The Challenge of Multipoint Communication

Jonathan S. Turner
WUCS-87-6

Abstract

The design of flexible communications systems, supporting a wide range of applications is the principal challenge facing the communications industry today. This paper focuses on the problem of multipoint communication, suitable for supporting such applications as entertainment video distribution, voice/video teleconferencing and LAN interconnection. We review the key issues involved in the design of multipoint communication networks, including switching system architecture, connection management, multipoint routing and congestion control. We conclude that flexible multipoint communications networks are technically feasible given current technology, and while there are many research issues requiring further study, it appears likely the cost of such networks may be only marginally higher than that of comparable point-to-point networks.

This work supported by the National Science Foundation (DCI 8600947), Bell Communications Research, Italtel SIT, NEC.

The Challenge of Multipoint Communication

Jonathan S. Turner

Abstract

The design of flexible communications systems, supporting a wide range of applications is the principal challenge facing the communications industry today. This paper focuses on the problem of multipoint communication, suitable for supporting such applications as entertainment video distribution, voice/video teleconferencing and LAN interconnection. We review the key issues involved in the design of multipoint communication networks, including switching system architecture, connection management, multipoint routing and congestion control. We conclude that flexible multipoint communications networks are technically feasible given current technology, and while there are many research issues requiring further study, it appears likely the cost of such networks may be only marginally higher than that of comparable point-to-point networks.

1. Introduction

The last decade has seen a tremendous growth in interest in flexible communications networks, capable of handling a wide range of different applications. Commercially, attention is currently focused on the Integrated Services Digital Network (ISDN), which represents a modest first step towards the goal of more flexible networks. Meanwhile, researchers around the world are exploring a variety of techniques that offer the potential of supporting a much wider range of applications, and equally important, may prove to be adaptable to the anticipated technology changes, that so often seem to render communications systems obsolete before they can be widely deployed.

The focus of most research efforts to date has been on techniques that can provide point-to-point connections at a variety of different rates. The spectrum of techniques studied have included multirate circuit switching at one end of the spectrum to fast circuit and fast packet switching at the other end. Hybrid techniques that combine conventional circuit and packet switching techniques have also been extensively studied. These techniques raise new issues in several areas of network design, including switching system design, connection management (or call processing), routing and overload control. While these issues are by no means fully understood, there is at least a growing recognition of the nature of the problems and approaches to solving them.

Jonathan S. Turner is an Associate Professor of Computer Science at Washington University, St. Louis, MO 63130. This work supported by grants from Bell Communications Research, Italtel SIT, NEC and the National Science Foundation (ECS 8600947).

This paper addresses the topic of networks for general multipoint communication. This is a natural extension of research on flexible point-to-point networks, and while it has received less serious attention to date, it promises to become a major thrust for new developments.

The motivation for multipoint communication comes from the recognition that there is a wide class of applications that requires it. The most obvious one is the distribution of entertainment programs, either audio or video. Several research organizations have organized programs designed around hybrid networks in which such "distributive services" are handled by a separate circuit-switched network. While such systems can effectively handle a few well-understood services, they provide little flexibility for supporting new services, since they can typically support only a few discrete channel rates and since they typically support only "one-to-many" connections, rather than general multipoint connections with multiple transmitters.

While the broadcast services such as entertainment video are obvious, there are many other services for which multipoint communication is important. A few of these are listed below.

- *Wire services.* News services like the Associated Press and Reuters distribute news reports from their bureaus to newspapers and radio stations throughout the world. A general multipoint connection would allow efficient transmission of this information from a moderate number of sources to a much larger number of receivers.

- *Multi-person conferences.* These services are currently handled by routing through a central point. They could be provided for efficiently and flexibly using a general multipoint connection.

- *Video lecture.* An important special case of a multi-person conference is a video lecture in which one speaker addresses a large audience, with provision for audience members to ask questions. Such a service could play an important role in education if it could be provided conveniently and inexpensively.

- *LAN interconnection.* Most large companies have local computer networks such as Ethernet at multiple locations. An extremely attractive service for them would be a multipoint connection that makes their geographically distributed LANs appear like one large network. Such a service would allow them to treat local and remote computers uniformly, allowing them to take advantage of the large base of networking software developed for LANs. The networking model offered by a multipoint-LAN is an attractive one for distributed operating systems and database applications. The extension of this model to geographically distributed networks would be very popular.

These examples indicate the possible range of services that multipoint connections can support. There is a much larger number of potential services that might become attractive if a flexible and economical multipoint connection capability were widely deployed.

If we accept the desirability of flexible multipoint communication as a goal, we are next faced with the question of whether or not such a capability is technically feasible and economically viable. While there remains a variety of challenging research problems, we contend that the answer to that question is yes. In section 2, several switching techniques that can be used to support flexible multipoint connections are reviewed. In section 3–5, research issues in connection management, routing and congestion control are discussed, key problems are identified and some potential solutions proposed. The reader should note that while the issues are discussed in the context of fast packet switching, most of the same issues arise in any network that provides multirate and multipoint communication.

2. Switching Techniques

In the last five or six years, several experimental switching system designs have been proposed that can support multirate communication in a flexible fashion. Several of these systems have been extended to support multipoint communication as well. Three such systems are discussed; the Starlite system originally developed by Alan Huang and Scott Knauer at AT&T Bell Labs and currently being developed further by a group at Bell Communications Research, the switching matrix for the Prelude experimental wide band switching system, developed by Coudreuse et. al. at CNET in France and the Broadcast Packet Switch being developed by this author at Washington University.

2.1. Starlite

Starlite is the name given to an experimental switching system developed by Alan Huang and Scott Knauer at AT&T Bell Labs [10,11,12]. The Starlite architecture was motivated by the observation that sorting networks, can be used to construct rearrangably non-blocking switching fabrics with distributed control. This observation was first put forward by Batcher [1] in 1968 in his seminal paper describing his *bitonic sorter* that sorts a set of n numbers using a network of approximately $(n/4)(\log n)^2$ simple comparison elements. For circuit switching applications, this observation leads to switching networks that are non-blocking, operationally very simple and eminently suited to VLSI implementation. To accommodate packet switching, mechanisms are needed to resolve contention between packets that arrive concurrently and are destined for the same output port. Multipoint communication requires additional mechanisms for packet replication. Huang and Knauer's contribution was the development of inexpensive VLSI implementations of Batcher's sorting network and the invention of a variety of supplementary networks which support packet switching and multipoint communication when used in concert with the sorting network.

While Huang and Knauer made no serious attempt to develop complete systems, they did develop a variety of useful tools that can be used for the construction of such systems and suggested ways in which they could be used. Figure 1 shows one possible implementation of a packet switch supporting multipoint communication. Packets arriving on external links enter a set of *Packet Processors* at left, which perform some address translation. For point-to-point packets this results in a destination PP number being placed in the packet header. This is used to guide the packet to the appropriate outgoing link.

For the moment, we will ignore the initial sort and copy networks at the top left and concentrate on what happens to packets when they enter the main sorting network at the middle of the figure. This network sorts packets in increasing order of their destination addresses, meaning that when the packets exit the sorting network, all packets with same destination address occupy a contiguous set of output links. The filters at the exit of the sorting network mark all but one packet destined for a particular address, by comparing the destination addresses of packets on adjacent links; if a packet has the same destination address as the packet on the next lower link, its *wait bit* is set. Packets for which the wait bit is zero are forwarded to the routing network at right which routes them to the proper outgoing links. Packets with the wait bit set are sent to one of a set of delay elements, which delays them for approximately one packet time, after which they are recirculated through the sorting network. It's useful to extend this basic scheme by adding a second field to each packet which records the number of times a packet has recirculated. By having the main sorting network use this field as a secondary sort key,

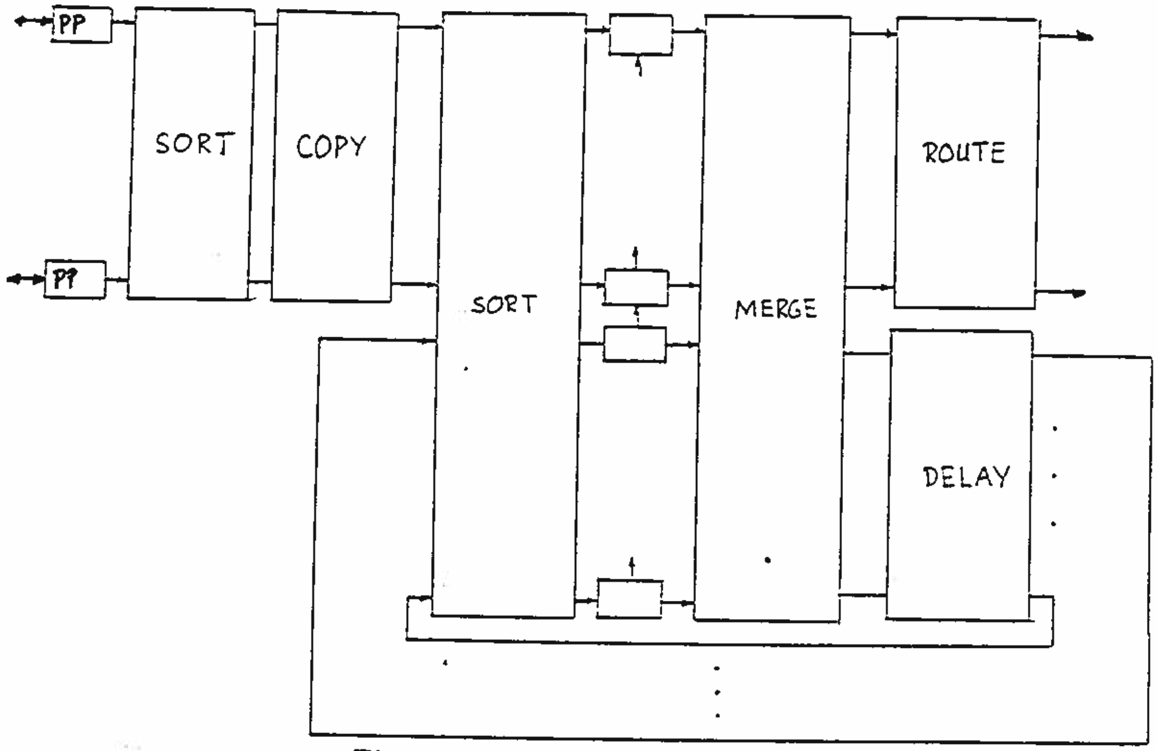


Figure 1: Starlite Switch Fabric

we can also order packets by their age, giving older packets priority over newer ones. This ensures that packets are transmitted in the same order in which they were received. If the network supports n external links and the main sort and merge networks have m input and output ports, up to $m - n$ packets may be recirculating at any time. Packets may be lost if during a cycle, more than $m - n$ packets have their delay bits set. The value of m is selected based on statistical considerations, to yield an acceptably low probability of packet loss.

We now turn to the issue of packet replication. The network is designed around the notion of a coordinated copy between source and destinations. That is, the source and the destinations must synchronize when a packet is to be copied. When the source PP sends a packet into the network, the destination PPs simultaneously send *blank packets* containing their address plus the address of the source in the headers. The initial sorting network sorts these packets on the source addresses, which places the original packet and associated blank packets on a contiguous set of links, upon exit from the sorting network. The copy network then copies the information from the source packets to each of the blank packets, a relatively straightforward process, given the sorted arrangement. When these packets enter the main sorting network, they are routed using destination addresses in the same way as point-to-point packets.

The Starlite system has some very attractive properties. The basic switch elements making up the various networks are simple and have a regular interconnection pattern, which makes the design of high speed VLSI implementations quite straightforward. The network is non-blocking and has a latency of only one bit time per stage of switching. It maintains packet sequencing, so that packets are received in the same order in which they are transmitted. The sharing of buffering across the switch fabric, rather than dedicated it to individual links provides more predictable performance in the face of statistical fluctuations in traffic.

The synchronization required for copying is a drawback of this approach, when used

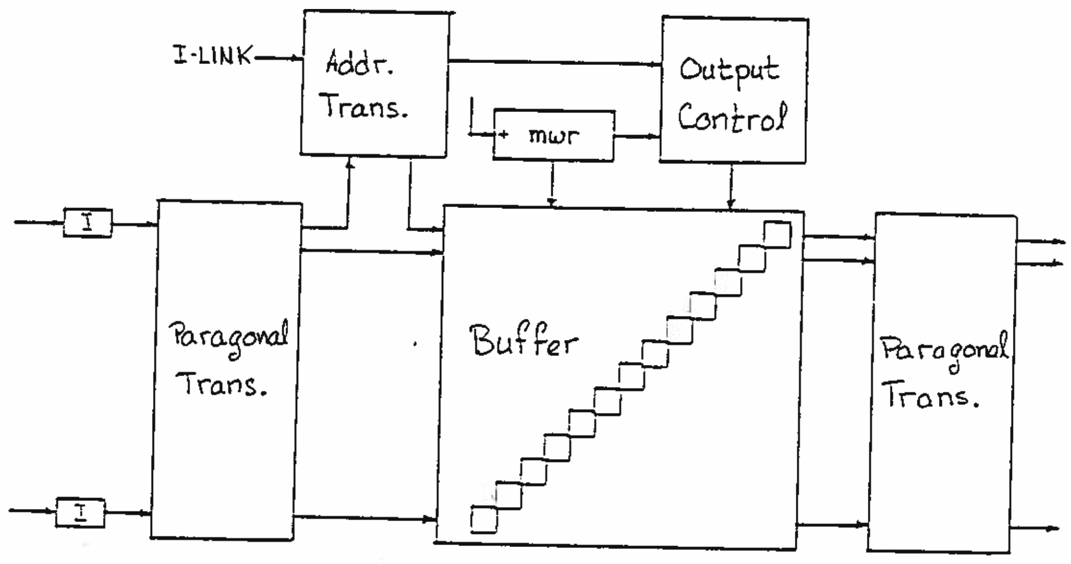


Figure 2: Prelude Switch Fabric

in a general packet switching environment. Some form of arbitration is required at the front end to schedule packets that must be copied, and while such a mechanism is probably feasible, no detailed proposal has been put forward. Also, while the switch elements are very simple, their interconnection is topologically complex relative to some competing proposals. Finally, the dimensioning of the main sort and merge networks is problematical; it appears likely that to achieve satisfactory performance in a general packet switching environment, these networks must have at least four to eight times as many inputs as there are external links. Nevertheless, the Starlite approach is a very promising one, and is a convincing demonstration of the power of a few simple ideas.

2.2. Prelude

The Prelude project began at CNET in France in the early eighties, with the objective of creating a flexible switching system that could provide point-to-point and multipoint communication at speeds up to a few hundred megabits per second [4,5,7]. It is based on a particularly simple form of fast packet switching referred to as *asynchronous time division multiplexing*, and uses a novel high speed switch fabric.

The basic structure of the Prelude switch fabric is shown in Figure 2. Packets enter at the transmission interfaces at left, which perform framing and synchronization functions. The packets are then passed through a rotative switch which transforms each packet to a so-called parallel-diagonal format (*paragonal*) in which each packet is distributed across the outputs of the rotative switch, with the first byte of each packet on the first output, the second byte on the second output, and so forth. This transformation places the headers of all packets on the first output of the switch, where they can all be processed by a centralized address translator. The address translator modifies the channel number in the header of the packet and then the modified packet is stored in a central buffer memory, still in the paragonal format. At the same time, the address translator passes

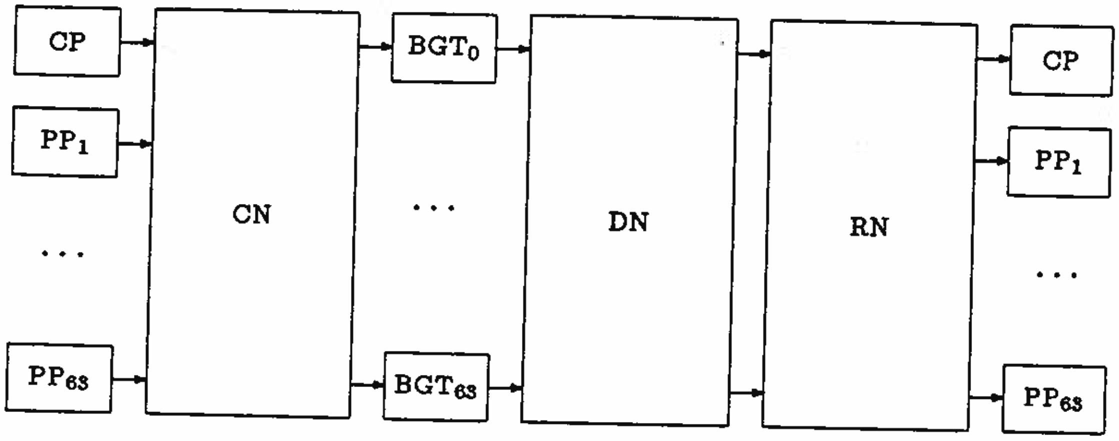


Figure 3: Broadcast Packet Switch Fabric

to the output control circuit, a bit vector defining which outputs are to receive copies of the packet. The output circuit stores the address at which the packet header was written in queues associated with the selected outputs. This information is used later to retrieve the packet from the central buffer. There is an output process that examines these queues in a cyclic fashion, initiating a new packet retrieval on each clock cycle. Broadcast is accomplished simply by reading the packet from the buffer once for each output that requires a copy. Note that these reads need not all take place during one packet cycle. Finally, a second rotative switch transforms the packets from the paragonal format back to the normal format so that they can be output on their respective links.

This design has several attractive features. The basic elements are simple; the rotative switches can be implemented as barrel shifters, requiring about $n \log n$ gates, the address translator and buffer are essentially just random access memories with a modest amount of control circuitry, and the output control consists of a fairly simple and regular collection of queues and address registers. As with Starlite it maintains packet sequencing and provides a single shared buffer rather than per line buffers. It is, on the whole simpler than the Starlite fabric and handles multipoint communication in a more satisfactory way.

The main drawback of this approach is its dependence on high speed memories, particularly in the central buffer. It must be possible to access this memory twice per clock time, once for reading and once for writing. There does not appear to be any architectural way to reduce the required memory cycle time for individual memory chips since the memory read-out process can access the memory chips in random order. Another drawback is that since channel translation takes place before packets are replicated, all the downstream copies of a multipoint packet carry the same channel number. This places operational restrictions on the assignment of channel numbers, that may be problematical, depending on the number of channels and multipoint connections. It is most troubling for general multipoint connections in which there are several transmitters. It appears that either all the links involved in such a connection must use the same channel number, or there must be a different channel number for every incoming port that can be the source of the packet. The latter solution requires that the downstream switches treat all those channel numbers similarly.

2.3. Broadcast Packet Switch

The Broadcast Packet Switch, proposed by Turner [26] is a switch fabric based on buffered binary routing networks. It is topologically simple and well-suited to VLSI implementation. The overall structure is shown in Figure 3. The system consists of a set of *Packet Processors* which interface to the external links and provide all per-packet protocol processing, a *Connection Processor* which sets up and maintains multipoint connections, and a switch fabric consisting of a *Copy Network*, a set of *Broadcast and Group Translators*, a *Distribution Network* and a *Routing Network*.

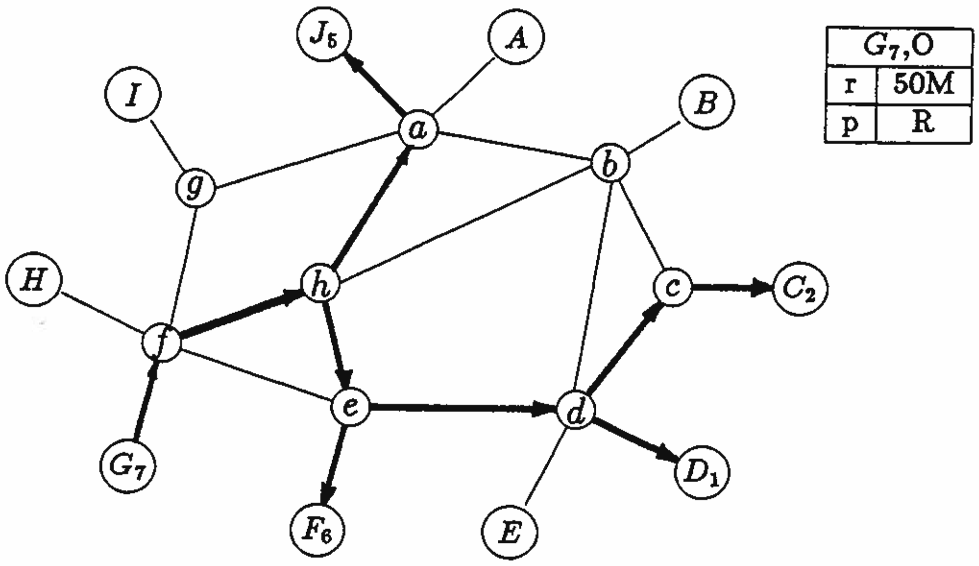
Packets enter one of the Packet Processors at left, where an address translation is performed. For point-to-point packets this yields an outgoing link number and an outgoing channel number. These are placed in the header of the packet, which then passes through the CN, one of the BGTs and the DN, following some arbitrary path. When the packet reaches the RN, it is routed using the outgoing link number. The RN is a conventional binary routing network with sufficient storage at each node to store a small number of complete packets. When the packet reaches the outgoing PP, the extra header information added at the incoming PP is stripped off and the packet is transmitted on the outgoing link. The role of the DN is to randomly distribute packets it receives across its outputs. This prevents congestion that can otherwise occur in the RN when subjected to traffic patterns with strong "communities-of-interest."

When a packet belonging to a multipoint connection is received at an incoming PP, it undergoes a similar translation process, but the information added to the packet header is different. It consists of two fields, a *Fanout* field which specifies the number of outgoing links which are to receive copies of the packet, and a *Broadcast Channel Number*, used by the BGTs. The CN replicates multipoint packets using the fanout field to guide its decisions. At each switch element where replication is performed, the fanout fields of the two copies are modified (essentially by halving the original fanout), so that a short time after the original packet enters the CN, the appropriate number of copies appears at its outputs. The BGTs then perform a translation similar to that done in the PPs, using the broadcast channel number in the copies to index a table, yielding a set of outgoing link and logical channel numbers. These are added to the packet header and used to guide the copies to the proper outgoing links.

This design is well-suited to implementation in a medium speed, high density technology like CMOS. While the individual switch elements are more complex than those in the Starlite and Prelude matrices, the topological complexity is very low, and the required speeds are relatively low. The only large memories are in the PPs and BGTs, and these need be accessed only once per packet cycle, permitting the use of high density memories with relatively long cycle times. While the switch elements are more complex than those in the the Starlite and Prelude fabric, this complexity has little impact on cost or performance.

The design does have some drawbacks. Unlike the Starlite and Prelude designs, the primary buffering is in the PPs, leading to less predictable performance in the presence of highly irregular traffic patterns. While the addition of some shared buffering is feasible, no detailed study of such an arrangement has yet been made. More seriously perhaps, is the fact that the system does not guarantee that packets are received in the same order as they are sent. While it is possible to modify the design to provide such a guarantee, the required changes impose operational constraints and degrade performance sufficiently to raise doubts about the merits of such changes. A more detailed analysis of these questions is required.

While none of the three switch fabrics reviewed above clearly dominates the others, all three represent technically feasible and economically viable solutions to the problem



G_7,O	
r	50M
p	R

Figure 4: One-to-Many Connection

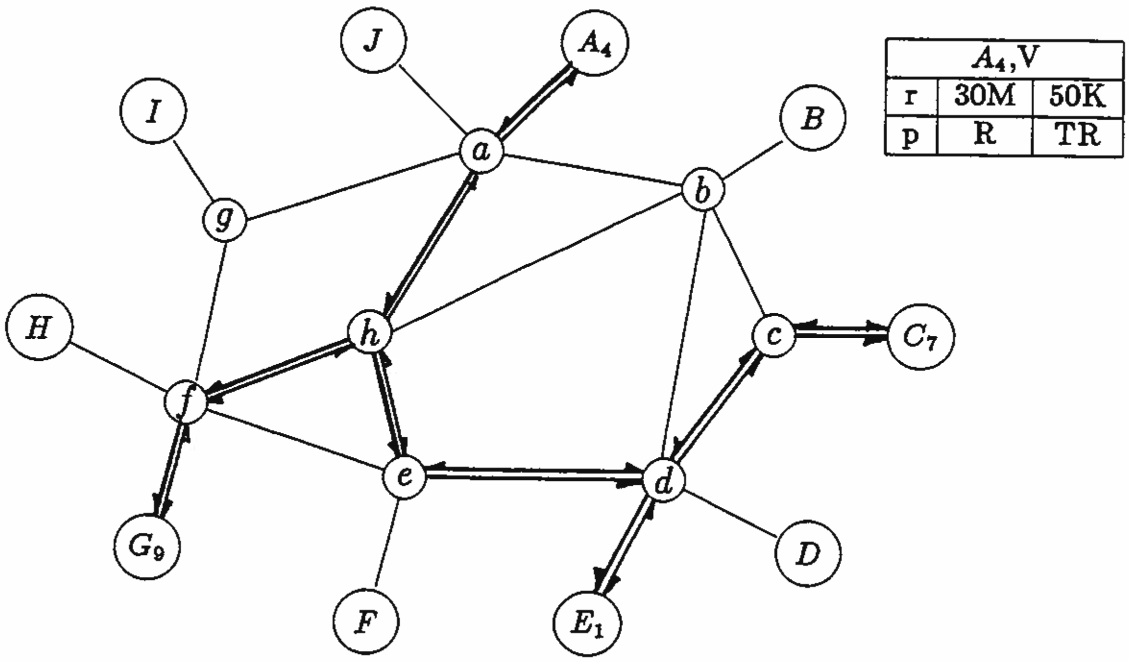
of flexible multipoint connections in a packet switched network. This observation makes it clear that the fundamental issue of switching system architecture can be solved in a variety of different ways. While there is a variety of other issues that remain to be resolved, this provides reason for being confident that the goal of a network providing flexible multipoint communication is attainable and worth pursuing.

3. Connection Management

Connection management refers to the collection of algorithms, data structures and protocols used to create and maintain connections among users. In conventional networks, connections join two endpoints. In multipoint networks, connections may join an arbitrary number of endpoints. Several types of connections appear to be useful, including point-to-point connections and simple broadcast connections having one transmitter and many receivers. Connections in which all participants can both transmit and receive are also useful, for conferencing and LAN interconnect applications among others.

As one considers applications of multipoint communication, one soon realizes that what is needed is a general multipoint connection capability that realizes point-to-point, broadcast and conference connections as special cases. This section briefly describes one approach to connection management, that provides a flexible way of defining multipoint connections supporting a wide variety of applications and which has the potential for being effectively implemented in large networks.

We start by describing a simple one-way broadcast connection, illustrated in Figure 4. The connection has a single transmitter G and receivers C, D, F and J. The subscripts in the diagram represent the channel numbers that the various terminals use to identify this particular connection among the set of connections present on the access links. The internal nodes in the diagram represent switching systems and at various points the stream of packets originating at G is replicated and forwarded to the appropriate points. The connection induces a tree in the network, in much the same way that a point-to-point connection induces a path in a conventional network. The table in the upper right-hand



A_4,V		
r	30M	50K
p	R	TR

Figure 5: Connection for Video Lecture

corner of the diagram summarizes some global information describing this connection. The G_7 at the top is the *connection identifier*, which is a globally unique name identifying this connection and distinguishing it from all other connections in the network; the motivation for having a connection identifier will be explained below. One simple way of providing such an identifier is to use the address of the *owner* of the connection together with an integer distinguishing this connection from others that the owner may also be participating in. The owner of the connection is just that terminal that is responsible for the connection and controls access to it. This scheme has been used in the example, implying that G is the owner, as well as the only transmitter in the connection. The 50M, denotes a *rate specification* of 50 megabits per second. In a real network, a more complex rate specification is required, allowing specification of peak rate, average rate and some measure of "burstiness," but for the moment, we will ignore this issue. Each endpoint participating in the connection can have *transmit-only* permission, *receive-only* permission or *transmit/receive* permission. The permission concept provides the basic mechanisms needed to allow the specification of general multipoint connections, that can be tailored to different applications. The network uses the permission information to allocate resources, (primarily link bandwidth) appropriately. The R at the bottom of the table defines the *default permission* to be receive-only, meaning that whenever an endpoint is added to the connection it is initially assigned receive-only permission; this can of course be changed by the owner if some other permission is required.

The example illustrates a connection that might be appropriate for distributing an entertainment video signal. To establish such a connection, G would send a control message to the network, describing the type of connection required. At that point it could begin transmitting on the connection, but initially there would be no one to receive the signal. Endpoints can be added to the connection in one of two ways. First G, as the owner, can send a message to the network asking that a particular endpoint be added. In response, the network would send a *connection invitation* to the specified endpoint

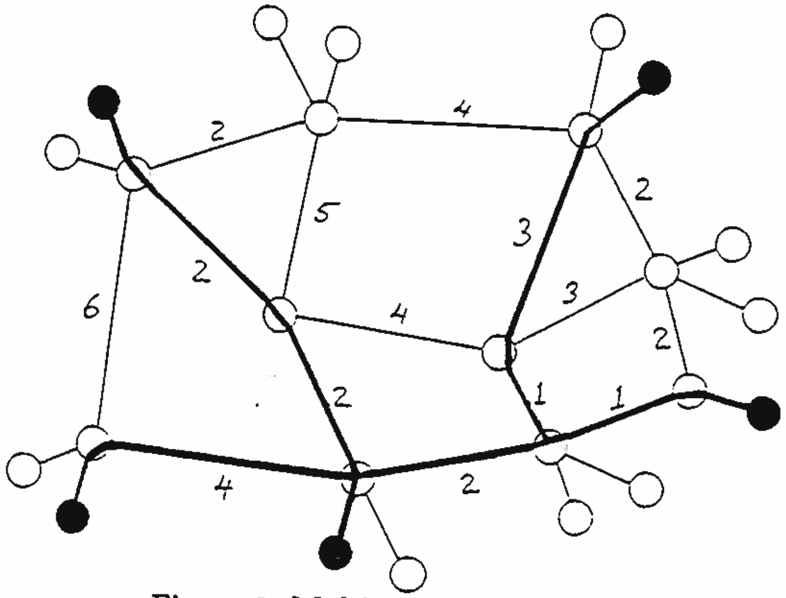


Figure 6: Multipoint Routing

and if the endpoint agrees (by exchange of control messages) to join the connection, the network would find an appropriate path and allocate the necessary resources to include the new endpoint in the connection. For entertainment video signals, a more appropriate way of adding an endpoint is at the endpoint's request. That is, an endpoint could send a control message to the network, requesting that it be added to a specified connection. To make such a request, the endpoint must specify the appropriate connection identifier. For entertainment video signals, this information would typically be widely available and could be built into terminal equipment, or programmed in, as appropriate. In response to such a request, the network would first search for the nearest place that the specified connection is available and attempt to add the new endpoint by creating a branch at that point.

Addition of new endpoints from "outside" the connection raises the need for some form of authorization. In the example, the O at the top of the figure specifies that this connection is *open*, meaning that anyone who wishes to join the connection may do so without explicit authorization from the owner. This would probably be the appropriate specification for a commercial video broadcast. Other options include *closed*, meaning that no one can join from outside and *verify*, meaning that outsiders may join, but only after getting explicit permission from the owner.

This example has illustrated the essential notions of the multipoint connection mechanism. A point-to-point connection for voice communication can be readily described using these mechanisms. In that case, the rate specification might be 50 kilobits per second rather than 50 megabits, the default permission would be TR, for transmit/receive and the outside access specification would be C (closed). Notice that in neither example, is there any need for the switching systems supporting the connection to have detailed knowledge of the connection, such as whether the signals carried are voice or video. Such information is required in the terminals and possibly network interface equipment (depending on the desired form of access), but is not needed anywhere in the core of the network. This *application-ignorance* is important for maintaining the flexible nature of the network. By keeping that information from the network core, we ensure that changes in the applications cannot affect the network's operation in any fundamental way.

Figure 5 gives an example of a more complicated connection requiring two related but separate channels within a single connection. This connection is intended for use in a

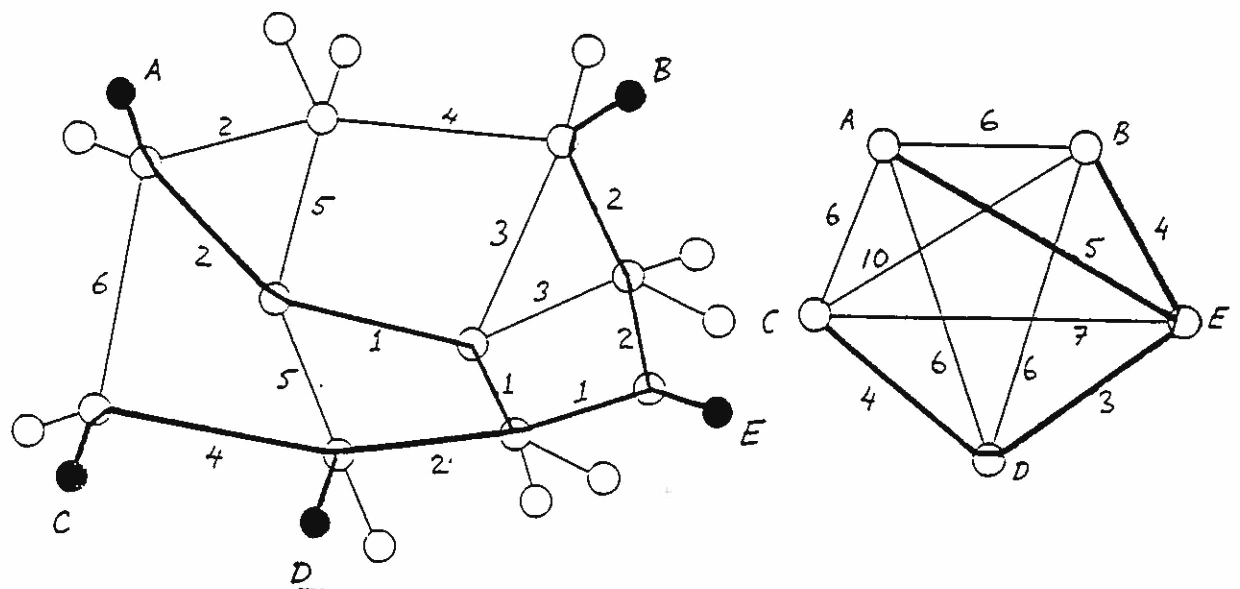


Figure 7: Minimum Spanning Tree Heuristic

video lecture situation. It includes a downstream channel from A to C, E and G, which has a rate of 30 megabits per second and a default permission of receive-only. It also has a second channel, which can be used for upstream audio, allowing the audience members to direct questions to the speaker. It has its own rate and default permissions specified. The network allocates link bandwidth separately for the two channels, but routes them through the same set of switches, for convenience in handling addition of new endpoints. The concept of multiple channels within a connection, is an important extension of the basic multipoint connection mechanism that can be useful in more complex applications.

4. Routing of Multipoint Connections

Routing connections in point-to-point networks is typically treated as a shortest path problem in a graph. Nodes represent switching systems, edges represent links and the edge lengths represent the costs associated with using a link. At the time a connection is established, one attempts to find the shortest available path connecting the desired pair of endpoints.

Routing a multipoint connection is comparable. Instead of the shortest path, one is interested in the shortest subtree of the network containing a given set of endpoints. This subtree may contain *transit nodes*, not in the specified set of endpoints. An example of a multipoint connection is given in Figure 6. The selection of transit nodes makes the problem more difficult than point-to-point routing. In fact, finding the shortest subtree connecting a set of points is a classical problem in graph theory (the Steiner tree problem) and is known to be NP-complete. This means that the existence of efficient algorithms for finding the optimal route is unlikely. Consequently, one is forced to turn to algorithms that may produce sub-optimal solutions, but which can be expected to work acceptably well in practice.

It turns out that for the Steiner tree problem, there are several known approximation algorithms that provide a good starting point for work on multipoint routing. The best known algorithm is called the *minimum spanning tree heuristic* or MST. It has been shown that the solutions produced by MST have a cost that is never more than twice optimal, and experimental results show that it is typically within five percent of optimal.

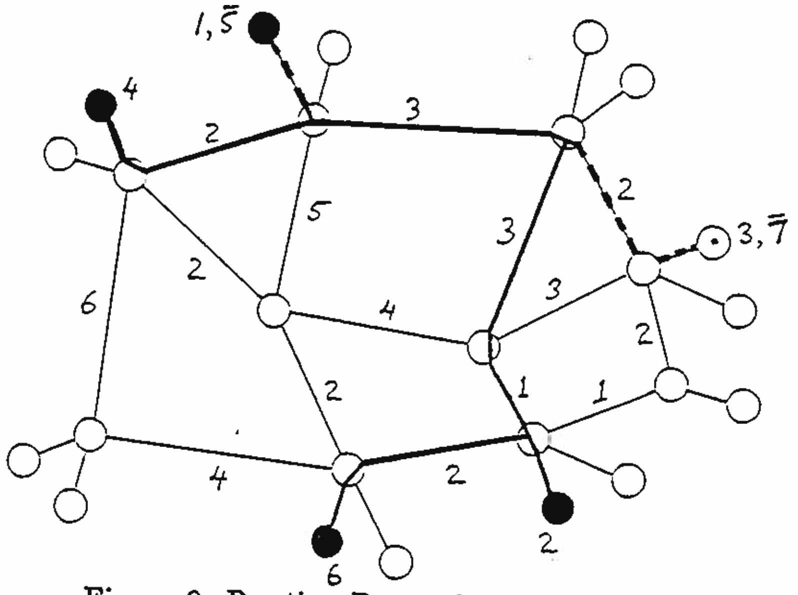


Figure 8: Routing Dynamic Connections

An example of the MST heuristic is given in Figure 7. Our objective is to find a subtree connecting nodes A, B, C, D and E. The first step is to construct another graph the *induced graph* containing just the nodes to be connected. All pairs of nodes in the induced graph are connected by an edge, with cost equal to the length of the shortest path in the original graph, connecting that pair. So for example, the length of the edge from A to E in the induced graph is 5 because the shortest path from A to E in the original graph has length 5. The second step in the algorithm is to find a minimum spanning tree in the induced graph; that is a shortest subtree that includes all the nodes in the induced graph. This can be done efficiently using well-known algorithms. For the example, the minimum spanning tree has a cost of 16, as indicated in the figure. Finally, the edges of the minimum spanning tree are mapped back onto paths in the original graph, giving the solution shown, which has a cost of 15. Note that this solution exceeds the cost of an optimal solution by one. The MST heuristic can be applied directly to routing of static multipoint connections, in which the identity of all endpoints is known in advance and in networks where centralized routing is feasible. Distributed implementations are also possible, but have yet to be developed. In any case, MST provides a useful measurement tool that can be used for comparison with other algorithms.

We now turn to the dynamic version of multipoint routing in which endpoints join and leave the connection over an extended period of time. Perhaps the simplest algorithm for this problem is to add a new endpoint to a connection, by using the shortest path from that endpoint to some node that is already in the connection. Similarly, one drops an endpoint, by simply removing that portion of the connection that serves only that endpoint. This simple *greedy algorithm* is illustrated in Figure 8. The numbers next to the solid nodes represent a sequence of events. Unbarred integers denote joining the connection and barred integers denote leaving the connection. So in the example, the connection initially consists of nodes B and D, joined later by nodes C and A (in that order). Then, node B leaves the connection, node E joins and node C leaves. Note that the algorithm can produce solutions that are far from optimal; in the example, the final arrangement could be reduced by 5 units. Experimental results have shown it to perform reasonably well, producing connections having costs that are typically within 30% of the cost of MST. One attractive feature of the algorithm is that it works incrementally, modifying the connection by addition or deletion, rather than making more complex

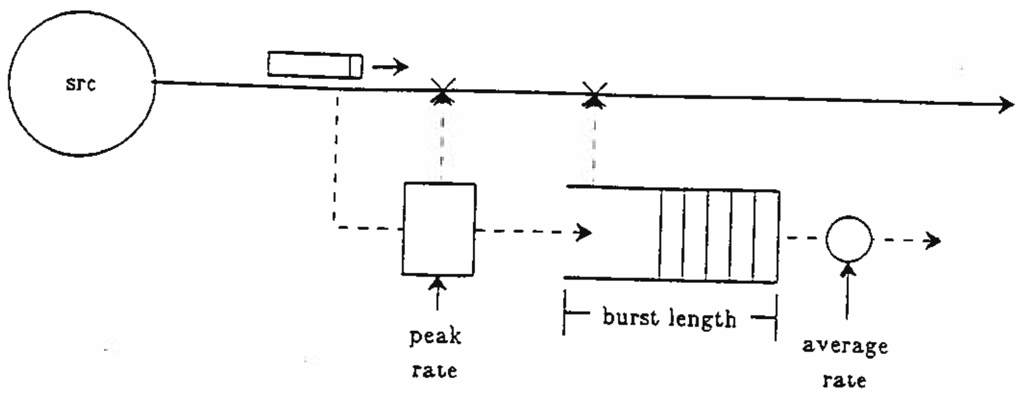


Figure 9: Simple Bandwidth Enforcement Mechanism

rearrangements. This mode of operation, while limiting the performance, may well be the only practical way to go.

5. Congestion Control

A principal advantage of packet switched networks is their ability to dynamically allocate bandwidth to the users who need it at a particular instant. Since networks are subject to rapid statistical variations in demand, care must be taken to ensure acceptable performance under conditions of peak loading. An effective congestion control system requires several specific methods, each acting on a different time scale. In connection-oriented packet networks, long term overloads can be prevented by the allocation of bandwidth to connections and the refusal of new connections unless the needed bandwidth is available. This means that the network must provide a mechanism for users to specify their bandwidth needs and an indication of the burstiness of their transmissions, and must enforce limits to prevent users from exceeding their allocations. Short term demand variations are handled by buffering within the network. To limit delay variability and keep memory buffer memory sizes at a reasonable level, the amount of buffering at each link should probably be limited to about 1 ms worth. For 100 Mb/s links this translates to about 100 Kb of memory.

When a user establishes a connection, he must provide a *rate specification* that in some sense describes the traffic characteristics of the connection. It must include the connection's *peak rate*, its *average rate* and some measure of how bursty it is. The network must be able to use the rate specification to determine what portion of each link's capacity should be allocated to the connection. In addition, the network must have a straightforward way of ensuring that the user does not exceed the allocation. One simple mechanism can be viewed as a *pseudo-buffer*, positioned at the edge of the network (this is sometimes called the "leaky bucket" method). Whenever a user sends a packet over a connection, the length of the pseudo-buffer at the network boundary is increased by one. So long as this doesn't cause the pseudo-buffer to overflow, the packet simply passes over the connection; if the pseudo-buffer does overflow, the packet is discarded (an alternative strategy is to buffer the packet and flow control the user). This idea is illustrated in Figure 9. The pseudo-buffer accepts packets at a certain maximum rate (the connection's peak rate), is drained at a constant rate (the connection's average rate) and has a maximum length (the connection's maximum burst length). These three parameters may all be specified by the user, allowing him in effect, to specify the characteristics of a kind of *virtual private link* to be associated with the connection. The network uses these parameters to determine a *bandwidth allocation* for the connection. The bandwidth allocation is selected to ensure acceptable performance (measured primarily in terms of packet loss rate) for all connections sharing links with the given connection. In general, the bandwidth allocation will lie somewhere between the peak and average rates. If the burst length and/or the peak rate are small, the bandwidth allocation can be close to the average length. If the burst length is long (meaning comparable in size to the link buffers) and the the peak rate is high (meaning comparable to the link rates), the bandwidth allocation may have to be close to the peak rate to ensure acceptable performance.

While bandwidth allocation and enforcement can limit long term congestion, other methods are needed to cope with congestion of intermediate duration. One class of methods, uses feedback to the sources. The feedback is triggered at the point where congestion occurs. Those connections contributing to the congestion are requested to reduce their rate of transmission, allowing the congestion time to clear up. In a high speed packet network, these methods appear of limited value, due to the time scales involved. Such feedback mechanisms may require 100 ms to take effect, but link buffers can fill and overflow in just a few milliseconds. For switching systems designed around shared buffers, such feedback mechanisms may be useful (since it takes longer for a large, shared buffer to overflow), but systems designed around per link buffers will likely find them of little value.

One promising method for coping with the shorter term overloads is the introduction of packet priorities. During high demand periods, the network can preferentially discard low priority packets. If half the packets on a specific FOL have low priority, that FOL can tolerate peak periods of arbitrary duration without losing any high priority packets. Priorities can be used to advantage for signals containing large amounts of redundant information. For example, video signals can be transmitted with the high order bits of each pixel carried in high priority packets and the low order bits carried in low priority packets. Occasional loss of low priority packets would probably be imperceptible. Periods of a few seconds during which many low priority packets are lost, are likely to be perceptible, but only mildly annoying to the viewer. Similar methods have been used very effectively for packet transmission of voice signals.

The above discussion applies to point-to-point connections and multipoint connections with a single transmitter. In these cases, it is straightforward to provide the bandwidth enforcement function entirely at the edge of the network. Connections with two or more transmitters and three or more receivers raise new issues, because in this case it becomes possible for packet streams from different transmitters to converge with one another inside the network, creating loads larger than are permitted at the boundary.

When one considers applications involving multiple transmitters, it becomes evident that in many cases, while there are many *potential transmitters* there is only one or a few *concurrent transmitters*. This suggests that the connection description should be augmented to include a specification of the number of concurrent transmitters. Given this information, the network can allocate sufficient bandwidth to support the specified number. It then of course, must ensure that this number is not exceeded.

There are several possible approaches to this last problem. If one wishes to control the problem entirely at the network boundary, one can apply the pseudo-buffer idea; the required change is that the length of the pseudo-buffer is incremented for every packet entering and leaving at a given boundary point. Doing this at all boundary points allows one to prevent long term congestion. The problem of course, is that the enforcement mechanism comes into play only after the congestion has occurred, when it can no longer do any good. One can modify the basic mechanism so that it "punishes" users who exceed their allocation, but this does not seem fully satisfactory.

Another approach is to have some explicit contention mechanism. The most obvious possibility is to require that each active transmitter hold a special packet called a *token* before being allowed to transmit. A connection may have several tokens, passed among the set of potential transmitters. The key drawback here is that it appears to require a mechanism for reliable token transmission, adding considerable complication to both the network and its use.

A third approach is the brute force one. Perform bandwidth enforcement throughout the network rather than just at the boundary. A bandwidth enforcer for several hundred channels is simple enough to implement on a single custom integrated circuit. The main constraint on chip area is the memory to store the information needed to simulate the pseudo-buffers for each channel. We have estimated that between 16 and 24 bytes per channel are required. Using the larger value of 24 bytes, we find that 256 channels can be handled by a chip with 50 Kbits of on-chip memory. This may be sufficient, even for links capable of carrying a much larger number of channels, since enforcement for low speed connections need be done only at the boundary. The added cost, while significant is not excessive.

6. Conclusions

This is an exciting period in the development of communications systems. Rapid improvements in technology now make it possible to implement flexible communications systems supporting a remarkably wide range of applications. The key challenge is to design systems that do not constrain that range unnecessarily and which allow predicted technology advances to be incorporated in such a way as to further extend the range.

We have discussed the problem of multipoint communication from several different angles, reviewing the problems of switching system architecture, connection management, routing and congestion control. While we by no means have a full understanding of all these issues, the nature of the problems is becoming apparent and there is good reason to be believe that viable solutions can be developed.

Acknowledgements. This work has been supported by the National Science Foundation and the Washington University Consortium for Research on Advanced Communications Systems, whose members include Bell Communications Research, Italtel SIT and NEC. I gratefully acknowledge their generous support. I particularly wish to thank Gil Devey and Steve Wolf of NSF, Eric Nussbaum and Neil Haller of Bell Communications Research, Maurizio Dècina and Anna Robrock of Italtel and Akihiro Kitamura and Noboru Suzuki of NEC. This paper describes work done collaboratively with several graduate students at Washington University. In particular, Kurt Haserodt, now at AT&T Bell Laboratories worked with me on connection management, Buddy Waxman is currently addressing the multipoint routing problem and Shahid Akhtar, the congestion control problem.

References

[1] Batcher, K. E. "Sorting Networks and Their Applications," *Proceedings of the Spring Joint Computer Conference*, 1968, 307–314.

[2] Beckner, M. W., T. T. Lee, S. E. Minzer. "A Protocol and Prototype for Broadband Subscriber Access to ISDNs," *International Switching Symposium*, 3/87.

[3] Bubenik, Richard and Jonathan S. Turner. "Performance of a Broadcast Packet Switch." Washington University Computer Science Department, WUCS-86-10, 6/3/86.

[4] Coudreuse, J. P. and M. Servel. "Asynchronous Time-Division Techniques: An Experimental Packet Network Integrating Videocommunication," *Proceedings of the International Switching Symposium*, 1984.

[5] Dieudonne, M. and M. Quinquis. "Switching Techniques Review for Asynchronous Time Division Multiplexing," *International Switching Symposium*, 3/87.

[6] Giorcelli, S., C. Demichelis, G. Giandonato and R. Melen. "Experimenting with Fast Packet Switching Techniques in First Generation ISDN Environment," *International Switching Symposium*, 3/87.

[7] Gonet, P., P. Adam, J. P. Coudreuse. "Asynchronous Time-Division Switching: the Way to Flexible Broadband Communication Networks," *Proceedings of the International Zurich Seminar on Digital Communication*, 3/86, 141–148.

[8] Hayward, G., L. Linnell, D. Mahoney and L. Smoot. "A Broadband Local Access System Using Emerging Technology Components," *International Switching Symposium*, 3/87.

[9] Hoberecht William L. "A Layered Network Protocol for Packet Voice and Data Integration," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 1006–1013.

[10] Huang, Alan and Scott Knauer. "Starlite: a Wideband Digital Switch," *Proceedings of Globecom 84*, 12/84, 121–125.

[11] Huang, Alan. "Distributed Prioritized Concentrator," U.S. Patent 4,472,801, 1984.

[12] Huang, Alan and Scott Knauer. "Wideband Digital Switching Network," U.S. Patent 4,542,497, 1985.

[13] Jenq, Yih-Chyun. "Performance Analysis of a Packet Switch Based on a Single-Buffered Banyan Network," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 1014–1021.

[14] Lea, Chin-tau. "The Load-Sharing Banyan Network," *IEEE Transactions on Computers*, 12/86.

[15] Karol, M. J., M. G. Hluchyj and S. P. Morgan. "Input vs. Output Queueing on a Space-Division Packet Switch," *Proceedings of Globecom*, 12/86.

[16] Kulzer, John J. and Warren A. Montgomery. "Statistical Switching Architectures for Future Services," *Proceedings of the International Switching Symposium*, 5/84.

[17] Montgomery, Warren A. "Techniques for Packet Voice Synchronization," *IEEE Journal on Selected Areas in Communications*, vol. SAC-1, no. 6, 12/83, 1022–1028.

[18] Muise, R. W., T. J. Schonfeld and G. H. Zimmerman III. "Experiments in Wideband Packet Technology," *Proceedings of the International Zurich Seminar on Digital Communication*, 3/86, 135–139.

[19] Rettberg, R., C. Wyman, D. Hunt, M. Hoffman, P. Carvey, B. Hyde, W. Clark and M. Kraley. "Development of a Voice Funnel System: Design Report," Bolt Beranek and Newman, Report No. 4098, 8/79.

[20] Richards, Gaylord and Frank K. Hwang. "A Two Stage Rearrangeable Broadcast Switching Network," *IEEE Transactions on Communications*, 10/85.

[21] Sincoskie, W. D. "Transparent Interconnection of Broadcast Networks," *Proceedings of the International Zurich Seminar on Digital Communication*, 3/86, 131–134.

[22] Staehler, R. E., J. J. Mansell, E. Messerli, G. W. R. Luderer, A. K. Vaidya. "Wideband Packet Technology for Switching Systems," *International Switching Symposium*, 3/87.

[23] Takeuchi, Takao, Hiroshi Suzuki, Shin-ichiro Hayano, Hiroki Niwa and Takehiko Yamaguchi. "An Experimental Synchronous Composite Packet Switching System," *Proceedings of the International Zurich Seminar on Digital Communication*, 3/86, 149–153.

[24] Turner, Jonathan S. and Leonard F. Wyatt. "A Packet Network Architecture for Integrated Services," *Proceedings of Globecom 83*, 11/83, 45–50.

[25] Turner, Jonathan S. "Fast Packet Switching System," United States Patent #4,494,230, 1/15/85.

[26] Turner, Jonathan S. "Design of a Broadcast Packet Network," *Proceedings of Infocom*, 4/86.

[27] Turner, Jonathan S. "New Directions in Communications," *IEEE Communications Magazine*, 10/86.

[28] Turner, Jonathan S. "Design of an Integrated Services *Packet* Network," *IEEE Journal on Selected Areas in Communications*, 11/86.

[29] Yeh, Y. S., M. G. Hluchyj and A. S. Acampora. "The Knockout Switch: a Simple Modular Architecture for High Performance Packet Switching," *International Switching Symposium*, 3/87.