is very desirable for safety-critical applications and it is easier to provide fault-tolerance in distributed implementations than centralized systems. All distributed real-time systems are based on an underlying communication network. These networks are expected to deliver messages in a timely fashion.

We can define real-time communication as one where the applications make specific quality of service demands to the communication network such as maximum delay, maximum loss rate, etc.; and the network once it accepts a connection guarantees the requested service quality. Traditional network protocols such as Ethernet are designed to deliver "best effort" performance. A best effort network strives to achieve good average performance, and makes no attempt to meet the individual deadlines of tasks. These networks are intended for use in applications where long delays and high data loss under heavy load conditions are acceptable. It is needless to say that such networks are insufficient for use in real-time applications.

In this chapter, we first discuss some basic concepts concerning real-time communication. Subsequently, we consider how real-time communication can be supported in multiple access networks. Finally, we discuss how real-time communication can be supported in packet-switched networks.

1 Examples of Real-Time Communication in Applications

In this section we discuss a few applications whose satisfactory functioning is heavily dependent on the underlying network's capability to support real-time communication. We should bear these applications in mind while trying to understand various issues in real-time communications discussed later in this chapter. Traditionally, real-time communications were used in hard and soft real-time applications running in distributed process control applications such as power plants, avionics, chemical process control, automobiles, etc. However, the current advancements in communication technologies has made it possible for supporting applications in many different areas such as: video broadcast and applications on the Internet such as banking, stock trading, e-commerce, etc. In the following, we review a few of these applications.

Manufacturing Automation: In an automated factory a set of robots carry out the manufacturing activities. Here, the network spans a small geographic area in a LAN environment. The robots communicate among themselves and with the controller using a real-time communication protocol implemented on a wireless medium (see Fig. 22). The controller coordinates the activities of the robot. The messages that the robots communicate with the controller range from non-critical and non-real-time logging information to highly critical and hard real-time control information. Naturally, the real-time communication system would be called upon to support these communication requests with widely varying service quality requirements.

Automated Chemical Factory: Consider a chemical factory which uses its existing LAN set up for controlling and

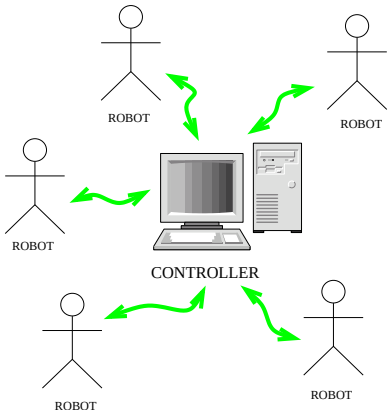

Figure 1: Robot Communication in an Automated Factory

monitoring the parameters of the chemical plant on a real-time basis. The sensors transmit the sampled parameter values at periodic intervals to the computer that acts as the controller. The controller computes the corrections necessary to the parameters and transmits these to the actuators as and when required. These critical activities proceed along with other non-critical activities such as logging, e-mails, and surveillance, etc.

Internet-based Banking Applications: In an Internet-based banking application, bank customers carry out their banking transactions using web browsers in the comfort of their homes. Though this is often considered to be very desirable, for this the Internet banking applications need to be extremely secure and reliable due to the sensitive nature of the financial transactions involved. These applications also have stringent delay requirements. Unless a transaction is completed in time, the server may time out leading to rejection of user requests and can cause user dissatisfaction.

Multimedia Multicast: In a multimedia multicast application, many receivers located at geographically different places receive multimedia information from a multimedia server. The multimedia information is usually in the form of streaming video and audio. Transmission and processing of these information have firm real-time requirements.

Internet Telephony: VoIP is all set to revolutionize voice communication. VoIP stands for Voice over IP, where IP refers to the Internet Protocol that underlies all Internet communication. By using VoIP phones and technology one can make phone calls through Internet. The benefit is that, as the actual voice traffic is carried by the Internet, VoIP is free or costs much less than an actual telephone call, especially over long distances.

The network encrypts voice signals into data packets at the sender end and decrypts it into voice calls at the receiver end. This encryption and decryption is from an analog signal (i.e. a voice call) into digital signal (data packets) that is places on the network and again this is decrypted into analog signals at the receiver end.

VoIP applications are normally used with a simple microphone and computer speakers, but IP telephones can also be used, providing an experience identical to normal telephoning. VoIP applications and services require firm real-time data transfer support.

2 Basic Concepts

In this section, we discuss a few basic concepts such as quality of service parameters and traffic categorization that are important to understanding our discussions in the rest of this chapter.

2.1 Types of Networks

In real-time communication, mainly three types of networks are relevant: Controller Area Networks (CANs), Local Area Networks (LANs), and Internet. This classification is made on the basis of the size of the network and the communication technology deployed. In the following, we briefly discuss these three types of networks.
Controller Area Network (CAN) CANs are very small networks. CANs are typically used to connect the different components of embedded controllers. In an automotive system the different components such as engine, brakes, etc. are connected and controlled through a CAN. The end-to-end length of a CAN is usually less than 50 meters. Since, the propagation time of a CAN is very small, functionally a CAN behaves like a local bus in a computer.

The present day automotive electronics is fairly sophisticated, with automated engine management systems, fuel injection, active suspension, braking, lighting, air-conditioning, security, and central locking. A considerable amount of information exchange among these components is required. The conventional method of networking such components was point-to-point wiring. This methodology was a simple evolution of the simple electrical systems used in earlier cars. However, as the sophistication of the cars grew, such a simple scheme would have required several kilometers of wiring, adding to the cost of manufacturing, weight, complexity, and at the same time resulting in severely reduced reliability.

This gave rise to the development of CAN. A special requirement on CAN is to handle noise. Automotive components such as electric motors, ignition systems, and RF transmissions, etc. are heavy producers of noise. Another distinguishing feature of CAN is the 12 volt power supply, that has been mandated by the conventional 12 volt automotive power supply. CAN specifies only the physical and data link layers of ISO/OSI model, with higher layers being left to the specific implementations.

Because of its robustness, CAN ha expanded beyond its automotive origins and can now be found in industrial automation systems, trains, ships, agricultural machinery, household appliances, office automation systems, elevators, etc. CAN now is an international standard under ISO11898 and ISO11519-2.
Local Area Network: A local area network (LAN) is within a building or a campus and is usually privately owned. A LAN is used to connect a number of computers within an organization to share data and other resources such a files, printers, FAX services, etc. LANs typically operate at data rates exceeding 10Mbps and many present day LANs (gigabit Ethernets) operate at 1 Gbps. LANs are usually implemented using broadcast networks.
Internet: The Internet, sometimes called simply "the Net," is a worldwide system of computer networks - a network of networks in which users at any one computer can, if they have permission, get information from any other computer and sometimes talk directly to users at other computers using VoIP. We must however be aware of the following two distinctions: the internet (lowercase i) is any collection of separate physical networks, interconnected by a common protocol, to form a single logical network. The Internet (uppercase I) is the worldwide collection of interconnected networks, which grew out of the original ARPANET [17], that uses Internet Protocol (IP) to link the various physical networks into a single logical network. The Internet began in 1962 as a resilient computer network for the US military under the ARPANET project, and over time has grown into a global communication tool of more than 12,000 computer networks that share a common addressing scheme.

2.2 Quality of Service (QoS)

Real-time applications need guarantees regarding the service quality from the underlying network for their satisfactory operations. The service quality expected of the underlying network is often expressed in terms of certain Quality of Service (QoS) parameters. Real-time applications usually have stringent requirements on the following QoS parameters: bandwidth, maximum delay in transmission, delay jitter, loss rate, and blocking probability. We elaborate these QoS parameters in the following.

Delay: A successful delivery of a packet by a communication network in a real-time application depends not only on receiving the packet intact, but also on the time at which it is received. If the time the network takes to deliver a

packet exceeds the specified delay bound, then the application times out and can result in a failure in case of a hard real-time application and a degradation of service quality in case of a soft real-time application.

Delay jitter: Jitter is defined as the maximum variation in delay experience by messages in a single session. In other words, it is the difference between the maximum and minimum delays that packets encounter in a packet-switched network. For example, if the maximum end-to-end delay delay experienced by a packet in a class is 1 ms, and the maximum delay is 10 ms, then the delay jitter of the call is 9 ms.

As a packets travel in a network, they may undergo different amount of queuing delays at different nodes. Also, packets may travel in different paths causing jitter. Jitter is unacceptable in many applications, especially in many hard real-time application, and in soft real-time applications such as multimedia applications. In a video application, jitter can show up as picture remaining still for certain time.

In many applications, buffers at the receiver can be used to control jitter. In this approach, removing jitter requires collecting packets in buffers and holding them long enough to allow the slowest packets to arrive in time to be played in correct sequence. The amount of buffer space required can be determined from the peak rate and the delay jitter. The size of the buffer would in fact be given by $peak/rate \times delay\ jitter$. For example, a single video source transmitting 30 frames per second, each containing 2Mb of data and experiencing a jitter of 1 second, would require 60Mb of buffer space at the destination to eliminate jitter. Providing a tight delay jitter bound can reduce the buffering required at the receiver.

Bandwidth: This parameter indicates the rate at which a connection would be serviced by the network. Sufficient bandwidth is required to sustain the required throughput for an application.

Loss rate: Loss rate denotes the percentage of all transmitted packets that may be lost during transmission. Packets in a connection may be lost due to a variety of reasons such as delay bound violation, delay jitter-bound violation, buffer overflow, and data corruption. Data corruption is insignificant in fibre optic media and can be ignored, whereas data corruption is significant in wireless media. Different applications are sensitive to loss rate to different extents. Applications like process control require zero loss rate while applications like multimedia can sustain a certain amount of loss-rate.

Blocking probability: Blocking probability is the probability of a new connection is rejected by the admission control mechanism of a network.

The requirements of different real-time applications with respect to the above QoS parameters may be very different. For example, certain applications such as non-interactive television and audio broadcasting require stringent bounds on jitter but are more or less insensitive to delay. Whereas, sensor data processing in a fly-by-wire aircraft is usually very sensitive to delay, i.e. only delays of sub-milliseconds range in receiving sensor signals are acceptable. In contrast, traditional computer communication applications such as file transfer, electronic mail, and remote login are non-real-time applications. The QoS metrics for such applications are very different from what we discussed for real-time applications, and typically average packet delay and average throughput are considered, rather than worst-case packet delay or worst-case throughput. These applications have also strict reliability requirements since any corruption of parts of a document can make the entire document unusable. Indeed much of the complexity of the traditional network protocols arises from the need for loss-free communication between data-oriented non-real-time applications.

Another notable example of a specific service requirement occurs in case of video transmissions. Video transmissions are very sensitive to packet losses since these show up as flickers or glitches on the display screen. Voice data packet sizes are therefore deliberately kept small to minimize the packetization delays and to limit the effect of packet losses. The standard 48-byte cell size for ATM network for example was chosen primarily for the benefit of the voice applications.

2.3 Traffic Categorization

It is important to categorize a traffic source based on its traffic generation characteristics. Such categorization becomes necessary for the network to be aware of the categories of the traffic from different sources for it to be able to give quality of service guarantees. The traffic generated by a source can be categorized based on the rate at which data are generated by the source for transmission on a network. The following are three important categories of traffic that are commonly encountered.

CBR Traffic: CBR traffic arises due to Constant Bit Rate data generation. Data transmission involved in had real-time applications are usually CBR traffic. For example, periodic data generated by sensors are examples of CBR traffic.

VBR Traffic: VBR traffic as the name suggests consists of different rates of data generation and transmission at different times. There are several types of VBR sources. In VBR traffic, the data source alternates between a period in which fixed sized packets are generated with deterministic spacing and an idle period. In another type of VBR traffic, the source periodically submits variable sized packets to the network. Compressed audio signals is an example of such VBR traffic. Typically in compressed audio, to reduce the size of voice traffic, no data is generated during periods of silence. This results in on-off sources where fixed sized data is generated during on periods and no data is generated during idle periods. While silence suppression can produce significant reductions in the bandwidth requirements, this is achieved at the cost of having to reconstruct the original traffic at the receiver. Another example of VBR traffic is compressed video signals. Redundancy in digitized video data is very high. It is therefore often compressed using such algorithms as MPEG, before being introduced into the transmission network. It should be clear that variable bit rate transmission is intended to make better use of bandwidth in applications such as transmission of compressed video or audio.

Sporadic Traffic: In sporadic traffic variable sized packets are generated in bursts. Sporadic traffic occur under very rare circumstances. For example traffic generated by alarm messages belong to this category.

3 Real-Time Communication in a LAN

Many hard real-time applications such as automated manufacturing systems, industrial process control applications, high-speed data acquisition systems, etc. span a small geographical areas. In such situations, local area networks (LANs) usually turn out to be the preferred choice for networking. In these situations, a dedicated point-to-point network is usually difficult to deploy and maintain and is also not cost-effective.

In a LAN, there is a single shared channel and only one node can transmit at any time. The **access arbitration** policy of a network determines when a node can transmit on the channel. The **transmission control** policy determines how long the node can transmit. These two policies are together called **access control techniques** and form the **Media Access Control (MAC)** layer protocol. In other words, the media access control (MAC) protocol in a LAN consists of two processes: an access arbitration process that determines when a node can use the channel and a transmission control process that determines for how long a node can continue to use the channel. Before we discuss how real-time communication can be achieved in a LAN, we discuss some basic aspects about LANs that are crucial to understanding our subsequent discussions.

3.1 LAN Architectures

There are two major LAN architectures that are normally used. These are the bus and the ring architectures [17, 9]. These two architectures use different access control techniques. We first briefly discuss these two LAN architectures, and subsequently we discuss the associated access control techniques.

Bus-based Architectures: In bus-based architectures, the nodes are connected to the network cable using T-shaped network interface connectors as shown in Fig. 2. Terminating points are placed at each end of the network cable.

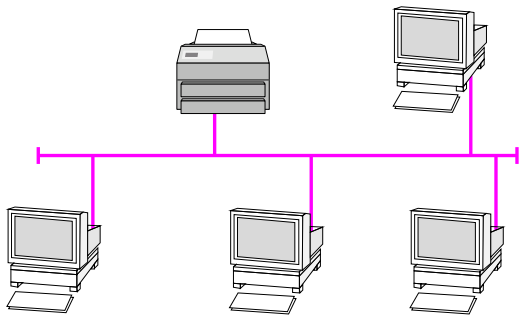

Figure 2: A Bus Architecture

There is a single shared channel for which the transmitting nodes contend. The most commonly used protocol for access control in traditional bus networks is the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) [17]. In bus networks, when two or more nodes transmit packets simultaneously, they overlap in time and the resulting signal gets garbled. Such an event is called a collision. A collision entails retransmission of the corrupted data.

In CSMA/CD networks, any node can at any time sense the channel to determine whether the channel is idle. A node transmits a packet only if it senses the channel to be idle. But, this does not guarantee that there will be no collisions. Several nodes might sense the channel to be idle at the same time instant and start transmitting simultaneously, resulting in a collision. The transmitting nodes can also detect a collision when it occurs. Therefore while transmitting a packet, a node would check for a collision and would immediately stop transmitting, if it detects one. Large propagation delays increase the probability of collisions.

Ethernet is a LAN standard based on CSMA/CD access control. Due to its ubiquity, high speed, simplicity and low cost, Ethernet has over the years emerged as one of the most preferred LAN protocol. It is therefore not surprising that many attempts have been made in the past to design protocols based on Ethernet to support real-time communication. However, the logical ring architecture possesses significant advantages over Ethernet due to its inherent deterministic access arbitration mechanism in contrast to the collision based mechanism of Ethernet. For example, in Ethernet the delay in message transmission increases as the traffic increases. Under high traffic situations, the throughput and maximum delay almost become unacceptable.

Ring-Based Architectures: A ring-based architecture has schematically been shown in Fig. 3. In Fig. 3, an multistation access unit (MSAU) is a hub or concentrator that connects a group of computers ("nodes" in network terminology) to a Token Ring local area network. In a ring-based architecture, the nodes are arranged in a ring and the nodes transmit in turn usually for certain predetermined periods of time. Therefore, the transmission delays are predictable and can be made sufficiently small as per requirement. Therefore, as already mentioned earlier, ring-based architectures are often preferred in real-time applications.

However, the ring architecture suffers from a few important problems. First, any break in the ring can bring the whole network down. This makes reliability of ring networks a major concern. Further, ring is a poor fit to the linear topology normally found in most assembly lines and other applications. This made researchers to look for an alternative technology which can have the advantage of both bus and ring architecture. This led to the development of the token bus architecture. A token bus architecture is a bus based architecture, where the stations on the bus are logically arranged in a ring with each station knowing the address of the station to its "left" and "right" (see Fig. 4).

When the logical ring is initialized, the highest numbered station gets a chance to initiate its transmission. After completing transmission for a predetermined time, the station passes the frame transmission permission to its immediate (left or right as per the convention adopted) neighbor by sending a special control frame called a token.

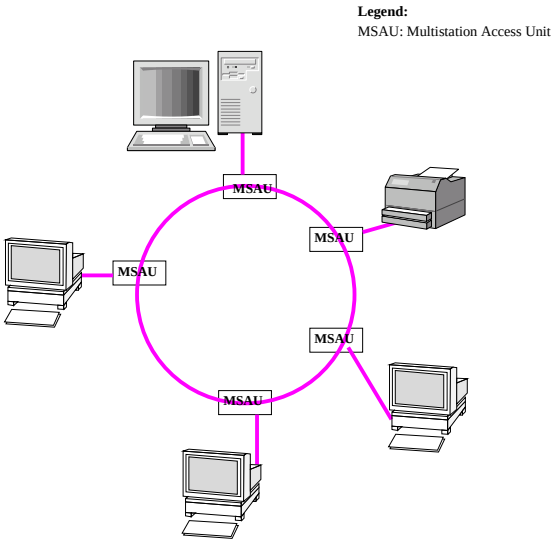

Figure 3: A Ring Architecture

The token propagates around the logical ring. At any time only the token holder is permitted to transmit frames. Since only one station at a time holds the token in a ring network, collisions can not occur. An important point to realize is that the physical order in which the stations are connected to the cable is not important. This is because the cable is inherently a broadcast medium, each station receives every frame transmitted, discarding those that are not addressed to it. When a station passes the token, it transmits a special token frame, specifically addressed to its logical neighbor in the ring, irrespective of where that station is physically located on the cable. It is also worth noting that many stations on the network may not be not in the ring (for e.g. stations 11, 15 in the Fig. 4). The MAC protocol also provides for adding stations to, and removing stations from the logical ring.

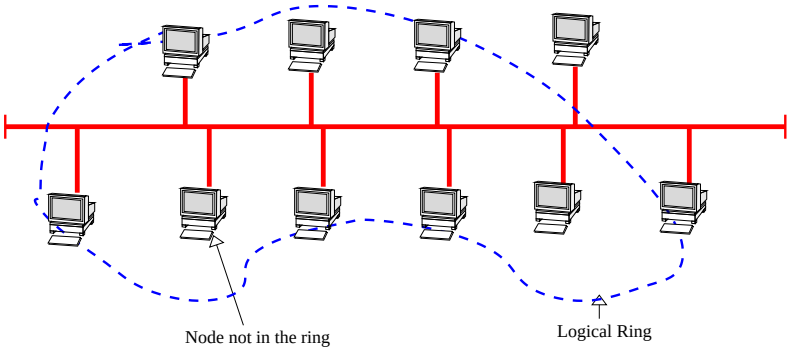

Figure 4: Logical Ring in a Token Bus

4 Soft Real-Time Communication in a LAN

Soft real-time communication can be used to support soft real-time applications in a LAN [8]. Soft real-time communication networks are not expected to provide any absolute QoS guarantees to the applications. They only ensure prioritized treatment for real-time messages so that the message deadline miss ratio (for real-time messages) can be kept to a minimum and the real-time messages can be provided statistical guarantees on delay bounds.

Soft real-time protocols usually assume that both soft real-time and non real-time messages would be transmitted over the network. Soft real-time traffic is usually assumed to be made up of CBR and VBR sources. Soft real-time message rates are assumed to be very low compared to the channel capacity. Non real-time messages arrive aperiodically in bursts. In presence of bursts, it becomes very difficult to sustain the guarantees to the real-time traffic. Therefore, the bursts need to be smoothed for sustaining the statistical guarantees on deadline bounds for real-time messages to hold.

Kweon and Shin developed a fixed-rate traffic-smoothing algorithm. This algorithm is based on the network wide transmission limits. From the transmission limits, the input limit for each node in the system was derived. The traffic smoother, which is placed between the MAC layer and TCP/IP layer, smooths a non-real-time stream so that guarantees to real-time messages is not be violated. The traffic-smoothing concept is a Leaky Bucket algorithm called as Credit Bucket Depth (CBD). It has two parameters CBD and Refresh Period (RP). These two parameters are fixed statically. CBD is the maximum number of credits that are added to the bucket at every refresh. It is also the maximum number of credits that a bucket can hold. RP is the refresh period with which the bucket is replenished with new credits. The ratio CBD/RP is the average guaranteed throughput for non-real-time messages. The number of credits in the bucket is denoted as the current network share (CNS). When the number of available credits is positive, but is smaller than the the size of a message to be transmitted, credits are allowed to be borrowed. So, the balance of credits can become negative at any time. The CNS may become negative if credits have been borrowed. At every refresh, the number of credits in the bucket (CNS) is updated as follows: CNS = min (CNS + CBD, CBD). That is, CBD amount of credits are replenished, subject to the limit that CNS does not exceed CBD. When a non-real-time message arrives from the TCP/IP-layer, the smoothing mechanism carries out the following steps:

```
if (CNS > 0)
            CNS = CNS - message.numberOfBytes
            /*  message.numberOfBytes is the size of the message */
            send message for transmission
else
            hold message in buffer until CNS > 0
```

The disadvantage of the fixed-rate traffic smoothing technique is that it is not very flexible as the network wide input limit is fixed once the delay requirements of the real-time messages are known. The transmission rate of the non-real-time sources is limited and gets reduced as the number of nodes increases on the LAN. Therefore, an increase in the number of stations, leads to a decrease in station traffic generations limits. This shows up as a corresponding decrease in throughput of non-real-time traffic. It is also very pessimistic since it is based on worst-case calculations on the total traffic arrival rates in the system. Kweon and Shin have proposed an adaptive traffic smoother that addresses the shortcomings of the CBD algorithm [10].

Adaptive Traffic Smoothing: In order to provide a reasonable throughput for non-real-time messages, the non-real-time traffic-generation rate can be allowed to adapt itself to the underlying network load conditions. That is, the nodes are allowed to increase their transmission limits, if the utilization of the bus is low. On the other hand, when the utilization of the bus becomes high, the nodes transmitting non-real-time messages have to decrease their transmission limits. In order to implement an adaptive-rate traffic smoother which would meet the delay requirements of real-time packets, and at the same time provide improved average throughput for non-real-time packets, the following two problems must be resolved:

- How to detect a change in network utilization?
- How to adapt the transmission limits to a detected change?

The solution that has been proposed is that the number of collisions per unit time can be used as a measure of network utilization. This is a simple and approximate way to determine network utilization at any time. In the

event of a collision, the credit bucket is immediately emptied causing suspension of non-real-time packets, except for those packets already under transmission. This increases the chance to deliver the real-time packets generated from other nodes sooner, and prevent delays caused by bursts of non-real-time packets generated from this node. For this reason, packet collision is used as a trigger to decrease throughout as well as to deplete the current credits.

The station input limit CBD/RP can be adapted by either changing the CBD or the RP parameter. The station input limit is increased periodically by decreasing RP periodically by a fixed number, if there has not been any recent collision. Of course, the lower bound of RP is fixed by some predefined value RPmin. When a collision is detected, all credit buckets are depleted and RP is doubled. The upper bound of RP is fixed by a certain predefined value RPmax.

The experimental results have shown that by using the adaptive traffic smoothing scheme the message deadline miss ratios (for real-time messages) can be kept well under the milli seconds range for any arbitrary arrival rate for the non-real time messages. This technique though efficient fails to provide deterministic real-time communication. Hence it is suitable only for soft real-time systems and can not support hard real-time communication.

5 Hard Real-Time Communication in a LAN

We had discussed several example applications in Section 6.1, for which hard real-time communication in a LAN environment is necessary. Hard real-time applications often involve transmission of CBR traffic such as periodic sensor signals. In hard real-time applications, deterministic predictability of network delay takes precedence over network utilization aspects. However, to improve the network utilization, often soft and non-real-time transmissions are allowed to be transmitted in those intervals during which hard real-time messages are not present.

In a LAN, hard real-time communications are normally supported using any of the following three classes of protocols [1, 11]: global priority-based, calender-based, and bounded-access protocols. We first discuss some important features of these three classes of hard real-time communication protocols. Subsequently, we discuss some important protocols belonging to these three classes.

Global Priority Protocols: In a global priority protocol, each message is assigned a priority value. This MAC layer protocol tries to ensure that at any time the channel is serving to the highest priority message in the network. This opens up the possibility of using RMA for scheduling messages in a network deploying a global priority-based protocol. However, two important problems arise while trying to implement RMA or EDF message scheduling algorithms using a global priority-based scheduling protocol:

1. When the transmission of a packet is underway, it cannot be stopped before the transmission is complete and the remaining bits of the packets transmitted at a later time. In other words, unlike the way tasks could be preempted from CPU usage, packet transmissions cannot be meaningfully preempted channel usage.
2. A global priority protocol can not instantaneously determine the message that has the highest priority. As we shall see in our subsequent discussions is that what is obtained in practice is based on somewhat outdated information.

Both these problems violate some of the basic premises that were made for RMA to optimally schedule tasks. Therefore, if RMA is used to schedule messages in a global priority protocol, the above mentioned two problems would restrict the achievable schedulable utilization of the channel to very low values.

Bounded Access Scheduling: In bounded access scheduling, real-time guarantees to messages are provided by bounding the access time of every node to the channel. This in turn implies that the time for which a packet may have to wait before it is transmitted is bounded. A local scheduling algorithm is used to determine the order in which packets are taken up by a node for transmission .

Calendar-based Scheduling: In a calendar-based protocol, every node maintains a copy of the calendar that indicates which node is permitted to transmit during which time period. Traffic sources can reserve time interval for packet transmission by broadcasting. When a node needs to transmit a message for which no reservation was made, it finds a free slot by consulting the calendar and reserves the required time interval by broadcasting the reservation information to all nodes. A calendar-based protocol is simple, efficient, and works very well when all messages in the system are periodic and predictable.

In the following, we discuss some important protocols from each of these three categories of protocols.

5.1 Global Priority-based Scheduling

A few important global priority arbitration based protocols are the following.

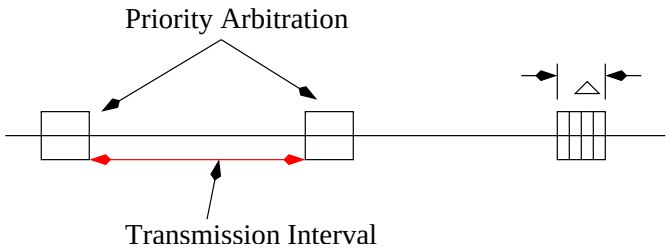

Figure 5: Priority Arbitration and Transmission Intervals

Countdown Protocol: In this protocol, the time line is divided into fixed intervals. At the start of every interval, priority arbitration is carried out to determine the highest priority message in the network. As soon as priority arbitration is complete, the node having the highest priority message is allowed to transmit. In other words, this scheme involves periodic priority arbitration followed by message transmission as shown in Fig. 5.

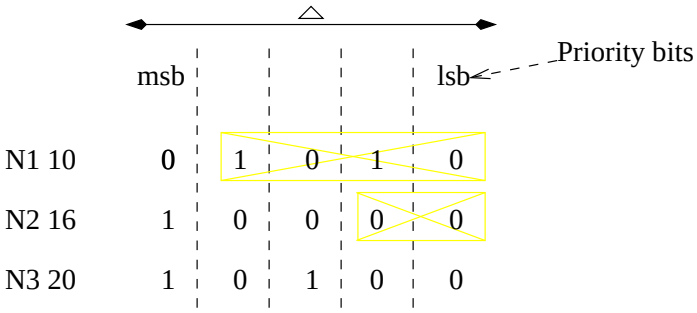

Figure 6: Priority Arbitration Example

Priority arbitration time is divided into fixed sized slots. The duration of each slot duration is typically made equal to end-to-end propagation delay of the medium. If the slot size is made any smaller than this, then the priority arbitration scheme would not work since a collision even when it occurs would not be detected. A slot size larger than this would lead to an increase in channel idle time during every slot. Over these slots, every node transmits the priority value of its highest priority pending message with msb first as shown in Fig. 5. Since simultaneous transmission follows *or* logic, a node that transmits a 0 and receives a 1, knows that there is at least one node that is having a higher priority pending message, and drops out of the contention.

To illustrate the working of this protocol, transmission of bits by different nodes in a hypothetical example during the arbitration interval has schematically been shown in the Fig. 6. In Fig. 6, three nodes N1, N2, and N3 are participating in the arbitration process. The priorities of highest messages at the nodes N1, N2, and N3 are 10, 16, and 20. As shown, node N1 drops out after transmitting the first bit as it transmits a 0 but listens a 1. Similarly, the node N2 would drop out after transmitting the third bit when it would transmit a 0 and listen to a 1. Finally, after transmitting the last bit, node N3 would conclude that it has the highest priority message and would begin its transmission.

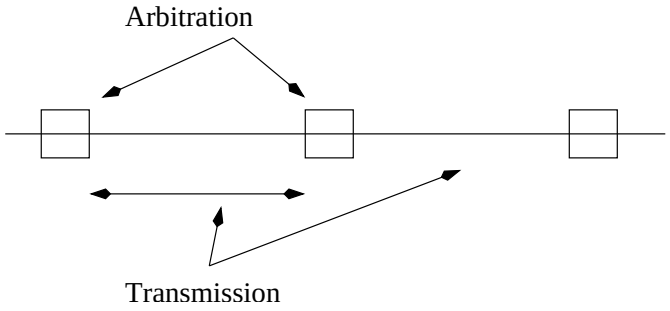

Figure 7: Priority Arbitration in Virtual Time Protocol

Virtual Time Protocol: In this protocol a node uses the state of the channel to reason about the pending packets residing at other nodes. Each node with a packet to send waits for an interval of time that is inversely proportional to the priority of highest priority message it has. That is the lower is the priority of a message that a node has, the more time it waits. At the expiration of the waiting time, the node senses the status of the channel. If the channel is busy, then this means that a higher priority message is being transmitted and it needs to wait until an idle period. So, if it finds that transmission is already taking place, then it waits for the next frame, otherwise it starts to transmit.

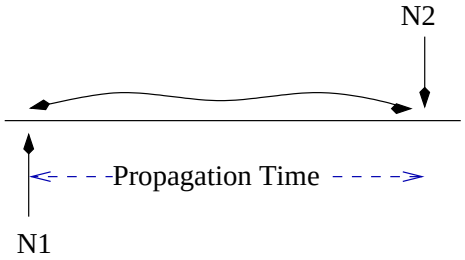

Figure 8: Problem in Virtual Time Protocol

What should be the difference in wait times of two nodes which have messages whose priorities differ by one? Because of the propagation delay in the network, a node can not instantaneously detect when another node starts transmission during arbitration. As a result, unless the wait time is at least as large as the propagation time, the following problem would occur. Assume that two nodes N1 and N2 have their priorities differ by one. After N1 starts transmitting, if N2 waits for any time lower than the propagation time, then it can not detect the transmission of N1 and would start transmitting. This would lead to collision and incorrect priority arbitration. This bounds the difference between the wait times of two nodes that have consecutive priorities.

5.2 IEEE 802.5:

IEEE 802.5 is a priority based token ring protocol. The header of a token contains two fields: a priority field and a mode field. In this protocol, token alternates between two modes: a reservation mode, and a free mode.

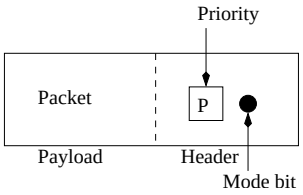

Figure 9: Structure of a Token in IEEE 802.5

In reservation mode, packet transmission occurs. The priority of the message that is being transmitted is registered in the header of the token as shown in Fig. 9. As messages pass through the ring, every node with pending messages inspects the reservation field in the token header. A node having a higher priority message, registers its priority in the priority field of the header. When the token returns to the sending node, it puts the token in free mode and releases it. As the token passes through the ring, the node that made the reservation seizes the token, puts it into reservation mode, and starts transmitting.

There are two important results about this protocol.

Theorem 1: *The minimum time to complete transmission of a packet (that is, the time to start transmission of next packet) is* $max(F, \theta)$*, where* F *is packet transmission time and* θ *is propagation time.*

Proof: A node does not start transmitting the next packet until:

- It has completed transmission of the last bit.
- It has received the header of token back.

Therefore, the transmission time is the higher of the two values. □

Theorem 2: *A higher priority packet might undergo inversion for at best* $2 \times max(F, \theta)$.

Proof: First, the reservation mode must complete before any other node transmits and secondly the node must receive the free node token.

For first, it takes $max(F, \theta)$ and for second also it at most takes $max(F, \theta)$ time. Therefore, the total time a higher priority packet may wait is given by $2 \times max(F, \theta)$.

□

5.3 Window-based Protocol

In this protocol, time is divided into frames as shown in Fig. 10. Every node maintains the current transmission window $(low, high)$, where low and $high$ refers to the lower and higher priority. A node that has a message whose priority is within low and $high$ starts transmitting. On a collision, every node increases the value of low (i.e. makes $low++$) and on a free frame every node reduces low (i.e. make $low--$).

5.4 Calender-based Protocol

An example of calendar-based scheduling is the dynamic reservation technique. In this protocol, each node maintains a calender data structure where information about the access time reservations of the guaranteed messages are

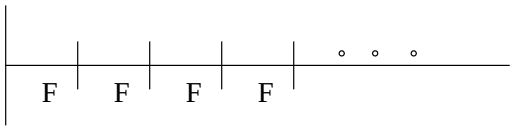

Figure 10: Frames in the Window-based Protocol

maintained. When a message for which no reservation has been done yet arrives at a node, the node attempts to reserve a future time interval by broadcasting a *control message* to all nodes. Each node up on receipt of the control message updates its calendar accordingly. This node can very efficiently handle deterministic periodic messages, and there is no overhead in priority arbitration unlike the other discussed protocols. However, it becomes very difficult to handle aperiodic messages and sporadic messages in this protocol. Therefore this protocol is used only in very simple systems.

6 Bounded Access Protocol

We discuss IEEE 802.4 and RETHER as two popular examples of bounded access protocols. In the following, we discuss these two protocols.

6.1 RETHER

RETHER stands for Real-time ETHERnet. RETHER is enhances TCP/IP and provides real-time performance guarantees to real-time applications without modifying existing Ethernet hardware [18]. Network transmissions can occur in two modes: CSMA/CD mode or RETHER mode. The network switches transparently to RETHER-mode when there are real-time sessions and back to CSMA/CD-mode when all real-time sessions terminate. Protocol switching done to minimize the performance impacts on non-real-time traffic when there are no real-time sessions. In RETHER mode token passing scheme is used.

Protocol Description: In the absence of any real-time messages at the nodes, nodes compete for the channel using the usual CSMA/CD protocol of the Ethernet. When a node receives a real-time request from a local application, it broadcasts a Switch-to-RETHER message on the Ethernet unless the network is not already in Rether mode. Every node that receives this message responds by setting its protocol mode to RETHER mode. The transmitting node waits for the ongoing packet transmission to complete. It then sends an acknowledgment back to the initiator, indicating its willingness to switch to RETHER mode and that there is no data left in the backoff-phase of CSMA/CD protocol. Upon receiving of all the acknowledgments, the initiating node creates a token and begins circulating it. This completes a successful switch to RETHER mode.

If more than one initiators try to initiate RETHER-mode at the same time, all nodes only acknowledge switch messages to the initiator with the smallest ID among those that have already been acknowledged. An initiator only sends an acknowledgment to another initiator if its node ID is smaller than its own. In the case of loss of the acknowledgment, or when some node does not receive the Switch-to-RETHER broadcast, or when some nodes are dead, then the initiator times out due to non-receiving of all the acknowledgments. After a fixed number of retries, it concludes that the nodes that did not acknowledge are dead which it conveys to all other live nodes.

The RETHER mode uses a timed token scheme to provide bandwidth guarantees. At any time there is only one real-time-request per node and each real-time request specifies the required transmission bandwidth in terms of the amount of data it needs to send during a fixed interval of time called MTRT - Maximum Token Rotation Time. The Maximum Token Holding Time is calculated for each node based on this information (amount of data and MTRT). An average constant bandwidth is reserved for each session.

Rether Protocol: The control token circulates among two sets of nodes, the real-time set (RTS) and the non-real-time set (NRTS). Only nodes that have made a bandwidth reservation belong to RTS. All other nodes belong to NRTS. During each token rotation, the token visits all nodes in real-time-set in order. When a node in real-time-set receives the token it sends one unit of real-time-data and then passes the token to its neighbor in the real-time-set. The last node in the RTS passes the token to the NRTS. Let MTRT be the mean token rotation time and $MTHT_i$ and let $MTHT_i$ be the mean token holding time for node N_i. The token is then tagged with a TimeToDeadline field such that:

$$TimeToDeadline = MTRT - \sum_{i \in RTS} MTHT_i$$

When an NRT node receives the token, it determines if there is sufficient time to send a packet before the token deadline. If there is, it sends the packet, decrements TimeToDeadline accordingly, and passes the token to the next node in NRTS. If there is no time to send a packet, it informs the last node in the real-time set that it should be the first node to get the token among NRTS nodes during the next round, and it passes the token to the first node in the RTS.

Every new real-time request goes through an admission control procedure that determines if it is possible to accept the request. Admission control is performed locally on each node: Admit real-time request if

$$\sum_{i \in RTS} MTHT_i + MTHT_{new} + TBNRT \leq MTRT$$

Where TBNRT is the bandwidth reserved for the NRT set. Admission control is not performed until the node receives the token. When a real-time node wants to terminate its real-time connection, it merely removes itself from the RTS information on the token.

6.2 IEEE 802.4

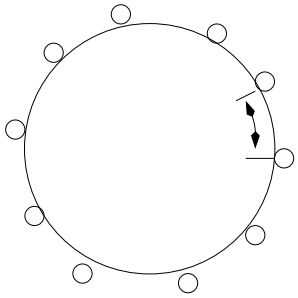

Figure 11: Token Ring in IEEE 802.4

IEEE 802.4 protocol is applicable to token ring networks. This protocol is often referred to as the *timed token protocol*. In this protocol, the amount of time each node holds a token is bounded. It has been incorporated in FDDI.

In IEEE 802.4, TTRT(Target Token Rotation Time) is used as design parameter. TTRT is the expected time between two consecutive visits of the token to a node. At the network initialization time, the expected token rotation time is specified as TTRT. Individual nodes are allocated a portion of TTRT, known as its *synchronous bandwidth* according to the timing characteristics of the periodic messages originating at each node. A station transmits non-real-time messsagesm only the token arrived earlier than expected. The time needed to transmit an asynchronous frame is called *asynchronous overrun* and it reduces the effective bandwidth available to transmit synchronous messages. Due to asynchronous overrun, the worst case time between two successive visits of the token to the node is $2 \times TTRT$. However, in a network that uses only synchronous mode, time between consecutive token arrivals is limited by TTRT.

TTRT can be represented as
$TTRT = \theta +$ *holding time at nodes*
where θ is the propagation time. Suppose of all messages that the different nodes in the network can originate, node N_i has the message that has the smallest deadline Δ. We can then fix TTRT to be:

$$TTRT \leq \frac{\Delta}{2}$$

Each node N_i is assigned a synchronous bandwidth H_i, which is the maximum time the node N_i is allowed to hold the token to transmit periodic messages. When a node receives the token, it first transmits its synchronous traffic for an time bounded by the synchronous capacity. Then it may transmit asynchronous traffic, only if the time since the previous token departure from the same station is less than TTRT.

Token holding times at individual nodes is the synchronous bandwidth alloted to the node. As soon as a node receives the token, it starts a timer set to its synchronous bandwidth, and releases the token upon expiry of the timer. The synchronous bandwidth allocated to a node N_i is given by the following expression:

$$H_i = TTRT * \frac{C_i/T_i}{\sum C_i/Ti} \qquad \ldots (6.1)$$

Where C_i is the size of the message (in bits) that node N_i requires to transmit over T_i interval, and $\frac{C_i}{T_i}$ is the channel utilization due to the node N_i.

Example 6.1: Suppose a network designed using IEEE 802.4 protocol has three nodes. Node N_1 needs to transmit $1MB$ of data every $300ms$. Node N_2 needs to transmit $1.2MB$ of data every $500ms$. Node N_3 needs to transmit $1.2MB$ of data every $500ms$. Select a suitable TTRT for the network and compute the token holding time for each node.

Solution: From an examination of the messages, the shortest deadline among all messages is $200ms$. Therefore, we can select $TTRT = \frac{200}{2} = 100ms$. The channel utilization due to the different nodes would be:

$$\frac{C_1}{T_1} = \frac{1 \times 8}{300} \text{ Mb/ms}$$
$$\frac{C_2}{T_2} = \frac{1.2 \times 8}{500} \text{ Mb/ms}$$
$$\frac{C_3}{T_3} = \frac{2 \times 8}{200} \text{ Mb/ms}$$
$$\frac{C_1}{T_1} + \frac{C_2}{T_2} + \frac{C_3}{T_3} = \frac{1 \times 8}{300} + \frac{1.2 \times 8}{500} + \frac{2 \times 8}{200} = \frac{377.6}{3000} \text{ Mb/ms}$$

Using Expr. 6.1, the token holding times of the different nodes can be determined to be:

$$H_1 = 100 \times \frac{8}{300} \times \frac{3000}{377.6} = 21.18ms$$
$$H_2 = 100 \times \frac{9.6}{500} \times \frac{3000}{377.6} = 15.25ms$$
$$H_1 = 100 \times \frac{16}{200} \times \frac{3000}{377.6} = 63.56ms$$

□

7 Performance Comparison

At the design time, a detailed knowledge of the message set is often unavailable. However, the designers of a real-time network usually have an estimate of the traffic generated by the different real-time sources in terms of the respective channel utilization. Therefore, the following two utilization-based metrics are meaningful while comparing the performance of different protocols.

- **Absolute Breakdown Utilization (ABU):** This metric indicates the expected value of utilization of a message set S at which messages start missing deadlines.

$$U = \sum_{i \in S} \frac{C_i}{T_i}$$

  where C_i is the size of message $i \in S$, and T_i is its period. We can alternately view this metric to indicate how much traffic can be accommodated in a network without the messages having to miss their deadlines.

- **Guarantee Probability(GP(U)):** The guarantee probability at channel utilization U (GP(U)) indicates the probability that a message set with utilization U meets all message deadline.

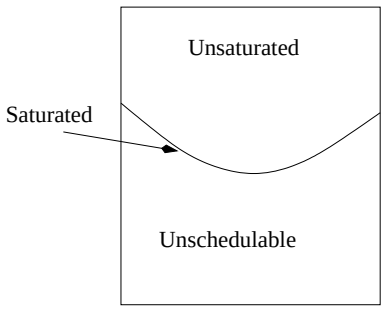

Figure 12: Schematic Representation of Message Sets

We now provide an intuitive explanation of the metrics. The entire population of synchronous messages can be divided into three classes depending on the schedulability offered as shown in Fig. 12. These three classes of messages are briefly explained in the following.

Unsaturated Schedulable: The message set in this class are schedulable and remain schedulable even after any message size is slightly increased.

Saturated Schedulable: The message set in this class are schedulable but any increase in the message size of a message would lead to some messages missing their deadlines.

Unschedulable: Unschedulable set refers to those messages for which deadlines of some messages would be missed.

We would expect GP(U) to be close to 1 for utilization lower than ABU and approach 0 as utilization increases beyond ABU.

In low bandwidth networks, priority-based protocol works better; whereas at higher bandwidths, the bounded access protocol works better. The performance of the priority-driven protocol initially improves as the bandwidth is increased, but starts to drop off beyond a certain point. This is against the intuition that performance of a protocol should improve as the bandwidth is increased.

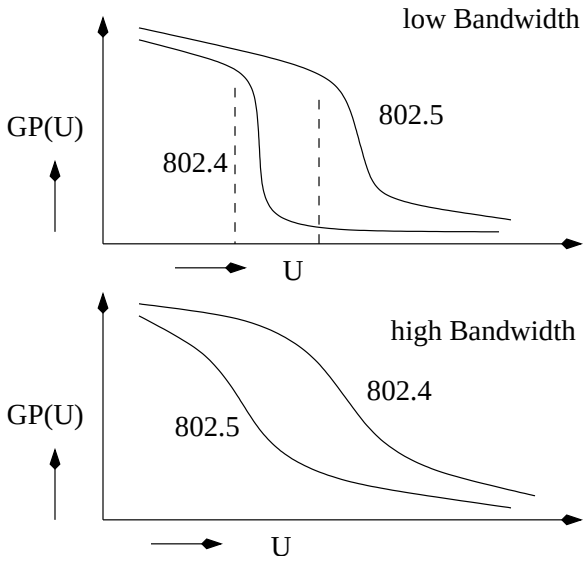

Figure 13: GP(U) versus Utilization at Low and High Bandwidths

When the bandwidth is increased beyond a certain value, the decrease in transmission time causes the frame transmission time F to be less than the token rotation time θ. In this case, before releasing a new token, the transmitting node has to wait for the token to return even after the transmission of the frame is complete to transmit the next packet. Thus, the effective frame transmission time in this case is θ and the fraction of the wasted bandwidth is (θ-F)/F. The propagation time is independent of the bandwidth and hence can be considered as a constant. The token transmission time decreases with increasing bandwidth. Therefore, the percentage of wasted bandwidth increases with increase in bandwidth.

The timed token protocol does not exhibit this anomaly, because a node is permitted to transmit continuously during its assigned time slot.

In all cases, it can be seen that the guarantee probability remains close to 1 as long as the utilization is less than ABU. There is a sharp drop in the guarantee probability at utilization values close to ABU. This demonstrates that ABU is a robust measure of average performance of real-time networks. Now, we can summarize our observations as follows. At low transmission speeds, the priority inversions caused by the round-robin scheduling approach tends to adversely impact messages with short deadlines. Thus, at low bandwidths priority-driven protocol is better suited than the timed token protocol for real-time applications.

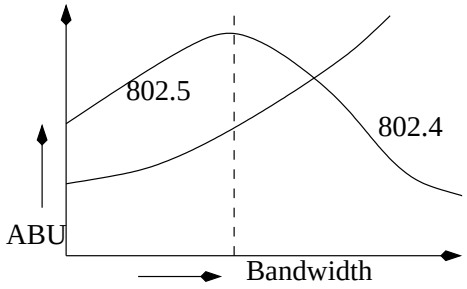

Figure 14: Absolute Breakdown Utilization Versus Bandwidth

8 Real-Time Communication Over Internet

From a modest beginning a little over a decade ago, Internet has become a vast repository of information, and enabler of several new applications such as e-commerce. Internet provides *best effort* service to applications, meaning that traffic is processed as quickly as possible but there is no guarantee of timeliness or actual delivery. With the increasing commercial usage of Internet, there is a need to support soft and firm real-time applications. In this section, we first provide some basic concepts about Internet. Subsequently, we review the recent trends in this area.

8.1 A Basic Service Model

The basic Internet service model has schematically been shown in Fig. 15.

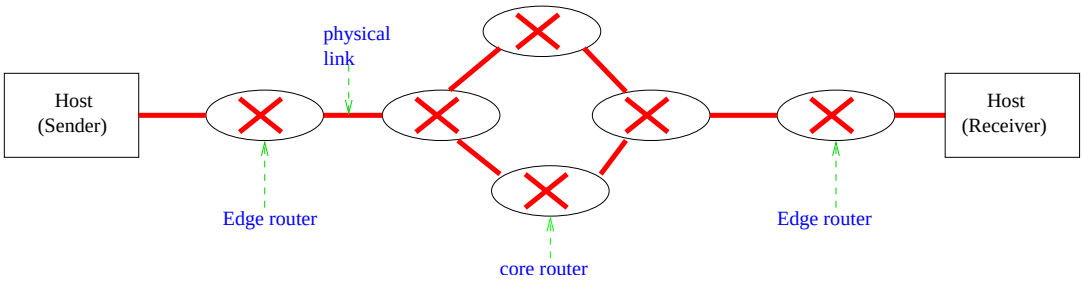

Figure 15: A Simple Model of the Internet

The Ingress router is the router attached at the sender's end. It is the first interface to the data sent by the sender host. The type of physical link used at a link may be a twisted pair cable, a fiber optic cable, and even wireless link. The Internet service is provided by an Internet Service Provider (ISP). Therefore, the data passes through the ISP's routers. These are termed as core routers. The density of these routers is very high at the ISP end, since huge amount of data and links typically converge at the ISP. From the core routers, the data is forwarded through the egress routers (at the receiver end) to the corresponding receivers.

Real-Time communication can be supported on the Internet by providing preferential treatment to data of certain senders that have entered into a contract with the ISP. But for that data packets should have been classified before transmission. At the connection set-up time, the sender provides the ISP with two parameters: the traffic specification and the service specification. The ISP uses this to determine whether it can accept or has to refuse the client request. It checks for the resources on the possible routes of that packet data. Alternatively, we can say it tries to find a path (unicast or multicast routing) that has the necessary resources to satisfy the required QoS for the specified traffic pattern to be generated by the source.

If the network accepts the connection request, then it promises that the data transmission service will meet (or exceed) the client-specified performance bounds. However, the given guarantee remains valid only if the client-generated traffic remains within the specified traffic bounds. If the client traffic violates the prespecified traffic bounds, then the network may drop packets, or reshape the traffic.

Another important component of this approach is the protection given to the real-time connections. Note that the network service will meet the given performance bounds as long as the client traffic remains within the traffic specification, irrespective of the behavior of other clients. Now, this performance cannot be guaranteed, if the connections are not protected from the misbehaving (or malicious) real-time and non real-time sources. To avoid violating the guarantees made to real-time connections, the network must either explicitly control the input rates on a per-connection basis i.e. shaping and policing of the data, or adopt scheduling algorithms that will do so

automatically in the nodes (e.g., Fair Queuing, Weighted Fair Queuing, etc.).

These techniques are a part of integrated software architectures like IntServ, DiffServ etc., which have been designed keeping in mind various issues pertaining to real-time communication. We discuss these issues in the following.

8.2 Traffic Characterization

Traffic characterization is a model of the data generation characteristics of a source. Normally a traffic is characterized by bounding its volume. A bound on the volume of the traffic can be used to bound the amount of network resources that may have to be reserved to provide the required quality of services. During connection establishment, resources are provided based on the traffic characterization and resource requirements. There are many specification models proposed in the literature for traffic characterization. We discuss a few important ones.

(X_{min},S_{max}) model:
The (X_{min},S_{max}) model bounds a traffic source with a peak rate. A connection satisfies (X_{min},S_{max}) model, iff the inter-arrival times between two packets is always less than X_{min}, and size of the largest packet is bound by S_{max}. Both the peak and average rates of traffic in this model is given by $\frac{S_{max}}{X_{min}}$.

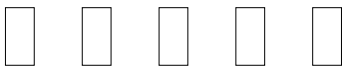

Figure 16: Constant Bit-Rate Traffic

This model provides a tight bound for CBR traffic. From Fig. 16 it can be seen that for CBR traffic both the worst case and the average packet arrival times are the same. However, for bursty traffic use of this model to specify traffic characteristic results in very conservative resource reservation, leading to low utilization of the reserved resources.

(r,T) model:
In this model the time axis is divided into intervals of length T each, called a frame. A connection satisfies (r,T) model, if it generates no more than r.T bits of traffic in any interval T. In this model, r is the upper bound on the average rate over the averaging interval T. This model is in some sense similar to the (X_{min},S_{max}) model. This model also provides a tight bound for CBR traffic. However, for bursty traffic it results in very conservative resource reservation, leading to low utilization of the reserved resources.

(X_{min},X_{avg},S_{max},I) model:
In this model, X_{min} specifies the minimum inter-arrival time between two packets. S_{max} specifies the maximum packet size and I is a certain interval over which the observations are valid. X_{ave} is the average inter-arrival time between packet over any I interval. So a connection satisfies this model if it satisfies (X_{min}, S_{max}) model and the average inter-arrival time of consecutive packets is larger than X_{ave} during any interval of length I. Here in this model the peak rate and the average rate of the traffic are S_{max}/X_{min} and S_{max}/X_{ave}. Notes that this model provides bounds for both peak rate and average rate of the traffic. This model provides better characterization of VBR traffic compared to both (X_{min},S_{max}) and (r,T) models.

(σ, ρ) model:
In this model σ is the maximum burst size and ρ is the long term average rate of the source traffic, respectively. Average traffic is calculated by (number of packets generated over large duration)/(large duration). A connection satisfies (σ, ρ), if during any interval of length t the number of bits generated by the connection in that interval is less than $\sigma + \rho$t. Burst may come over t duration or not because it is very infrequent so no of bits generated is less

than or equal $\sigma + \rho$t. This model can be satisfactorily be used to model bursty traffic sources.

Multiple rate bounding: A more accurate approximation for bursty traffic sources can be obtained by characterizing the traffic with multiple bounding average rates, each over different averaging intervals. A traffic would satisfy {(r1,T1)(r2,T2)...}, if $T1 < T2 < T3$..., and over any interval I the number of bits generated is bounded by ri*Ti if $Ti - 1 < I < Ti$. As the averaging interval gets longer, a source is bounded by a rate lower than its peak rate and closer to its long term average rate.

9 Routing

Once a traffic source has specified its traffic characteristics and its QoS requirements to the network, it waits for the network to accept the request before it can start its transmissions. The network on its part checks if adequate resource is available along any path from source to destination to meet the demand. Route selection (unicast or multicast) for a session during the connection establishment phase.

9.1 QoS Routing

Current Internet protocols use routing algorithms such as the shortest path routing in which the routing can be optimized for some metric without taking into account whether the required resource is actually available. Consequently, the flows might be routed over the paths that are unable to support their requirements while other paths might exist which could have satisfied the specified requirements. This would lead to breach of committed QoS guarantees to connections. Researchers have now proposed *QoS routing* or *constraint-based routing* which overcome these problem. The primary goals of QoS routing are:

- To select routes that can meet certain QoS requirements,
- To increase the utilization of the network.

While determining a route, QoS routing schemes not only consider the topology of a network, but also the requirement of the flow, the resource availability of the links, and other policies that may be specified by the network administrators. Therefore, QoS routing would be able to find a longer and lightly-loaded path rather than the shortest path that may be heavily-loaded. QoS routing schemes are therefore expected to be more successful than the traditional routing schemes in meeting QoS guarantees.

9.2 Routing Algorithms

The complexity of a QoS routing algorithm depends on the chosen QoS constraints such as hop count, bandwidth, delay, delay-jitter etc. The QoS constraints can be divided into three broad classes [19].

Let d(i,j) denote a chosen constraint for the link (i,j). For any path P = (i, j, k,... l, m), a constraint d would be termed *additive*, if d(P) = d(i,j) + d(j,k) + ... + d(l,m). For example, end-to-end delay is an additive constraint and is equal to the sum of the delay of the individual links in the path. Jitter, cost and hop count are also additive constraints.

A constraint is termed *multiplicative*, if $d(P) = d(i,j) \times d(j,k) \times ... \times d(l,m)$. For example, reliability constraint is multiplicative. The reliability of a path is given by the product of the reliability of the individual links.

Similarly, a constraint is termed *concave*, if d(P) = min{d(i,j), d(j,k), ..., d(l,m)}. An example of a concave constraint is bandwidth. Bandwidth on a path is equal to the minimum bandwidth of a link on that path. Wang and Crowcroft proved that the problem of finding path subject to two or more additive and/or multiplicative constraints in any possible combination is NP-complete [21]. However, the proof of NP-Completeness is based on the assumptions

that: 1) all the considered constraints are independent; and 2) the delay and jitter of every link are known a priori. Although the first assumption is normally true for circuit-switched networks, bandwidth, delay, and jitter are not independent parameters in case of packet switched networks. As a result, polynomial algorithms like Bellman-Ford and the extended Dijkstra's algorithm [17] exist for computing routes with hop count, delay, and jitter constraints. Interestingly, bandwidth and hop count constraints are more important than the delay or jitter constraints, since these two are to a large extent determined by the bandwidth and hop count constraints and also most real-time services require bandwidth as an essential constraint while the hop count constraint determines effective resource utilization.

In the last two decades, several algorithms have been proposed to address the various problems associated with unicast and multicast routing. Comprehensive reviews of unicast and multicast algorithms for QoS routing are available in [2, 6]. Multimedia applications, which require multicast QoS routing, are becoming very important and are widely being used in many soft real-time applications. In the following, we discuss some issues relevant to multicast QoS routing, and then investigate a few important protocols.

10 Multicast Routing

Internet is finding growing use in many soft real-time applications. In addition to video-on-demand, and teleconferencing, many distributed applications require multicast communication. Giving each source an independent point-to-point connection to each destination is an unacceptable approach because of the potential wastage of resources. The common approach is to construct for each source a multicast tree connection. Each vertex on a multicast tree represents either a router or a host.

One can save a great deal of resources by having the sources share resources whenever possible. For example, in an audio conference, one (or at best a few) people speak at any time. In such a scenario, we can define a multicast group as one where traffic from a set of sources traverse some common routers and transmission links.

The QoS routing function in a multicast application involves finding a tree rooted at a source and covering all the receivers, with every path from the sender to the receivers satisfying the QoS requirement. The tree construction in multicasting are usually performed in two basic ways: source-based and core-based. We now discuss a few routing protocols belonging to these two broad categories.

10.1 Source-based Routing Protocols

In the following, we discuss two important source-based routing protocols.

DVMRP (Distance Vector Multicast Routing Protocol)
DVMRP [3] implements source-based trees with reverse path forwarding, pruning and grafting. It relies on the exchange of "distance vector updates" among nodes. Each vector entry contains the destination and distance information expressed in terms of number of hops. These updates are sent on each link. The multicast distribution tree is then created.

MOSPF (Multicast Open Shortest Path First) Routing Protocol
MOSPF [12] is a multicast extension to the OSPF unicast link-state routing protocol. In OSPF each router periodically sends link states to all other routers in the networks. Each router builds up a complete network map. With this information each router is able to compute the shortest path to every destination in the network using the Dijkstra's shortest path algorithm [17]. MOSPF extends these link states to also include information about group memberships. Each router advertises the presence of multicast group receivers attached to it. Therefore MOSPF can construct a shortest-path multicast tree, i.e. the path from the source to each receiver in the multicast tree is

the unicast shortest-path. Instead of relying on flooding/pruning information like DVMRP, in MOSPF each router keeps information about all group members at all routers in the network. A limitation of MOSPF is that it does not scale well to large networks.

10.2 Core-Based Tree (CBT) Multicast Routing

In the core-based tree citeball,ball1 approach, first a core for a multicast tree is selected. The different receivers send their subscribe request to this core. Each intermediate router remembers the interface from which this request has been received in order to include it into the distribution tree. When a source sends data to the multicast group, the packet reaches a first on-tree router which then replicates this packet on all the on-tree interfaces except the one from which the packet came from.

PIM-DM/SM routing Protocols
Protocol Independent Multicast (PIM) [4] as the name itself suggests is independent of any underlying protocol. It consists of two multicast routing protocols: Dense Mode (DM) PIM and Sparse Mode (SM) PIM. We now discuss these two protocols.

Dense Mode (DM) PIM: It is intended for inter-domain routing and assumes that group members are densely distributed in the network. Hence, it is reasonable to assume that each and every router should be involved in multicast. Thus, an approach like Reverse Path Flooding that floods datagrams to every multicast router.

Sparse Mode (SM) PIM: It was first intended for wide-area multicast routing. In SM, multicast group members are sparsely distributed in the network. An assumption that is made is that a router should not have to do any work unless it wants to join a multicast group. This is a center-based approach, where routers send explicit "join "messages, but are otherwise not involved in multicast forwarding.

Rather than "core", PIM-SM uses the notion of a rendezvous point (RP). Each group is associated one (or more) RP. New group receivers send "join" messages to the RP. Like CBT, each intermediate router takes advantage of these "join" messages to update the distribution tree. Data issued by a source is first encapsulated in unicast and then sent to the RP before being distributed in the multicast tree (the tree is unidirectional). For those very active sources, and because going through a RP before joining a receiver is moderately efficient, PIN/SM enables the creation of a RPF tree rooted at this source. It is thus less dependent on the center location than CBT.

10.3 QoS-based Multicast Routing

Kompella et al. [14] proposed a distributed heuristic algorithm for constructing a constrained Steiner tree. The algorithm requires every node to maintain a distance vector which stores the minimum delay to every other node. Starting with the source node, the algorithm constructs the multicast tree iteratively by adding a link into the tree each time. Each iteration of the algorithm consists of three phases of message passing. In the first phase, the source node broadcasts a *find message* down the partially constructed tree. When a node receives the message, it finds out the adjacent link which:

- leads to a destination out of the tree,
- does not violate the delay constraint, and
- minimize a selection function.

In the second phase, the selected links are sent to the source node, where the best link l which minimizes the selection function is chosen. In the third phase, an ADD message is sent to add l to the tree. This procedure continues until every destination is included in the tree. This algorithm requires intensive multiple message exchanges. Two important problem with KPP is that the communication overhead is very high and that every node needs to maintain a global state.

10.4 QoS extension to CBT

Hou *et al* concentrated on enhancements to join/leave message processing on routers, aiming at the guarantee of different QoS requirements, one at a time [20]. Each router keeps information regarding the QoS level of its downstream members. When it receives a join message, it checks if the QoS level required for the new member can be provided, and at the same time whether the QoS level of the other members can still be maintained (for join messages only). The join message has to travel to the core of the tree before the join can succeed, since each router keeps QoS information about downstream members only. The delay and loss requirement of members may be different, but the bandwidth requirement is assumed to be the same. The QoS requirements are not considered in the routing, they are just guaranteed during the tree construction and then it is maintained by restricting the success of join messages.

11 Resource Reservation

A network can provide QoS guarantees to connections, only if it has reserved appropriate resources along the identified routes. Merely finding a route that has sufficient resources, may not automatically solve the problem of providing QoS guarantees. If no reservation scheme is undertaken, then the traffic being dynamic, can increase and congest the system that in turn will hurt the QoS guarantees badly. So, definite allocation of resources helps in safeguarding the QoS guarantees. In this section we discuss a popular and widely used protocol named Resource reSerVation Protocol (RSVP) [16, 22].

We can sometimes reduce the bandwidth and buffer space reserved for a multicast group at the routers and links traversed by more than one multicast tree using a resource reSerVation protocol that considers the aggregate requirement of the multicast group as a whole. In deed, this is the RSVP protocol.

11.1 Resource Reservation Protocol (RSVP)

RSVP is solely a resource reservation protocol. It assumes that the multicast tree has been set up by the routers of the network. Admission of a new multicast connection is handled by the admission control module in a lower layer.

RSVP was designed in 1993 at Xerox Palo Alto Research Center. The idea was to design a resource reservation protocol that can scale to multicast communication and can incorporate heterogeneous receivers.

RSVP establishes and maintains a *sink tree* for each destination and uses it to send control messages from each destination. A sink tree is a spanning tree that is rooted at the destination and connects all the sources in a multicast group. It is obtained by tracing in the reverse direction from the destination to every source along the paths in a multicast tree.

Protocol Operation: Each router used by a multicast group maintains two types of information: **path state** *reservation state.* The path state of each source is concerned about the path used by the source. RSVP uses this information to maintain the sink trees of the destination.

RSVP is a receiver-oriented protocol. That is, the receiver of a data flow initiates and maintains the resource reservation used for that flow. Note that each router receives a reservation message from each of the downstream links in the multicast tree and sends only one reservation message into its upstream link. The reservation state of each destination is concerned with resources set aside for the destination to support the required QoS. The name of the destination (reserver) and the amount reserved are among the information provided by the reservation table.

A path message is a control message sent by a source and forwarded by the routers in a multicast tree. Each source sends its first path message before it commences transmission and sends subsequent path messages periodically. Among the information carried in each path message of a source is the flow specification of its message stream.

A router updates the path state upon receiving the path message from a source.

Each destination sends reservation request message (resv message for short) which are forwarded along the sink tree of the destination. Among the information contained in each reservation message is the flow specification giving the QoS requirement of the sender.

It is clear that the amount of bandwidth that a router reserves should not exceed the link's capacity. Towards this, whenever the router receives a new reservation message, it determines if its downstream links on the multicast tree can provide the required reservation. If the admission test fails, the router rejects the reservation and returns an error message to the appropriate receiver(s). RSVP does not define any specific admission test, but it assumes that the router performs such a test and that the RSVP can interact with the test.

The destination sends its first reservation message when it receives a path message from a source whose message stream it wishes to receive. Afterwards it sends reservation messages periodically to maintain reservation. The maintenance of reservation through periodic messages is what sets RSVP apart from most of the other resource reservation protocols. This is the so called *soft state* and is the preferred design choice for applications that are are dynamic in nature. However, the need for each host to periodically transmit messages for the purpose of refreshing its states maintained by routers leads to a higher protocol overhead. This overhead is somewhat lessened by the fact that control messages require lower overhead than data messages. Merging of control messages from all hosts and forwarding whenever there is no new state information, the protocol overhead can further be reduced.

A filter is a list of names of sources whose message streams can use the resources reserved for the destination. A destination that wants a filter includes the filter in its reservation message. When no filter is applied, all sources in the multicast group may use its resources.

12 Traffic Shaping and Policing

As already discussed, some mechanism to regulate the flow of packets in the network is needed, so that the greedy or malicious real-time or non real-time sources do not negatively affect the QoS guarantees given to other connections. Shaping and policing are the mechanisms used to ensure this. Table 6.1 summarizes the key differences between shaping and policing. The choice between which one to use depends on the type of application and its specific QoS requirements.

	Shaping	**Policing**
Objective	Buffers and queues the excess packets above the committed rates.	Drops the excess packets over the committed rates. Does not buffer.
Handling Bursts	Uses a leaky bucket to delay traffic, achieving a smoothing effect.	Propagates bursts. Does no smoothing.
Advantage	Avoids retransmission due to dropped packets.	Avoids delay due to queuing.
Disadvantages	Can introduce delays due to queuing.	Can reduce throughput of affected streams.

Table. 6.1: Traffic Shaping versus Policing

Several traffic shaping and policing techniques are available. In the following, we discuss an important traffic shaping and policing technique.

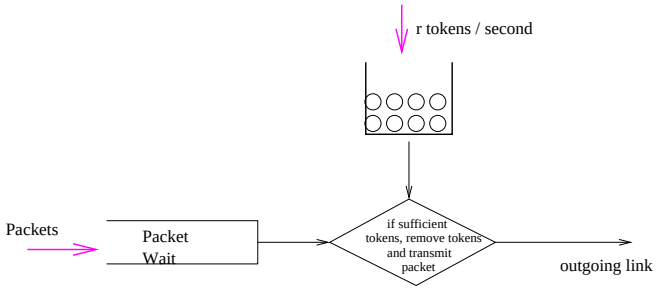

Figure 17: Token Bucket With Leaky Rate Control

12.1 Token Bucket with Leaky Rate Control

A popular technique used in shaping and policing is token bucket with leaky rate control [17]. As shown in Fig. 17, a bucket can hold up to certain prespecified number of tokens. Tokens are added to this bucket as follows. New tokens that might be added to the bucket, are always generated at a rate of r tokens per second. If the bucket contains less then b tokens when a token is generated, the newly generated token is added to the bucket; otherwise the newly generated token is ignored, and the token bucket remains full with b tokens.

In leaky rate control, a waiting packet is taken for transmission only if there is at least one token in the bucket. If enough tokens are not available in the bucket, then the packet either waits until the bucket has enough tokens in case of a shaper. In case of a policer, the packet is either discarded or is marked down. The token is removed, if transmission of the packet takes place. Because there can be at most b tokens in the bucket, the maximum burst size of a leaky-bucket-policed flow is b packets. Furthermore, because the token generation rate is r, the maximum number of packets that can enter the network of any interval of time of length t is rt + b. Thus, the token generation rate, r, serves to limit the long-term average rate at which the packet can enter the network. This technique can also be used to police the peak rate along with long-term average rate.

13 Scheduling Mechanisms

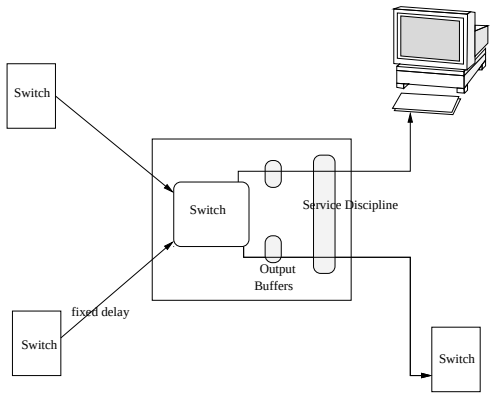

Figure 18: A Model of A Packet Switched Network

A simple model of an Internet is a collection of sending and receiving nodes and network switches. At every switch, there may be many incoming and many outgoing communication links as shown in Fig. 18. Depending

upon destination of a packet, it is queued on buffer queue of particular outgoing link. A scheduler selects packets to be transmitted from the buffer at the output link based on some scheduling policy. A service discipline schedules incoming packets for transmission. For real time connection, rate based service disciplines are used to protect the guarantees given to connections from network load fluctuations. This guarantees a minimum service rate to individual connections regardless of the traffic generated by other connections.

The Buffer requirements at a switch depends on maximum packet arrival rate, maximum packet size and the expected sojourn time of packets. On the other hand, the buffer requirements at the receiving end is equal to the product of delay jitter, and the maximum packet arrival rate and packet size.

The service discipline at any time decides which output buffers would be taken up for transmission next. The switch/router does the switching of the data packets from its input port to the desired output port, so that it reaches the destination address. Every switch/router can be modeled as a queue, where packets from different flows compete for the switching processing time and output link. There may be many incoming links to a switch, there may be a number of virtual connections for each link, and there may be a number of packets in each connection to be served. These packets may compete for the same output link. The manner in which the queued packets are selected for the transmission on the link is known as link or traffic scheduling discipline. The scheduling discipline at each switch selects which packet to transmit next by discriminating packets based on their QoS requirements.

13.1 Traffic Distortion

Traffic gets burstier and burstier after it passes through each switch, and it may no longer satisfy its source characterization. This is because, the queuing at each switch has a bunching effect on the packets of a session.

For example, if a connection satisfies (σ, ρ) at the source, then its worst case traffic characterization just before the ith switch becomes $(\sigma + \Delta\sigma, \rho)$, where $\Delta\sigma = \sum_{j=1}^{i-1} \rho.{d_j}^{max}$ and ${d_j}^{max}$ is the local delay bound for the connection at the jth switch.

Controlling distortion: The role of the traffic distortion handling is then to modify the input traffic either by reconstructing it to its original source characteristics or by deriving the new traffic characterization at the entrance to the switch assuming the worst case multiplexing. Another way to handle the traffic pattern distortion is to control it at every switch by reconstructing the traffic of a connection to its source characterization. All non-work-conserving disciplines proposed in the literature use this technique. They separate the server into two components

1. A rate controller:
   It controls the distortion on each connection using regulators and allocates bandwidth to them.

2. A scheduler:
   It order transmission of packets on different connections and provide over connection delay bounds.

A service discipline helps isolate well-behaved guaranteed traffic sources from ill-behaved sources, network load fluctuation and best-effort traffic. The ill-behaving user and malfunctioning equipment may send packets to the switch at a higher rate then declared. The network load fluctuation may cause higher instantaneous arrival rate even from a source, which has satisfied the traffic constraint at the entrance of the network. In the case of best effort traffic, since the traffic is not constrained they can very well cause congestion and disrupt the service to the well-behaved guaranteed traffic sources.

13.2 Traffic Scheduling Disciplines

For supporting best effort service, conventional service disciplines such as FCFS or round robin services are sufficient. However, per connection performance guarantees need more elaborate service disciplines. A service discipline can be

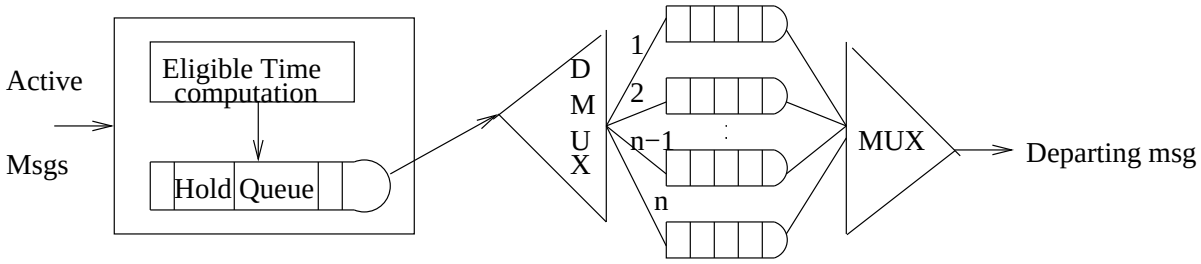

Figure 19: Processing in a Non-Work-Conserving Service Discipline

classified as either work-conserving or non-work-conserving. In a work-conserving discipline, a server is never idle when there is a packet to send. In a non-work-conserving discipline, each packet is assigned an eligibility time, a time after which a packet becomes eligible to be serviced. A server is allowed to send only the packets which are eligible and it is never idle as long as there are eligible packets to send. Among the proposed service disciplines,virtual clock,WFQ, SCFQ,Delay-EDD are work-conserving, where as Jitter-EDD, Stop-and-Go, HRR, RCSP are non-work-conserving discipline.

To provide end-to-end QoS guarantees to individual connections, real time service disciplines need to supply solutions for: per connection bandwidth and delay allocation in a single switch.
It provides local performance guarantees to a connection at a switch. There are essentially three approaches to bandwidth and delay allocation to different connections in a single switch:

1. **Multilevel Framing Strategy:** It is used both in Stop-and-Go and HRR service disciplines. Let us consider one level framing first. In one level framing, at each node, local time axis is divided into fixed intervals T, called a frame. Although convenient, frames of different nodes need not be synchronized. Bandwidth is allocated to each connection as a certain fraction of the frame time, which bounds the average rate of the traffic of a connection, where the averaging interval is T. Disadvantage of one level framing is the trade-off between the granularity of bandwidth allocation and the delay bound. These factors are inversely proportional.

   To overcome this multiple frame size is introduced. But in these scheme also there are so many problems as lower utilizations of bandwidth. Another disadvantage of framing technique is its inefficiency in handling the connection with bursty traffic. But framing strategy is easy to implement and fast enough to be used at high speed transmission rates.

2. **Sorted Priority Queue:** It is used to to implement dynamic priority schedulers such as EDD used in delay-EDD and jitter-EDD and also used in virtual clock, WFQ, SCFQ. It does not have the any of the shortcomings of framing strategies. However, it suffers from a high degree of complexity. Insertion into a sorted priority queue is an O(log N) problem, which can be unacceptably slow for high speed network.

3. **Multilevel FCFS Queue:** It is used to implement static priority schedulers. Levels of the FCFS queues correspond to priority levels. Each connection is assigned a priority and all packets from that connection are inserted into the FCFS queue of that priority level. Multiple connections can be assigned to the same priority level. Packets are scheduled in FCFS order from the highest-priority non-empty queue for transmission. These scheme is used in the RCSP discipline.
   The only restriction in using multilevel FCFS queue for static priority schedulers is that the number of different delay bound values that can be provided to the connections are bounded by the number of priority levels of the system.

There are essentially two types of traffic scheduling disciplines:

- **Work Conserving Discipline:** In work conserving scheduling schemes, the server is never idle if there is data to be sent.

- **Non-Work Conserving Discipline:** In a non-work conserving discipline, each packet is assigned, either explicitly, or implicitly, an eligibility time. The server idles (not transmitting any data) when no data is eligible.

Whether a service discipline is work conserving or non-work conserving affects buffer requirements, delay and delay jitter. Work conserving disciplines need less buffer and cause shorter delay but cannot bound delay jitter tightly. The opposite is true for non-work conserving disciplines: they need a larger buffer, cause longer delay, but can bound delay jitter tightly. The following are a few criteria to judge the suitability of a particular service discipline.

Whether it is required to provide statistical or deterministic guarantees. The properties of a flow that are required to be guaranteed by the scheme. The four common flow properties that are often guaranteed are upper bound on loss rate, the worst-case bandwidth, the upper bounds on queuing delay, and delay jitter. Some schemes only guarantee one or some of these properties. A comprehensive analysis of various traffic disciplines (both work conserving and non work conserving) and their relative comparison can be found in [25].

In the following, we first discuss a few work conserving disciplines, and subsequently we discuss a few non-work-conserving service disciplines.

13.3 Work Conserving Discipline

In this subsection, we discuss a few important work conserving service disciplines.

Weighted Fair queuing (WFQ): WFQ [7] has evolved from Fluid Fair Queuing (FFQ), also known as Packet Generalized Processor Sharing (PGPS). In FFQ there is a separate FIFO queue for each connection sharing the same link. During any time interval when there are exactly N non-empty queues, the server serves the N packets at the head of the queues simultaneously, each at the rate of 1/Nth of the link speed. FFQ allows different connections to have different service shares. A FFQ is characterized by N positive real numbers $\phi_1, \phi_2 ... \phi_N$, each corresponding to one queue. At any time the service rate for a non-empty queue i is exactly $(\phi_i / \sum_{j \in B(\tau)}) * C$ where B(τ) is the set of non-empty queues and C is the link speed. Therefore, FFQ serves packets in non-empty queues according to their service shares. FFQ is impractical as it assumes that the server can serve all connections with non-empty queues simultaneously and that the traffic is infinitely divisible. In a more realistic packet system, only one connection can receive service at a time and an entire packet must be served before another packet can be served.

13.4 Non-Work Conserving Disciplines

Non-work conserving disciplines were not given due importance earlier because of two main reasons.

- Previously the major performance indices were the average delay of all the packets and the average throughput of the server. With a non-work conserving discipline, a packet may be held in the server even when the server is idle. This may increase the average delay of the packets and decrease the average throughput of the server.

- Queuing analysis so far assumed only a single server environment. The potential advantages of non-work conserving disciplines in networked environment were never realized In guaranteed performance service, the more important performance index is the end-to-end delay bound rather than the average delay. In addition, delay needs to be bounded in a networking environment rather than just in a single node. Therefore, the above reasons for not using non-work conserving disciplines do not hold any more.

In the following, we discuss a few important non-work-conserving service disciplines.

Jitter-Earliest Due Date: The Jitter-EDD [24] provides delay-jitter bounds. After a packet has been served at each server, a field in its header is stamped with the difference between its deadline and the actual finishing time. A regulator at the entrance of the next server holds the packet for this period before it is made eligible to be scheduled.

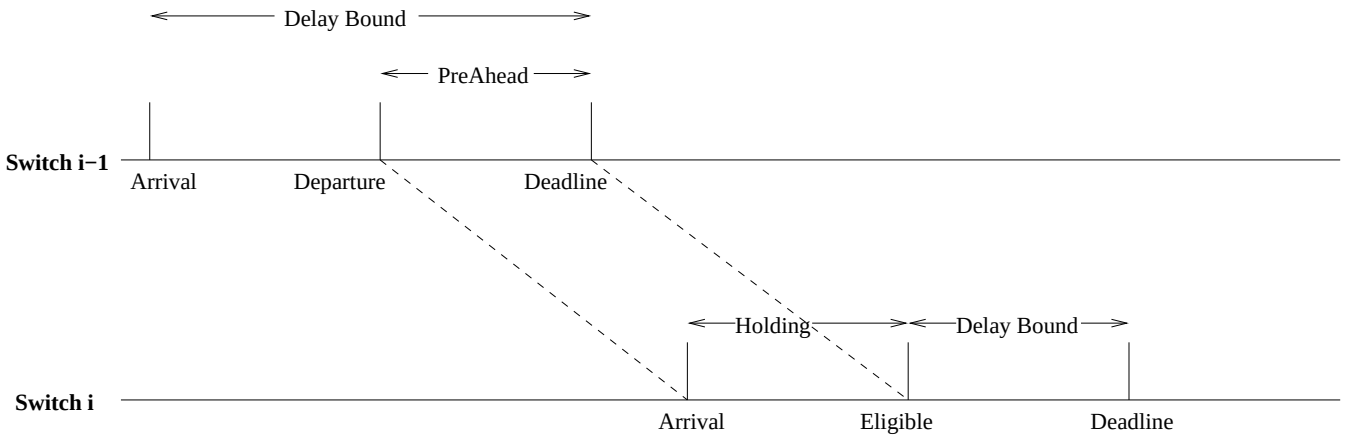

Figure 20: Packet Service in Jitter-EDD

Jitter- EDD is illustrated in Fig. 20, which shows the progress of a packet through two adjacent servers. In the first server, the packet got served PreAhead seconds before its deadline. So, in the next server, it is made eligible to be sent only after PreAhead seconds. Since there is a constant delay between the eligibility times of the packet at two adjacent servers, the packet stream can be provided a delay jitter bound. Assuming there is no regulator at the destination host, the end-to-end delay jitter bound is the same as the local delay bound at the last server.

Stop-and-Go: Stop-and-Go discipline [5] is based on multilevel framing strategy. First we describe this discipline using one level framing, and then we extend it to multi-level framing. In a framing strategy, the time axis is divided into frames, which are periods of some constant length T. Stop-and-Go defines departing and arriving frames for each link. At each switch, the arriving frame of each incoming link is mapped to the departing frame of the output link by introducing a constant delay θ, where $0 \leq \theta < T$. According to the Stop-and-Go discipline, the transmission of a packet that has arrived on any link l during a frame f should always be postponed until the beginning of the next frame, thus the server remains idle until then even if there are packets queued for transmission.

The framing strategy introduces the problem of coupling between delay bound and bandwidth allocation granularity. The delay of any packet at a single switch is bounded by two frame times. To reduce the delay, a smaller T is desired. However, since T is also used to specify traffic it is tied to bandwidth allocation granularity. Assuming a fixed packet size P, the minimum granularity of bandwidth allocation is P/T. To have more flexibility in allocating bandwidth, or smaller bandwidth allocation granularity, a larger T is preferred. It is clear that low delay bound and fine granularity of bandwidth allocation cannot be achieved simultaneously in a framing strategy like Stop-and-Go.

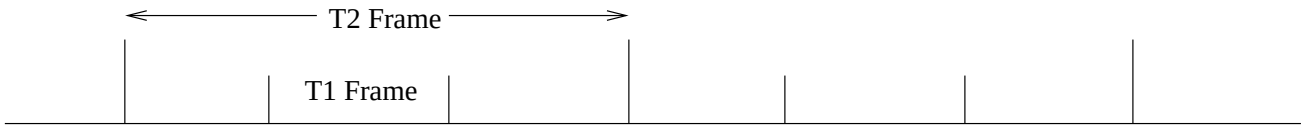

Figure 21: Two Level Framing with $T_2 = 3 \times T_1$

To get around this coupling problem multi-level framing is used. In this, the time axis is divided into a hierarchical framing structure shown in Fig. 21. For a n level framing with frame sizes T1,.., Tn and Tm+1= Km.Tm for m=1, ., n-1, packets on a level p connection need to observe the Stop-and-Go rule with frame size Tp, i.e., level p packets which arrived at an output link during a Tp frame will not become eligible for transmission until the start of next Tp frame.

Also, for two packets with different frame sizes, the packet with a smaller frame size has a non-preemptive priority over the packet with a larger frame size. With multi-frame Stop-and-Go, it is possible to provide low delay bounds to some channels by putting them in frames with a smaller frame time, and to allocate bandwidth with fine granularity to other channels by putting them in levels with a larger frame time. However, the coupling between delay and bandwidth allocation granularity still exists within each frame. A scheme is proposed to add a separate shaping mechanism at the network entry point for networks with framing based disciplines. With traffic shaping at the entrance to the network, it is possible to multiplex several connections on a single slot of a frame, therefore avoid the problem of coupling between frame size and bandwidth allocation granularity.

Rate-Controlled Static-Priority: The previously discussed service-disciplines were either based on sorted priority queue mechanism (e.g. WFQ) or framing strategy (e.g. Stop-and-Go). Sorted priority service-disciplines are complex and difficult to implement and framing strategy suffer from the dependencies between queuing delay and granularity of bandwidth. To get around these problems, zhang and Ferrari [26], proposed Rate-Controlled Static-Priority wherein the function of rate control from delay control has been decoupled in the design of server.

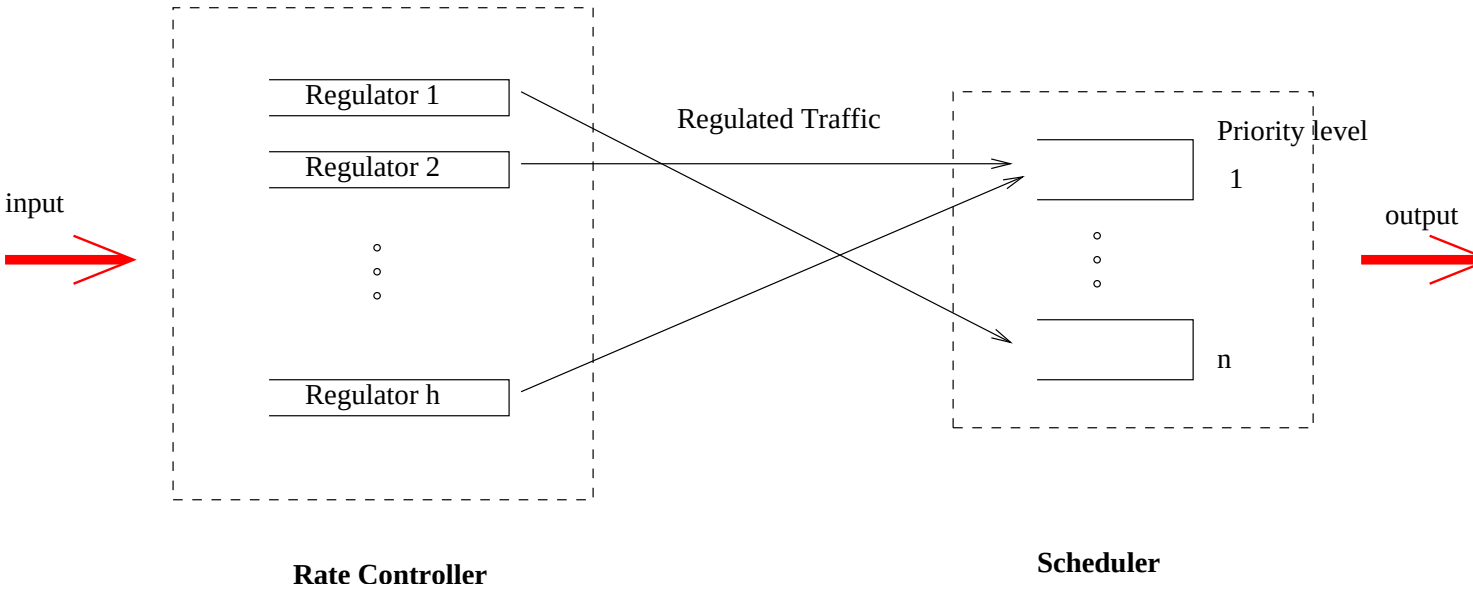

Figure 22: Rate Controlled Static Priority

As shown in **??**, a RCSP server has two components: a rate-controller and a static priority scheduler. Conceptually, a rate controller consists of a set of regulators corresponding to each of the connections traversing the server; each regulator is responsible for shaping the input traffic of the corresponding connection into the desired traffic pattern. Upon arrival of each packet, an eligibility time is calculated and assigned to the packet by the regulator. The packet is held in the regulator till its eligibility time before being handed to the scheduler for transmission. The static priority schedulers provide guarantees on delay and delay-jitter bound. The scheduler is implemented by a multilevel FCFS queue, which serves the packets according to its priority i.e. first packet in the highest priority non-empty queue. There are two types of regulators:

- Rate-jitter (RJ) regulator, which controls rate-jitter by partially reconstructing the traffic.

- Delay-jitter (DJ) regulator, which controls delay-jitter by fully reconstructing the traffic. We assume that the RCSP satisfy the (Xmin, Xave, I) characterization. The (Xmin, Xave, I) RJ regulator ensures that the output of the regulator satisfy the (Xmin, Xave, I) traffic model, while the DJ regulator ensures that the output traffic of the regulator is exactly the same as the output traffic of the regulator at the previous server. Thus, if the traffic satisfies the (Xmin, Xave, I) characterization at network entrance, both types of regulators will ensure that the output of the regulator, which is the input to the scheduler, will satisfy the same traffic characterization.

The scheduler in a server RCSP uses a non-preemptive static priority policy; it always selects the packet at the head of highest priority queue that is not empty. The SP scheduler has a number of priority levels with each priority level corresponding to a delay bound. Each connection is assigned to a priority level during connection establishment time. Multiple connections can be assigned to the same priority level, and all packets on the connections associated with a priority level are appended to the end of the queue for that priority level.

The scheduler in a server RCSP uses a non-preemptive static priority policy; it always selects the packet at the head of highest priority queue that is not empty. The SP scheduler has a number of priority levels with each priority level corresponding to a delay bound. Each connection is assigned to a priority level during connection establishment time. Multiple connections can be assigned to the same priority level, and all packets on the connections associated with a priority level are appended to the end of the queue for that priority level.

RCSP is a better scheduling algorithm since it is easy to implement and is devoid of problems faced by other schedulers.

14 QoS Models

Internet Engineering Task Force (IETF) [1] proposed several models for providing QoS assurances to Internet-based applications. These models are called *QoS models* or *QoS architectures*. The models require that the connections requesting QoS must follow certain procedures to avail the QoS. There are three major QoS models that have been proposed. These are Integrated Services (IntServ), Differentiated Services (DiffServ) and lastly Multi-Protocol Label Switching (MPLS). DiffServ is an improvement on IntServ while MPLS is a packet-forwarding scheme having the advantages of both IntServ and DiffServ.

14.1 Integrated Services

Integrated Services (IntServ) Model **??** includes two types of services targeted toward real-time traffic: guaranteed and predictive service. It integrates these services with controlled link sharing, and it is designed to work well with multicast as well as unicast traffic. Intserv requiers explicit resource reservation to be done, which in turn requires flow-specific state in the routers.

The Internet has to be used as a common platform for both real-time and non-real-time traffic and thus precludes the need to design a parallel infrastructure for real-time traffic. Use of existing Internet-layer protocol (e.g., IP) for real-time data is proposed so that economy of communication and interoperability between IntServ and non-IntServ networks can be realized. There should be a single service model but packet scheduling and admission control mechanism can vary.

[1] The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet.

IntServ implementation has four important components: the signaling protocol (RSVP), the admission control routine, the classifier and the packet scheduler. After the resource reservation and call admission processes are complete, when a router receives a packet, the classifier performs a classification and inserts the packet in a specific queue based on the classification result. The packet scheduler will then schedule the packet accordingly to meet its QoS requirements.

Fig. 11 shows, how these components might fit into an IP router that has been extended to provide integrated services. The router has two broad functional divisions: the forwarding path below the double horizontal line, and the background code above the line. The forwarding path of the router is executed for every packet and must therefore be highly optimized. Indeed, in most commercial routers, it is implemented in hardware.

The forwarding path is divided into three sections: input driver, Internet forwarder, and output driver. The Internet forwarder interprets the IP header. For each packet, an Internet forwarder executes a suite-dependent classifier and then passes the packet and its class to the appropriate output driver. A classifier must be both general and efficient. For efficiency, a common mechanism should be used for both resource classification and route lookup. The output driver implements the packet scheduler.

The background code is simply loaded into router memory and executed by a general-purpose CPU. These background routines create data structures that control the forwarding path. The routing agent implements a particular routing protocol and builds a routing database. The reservation setup agent implements the protocol used to set up resource reservations. If admission control gives the "OK" for a new request, appropriate changes are made to the classifier and packet scheduler database to implement the desired QoS. Finally, every router supports an agent for network management. This agent must be able to modify the classifier and packet scheduler databases to set up controlled link sharing and to set admission control policies. The implementation framework for a host is generally similar to that for a router, with the addition of applications. Rather than being forwarded, host data originates and terminates in an application. An application needing a real-time QoS for a flow must somehow invoke a local reservation setup agent.

As stated earlier IntServ has been designed to provide two types of services:

- Guaranteed Service for applications requiring fixed delay bound

- Predictive Service for applications requiring probabilistic delay bound. The implementations of the guaranteed service and the predictive service are defined in the guaranteed service RFC [13] and the controlled load service RFC [23], respectively.

The IntServ architecture represents a fundamental change to the Internet architecture but it is beset with many problems, which makes it very difficult to deploy. Some of the important problems are the following:

- The amount of state information increases proportionally with the number of flows. This places a huge storage and processing overhead on the routers. Therefore, this architecture does not scale well in the Internet core;

- The requirement on routers is high. All routers must implement RSVP, admission control, MF classification and packet scheduling;

- Ubiquitous deployment is required for guaranteed service. Incremental deployment of Controlled-Load Service is possible by deploying Controlled-Load Service and RSVP functionality at the bottleneck nodes of a domain and tunneling the RSVP messages over other part of the domain.

14.2 Differentiated Services

To overcome the limitations of IntServ, IETF proposed Differentiated Services (DiffServ) [15], which is simpler and more scalable. DiffServ redefines the IPv4 Type of Service (TOS) field or IPv6 Traffic Class Byte as differentiated service (DS) field. The first six bits of the field are called as DS Code point, which when marked differently indicates the behavior (known as per-hop behavior or PHB) each router is required to apply to individual packets.

The information required by the buffer management and scheduling mechanisms is carried within the packet. Therefore, differentiated services do not require special signaling protocols to control the mechanisms that are used to select different treatment for the individual packets. Consequently, the amount of state information, which is required to be maintained per node, is proportional to the number of service classes and not proportional to the number of application flows, which is major improvement over IntServ.

A customer must have service level agreement (SLA) with its ISP in order to receive DiffServ. An SLA specifies the class of service supported and the amount of traffic allowed in each class. An SLA can be static or dynamic. Static SLAs are negotiated on a regular, e.g. monthly and yearly, basis. Customers with Dynamic SLAs must use a signaling protocol, e.g. RSVP, to request for services on demand.

14.3 Functional Elements of DiffServ Architecture

The Differentiated Services architecture [46] is composed of a number of functional elements, namely packet classifiers, traffic conditioners and per-hop forwarding behaviors (PHB). These are described below:

Packet Classifiers: Packet classification is normally done at the edge of the DS domain by packet classifiers, which select packets based on the content of packet headers. Two types of classifiers are currently defined: the Behavior Aggregate (BA) classifier, which selects packets based on the DS CodePoint only, and the Multi-Field (MF) classifier, which performs the selection based on the combination of one or more header fields, such as source address, destination address, type-of service byte, protocols ID, source port number and destination port number.
Traffic Conditioners: Traffic conditioners form the most vital part of a differentiated services network. Their goal is to apply conditioning functions on the previously classified packets according to a predefined Traffic Class Specification in SLA. A traffic conditioner consists of one or more of the following components:

Meter: A device which measures the temporal properties of a traffic stream selected by a classifier.
Marker: A device that sets the DS CodePoint in a packet based on well-defined rules.
Shaper: A device that delays packets within a traffic stream to cause the stream to conform to some defined traffic profile (See Sec- 3.6).
Dropper/Policer: A device that discards packets based on specified rules (See Sec. 3.6).

Per-Hop Behavior (PHB): A PHB is a description of the externally observable forwarding behavior of a differentiated services node, applied to a collection of packets with the same DS CodePoint that are crossing a link in a particular direction (called differentiated services behavior aggregate). Each service class is associated with a PHB. PHBs are defined in terms of behavior characteristics relevant to service provisioning policies, and not in terms of particular implementations. PHBs may also be specified in terms of their resource priority relative to other PHBs, or in terms of their relative observable traffic characteristics. Currently there are three proposed PHBs that are briefly described below.

- The Default (DE) PHB is the common, best-effort forwarding available in today's Internet. IP packets marked for this service are sent into a network without adhering to any particular rules and the network will deliver as many of these packets as possible and as soon as possible but without any guarantees.
- Assured Forwarding (AF) PHB is a means for a provider differentiated services domain to offer different levels of forwarding assurances for IP packets received from a customer differentiated services domain.

- The Expedited Forwarding (EF) PHB is a high priority behavior and is defined as a forwarding treatment for a particular differentiated services aggregate where the departure rate of the aggregate's packets from any DS-compliant node must equal or exceed a configurable rate. The EF traffic should be allocated this rate independently of the intensity of any other traffic attempting to transit the node.

SUMMARY

- Real-time communication requirements of applications are expressed in terms of certain quality of service (QoS) parameters. A real-time protocol guarantees the QoS requirement of an application, once it accepts a connection request.
- Real-time communication protocols allow both non-real-time and real-time messages to be handled in the same framework.
- In the Internet environment, only soft real-time communication is supported.

EXERCISES

1. State whether you consider the following statements to be **TRUE** or **FALSE**. Justify your answer in each case.
   (a) Streaming compressed video transmission is an example of real-time VBR (variable bit rate) traffic.
   (b) The virtual time protocol is an example of a bounded access type of scheduling in a multiple access network.
   (c) In a bounded access token ring protocol using either the proportional or the local transmission time allocation schemes, the sums of the synchronous bandwidths ΣH_i equals 1.
   (d) Under normal real-time traffic conditions, the maximum delay suffered by the highest priority packets at a switch when a multi-level FIFO queue-based service discipline is deployed would be lower compared to the case when a framing-based service discipline is deployed.
   (e) Real-time computer communication is essentially communication at high data rates.
   (f) Under normal traffic conditions, a priority queue-based service discipline would incur less processing overheads at a switch compared to a multi-level FIFO queue-based one.
   (g) In the virtual-time protocol, suppose that the highest priority packet that a node needs to transmit at some instant of time is m. Then for efficient working of the protocol, during priority arbitration interval the node should wait for $2 * \delta * m$ time units before starting transmission. (δ is the propagation delay in the network.)
   (h) The countdown protocol should work successfully irrespective of whether the priority arbitration transmissions start with either the msb or lsb end of the priority value, as long as all nodes agree on the convention.
   (i) Even when a global priority-based message transmission is supported by a real-time communication network, RMA (rate monotonic algorithm) scheduling of real-time messages is ineffective.
   (j) For transmitting VBR real-time traffic among tasks over a LAN, a calendar-based reservation protocol would yield higher GP(U) than either a global priority-based, or a bounded access type of protocol. (GP(U) is the guarantee probability at utilization U)
   (k) In multiple bounding average rate characterization of bursty traffic, the larger is the averaging interval the lower is the rate with which the source is bound.

(l) The (X_{min}, S_{max}) model for real-time traffic can be satisfactorily used for resource reservation to provide QoS guarantee for compressed audio and video signals.
(In the (X_{min}, S_{max}) model the minimum interarrival time between packets is bounded below by Xmin and the largest packet size is bounded above by Smax)

(m) The maximum delay suffered by packets under a multilevel FCFS queue based service discipline would be lower than that in a framing-based service discipline under similar traffic conditions.

(n) In any practical real-time communication protocol, we can expect that the guarantee probability (GP) to be close to 0 for utilization lower than the Average Breakdown Utilization (ABU) and GP would approach 1 as the utilization increases beyond ABU.

(o) If a certain token-passing network has a channel capacity of 100Mbps, then the IEEE 802.5 (priority-based protocol) would provide higher guarantee probability compared to IEEE 802.4 (timed token protocol) at very low channel utilizations.

(p) The countdown real-time global priority communication protocol should work successfully irrespective of whether the priority arbitration transmissions start with either the msb or lsb end of the priority value, as long as all nodes agree on the convention.

(q) Even when a global priority-based message transmission is supported by a real-time communication network, RMA (rate monotonic algorithm) scheduling of real-time messages is not very effective.

2. Dynamic changes to the reserved resources is an important distinguishing characteristic of the RSVP protocol. Using four or five sentences explain how RSVP protocol is capable of dynamically changing the reservation status of a connection during the life time of the connection.

3. In an IEEE 802.4 network, if δ is the shortest deadline of all messages required to be transmitted over the network, identify the pros and cons of making TTRT equal to
(a) δ (b) $\frac{\delta}{10}$ (c) $\frac{\delta}{2}$ (d) $2 \times \delta$

4. Answer the following in the context of a Chemical manufacturing company that wishes to automate its process control application:

   (a) What problems might arise if an attempt is made to implement the chemical plant control software using the Ethernet LAN available in the factory?

   (b) How can a global priority protocol be supported in a LAN with collision-based access?

   (c) If RMA scheduling of of packets is to be supported, what is the maximum channel utilization that can be achieved?

   (d) What are the main obstacles to efficient implementation of RMA in this set up?

5. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit upto b_i Kbits over each period of duration P_i milliSeconds, where b_i and P_i are given in the table below. Assume that the propagation time is negligible compared to TTRT and that the network bandwidth is 1Mbps.

Node	b_i	P_i
n1	100Kb	10 msec
n2	200Kb	15msec
n3	500Kb	100msec

   (a) Choose suitable TTRT (target token rotation time).

   (b) What is the maximum time for which a real-time message may suffer inversion?

   (c) obtain suitable values of f_i (total number of bits that can be transmitted by the nodes n1, n2, and n3 over every cycle.) under proportional and local allocation schemes.

(d) Determine the worst case times by which n1 completes transmitting 100Kb of real-time message under each of the two bandwidth allocation schemes.

(e) Determine the worst case time by which n1 completes transmitting 100Kb of non-real-time message under each of the two bandwidth allocation schemes.

6. Consider a calendar-based reservation protocol to transmit real-time messages over a collision-based network:

   (a) Explain how transmission of asynchronous messages by nodes can be handled. (Asynchronous messages have probabilistic arrival times and do not have any specified time bounds).

   (b) Explain with proper reasoning the types of traffics for which a calendar-based protocol would perform satisfactorily and the types for which it will not.

7. The bandwidth of a priority-based token ring network (IEEE 802.5) is 100Mbps. Assume that the propagation time of the network is 10 milliSeconds. Determine the fraction of bandwidth wasted during every frame transmission, when the frame size is 2 KBps (Kilo Bytes per second) Under what conditions would the wasted bandwidth be zero?

8. Identify the factors which contribute to delay jitter in real-time communications in packet-switched networks. Assume that a certain real-time application receives data at the rate of 10Mbps. The QoS guarantee to the application permits a delay jitter of 20mSec. Compute the buffer requirement at the receiver.

9. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit upto b_i bits over each period of duration P_i milliSeconds, where b_i and P_i are given in the table below.

Node	b_i	P_i
N_1	1K	10,000
N_2	4K	50,000
N_3	16K	90,000
N_4	16K	90,000

   Choose suitable TTRT (target token rotation time) and obtain suitable values of f_i (total number of bits that can be transmitted by node N_i over every cycle.) Assume that the propagation time is negligible compared to TTRT and that the system bandwidth is 1Mbps.

10. If you are constrained to use an existing Ethernet LAN for a factory automation application, what are the different protocols that you can use to support communicating real-time tasks? Which of the alternatives that you have identified you think would work best? Give your reasonings.

11. Consider a 10Mbps token-ring network operating under the priority-based protocol (IEEE 802.5). For this network, the walk time is 2 mSec and the frame length is 1024 Bytes. Determine the fraction of wasted bandwidth due to the wait time required for the token to return to the transmitting node after transmission of a frame is complete. At the given bandwidth would a bounded access protocol perform better? Explain your answer.

12. Consider a 10Mbps token ring network. The walk time is 1 mSec. The frame size is 512 bytes. Determine the maximum time for which a message may undergo priority inversion under IEEE 802.4 and IEEE 802.5 protocols.

13. (a) Identify at least two factors which contribute to delay jitter in real-time communications and explain how they cause jitter.

    (b) What is the difference between execution time and response time of a task? In what circumstances can they be different?

14. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit upto b_i bits over each period of duration P_i milliseconds, where b_i and P_i are given in the table below.

Node	b_i in KBytes	P_i in milli Seconds
N_1	4	10
N_2	10	50
N_3	10	90
N_4	20	100

Choose suitable TTRT (target token rotation time) and obtain suitable values of f_i (total number of bits that can be transmitted by node N_i over every cycle.) Assume that the propagation time is 1 milliSec and that the bandwidth of the network is 10Mbps.

15. A (σ, ρ) traffic characterization of a certain packet switched real-time traffic is characterized by a peak traffic (σ) of 100 packets and average traffic rate (ρ) of 10 packets per second. The packet size is 512 bytes. Assuming that the packets undergo a maximum queuing delay of 50mSec at each of the switches, what would be the traffic characterization after the 10th switch on the route.

16. Consider a 10Mbps token-ring network operating under the priority-based protocol (IEEE 802.5). For this network, the walk time is 2 mSec and the frame length is 1024 Bytes. Determine the fraction of wasted bandwidth due to the wait time required for the token to return to the transmitting node after transmission of a frame is complete. At the given bandwidth would a bounded access protocol perform better? Explain your answer.

17. Consider a 10Mbps token ring network. The walk time is 1 mSec. The frame size is 512 bytes. Determine the maximum time for which a message may undergo priority inversion under IEEE 802.4 and IEEE 802.5 protocols.

18. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit upto b_i bits over each period of duration P_i milliseconds, where b_i and P_i are given in the table below.

Node	b_i in KBytes	P_i in milli Seconds
N_1	4	10
N_2	10	50
N_3	10	90
N_4	20	100

Choose suitable TTRT (target token rotation time) and obtain suitable values of f_i (total number of bits that can be transmitted by node N_i over every cycle.) Assume that the propagation time is 1 milliSec and that the bandwidth of the network is 10Mbps.

19. Would it be advisable to use an Ethernet LAN in a hard real-time application such as factory automation? Justify your answer. Evaluate the pros and cons of using an Ethernet-based protocol in such an application.

20. Suggest a scheme that can help handle aperiodic and sporadic messages in a reservation (calendar-based) protocol without making major changes to the protocol.

21. Identify the factors which contribute to delay jitter in real-time communications. Assume that a certain real-time application receives data at the rate of 10Mbps. The QoS guarantee to the application permits a delay jitter of 20mSec. Compute the buffer requirement at the receiver.

22. Explain why traffic gets distorted in a multisegment network and how traffic reshaping is achieved for providing QoS guarantee.

23. Compare the performance of IEEE 802.4 protocol with IEEE 802.5 protocol for real-time applications at high, medium, and low bandwidths.

24. Explain using four or five sentences, how global priority arbitration is achieved using virtual time protocol. At which ISO layer would such a protocol operate?

25. Suppose that we have three periodic messages m1, m2, and m3 to be transmitted from three stations S1, S2, and S3, respectively on an FDDI network. Let the TTRT (token rotation time) be 10 msec and the walk time be 1msec . The transmission time ($C_i$msec) and the period of the messages (T_i) are as follows: (C_1=9msec,T_1=110msec); (C_2=12msec,T_2=150msec)(C_3=15msec,T_3=160msec). Determine the synchronous bandwidth that needs to be allocated to each station for successful transmission of the messages.

26. Explain the important shortcomings of the IntServ architecture in supporting real-time communication, and how DiffServ overcomes it.

27. Compare the advantages and disadvantages of using a ring network and a collision based network for real-time communication.

28. What do you understand by QoS routing. Explain the different types of QoS routing algorithms.

29. Explain what are the different QoS constraints that are considered during QoS routing. Explain the features of these constraints based on which the constraints can be classified. What are the implications of these features on the routing algorithms?

30. What do you understand by QoS routing? Give some examples of additive, multiplicative, and concave constraints that are normally considered in QoS routing schemes. Explain how these features are considered in QoS routing protocols.

31. Explain the difference between traffic shaping and policing. Name a traffic shaping and policing protocol and briefly describe its operation.

32. What is a controller area network (CAN)? Explain a real-time communication protocol that can be used in a CAN.

References

[1] Ashok Agarwala Ardas Cilingiroglu, Sung Lee. Real-time communication. *University of Maryland Institute for Advanced Computer Studies. Dept. of Computer Science, Univ. of Maryland*, January 1997.

[2] Shigang Chen and Klara Nahrstedt. An overview of quality-of-service routing for the next generation high-speed networks: Pro blems and solutions. *IEEE Network: Special Issue on Transmission and Distribution of Digital Video*, Nov./Dec. 1998.

[3] D. Waitzman C. Partridge S. Deering. Distance vector multicast routing protocol. *RFC:1075, Network Working Group, Stanford University*, November 1988.

[4] A. Helmy D. Farinacci et. al. Protocol independent multicast-sparse mode (pim-sm): Protocol. *RFC: 2362, Category: Experimental*, June 1998.

[5] S. J. Golestani. A stop-and-go queuing framework for congestion management. *SIGCOMM'90 Symposium, Communications Architecture and Protocol, Philadelphia, PA*, pages 8–18, September 1990.

[6] I. Viniotis H. Salama, D. Reeves and T. Sheu. Comparison of multicast routing algorithms for high-speed networks. *IBM Technical Report, IBM-TR29.1930*, 1994.

[7] A. Demers S. Keshav and S. Shenker. Analysis and simulation of a fair queuing algorithm. *Proceedings of ACM SIGCOMM '89, Austin TX*, pages 1–12, September 1989.

[8] Kweon S.K. Shin K.G. and Zheng Q. Statistical real-time communication over ethernet for manufacturing automation systems. *Proceedings of IEEE Real-Time Technology and Applications Symposium*, pages 192–202, 1999.

[9] James F. Kurose and Keith W. Ross. *Computer Networking-A Top down Approach.* Pearson Education Asia, 2002.

[10] K.G. Kweon, S.K. Shin and Workman G. Achieving real-time communication over ethernet with adaptive traffic smoothing. *Proc. 6th IEEE Real-Time Technology and Applications Symposium (RTAS'2000), Washington D.C., USA*, pages 90–100, 31 May - 2 June 2000.

[11] N. Malcolm and W. Zhao. Hard real-time communication in multiple access networks. *Journal of Real-Time Systems*, Vol. 9:75–107, 1995.

[12] J. Moy. Multicast extensions to ospf. *RFC: 1584, Category: Standards Track, Network Working Group, Proteon Inc.*, March 1994.

[13] S. Shenker C. Partridge and R. Guerin. Specification of guaranteed quality of service. *RFC 2212*, Sept. 1997.

[14] V. P. Kompella J. C. Pasquale and G. C. Polyzos. Multicast routing for multimedia communication. *IEEE/ACM Transactions on Networking*, Vol. 1(No. 3):286–292, June 1993.

[15] et. al. S. Blake D. Black M. Carlson. An architecture for differentiated services. *RFC 2475*, Dec. 1998.

[16] L. Zhang S. Deering D. Estrin S. Shenker and D. Zappala. Rsvp: A new resource reservation protocol. *IEEE Network Magazine*, September 1993.

[17] Andrew S. Tanenbaum. *Computer Networks.* Prentice Hall of India, 1997 (Third Edition).

[18] C. Venkatramani and T. S. Chiueh. Supporting real-time traffic on ethernet. *Proc. 15th IEEE Real-Time Systems Symposium*, pages 282–286, 1994.

[19] Bin Wang and Chao-Ju Hou. A survey on multicast routing and its qos extension: problems, algorithms, and protocols. *IEEE Network Magazine*, Vol. 14(No. 1), February 2000.

[20] Hung-Ying Tyan Chao-Ju Hou Bin Wang and Yao-Min Chen. Qos extension to cbt. *IEEE/ACM Trans. on Networking*, July 2002.

[21] Z. Wang and J. Crowcroft. Qos routing for supporting resource reservation. *IEEE Journal of Special Areas of Communication*, Sept. 1996.

[22] Paul P. White. Rsvp and integrated services in the internet: A tutorial. *IEEE Communications*, May 1997.

[23] J. Wroclawski. Specification of the controlled-load network element service. *RFC 2211*, Sept. 1997.

[24] D. Verma H. Zhang and D. Ferrari. Guaranteeing delay jitter bounds in packet switching networks. *Proceedings of Tricomm'91, Chapel Hill, North Carolina*, pages 35–46, April 1991.

[25] H. Zhang. Service disciplines for guaranteed performance service in packet switching networks. *Proceedings of the IEEE*, Vol. 85(10), October 1995.

[26] H. Zhang and D. Ferrari. Rate-control static-priority queuing. *Proceedings of IEEE INFOCOM'93, San Francisco, CA*, pages 227–236, 1993.