RESERVATION APPROACH TO REAL-TIME COMMUNICATION SERVICES

K. Arvind, K. Ramamritham and J. A. Stankovic
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

COINS Technical Report 90-128

Reservation Approach to Real-Time Communication Services

K. Arvind (arvind@cs.umass.edu)
Krithi Ramamritham (krithi@cs.umass.edu)
John A. Stankovic (stankovic@cs.umass.edu)

Abstract

We examine an important shortcoming, viz., *pessimism*, of the various protocols that we have looked at so far for supporting real-time virtual circuit services. We identify the lack of global knowledge in these protocols as a cause for their pessimism. We then look at several reservation-based protocols to see whether any of these protocols can be used to overcome this shortcoming. The class of explicit reservation-based protocols make use of global knowledge in order to schedule messages on the channel. However these protocols have a major drawback, viz., they may lose scheduling synchronisation in the presence of noise. Since loss of scheduling synchronization could result in the violation of guarantees, these protocols require special fault-tolerance mechanisms for implementing guarantee-based services.

1 Introduction

We introduced the notion of a real-time virtual circuit at the medium access control layer, in the context of a local area network in ([2],[3]). We defined a real-time virtual circuit to be a logical channel with a bounded packet delivery time. We examined different approaches to implementing real-time virtual circuits. While protocols like the TDMA protocol and the token-passing protocols are collision free protocols, the CSMA window protocols that we presented make a deliberate use of collisions to arbitrate access to the shared channel. We argued that neither the collision-free TDMA class of protocols nor the collision-based CSMA class of protocols is inherently superior to the other. Both classes of protocols have some form of overhead associated with them (e.g., the token passing delay in token passing protocols, the delay awaiting one's turn to transmit in TDMA and the wasted time due to collisions in the window protocol) and consequently the difference in performance between the two classes of protocols in terms of the performance measure of interest, viz., *guarantee ratio*, is not substantial. This is because both classes of protocols always make use of the worst case channel access time (which is roughly equal to the number of real-time virtual circuits times the packet transmission time) in determining whether a packet that has just arrived can be guaranteed or not; i.e., they assume that all the other nodes will be constantly busy and hence a node will be able to transmit only one packet every cycle. Thus these are *pessimistic* protocols which can reject messages that may actually be able to make their deadlines. The pessimistic nature of these protocols arises because of two reasons:

1. The notion of *guarantee*: These protocols are expected to support the notion of guarantee, which means that they can accept a packet for transmission only if they are *absolutely* certain that the packet can be transmitted before its deadline.
2. *Local view*: These protocols lack global knowledge, i.e., each station knows only the state of its own transmission queues when it guarantees a message. Hence it is forced to assume that

***This work is part of the Spring Project at the University of Massachusetts funded in part by the Office of Naval Research under contract N00014-85-K-0398 and by the National Science Foundation under grant DCR-8500332 and CCR-8716858.**

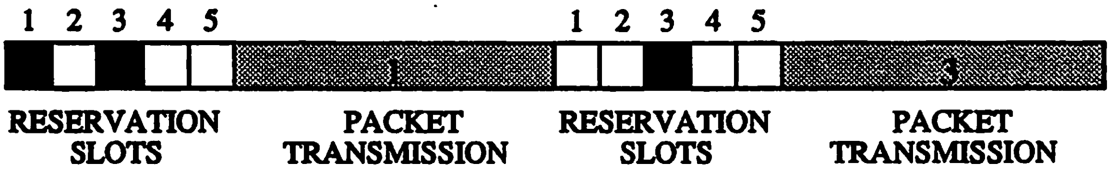


Figure 1: Bit Mapped Reservation Protocol

other stations will always have a packet to transmit. A protocol in which each station knows the state of the message queues in all the other stations can be expected to be able to provide better performance.

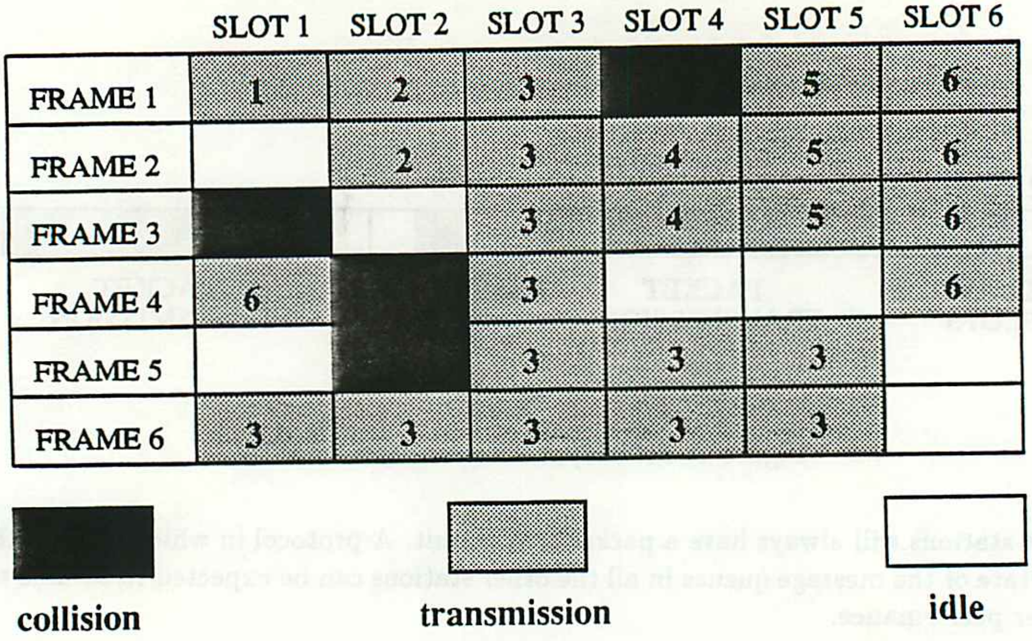
The notion of guarantee is a characterizing feature of real-time virtual circuits and hence has to be accepted. However, the second reason, viz., the local view of the protocols, appears to be merely a restriction that arises because of the nature of the protocols that we have been using. Is it possible to eliminate this cause for pessimism through the use of a suitable reservation protocol? We examine various reservation protocols that have been proposed in the literature to try to answer this question.

2 Reservation Protocols

Various reservation protocols have been proposed in the context of broadcast satellite-based data communication and integrated voice data local networks. These come in different flavors. We examine below three classes of reservation protocols that have been proposed in the literature below.

2.1 Single Packet Reservation

In one scheme proposed by Kleinrock and Scholl [10], each packet transmission is preceded by a set of reservation slots (Figure 1). The length of each reservation slot is equal to the end-to-end propagation delay, and there is one reservation slot for every station in the system. If a station has a packet to transmit, it makes a reservation by broadcasting a burst of noise during its reservation slot. This burst is a signal to all the stations with a higher address to refrain from transmitting in the packet transmission slot following the reservation slots. Thus this protocol selects the station with the lowest address that has a packet for transmission. Figure 1 depicts a system with 5 nodes. In the first transmission cycle shown, there are two nodes (1 and 3) that have a packet to transmit. Node 1 being the station with the smaller address gets to transmit its packet. In the following transmission cycle, node 3 is the only node with a pending packet and is granted transmission rights by the protocol. This protocol makes a very limited amount of local information, viz., the information as to which nodes have a packet to transmit, global. However this information is not sufficient for a node to determine whether a packet that has just arrived will be able to meet its deadline or not. Hence this protocol is *not suitable* for providing real-time virtual circuit services.



	SLOT 1	SLOT 2	SLOT 3	SLOT 4	SLOT 5	SLOT 6
FRAME 1	1	2	3	■	5	6
FRAME 2		2	3	4	5	6
FRAME 3	■		3	4	5	6
FRAME 4	6	■	3			6
FRAME 5		■	3	3	3	
FRAME 6	3	3	3	3	3	

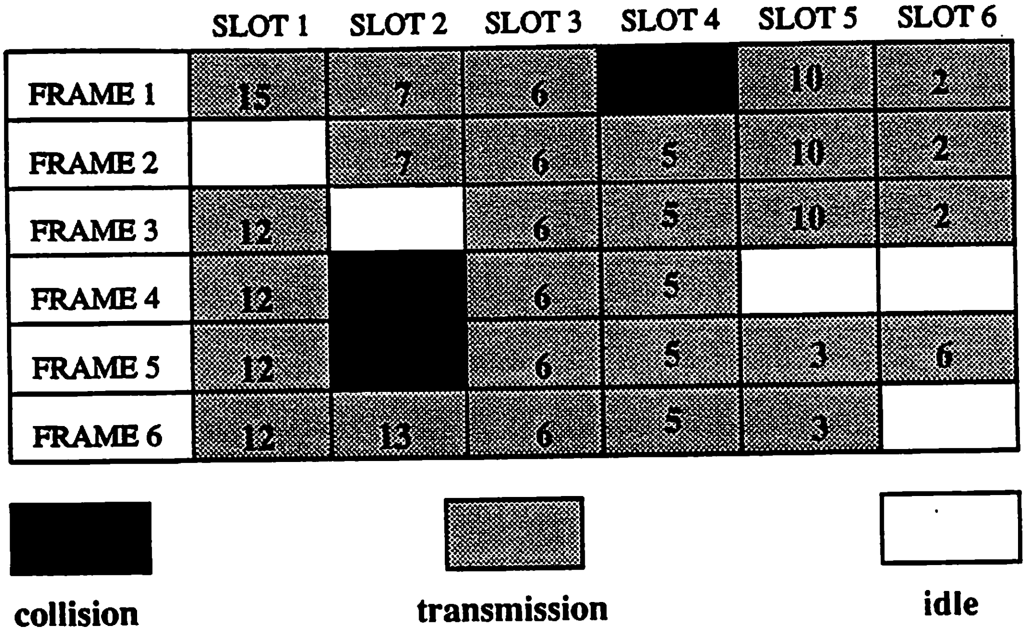
Figure 2: Binder's broadcast satellite protocol

2.2 Multiple Packet Reservations

The protocols that we look at in this section make reservations for multiple packets, thereby avoiding contention for the packets that follow the first one in a sequence of packets.

Binder [6] has proposed a protocol that normally operates like a TDMA protocol, where each slot is associated with a particular station (the "owner" of the slot). However, when a station has nothing to transmit, its slot is made available to other stations. Any station that requires the slot may capture it through a process of contention. This station then holds on to the slot as long as it has a packet to transmit (i.e., it has reserved this slot for its own use in the subsequent frames). If the owner of the captured slot needs the slot back in the mean time, it induces a collision by transmitting during the slot. The collision is a signal to the capturer that the owner needs the slot back. The capturer relinquishes the slot immediately. Figure 2 depicts a system with 6 nodes and six corresponding slots. Node 1 has only one packet to transmit, which it transmits during its slot in frame 1. The fact that slot 1 is idle in the second frame is an indication to the nodes that the owner of this slot (node 1) has no more packets to transmit. So in frame 3, nodes 3 and 6 contend for this slot resulting in a collision. Due to the randomized back off algorithm, node 6 successfully captures the slot in frame 4. Node 6 has no packets to transmit in frame 5. So slot 1 again becomes available for contention in frame 6, when node 3 captures it. Note that, since node 3 always has a packet to transmit, slot 3 never comes up for contention. Slot 4 was presumably used by some other node prior to frame 1. In frame 1, node 4 recaptures its slot by causing a collision. Finally, slot 6 is empty in frame 6 because no nodes have a packet to transmit.

While Binder's protocol requires that the number of stations be known in advance, a related protocol, reservation ALOHA, proposed by Crowther et al. [8] does not have such a requirement. In this protocol, slots do not have owners associated with them. Instead, whenever a station needs a slot, it waits for a slot to become idle, i.e., it waits for some station that is currently transmitting packets to complete its transmission. It then tries to capture the slot released by this station through a process of contention. The station that succeeds in capturing the slot keeps the slot until it has no more packets to transmit, at which point the slot becomes available again to anyone who needs it. This protocol has the potential problem that if a significant traffic imbalance exists



	SLOT 1	SLOT 2	SLOT 3	SLOT 4	SLOT 5	SLOT 6
FRAME 1	15	7	6	■	10	2
FRAME 2	□	7	6	5	10	2
FRAME 3	12	□	6	5	10	2
FRAME 4	12	■	6	5	□	□
FRAME 5	12	■	6	5	3	6
FRAME 6	12	13	6	5	3	□

■ collision ▨ transmission □ idle

Figure 3: Reservation ALOHA

even for a short time, it is possible for one station to capture the entire channel starving the other stations. Figure 3 illustrates the reservation ALOHA protocol. The system depicted in the figure has 6 slots and 15 nodes. Station 15 has completed its transmission in frame 1. The fact that slot 1 is idle in frame 2 indicates to all the stations that the previous holder of this slot has completed its transmission. This slot comes up for contention in frame 3, when station 12 captures it successfully. Slot 2, held by station 7 till frame 2, comes up for contention in frame 4. After two collisions, the slot is captured by station 13 in frame 6. Slot 3 is never relinquished by station 6, since it always has a packet to transmit. The events in the other slots and frames may be explained similarly.

Another protocol that is similar in spirit to the above protocols is the protocol proposed by Limb and Flamm for the Fasnet [12] local area network. Fasnet is a high bandwidth unidirectional dual bus system capable of supporting both voice and data traffic. In the protocol proposed in [12], when a user makes a phone call, the station tries to capture an empty voice slot and holds onto the slot for the duration of a talk spurt (A conversation can be modeled as a series of talk spurts with intervening silence spurts). This slot, which constitutes a virtual voice channel, is relinquished during a silence spurt at which point it is made available to a data station. At the next talk spurt, the voice station captures another slot and this continues till the end of the call. The capture and release of slots is mediated through the two end stations on the bus. This protocol guarantees that once a call has been set up, the source station will always be able to find a free slot in each cycle of slots, whenever it needs one.

Note that, in fact, all the reservation protocols that we have described in this section, assume a traffic model similar to that of the Fasnet protocol, namely the model of a phone call. They assume that the traffic that originates at a node consists of transmission spurts followed by idle spurts (the Fasnet protocol, in addition, takes into account the real-time constraints of voice data). During a transmission spurt, a station has a sequence of messages or packets to transmit. Thus once a slot has been allocated to a station, it makes little sense to throw the slot open for contention until the station has transmitted all its packets (in an application like voice communication, owing to the real-time requirements of the data, such an action may not even be feasible). Therefore these protocols allocate a slot to a station for one transmission spurt, which may be of any length. The protocols do not involve the broadcast of any local information. Hence each station has to

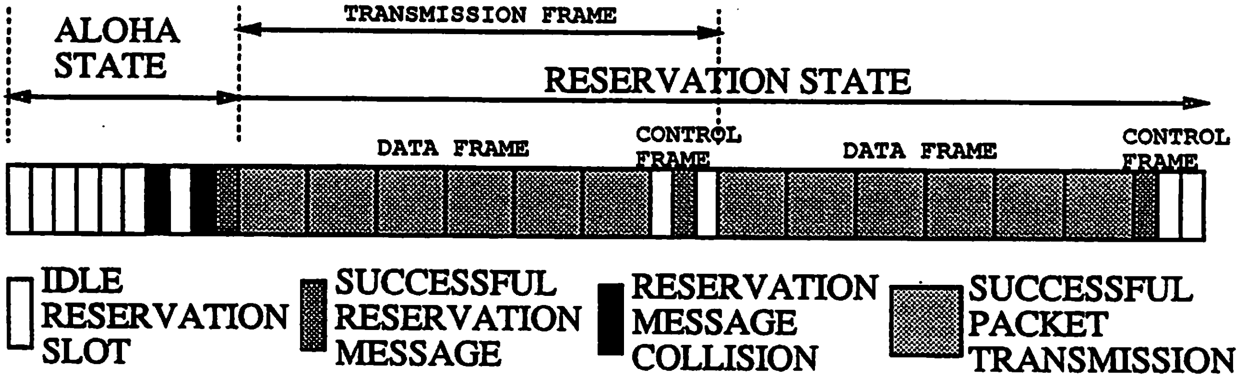


Figure 4: Roberts' Reservation Protocol

make worst case assumptions based only on local knowledge, if it wishes to guarantee asynchronous time-constrained messages.

2.3 Explicit Reservation Protocols

We next examine some protocols that make use of explicit reservation messages to arbitrate access to the channel. These protocols were proposed in the context of satellite-based packet-switched communication. All of them try to improve channel utilization and reduce the average delay experienced by a packet. The approach used basically consists of broadcasting short control messages requesting channel time prior to transmitting the actual queued messages.

The first protocol in this category that we examine was proposed by Roberts [13]. In this protocol (Figure 4), the channel may be in one of two states (i) the *ALOHA state* or (ii) the *reservation state*. The channel is in the ALOHA state when there are no pending reservations waiting to be serviced. In this state, stations may transmit new *reservation* messages according to the slotted ALOHA protocol [14]. When a reservation message gets transmitted successfully, the channel shifts to the reservation state in which reservations get serviced. In the reservation state, each transmission frame (or cycle) consists of a data frame and a control frame. The data frame is divided into data slots each equal to one data packet transmission time in length. The control frame is divided into reservation slots each equal to one reservation message transmission time in length. The data slots are used by stations that have made reservations, while the reservation slots are used by stations to make new reservations. Each reservation message consists of a number between 1 and 8 that indicates how many packets the station has to transmit. Since the channel is a broadcast channel, every reservation message is received by all stations. Each station keeps track of the total number of packets for which reservations have been made and the position of its own reservation message (according to the FIFO ordering) relative to the reservation currently being serviced. When a station's reservation request is ahead of all the pending reservation messages, the station transmits the corresponding data packet (for which the reservation was made).

The reservation TDMA protocol (Figure 5), proposed by Weissler et al., is a modification of the protocol proposed by Binder that we described in Section 2.2. In the modified protocol, each

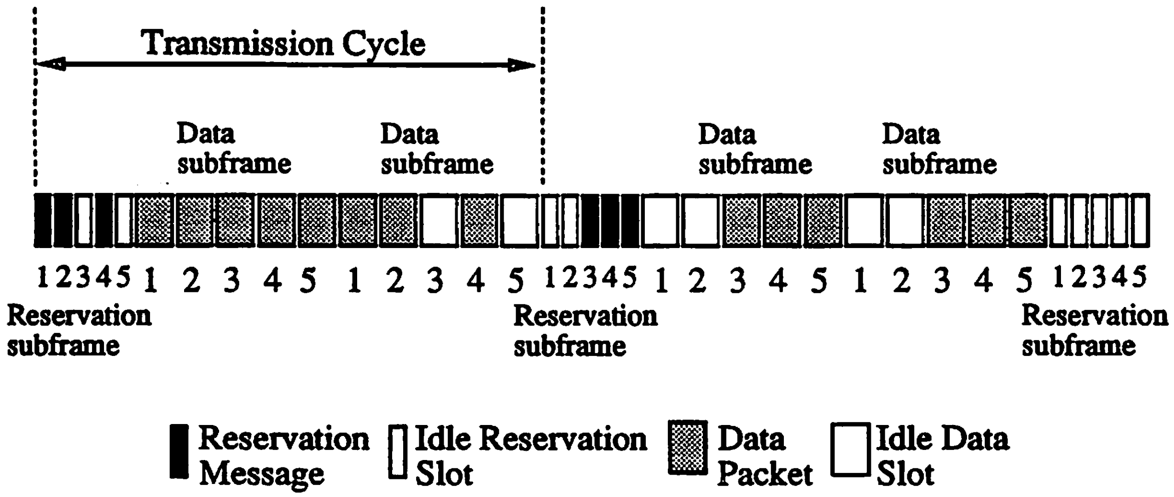


Figure 5: Reservation TDMA

transmission cycle consists of *k* data subframes preceded by a reservation subframe, where *k* is a system parameter (k=2 in Figure 5). Each of these subframes consists of one slot per station (of which the corresponding station is the "owner"). The length of a slot in the reservation subframe is equal to the transmission time of a reservation packet, while the length of a message subframe slot is equal to that of a data packet. During its reservation slot, a station *broadcasts* a reservation message that contains a "new message count", i.e., the number of new packets (which may be zero) that have arrived since the last time it made a reservation. At a commonly agreed upon time known as the update slot, each node updates its local copy of a *reservation table* based on the received new message counts. This table is used to assign data slots to the nodes according to the following algorithm (which every station executes). If the owner of a slot has pending reservations in the table, then the slot is assigned to the owner; otherwise, the slot is assigned to other stations that have made reservations in a round-robin manner. In Figure 5, stations 1, 2 and 4 transmit reservation request during their respective reservation subframe slots in the first transmission cycle[1]. Station 1 makes a request for 4 data slots, while stations 2 and 4 make requests for 2 data slots each. Since stations 3 and 5 do not have any packets to transmit, the slots owned by them in the first data subframe are allocated to station 1 to satisfy its requirements. In the second transmission cycle, stations 3, 4 and 5 each make reservation requests for two packets and are allocated the slots owned by them.

The last explicit reservation protocol that we look at was proposed by Jacobs et al. [9]. This protocol known as PODA (priority-oriented demand assignment) is a general-purpose protocol that is capable of supporting both synchronous (stream) and asynchronous (datagram) traffic, multiple delay classes and priorities, and variable message lengths. In this protocol, channel time is divided into frames each of which consists of an *information subframe* and a *control subframe* (Figure 6). The information subframe is used for scheduled datagram and stream[2] transmissions.

[1] This figure is merely meant to be illustrative. In the actual protocol, due to the large end to end propagation delay (≈ 250 millisec) in satellite based communication, reservations for one transmission cycle are made in the reservation subframe of the preceding transmission cycle

[2] A stream is a virtual channel, e.g., a voice channel, with a constant message arrival rate. Only one reservation message is required for all the messages that are part of the stream and is broadcast when the stream is set up.

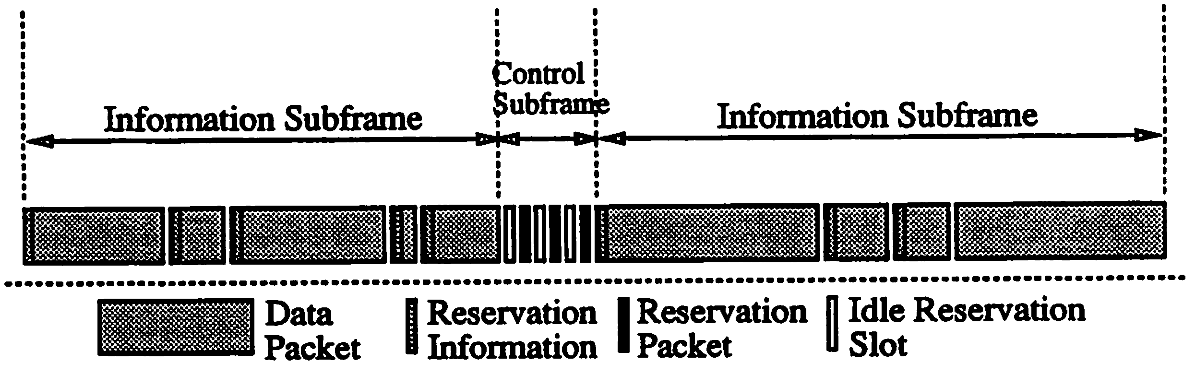


Figure 6: Priority-Oriented Demand Assignment

These transmissions may contain additional reservation requests in their headers. The control subframe is used to broadcast reservations that cannot be sent as part of the data transmissions in a timely manner (e.g., when a station wants to make a reservation for a high priority message or when a station wants to join the system). The PODA protocol comes in two flavors, each characterized by a different access method during the reservation subframe. The CPODA (contention PODA) protocol makes use of a random access scheme similar to that of Robert's protocol to arbitrate access to the channel for reservation messages. The FPODA (fixed assignment PODA) uses a TDMA scheme similar to that in the protocol proposed by Weissler et al. for this purpose.

In the PODA protocol, all stations[3] maintain a local copy of a scheduling queue ordered by reservation urgency, which is a function of the delay class and priority of a message. Reservations with the same urgency are ordered further to ensure fairness to all the stations involved. Channel time in the information subframe is allocated to the stations according to this queue. Whenever a reservation request for a asynchronous message is broadcast, all the nodes that successfully receive the message enter the request in their respective local scheduling queues (on the basis of its reservation urgency). A reservation for a stream is made when the stream is set up and is entered into a separate stream queue that is maintained at each station. A stream reservation message contains information regarding the stream repetition interval, desired maximum delay relative to this interval, and priority. Whenever the repetition time is near, each station creates a new reservation request for the stream's next message and enters it into its local scheduling queue. The PODA scheme is the most general scheduling scheme (based on individual message requirements) among the three explicit reservation schemes that we have looked at and does *scheduling based on a global view of all messages in the system*. This property is exactly what we would like to have in a protocol used to implement real-time virtual circuit service.

Note that, all the reservation protocols that we have examined so far are essentially non-real-time protocols. These protocols evolved from a desire to overcome the disadvantages of TDMA at low loads and ALOHA at high loads. Their main goal is to improve *channel utilization* (the throughput achievable with the slotted ALOHA protocol is limited to a maximum of 36 %) and

[3] The protocol also permits only a subset of stations to perform the distributed scheduling - the other stations depend on centralized assignment in this case.

reduce *average delay*. They are based on the well known performance improvement (in terms of average delay) resulting from combining multiple single server queues into a single server queue with the combined service rates of all the queues. However it should be kept in mind that we are interested in *real-time* protocols; the performance measure that is of interest to us is the *guarantee ratio* for asynchronous real-time virtual circuit messages.

3 Guarantee-Based Reservation Protocols

The individual-message-based global scheduling approach used in the PODA protocol appears to be a good basis for developing a guarantee-based reservation scheme. For example, each station can transmit reservation messages periodically and as part of data messages. These reservation messages will contain the arrival time and deadline requirements of packets that have arrived since the last reservation. Each station then applies the same guarantee algorithm on these reservation requests. If it is possible to guarantee the deadline of a packet, then a reservation is made for the packet in the local copy of the channel schedule at each station. Stations then transmit packets on the basis of their local copies of the channel schedule.

Unfortunately such a guarantee-based protocol based on a common channel schedule is not realizable in practice for the following reason. In all explicit reservation schemes, reservation messages may be corrupted by noise on the channel. This may result in different stations receiving different messages. As a result, the local scheduling queue or reservation table at each station may become mutually inconsistent. Since each station decides when to transmit on the channel based on this table, this loss of scheduling synchronization could potentially lead to collisions (when two different stations believe it is their turn to transmit) or unutilized reserved time (when each station believes it is some other station's turn to transmit). This problem is typically solved by one of the following schemes:

1. Each data message carries control information about the source station's current view of the scheduling information. For example, in Robert's scheme, information about the number of reservations that have not yet been serviced is carried in each data packet. If a station detects an inconsistency between this information and its own local view, then it remakes reservations.

2. Each station monitors the channel and compares the actual transmissions to its own view of the channel schedule, as is done in PODA. If it detects inconsistencies, it switches to a state where it suspends its transmissions and tries to reestablish a local view that is consistent with the local view of the schedule at the other nodes. In order to reestablish consistency, it may have to cancel its existing reservations and remake reservations.

Cancellation and remaking of reservations will result in increased delay. Thus the presence of channel noise has the effect of merely increasing the average delay of packets in these schemes.

For a communication scheme based on the notion of *guarantee*, once a station loses scheduling synchronization, it may transmit during the turn of another station resulting in a collision and thus cause a *violation of the guarantee* (an event that contradicts the notion of guarantee) that was provided to the colliding packets. The guarantees provided to the packets that were scheduled to be transmitted during the period when the station tries to reacquire synchronization are also violated, since the station cannot transmit any packets during this period. Such potential guarantee violations cannot be compensated for by guaranteeing multiple (redundant) packets, since once scheduling synchronization is lost the guarantees for the redundant copies may also be violated. Thus reservation-based guarantee schemes are difficult to realize in practice, since they require the *continuous* maintenance of the consistency of the local copies of a common data structure (the channel schedule); this requires that *all* the stations always receive *every* update (reservation) message *without errors*, which it may not be possible to guarantee in practice without the use of special schemes for fault-tolerance.

One approach to improving the system's tolerance of errors and failures is the standard approach based on redundancy. The shared bus is replicated n times with the same MAC protocol being employed on all the buses. There are n copies of the global queue at each node, one associated with each bus. The queue associated with a bus is constructed entirely using the reservation requests that are received on the bus. This ensures that as long as there is at least one copy of the queue that is consistent across all the nodes, the system will function correctly.

An alternative approach is to employ a synchronous atomic broadcast protocol [7], again based on redundancy. Such a protocol guarantees the following three properties:

1. *Atomicity*: Every message whose broadcast is initiated by a sender is either delivered to all receivers or to none.

2. *Order*: All delivered messages are received in the same order at all receiving nodes.

3. *Termination*: Every message broadcast by a sender and delivered to some correct receiver, is delivered to all correct receivers after some known time interval.

These properties and the fact that scheduling algorithms executed at each node are identical ensure that the local copies of the global queue are mutually consistent in spite of errors.

The reservation approach seems to be a good way of avoiding the pessimism inherent in the real-time virtual circuit approach that makes use of worst case assumptions. However, as we saw above, it incurs high implementation complexity in the form of special mechanisms for fault-tolerance.

4 Conclusion

We identified an important shortcoming of various protocols that we previously considered, viz., their pessimism that arises because of their support for the notion of guarantee and their localized knowledge. We examined three classes of reservation-based protocols to identify protocols that operate on the basis of global knowledge. The explicit reservation protocols, especially the PODA protocol, possess this property, and hence could be potential candidates for implementing a guarantee-based protocol that operates on the basis of global knowledge. However there is one problem that is common to all explicit reservation based protocols, viz., the possibility of loss of scheduling synchronization due to channel noise. In the case of non real-time protocols in which the quality of service is measured by the average delay, loss of scheduling synchronization may result in increased delays. However, in the case of real-time protocols that support the notion of guarantee, loss of scheduling synchronization is not permissible, since this may result in the violation of existing guarantees. Therefore, explicit reservation-based schemes that make use of global knowledge are not suitable for implementing guarantee-based services, unless special and complex schemes for fault-tolerance are employed.

References

[1] Abramson, N., The Aloha System — Another Alternative for Computer Communications, *Proceedings of the AFIPS Fall Joint Computer Conference*, **37**, Fall 1970.

[2] Arvind, K., Ramamritham, K., Stankovic, J.A., Window MAC Protocols for Real-Time Communication Services, *COINS Technical Report 90-127*. Submitted for publication.

[3] Arvind, K., Ramamritham, K., Stankovic, J.A., A Local Area Network Architecture for Communication in Distributed Real-Time Systems, *Real-Time Systems*, Vol. 3, No. 2, May 1991.

[4] Bertsekas, D., Gallager, R., *Data Networks*, Prentice-Hall, New Jersey, 1987.

[5] Binder, R., Packet Protocols for Broadcast Satellites, *Protocols and Techniques for Data Communication Networks*, F.F. Kuo, Ed., Prentice-Hall, New Jersey, 1981.

[6] Binder, R., A Dynamic Packet Switching System for Satellite Broadcast Channels, *Proceedings ICC*, pp. 41-1 to 41-5a, 1975.

[7] Cristian, F., Synchronous Atomic Broadcast for Redundant Broadcast Channels, *Real-Time Systems*, Vol. 2, No. 3, September 1990.

[8] Crowther, W., Rettburg, R., Walden, D., Ornstein, S., Heart, F., A System for Broadcast Communication: Reservation Aloha, *Proc. 6th Hawaii International Conference on System Sciences*, 1973, pp. 371-374.

[9] Jacobs, I.M., Binder, R., Hoversten, E.V., General Purpose Packet Satellite Networks, *Proceedings of the IEEE*, 1978, pp. 1448-1468.

[10] Kleinrock, L., Scholl, M.O., Packet Switching in Radio Channels: New Conflict-Free Multiple Access Schemes, *IEEE Transactions on Communications*, COM-28, 7, July 1980, 1015-29.

[11] Kuo, F.F., *Protocols and Techniques for Data Communication Networks*, Englewood Cliffs, N.J.: Prentice-Hall, 1981.

[12] Limb, J.O, Flamm, L.E., A Distributed Local Area Network Packet Protocol for Combined Voice and Data Transmission. Advances in Local Area Networks, IEEE Press, 1987.

[13] Roberts, L., Dynamic Allocation of Satellite Capacity through Packet Reservation, *Proceedings NCC*, pp. 711-716, 1973.

[14] Roberts, L., ALOHA Packet System with and without Slots and Capture, ACM SIGCOMM *Computer Communications Review*, Vol. 5, No. 2, April 1975.

[15] Tanenbaum, A.S., *Computer Networks*, Prentice-Hall, New Jersey, 1989.