OPTIMAL NONBLOCKING MULTIPOINT VIRTUAL CIRCUIT SWITCHING

Jonathan S. Turner
Computer Science Department
Washington University, St. Louis

WUCS-93-47

September 25, 1993

Department of Computer Science
Campus Box 1045
Washington University
One Brookings Drive
St. Louis, MO 63130-4899

Abstract

Many multipoint virtual circuit switching systems have been proposed for use in ATM networks. The inherent flexibility of ATM networks and the desire to use them for a wide range of different applications makes it impractical to rely upon statistical traffic models, motivating a renewed interest in nonblocking networks. While the various multipoint architectures can all be configured to be nonblocking, for most, the cost quickly becomes prohibitive. We examine the complexity of several multipoint switch architectures, taking into account the switching network, the memory needed for routing and the effort required to create or modify a virtual circuit. We show that a simple nonblocking architecture, based on the concept of cell recycling, achieves optimal complexity. This architecture exemplifies a class of networks which we refer to as *reroutably nonblocking*.

[0]This work was supported by the National Science Foundation (grant DCI 8600947), Ascom Timeplex, Bell Communications Research, Bell Northern Research, Goldstar Information and Communications, Italtel SIT, NEC, NTT and SynOptics.

OPTIMAL NONBLOCKING MULTIPOINT VIRTUAL CIRCUIT SWITCHING

Jonathan S. Turner
Computer Science Department
Washington University, St. Louis

1. Introduction

Multipoint virtual circuit networks support communication paths from a sender to an arbitrary number of receivers. Multipoint virtual circuits induce a tree in the network connecting the sender to the receivers. Switching systems participating in the virtual circuit replicate received cells using *virtual circuit identifiers* in the cell headers to access control information stored in the switching system's internal control tables, then use this information to identify the outputs the cells are to be sent to and relabel the copies before forwarding them on to other switching systems. Figure 1 illustrates the function of a multipoint virtual circuit switch. The switch includes control information, shown here as a table, which for each incoming virtual circuit provides a list of outputs and outgoing virtual circuit identifiers. For a cell received on input link i and virtual circuit z, the switch forwards copies to outputs $j_1, j_2, \ldots$ after relabeling them with new virtual circuit identifiers, $y_1, y_2, \ldots$

This paper explores the question of how to design a multipoint virtual circuit switch so that no legitimate request to establish a new virtual circuit or extend an existing virtual circuit (adding an additional endpoint) need be refused for lack of switching system resources. That is, we are interested in systems that are nonblocking with respect to virtual circuit requests and have minimal complexity, where the complexity measures we are concerned with are (1) the hardware required for the switching network, (2) the memory required for defining the collection multipoint virtual circuits and (3) the amount of work that must be done to respond to a requested incremental change to a virtual circuit. For a system with n inputs and outputs and capable of supporting m virtual circuits per input/output, an optimal system will has a network complexity of $O(n \log n)$, contains $O(mn)$ words of memory with $O(\log mn)$ bits per word and can be reconfigured in response to an incremental change request by modifying a fixed number of memory words.

We study this question by reviewing the architecture of several multipoint virtual circuit switching systems, showing how they can be configured to provide nonblocking multipoint operation and determining the complexity of these configurations. We conclude with a

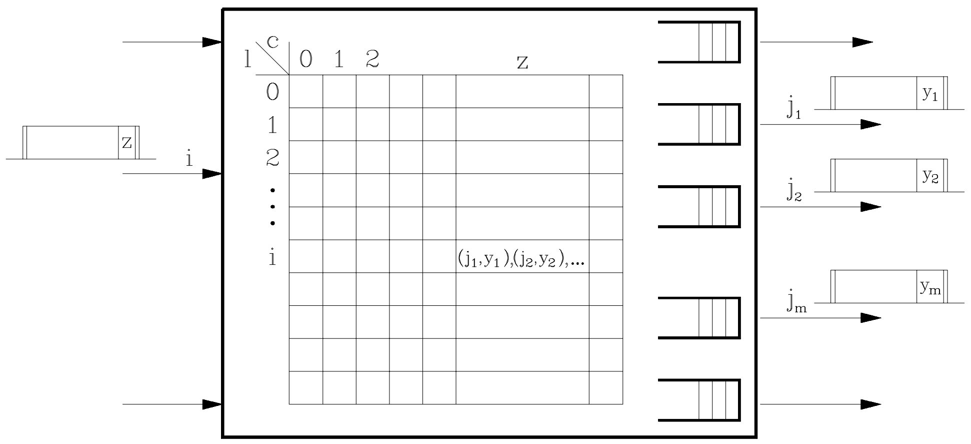

Figure 1: Multipoint Switch Functionality

novel multipoint switch architecture that is conceptually very simple and is simultaneously optimal in all three complexity dimensions.

2. Lee's Multipoint Switch Architecture

Lee [4] describes a multipoint switch architecture that uses unbuffered switching components and a novel contention resolution scheme to handle the fanout contention associated with multipoint virtual circuits. The design is shown in Figure 2. It includes a point-to-point switching system on the right preceded by several additional components to handle multipoint. Cells are received at the *Port Processors* (PP) on the left where they are assigned a *Fanout* and a *Broadcast Channel Number* (BCN). The cells then pass through a concentrator network, which places the cells on consecutive outputs so as to ensure non-blocking operation of the subsequent networks. Next, the cells pass through a running adder network, which for the cell on output port i computes the sum of the fanouts of all cells entering on ports 0 through $i-1$ and places this sum in a field of the cell (this is done for all output ports, using a network with $n \log_2 n$ simple processing elements). Following the adder, the cells enter a set of *dummy address encoders* (DAC), which perform two functions. First, the DAC at output i sets a variable lo to be the sum computed by the adder network for output i and lets $hi = lo + f_i - 1$ where f_i is the fanout of the cell at output i. If hi exceeds the number of outputs of the entire network, the cell is discarded. The DACs return an acknowledgement to the sending input for each cell that is not discarded. The discarded cells are retransmitted later.

For the cells that are not discarded, lo and hi are inserted into the cell header as the cell is sent to the copy network. The copy network sends copies of each cell to all the outputs in the range from lo to hi. The copy network also labels the copies and when each copy reaches a *Broadcast Translation Circuit* (BTC) its broadcast channel number and the copy number are used to produce an output address that the point-to-point switch uses to guide the cell

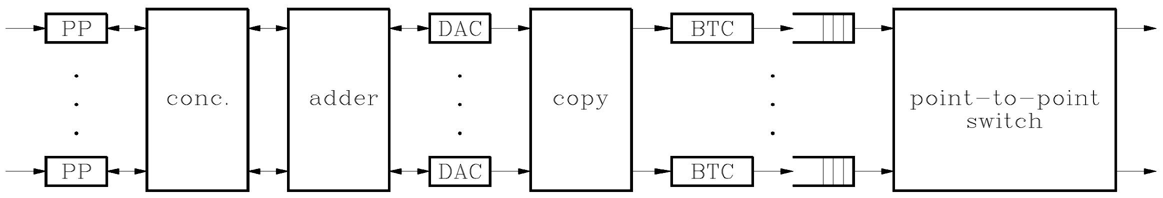

Figure 2: Lee's Multipoint Switch Architecture

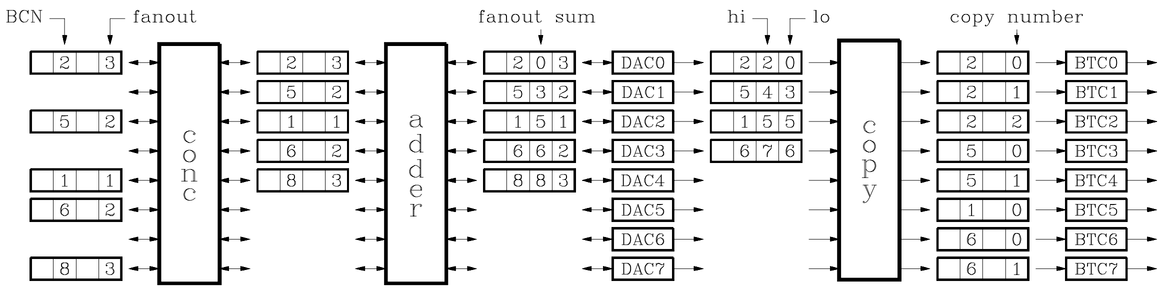

Figure 3: Example of Operation of Lee's Multipoint Architecture

to its ultimate destination. An example, illustrating the operation of Lee's architecture is shown in Figure 3.

While a variety of point-to-point switches can be used in this architecture, the original intent was that it would be used with an unbuffered switching network based on Batcher's bitonic sorter [1]. This leads to a complexity of $O(n(\log n)^2)$ for the switching network As Lee describes it, the architecture requires $O(mn^3)$ words of memory to support nonblocking operation and requires changes $O(n^2)$ words of memory in order to modify a virtual circuit. However [10] develops a series of improvements which taken together lead to version that can be configured for nonblocking operation with $O(mn^2)$ words of memory. For this configuration, the number of memory words that must be changed to modify a multipoint virtual circuit is $O(n)$.

3. Buffered Multistage Networks with Fixed Path Routing

The Prelude project of the early 1980s [3] introduced a multipoint switch architecture based on a natural extension of classical synchronous time division switching systems. Several manufacturers have adopted variations of this architecture in their current ATM switching products. In these architectures, a buffered multistage network carries user information cells through the switching system along a fixed path. For multipoint virtual circuits, the cells are replicated at various points within the switching system, inducing a tree that uses a fixed set of links in the switching system. This means that bandwidth must be allocated along the system's internal data paths to accommodate the bandwidth requirements of the various virtual circuits routed through those data paths. It also means that each of the basic switching elements making up the multistage network requires a lookup table which specifies which output links should receive copies of cells belonging to each virtual circuit.

There are many specific multistage interconnection network topologies that can be used. The *extended delta networks* [10] form a general class that includes the well-known delta network and the Beneš network as special cases. Let $d \geq 2$, $k \geq 0$, $0 \leq h < k$ be integers and let $n = d^k$. The extended delta network $D^*_{n,d,h}$ is constructed from $d \times d$ switch elements and has $k+h$ stages. The last k stages are topologically equivalent to a d-ary delta network. The initial h stages can be thought of as distribution stages which provide d^h distinct routes between any input and any output. When $h = 0$, the extended delta network is equivalent to the ordinary delta network and when $h = k-1$, it is equivalent to a Beneš network.

All of these networks have $O(n \log n)$ complexity for fixed d. To achieve nonblocking operation, the speed of the internal data paths must be higher than that of the external links. We denote the ratio of the internal data path speed to the external link rate by σ and refer to it as the *speed advantage* of the system. Since for any fixed clock rate an increase in speed advantage requires increasing data path width, the cost of a system grows with σ giving an overall system cost that is $O(\sigma n \log n)$. In [5], it is shown that for point-to-point virtual circuits it is necessary to have $\sigma \geq 1 + 2(1 + (d-1)(k-1))/d$ when $h = k-1$ (that is, for the Beneš network, which provides the best performance for this class of networks) which implies that the system complexity is $O(n(\log n)^2)$. For multipoint virtual circuits, the situation is worse. Here, we require a speed advantage of approximately $(n/d)(1-1/d)$ to avoid blocking, making the complexity $O(n^2 \log n)$ for any fixed d. Adding an endpoint to an existing virtual circuit requires changing $O(\log n)$ table entries.

To determine the memory requirements we need to consider two options. The conceptually simpler option is to have the lookup table at each switch store a bit vector of d bits (specifying a subset of the switch outputs) for each of the different multipoint virtual circuits that could be simultaneously active ($mn/2$ in the worst-case). This leads to a worst-case memory complexity of $O(mn^2 \log n)$. The second option is to have the lookup table specify not only the output ports to be used for a given multipoint connection but new virtual circuit identifiers for each copy as well. In order for each of the internal data paths to accommodate m virtual circuits, the total number of table entries would be $O(mnk)$, and the size of each entry would be $d \log m$ bits instead of a single bit. This leads to a complexity of $O(mn \log n)$ words. We note that these memory estimates underestimate the real system complexity since the required speed advantage would force a more costly memory organization as well. This can increase the real cost of the lookup tables by another factor of n.

In [6], it is shown that by cascading two Beneš networks together, one can achieve nonblocking operation for new virtual circuits (but not necessarily for extending an existing virtual circuit) with the same speed advantage required for point-to-point virtual circuits. This means that by rearranging just the virtual circuit we are seeking to extend, it is always possible to add a new endpoint. We use the term *reroutably nonblocking* for networks for which it is possible to avoid blocking in this way. The amount of memory required for this network is roughly double that needed for the single Beneš network. The number of words that must be changed in the worst-case when adding an endpoint to a virtual circuit with fanout f is $O(f \log n)$, including the time needed for rerouting. If no rerouting is required, the time is $O(\log n)$.

4. Buffered Multistage Networks with Per Cell Routing

Buffered multistage switching networks can also be constructed so that cells are routed individually, allowing different cells in a given virtual circuit to take different paths through the switching network. This means that cells may exit the network in a different order from that by which they entered, requiring a resequencing buffer at each output port of the network. The depth of the resequencing buffers must be proportional to the network delay to ensure that cells that pass through the network quickly can be held up long enough to allow cells that take a longer time to get through a chance to catch up. Since delay grows in proportion to the number of stages in a buffered multistage network, this does not change the growth rate of the system complexity, so we need not consider the resequencing cost further.

Networks which route cells independently, distribute the traffic as evenly as possible across the switching network's internal data paths allowing the system to operate with a much smaller speed advantage that when fixed path routing is used. To determine the necessary speed advantage, we can apply an analytical technique developed in [7]. We denote a virtual circuit by a triple (x, Y, ω), where x is an input to the network, Y is a set of outputs and $0 < \omega < 1$ is a *weight*, which represents the data rate of the virtual circuit, taken as a fraction of the network's internal data path speed.

We say that a virtual circuit *induces a load* on the links that lie on paths joining the virtual circuit's input and output ports. In particular, we assume that the load is distributed evenly whenever there are multiple paths to the desired destination. We let $0 < \alpha \leq 1$ denote the maximum load allowed on any of the network's input or output ports. A *virtual circuit assignment* C is a set of virtual circuits that satisfies this constraint and the load induced by C on link ℓ is denoted $\lambda_\ell(C)$. We say a network with per cell routing is nonblocking if for all virtual circuit assignments, $\lambda_\ell(C) \leq 1$ for all links ℓ.

A network with the extended delta topology can be used for switching multipoint virtual circuits by having it perform traffic distribution in the first h stages and then copying the cells to the required outputs in the final k stages. Hence, for $h \geq 1$, if a virtual circuit places a load ω on some input x of the network, the load on the links exiting from x's first stage switch is $1/d$, the load on the links exiting from the subsequent second stage switches is $1/d^2$ and so forth. This spreading of the load continues for the first h stages and then stops. In the last h stages, the load builds up again, increasing by a factor of d at every stage. Let ℓ be a link in stage i of an extended delta network and let $c = (x, Y, \omega)$ be a virtual circuit. From the above discussion, if there is a path in the network from x to ℓ and a path from ℓ to any output in Y, then

$$\lambda_\ell(c) = \begin{cases} \omega_j d^{-i} & 0 \leq i \leq h \\ \omega_j d^{-h} & h \leq i \leq k \\ \omega_j d^{-(k+h-i)} & k \leq i \leq k+h \end{cases}$$

Let C be any set of virtual circuits on $D^*_{n,d,h}$ and let C_ℓ be the subset of virtual circuits $c_j = (x_j, Y_j, \omega_j)$ for which there is a path from x to ℓ and a path from ℓ to some output in Y. Note that there are paths to ℓ from at most d^i inputs and from ℓ to at most d^{h+k-i} outputs. Because the load on every input and output is limited to α,

$$\sum_{c_j \in C_\ell} \omega_j \leq \alpha d^i \quad \text{and} \quad \sum_{c_j \in C_\ell} \omega_j \leq \alpha d^{k+h-i}$$

Thus, for $0 \leq i \leq h$,

$$\lambda_\ell(C) = \sum_{c_j \in C_\ell} \lambda_\ell(c_j) \leq d^{-i} \sum_{c_j \in C_\ell} \omega_j \leq \alpha$$

For $k \leq i \leq k+h$,

$$\lambda_\ell(C) = \sum_{c_j \in C_\ell} \lambda_\ell(c_j) \leq d^{-(k+h-i)} \sum_{c_j \in C_\ell} \omega_j \leq \alpha$$

For $h \leq i \leq k$,

$$\begin{aligned} \lambda_\ell(C) &\leq d^{-h} \sum_{c_j \in C_\ell} \omega_j \leq \alpha d^{-h} \max_{h \leq r \leq k} \min \left\{ d^r, d^{k+h-r} \right\} \\ &= \alpha d^{-h} d^{\lfloor (k+h)/2 \rfloor} = \alpha d^{\lfloor (k-h)/2 \rfloor} \end{aligned}$$

Hence, for any assignment C on $D^*_{n,d,h}$, $\lambda_\ell(C) \leq \alpha d^{\lfloor (k-h)/2 \rfloor}$ for all links ℓ. For $h = k-1$, this is simply $\lambda_\ell(C) \leq \alpha$. That is, the load on the internal links of the network is bounded by the load on the inputs and outputs. Hence, the only speedup needed in the network is that required to accommodate queueing effects. From [2] it is apparent that if the network is constructed from shared buffer switches with $d \geq 8$ and $4d$ buffer slots, a speed advantage as low as 20% can suffice.

This analysis implies that the switching network complexity needed to achieve nonblocking operation is $O(n \log n)$. As in fixed path networks, lookup tables are needed in at least the last k stages of the network. Because cells are routed independently, the size of the routing tables must be larger. In particular, the number of memory words required for nonblocking operation increases to $O(mn^2 \log n)$. The number of words that must be changed to modify a virtual circuit is large but the number of control cells needed to effect this change can be reduced by having the network copy the control cells. Taking this into account, the effective cost of modifying a virtual circuit is $O(\log n)$.

5. Broadcast Packet Switch

The broadcast packet switch architecture [8] is a multipoint switch architecture which is also based on the principle of per cell routing. There is a variety of specific design choices one can make with this architecture, but stated in a general way, it consists of a pair of extended delta networks, cascaded one after the other. The first network (called the copy network) is responsible for making copies of cells belonging to multipoint virtual circuits, using a fanout field in the cell header to produce the appropriate number of copies. The second network (called the routing network) is responsible for routing the individual copies to the proper network outputs. In between the two sections is a set of lookup tables which assign each of the copies an outgoing link number. The networks themselves do not require any lookup tables but rely entirely on information carried in the cell headers. By an analysis similar to the one given above, it has been shown that the networks require only a small constant speed advantage (with the exact speed advantage depending on the number of traffic distribution stages in the copy and routing networks) [7]. Hence, the network complexity is $O(n \log n)$.

Each of the n lookup tables requires an entry for each of the mn multipoint virtual circuits and each entry stores information relating to a single output of a virtual circuit. This means that the memory requirement is $O(mn^2)$ and $O(n)$ words may need to be changed to modify a virtual circuit. By having the copy network replicate control cells, the number of control cells that must be sent to effect this change can be reduced to $O(f)$ for virtual circuits with fanout f.

The size of the lookup tables can be dramatically reduced if the copying process is modified so that cells with smaller fanouts are delivered only to a subset of the lookup tables. Let $f_0 = 1 < f_1 < f_2 < \cdots < f_r = n$. We assign a virtual circuit belongs to *fanout class* i if its fanout exceeds f_{i-1} but not f_i. If the copy network routes virtual circuits in class i to one of a set of nonoverlapping ranges of f_i outputs, the number of multipoint virtual circuits of class i that can be handled with a given amount of memory can be increased by a factor of n/f_i. If i maximizes $f_i/(f_{i-1}+1)$, all virtual circuits have fanout $f_{i-1}+1$ in the worst-case. In this case, we can have $mn/(f_{i-1}+1)$ different virtual circuits and we use a total of f_i lookup table entries for each one. To handle the worst-case then, $mn(f_i/f_{i-1})$ total entries are enough. (In the original architecture, there is only one class and we need $mn^2/2$ table entries to handle the worst-case situation.) If f_i/f_{i-1} is the same for all i, then the number of classes is $\log_{(f_i/f_{i-1})} n$.

Thus, we can cut the amount of memory needed to $2mn$, using $\lg n$ levels (where lg denotes the base 2 logarithm). Unfortunately, fanout f virtual circuits now require consecutive sets of lookup tables with sufficient unused bandwidth. To prevent excessive loading at the lookup tables, an additional speed advantage is needed which grows in proportion to the number of fanout classes. The trade-off between the number of table entries and the network complexity (resulting from the increased speed advantage) is illustrated for several examples in the following table.

r	table entries	network complexity
1	$O(mn^2)$	$O(n \log_d n)$
2	$O(mn^{3/2})$	$O(n \log_d n)$
$\lg\lg n$	$O(mn^{1+1/\lg\lg n})$	$O(n(\log_d n)(\lg\lg n))$
$\lg n$	$O(mn)$	$O(n(\log_d n)(\lg n))$

For $n = 2^{16}$, $\lg\lg n$ is 4 and $n^{1/\lg\lg n}$ is 16.

In practice, this analysis may be too pessimistic, since we would expect most multipoint virtual circuits to have small fanout. We can get better performance in practice by designing for this case. To determine the amount of memory needed in the "expected case," we need a probability distribution for the frequency of virtual circuits with different fanouts. There are two natural distributions that are worth considering. First, assume that for any i, the number of virtual circuits with fanout in the interval $[2^{i-1}+1, 2^i]$ is twice the number in

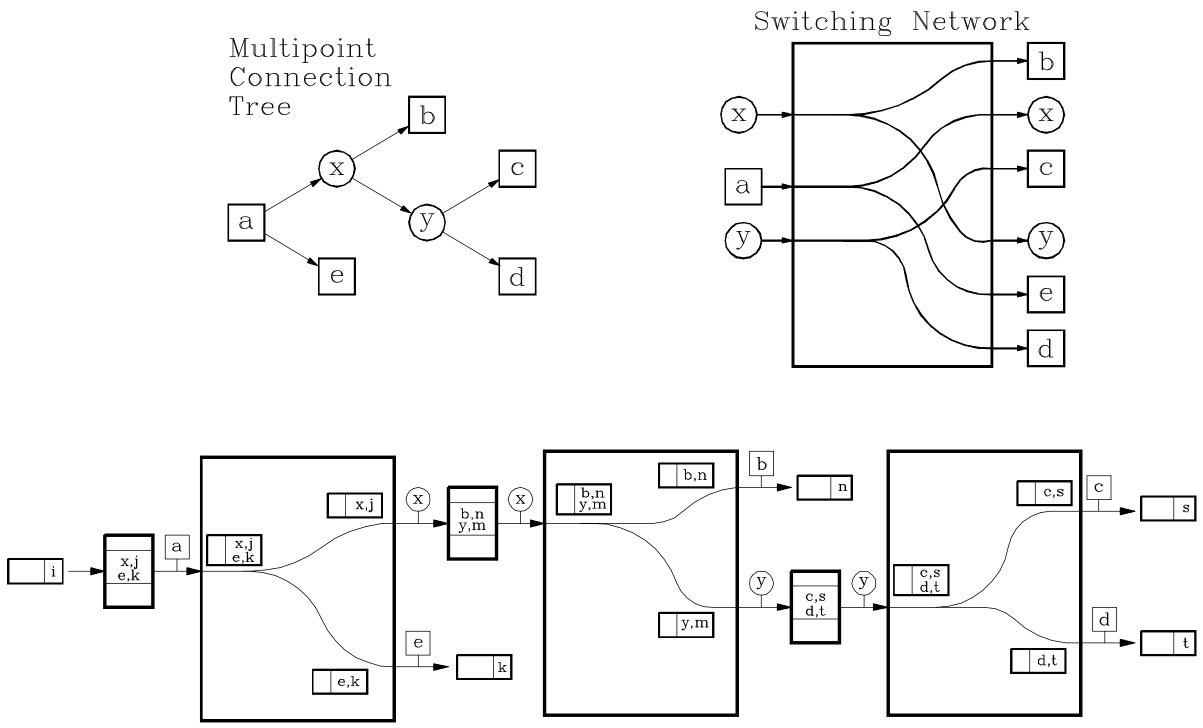

Figure 4: Multipoint Switching by Recycling Cells

the interval $[2^i+1, 2^{i+1}]$. The justification for this is that because the virtual circuits in the larger fanout class use roughly twice the output bandwidth of virtual circuits in the other class, there is "only room" for half as many of them.

Because smaller fanouts are more common, we can get better memory utilization by making the boundaries defining the small fanout classes closer together at the expense of wider spacing for the larger fanout classes. For example, if fanout classes are defined by the intervals $\left[2^{2^{i-1}}+1, 2^{2^i}\right]$, we can handle the expected case with $O(mn)$ table entries and network complexity of $O(m(\log n)(\lg\lg n))$.

Another natural fanout distribution is obtained by assuming that each of the mn output virtual circuits selects with equal probability from among the mn input virtual circuits (where each input virtual circuit can be chosen multiple times). It's straightforward to show that under this assumption, when mn is not too small, the expected number of virtual circuits with fanout $> \lg mn$ is less than 1. Hence, two fanout classes with a boundary at $\lg mn$ requires $O(m \log mn)$ entries per lookup table or a total of $O(mn \log mn)$ words and a network complexity that is $O(n \log n)$. By going to more levels, we can reduce the number of lookup table entries required. In particular, with $\lg\lg\lg mn$ levels the number of lookup table entries needed for nonblocking operation is $O(mn)$ and the network complexity $O(n(\log n)(\lg\lg\lg mn))$.

6. Multipoint Switching with Cell Recycling

Another approach for switching multipoint virtual circuits is based on the concept of *cell recycling* [10]. This is illustrated in Figure 4. To implement a multipoint virtual circuit, a binary tree is constructed with the source port at its root and the destination ports at its leaves. Internal nodes represent ports acting as relay points, which accept cells from the switching network but then recycle them back into the network after relabeling the cells with a destination pair identifying the next two ports they are to be sent to. In this example, a multipoint virtual circuit delivers cells from input a to outputs b, c, d and e, using ports x and y as relay points. In the lower part of the diagram, the implementation of the virtual circuit is shown in an 'unrolled' form, to clarify the flow of cells through the system. It should be understood however, that this is purely illustrative. There is in fact just one switching network, not three, and cells are simply sent through it multiple times in order to reach all the destinations. In the example, cells entering at input a with virtual circuit identifier (VCI) i, are forwarded to output e, VCI k and output x, VCI j. At x, the cell is recycled, with VCI j used to select a new table entry from x's VXT. The resulting information causes the cell to be forwarded to output b, VCI n and output y, VCI m. At y, the cell is recycled again, with the resulting copies delivered to c and d. Note that all the information needed by the switching network is placed in the cell header by the lookup table at the switch port, eliminating the need for any lookup tables within the switching network.

To add an endpoint to a multipoint virtual circuit, some rerouting of the virtual circuit is needed. In the example shown in Figure 4, adding a new output f, could be done by replacing e with a new internal node z and making e and f children of z. In general, some leaf in the tree is replaced with a new internal node and the old leaf and a new leaf (correponding to the new output) are made children of the new internal node. The only constraint on the selection of the leaf to be replaced is that we avoid increasing the depth of the tree unnecessarily. The only constraint on the choice of a new internal node is that it corresponds to a port with sufficient unused capacity on the recycling data path to accommodate the recycled virtual circuit. Dropping an endpoint is similar. The only special requirement is that we restructure the tree as necessary so that every internal node has two children. This can be accomplished by changing a single table entry.

To determine the switching network cost, we need to consider the impact of the recycling strategy on the total traffic in the network. A binary tree with r leaves and in which every internal node has two children, has exactly $r-1$ internal nodes. Hence, a multipoint virtual circuit with rate ω and r leaves places a total load of ωr on the outgoing links and generates a recycling load of $(r-2)\omega$. Thus, the recycling load never exceeds the exiting load. If σ is, then the total weight over all the output ports cannot exceed $2n/\sigma$ and there must always be a port at which the weight associated with recycling traffic is $\leq 1/\sigma$. if $\sigma \geq 3$, there is always some port that has sufficient unused capacity to act as a relay port for a multipoint virtual circuit. A more detailed analysis can reduce this further in many cases of practical interest, but for our purposes here, it is sufficient to know that the data rate required at the switching network ports is only a constant times the data rate of the external links. This means that if we use a Beneš network with per cell routing, the network complexity is

	network	memory	setup
recycling architecture	$O(n \log n)$	$O(mn)$	$O(1)$
broadcast packet switch 1	$O(n \log n)$	$O(mn^2)$	$O(n)$
broadcast packet switch 2	$O(n \log n \lg\lg n)$	$O(mn^{1+1/\lg\lg n})$	$O(f)$
per cell Beneš	$O(n \log n)$	$O(mn^2 \log n)$	$O(\log n)$
fixed path Beneš	$O(n^2(\log n))$	$O(mn \log n)$	$O(\log n)$
fixed path cascaded Beneš	$O(n(\log n)^2)$	$O(mn \log n)$	$O(f \log n)$
Lee 1	$O(n(\log n)^2)$	$O(mn^3)$	$O(f)$
Lee 2	$O(n(\log n)^2)$	$O(mn^2)$	$O(f)$

Figure 5: Complexity of Multipoint Switching Network Architectures

$O(n \log n)$.

The amount of memory needed in the lookup tables is determined by the same analysis used to bound the recycling traffic. Because the number of internal nodes in a binary tree is smaller than the number of outputs, the number of table entries used for a multipoint virtual circuit is bounded by its fanout, implying that the amount of memory needed to make the system nonblocking is $O(mn)$.

To keep cells in order, a resequencer is needed at the output of the Beneš network. Cells pass through the resequencer on every pass through the network. This raises one subtle issue when an endpoint is dropped from a virtual circuit. This reduces the number of passes that some cells take through the network by one, which means that cells passing over this virtual circuit immediately before and after the transition can become misordered if the transition is not handled properly. To ensure that cell ordering is maintained, cells taking the shorter path must be delayed for up to twice the usual amount of time, to give cells on the longer path time to catch up. A mechanism implementing this is described in [10]. From the system complexity standpoint, the main impact is to double the size of the resequencer over what would be required otherwise.

Hence the recycling approach results in a network that achieves optimal complexity in terms of the network, the amount of memory required and the number of words that must be changed to modify a multipoint virtual circuit. However, there is of course an added delay involved in the recycling architecture since cells must pass through the network up to $\lg F$ times, where F is the maximum fanout. For $F = n$, this results in an $O((\log n)^2)$ delay which is clearly not optimal. While in some contexts, this delay could be an issue, it is shown in [10] that for practical systems with $n = F = 2^{16}$, the worst-case delay associated with recycling be kept below 500 μs, making it smaller than the typcial delay in a digital time division circuit switch and completely negligible relative to the transmission delays associated with wide area network applications. It should also be noted that the recycling approach leads to neither a strictly or wide-sense nonblocking system, but rather a reroutably nonblocking system. A more detailed account of the recycling architecture, including a number of practical design refinements can be found in [10].

7. Conclusions

Figure 5 summarizes the results discussed in the previous sections. As we have seen, only the recycling architecture offers optimal complexity in all three dimensions. The first entry for the broadcast packet switch architecture refers to the original design and the second, to the improved version in which virtual circuits in different fanout classes are handled differently. Similarly, the second entry for Lee's architecture includes the refinements that reduce the amount of memory required.

This paper concentrates on asymptotic complexity, but it should be noted that there are no large constants lurking in any of these analyses. While with modest-sized systems (a few hundred ports), lower level design details can have a large impact on the relative merits of different architectures, we believe that for large switching systems (those with over 1000 ports), the asymptotic complexity is likely to dominate the cost equation.

References

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

[2] Bianchi, Giusseppe and Jonathan Turner. "Improved Queueing Analysis of Shared Buffer Switching Networks," *Proceedings of Infocom*, March 1993.

[3] Coudreuse, J. P. and M. Servel. "Prelude: An Asynchronous Time-Division Switched Network," *International Communications Conference*, 1987.

[4] Lee, Tony T. "Non-Blocking Copy Networks for Multipoint Packet Switching," *IEEE Journal on Selected Areas in Communications*, 1455–1467, 12/88.

[5] Melen, Riccardo and Jonathan S. Turner. "Nonblocking Multirate Networks," *SIAM Journal on Computing*, 4/89.

[6] Melen, Riccardo and Jonathan Turner. "Nonblocking Multirate Distribution Networks," *IEEE Transactions on Communications*, 2/93.

[7] Turner, Jonathan S, "Fluid Flow Loading Analysis of Packet Switching Networks," *Proceedings of the International Teletraffic Congress*, June 1988.

[8] Turner, Jonathan S., "Design of a Broadcast Packet Network," *IEEE Transactions on Communications*, June, 1988.

[9] Turner, Jonathan S., "Resequencing Cells in an ATM Switch," Technical Report WUCS-91-21, Department of Computer Science, Washington University, St. Louis, Missouri, 1991.

[10] Turner, Jonathan S., "An Optimal Nonblocking Multipoint Virtual Circuit Switch," Technical Report WUCS-93-30, Department of Computer Science, Washington University, St. Louis, Missouri, 1993.