NOW (NO WAITING) VIRTUAL CHANNEL ESTABLISHMENT IN ATM-LIKE NETWORKS

by

Robert T. Olsen
Lawrence H. Landweber

Computer Sciences Technical Report #1082

February 1992

NoW (No Waiting) Virtual Channel Establishment in ATM-like Networks

Robert T. Olsen
Lawrence H. Landweber

Computer Sciences Department
University of Wisconsin-Madison
Madison, WI 53706

Abstract

We describe a method for virtual channel establishment in ATM-like networks such that data may be sent over a virtual channel (VC) as it is being established. The NoW (No Waiting) VC establishment method eliminates the need for the source to wait one round trip time for confirmation that a VC has been established before data can be sent on the channel. Rather, a source can begin sending data immediately (or very soon) after it chooses the virtual channel identifier (VCI) it will use to represent the VC. We characterize ATM-like networks by three parameters; topology, size of VCI space, and expected VC pattern (creation rate and duration) between router (i.e., source/destination) pairs. We examine the performance of the NoW method on networks characterized by different values of these parameters, with emphasis on large-scale networks with realistic parameter values. We show that with the NoW method, VCs can usually be established with no special support on the part of routers or switches (i.e., intermediate nodes). We use *weak VCI collision* and *strong VCI collision* to denote conditions in which special support in the routers and switches is required to use the NoW method. We show the likelihood of occurrence of weak and strong VCI collisions is a function of the parameters characterizing the network. Thus, careful design of new ATM-like networks (i.e., appropriate parameter choices) can lead to increased performance.

1. Introduction

Future, high-performance, wide-area networks will carry a wide variety of traffic, including voice, video, and data. To facilitate the evolution to such networks, numerous broadband-ISDN protocols are being developed. It is likely that many future networks will use asynchronous transfer mode (ATM), which is tightly linked with broadband-ISDN. In ATM networks all information is transferred in 53 byte cells, the first 5 bytes of which contain a header. To transfer information from source A to destination B in an ATM network, a connection, or virtual channel (VC), must be established. All communication from A to B then follows the same path through the network, as defined by the VC. At intermediate nodes along the path, a virtual channel identifier (VCI) in each cell header identifies each cell as belonging to a particular channel and is used to route the cell along its fixed path.

This work was supported by the DARPA/NSF Gigabit Testbed program via CNRI and by the AT&T Bell Laboratories XUNET project.

There are two basic approaches which can be taken when establishing a virtual channel. In both cases the source initiates the establishment of the channel. In the first approach, the source does not begin sending data over the channel until it receives confirmation that a VC has been established. In the second approach, the source sends data at the same time, or shortly after, it initiates the establishment of the VC. The first approach has the obvious disadvantage of the source having to wait a minimum of one round trip time before sending data. If the distance between the source and destination is long and the capacity of the connecting link(s) is high, then a large amount of data could have been sent during that round trip time. In fact, if the source had been able to send data immediately, it may have even finished transferring its data in less than one round trip time. Also, with the first approach the source must keep track of which VCs are currently in the process of being established. The second approach to VC establishment eliminates the need to wait one round trip time before sending data, as well as the requirement that the source keep track of VCs currently in the establishment process. Furthermore, as link capacities increase in the future, the advantage of the second approach will grow, since, in many cases, the ratio of setup time to VC duration will become greater (e.g., fixed size file transfer). Since round trip latency cannot be reduced, the only way to substantially reduce VC setup time is to eliminate the need to wait one round trip time. The major disadvantage of the second approach is the added complexity (hardware and/or software) required on the part of the routers and switches to support the setup mechanism.

In this paper we describe the NoW method for VC establishment, which takes the second approach mentioned above. We consider mainly the establishment of VCs which carry traffic that does not have real-time performance requirements (e.g., throughput, delay, jitter). In the section on related work we compare the NoW method to a fast channel establishment method for real-time applications and discuss how the principles of the NoW method would apply to the fast establishment of VCs which guarantee a certain level of performance.

The goal of the NoW method is to *make the common case go fast*. The underlying principle of the NoW method is that, in general, an attempt to create a VC will succeed. If it is true that most

VC creation attempts succeed, then it is logical to optimize for that case (i.e., make the common case go fast). Waiting one round trip time to establish all VCs makes the performance of the common case equal to that of all other cases. The NoW method optimizes for the common case at the expense of potentially poorer performance for the uncommon case. The NoW method allows VCs to be established at the same time the first data cell traverses the network, typically with very little effort on the part of either the routers or switches within the network. Our method for VC establishment is simple and fast. The usefulness of the method is justified by the results of analysis and simulation, with parameters to the analysis and simulation derived from the results of a TCP traffic study. The percentage of VC establishments for which very little effort is required on the part of routers and/or switches is dependent on the parameters characterizing the network, namely, the topology, the size of the VCI space, and expected VC patterns between router pairs. We define the terms *weak VCI collision* and *strong VCI collision* to indicate conditions under which routers and switches must become more involved in the establishment process in order for it to complete successfully. Analysis and simulation show the probability of weak and strong VCI collisions within a router, given values for the parameters characterizing the network. Simulation shows the probability of weak and strong VCI collisions within a switch, given values for the parameters characterizing the network. Some parameters to the analysis and simulation are derived from measured TCP traffic characteristics between the University of Wisconsin-Madison and the rest of the Internet. The results of the analysis and simulation show the NoW method to perform very well (i.e., very low probability of weak and strong VCI collision) in ATM networks with characteristics analogous to those observed to currently exist in the Internet.

The second phase of the eXperimental University NETwork (XUNET) [Fras 91] will carry all traffic via ATM-like cells. At year-end 1991, hardware and software for routers (i.e., sources/destinations) and switches (i.e., intermediate nodes) was near completion. The UW-Madison is a part of the XUNET project and we intend to test the NoW approach for VC establishment within the XUNET environment. Such tests will help decide whether it is desirable to modify current router and switch VC setup mechanisms to better support the NoW method in XUNET.

Section 2 of this paper describes the XUNET environment in more detail, and in general discusses the type of environment for which the NoW method is suited. Section 3 provides more detailed motivation for why fast virtual channel establishment is important. Section 4 describes the details of the NoW VC establishment method. Section 5 contains the results of a TCP traffic study from which we derive parameters to simulations which show the performance of the NoW method within routers and switches. Section 6 contains simulation and analytical results which show how well the NoW method works within routers, under various assumptions for network topology, size of VCI space, and VC patterns between router pairs. Section 7 describes simulation results which show how well the NoW method works within switches, under various assumptions for network topology, size of VCI space, and VC patterns between router pairs. Section 8 discusses related work and Section 9 describes the direction of our future work and concludes.

2. Environment

There are two major hardware components in the XUNET network environment, routers and switches. Routers connect workstations on local area networks to the backbone network which carries ATM cells. Routers are able to support a total bandwidth of approximately 200 Mbits/sec. Each router will be connected to at least one switch. Switches route ATM cells through the network all the way to the destination router. Switches can support a total bandwidth of approximately 575 Mbits/sec. In order to support very high data transfer rates between hosts, a third component will be developed to allow computers equipped with a HiPPI interface to emit traffic into the ATM network at rates up to 800 Mbits/sec. Figure 1 gives the topology of the XUNET testbed as of year-end 1991. All of the links in the network operate at DS3 (44.736 Mbits/sec) transmission rates. In the future, some of the links will be updated to 600 Mbits/sec. We will test the NoW method within the XUNET testbed.

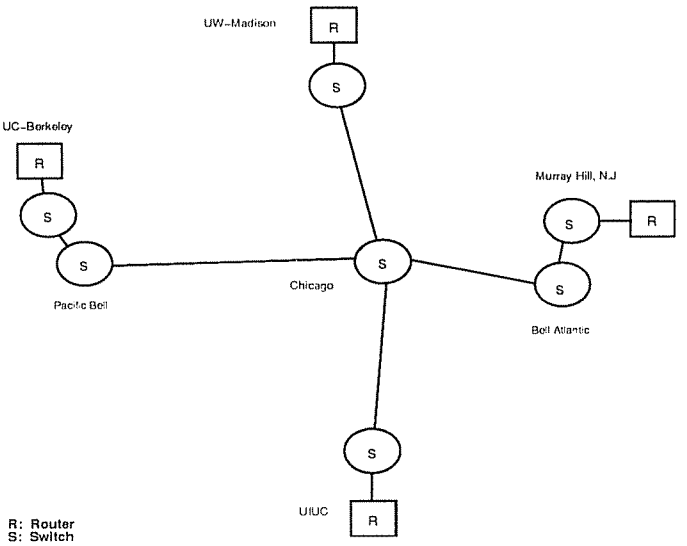


Figure 1. XUNET Topology 1991

It should be noted that the NoW method is not limited to the XUNET environment. Our method is designed for a general network architecture in which units of data identified by VCIs are carried between source and destination entities (e.g., routers). The router provides hosts, located on LANs, with access to a high performance ATM-based WAN. Intermediate nodes (e.g., switches) in the WAN provide for the routing of the data units.

When designing hardware for a network using virtual channels, one must consider the method that will be used to establish the channels and whether special hardware assistance is required to support the method. Designing hardware to use one particular method may make it difficult to use a different method. In this

paper we discuss the problems that network components must handle in order to use the NoW method for VC establishment.

3. Motivation

This section discusses the benefits of fast VC establishment Although it may seem intuitive that fast VC setup is beneficial, we provide explicit examples of how fast VC setup is useful in order to verify such intuition. We consider the implications of fast VC setup for sources, switches, and destinations. We consider sources/destinations as both hosts and routers.

3.1. Sources

There are several examples of how fast VC establishment can be useful to user-level applications. So called time-critical applications that send out alarms, or signals, which require immediate attention could certainly utilize a fast setup mechanism. Note, however, that any application which absolutely cannot tolerate any possibility of failed communication would likely have an established VC at all times, with reserved resources along the path, and thus would not need a fast setup mechanism. Applications which use remote procedure calls (RPC) can also benefit from fast VC setup. If the round trip time (RTT) between the caller and the callee is such that the RTT latency makes up most of the RPC time, then the time to perform an RPC is essentially cut in half when doing fast VC setup.

Some RPC packages use specially designed transport protocols which do not require a time-consuming connection setup phase (e.g., [BiNe 84]), such as the three-way handshake of TCP. If such special protocols run over ATM, then a fast VC establishment method is needed or the advantages of the fast transport layer setup mechanism are lost.

The existence of a fast VC establishment mechanism simplifies the tasks required of routers. Without fast VC setup, routers must queue all data sent from hosts until the VC is established. As line speeds increase, the opportunity cost of not sending data becomes larger and the amount of data to queue may also become larger. Such queueing could be avoided by having hosts block (i.e., not send data) until the router informs them that the VC is established. This does not solve the problem, however, since now the host operating system must perform the buffering, or the application must actually stop generating data. With fast VC setup, all queueing is avoided since the data can be sent without having to wait for one round trip time. In addition, the router need not necessarily keep track of those VCs which are currently in the process of being established. For example, if a pure datagram service is being provided, then there is no need for routers to keep track of those VCs which are currently in the process of being established. In the case of VC establishment failure, we assume a higher level protocol will retransmit any lost data. However, our method is designed to make the probability of such failure very low.

3.2. Switches

In ATM-based networks it is reasonable to perform buffer allocation and management within switches on a VC basis (i.e., each VC is allocated some amount of buffer space). This section contains the results of a simulation which measures the effects of fast VC setup in switches, given a particular buffer management strategy. We assume a buffer management strategy which allocates one full round trip window size to each active VC at each switch. A VC establishment request is rejected if there is insufficient buffer space at a switch. Clearly this is a conservative strategy and we are not advocating it as the best method. It can, however, guarantee no loss of data within the network and is simple to simulate. Note also that less conservative methods will still exhibit the type of behavior described below. It simply requires a higher load (i.e., more VCs) before such behavior sets in.

The details of the simulation are as follows. A single switch is simulated, with VC establishment requests (and data) arriving at the switch over one of multiple lines. The exact number of lines which a switch can support is dependent on the buffer space within the switch, which is a parameter to the simulation (i.e., the number of lines a switch can support is equal to the buffer space size divided by the round trip window size). It is assumed that if a VC has data to send and there is an available (i.e., unused) line, then the data flows over that line. This assumption leads to maximum utilization of switch resources for a given establishment request arrival rate. Buffer resources are allocated to a VC for the entire time it takes to perform VC setup and data transfer. Immediately upon completion of setup, each VC sends 1 megabyte of data. Upon completion of the sending of the data, each VC is immediately torn down (i.e., do not wait one round trip time for teardown). The transmission rates of the lines are the same as the operating rate of the switch, which is a parameter to the simulation.

Tables 1 and 2 contain probabilities of rejecting VC establishment requests within a switch due to lack of buffer space. The numbers in Table 1 were obtained from a simulation of a switch operating at 1 gigabit/second. Table 2 contains numbers from a simulation of a switch operating at 10 gigabits/second. Variable parameters in the simulation were the arrival rate of VC establishment requests, the amount of buffer space in the switch, and the total VC setup time (i.e., time until source router can start sending data). The source and destination routers for the VCs were assumed to be on opposite sides of North America. Thus the 50ms VC setup time seen in both tables represents a best case time for any VC setup method requiring confirmation of establishment.

There are several points to be made with respect to the data in Tables 1 and 2. From Table 1 it can be seen that a reduction in the VC setup time can be accompanied by a reduction in the amount of buffer space without a resulting increase in the probability of rejecting a VC establishment request. In Table 1, a reduction in VC establishment time from 50ms to 10ms was accompanied by a reduction in switch buffer space from 60MB to 36MB, with the probability of VC establishment request rejection remaining approximately stable. Also in Table 1, it can be seen that at lower

arrival rates, the probability of rejection is slightly higher at a lower setup time and smaller buffer size. At higher arrival rates, the probability of rejection becomes lower at a lower setup time and smaller buffer size. This is because at lower arrival rates there are fewer VCs simultaneously in the setup phase, making the time to perform setup less important. That is, it is unlikely that an establishment request will be rejected due to a large number of requests currently in the establishment phase. At higher arrival rates, more VCs are simultaneously in the setup phase, making a low setup time more important. In Table 2, which has a higher data transfer rate, a similar pattern holds. However, at the higher rate, an even larger reduction in buffer space size is possible. With a decrease in setup time from 50ms to 10ms, a decrease in buffer size from 420MB to 180MB (57% reduction) resulted in approximately the same or lower rejection probabilities. In Table 1 only a 40% reduction in buffer size was possible, (any further reduction resulted in rejection probabilities higher than with the slower setup time). In Table 2, a larger reduction was possible and even better performance in terms of rejection probabilities resulted. In other words, as line speeds increase, so do the advantages of fast VC establishment. The results of Tables 1 and 2 are not surprising and should be intuitive. They simply point out that as line speeds increase, the cost of setup becomes more important.

Another point to notice from Table 1 is that simply increasing the amount of buffer space within a switch is not always an acceptable solution to the problem of high rejection probability. At an arrival rate of 125 requests/second the probability of rejection is nearly 1. Increasing the buffer space will reduce this probability by allowing more connections to exist simultaneously. However, the reason the probability of rejection was so high was because the switch was fully utilized. Accepting more VCs will simply lead to longer queues and higher response times. In queueing theory terminology, the queue has become *unstable*.

	Buffer Space - VC Setup Time	
Arrivals/sec	**60MB - 50ms**	**36MB - 10ms**
50	0.003	0.008
75	0.069	0.074
100	0.396	0.327
125	0.999	0.976

Table 1. VC Establishment Rejection: 1 Gb/sec

3.3. Destinations

The same types of problems which occur at switches could also occur at destinations (both hosts and routers) if buffer space is allocated on a VC basis. Thus the results of the previous section may apply to destinations as well as switches. In some hosts (or servers), however, buffer space may not be a critical resource, but the VCI space may be. In such cases, if the server sits idle, waiting for the establishment of VCs to complete which were rejected during a busy period (i.e., no available VCIs), then fast VC setup would increase throughput at the server by eliminating such idle time.

	Buffer Space - VC Setup Time	
Arrivals/sec	**420MB-50ms**	**180MB-10ms**
50	0.015	0.018
75	0.099	0.050
100	0.323	0.106
125	0.719	0.179

Table 2. VC Establishment Rejection: 10 Gb/sec

4. Virtual Channel Management

The establishment of a virtual channel involves every hardware component which will take part in transferring data between the source and destination routers. Decisions regarding channel establishment need to be made at every hop along the path. In this section we describe the details of NoW VC establishment. We discuss the problems which can arise when using the NoW method and propose various solutions. We also discuss virtual channel teardown. Throughout this section the terminology used refers to the XUNET environment but all discussion is applicable to the general network environment described earlier in Section 2.

4.1. NoW VC Establishment

The primary weakness of the usual VC establishment method is that the source router must wait a minimum of one round trip time before it can begin sending data. In that time the router and/or host OS may have to buffer data being sent by the source application. It would be nice if such data could be sent to the destination immediately. The NoW VC establishment method allows the router to do just that. The crucial observation which led to the design of the NoW VC establishment method is that, in general, it is likely there are available VCIs within each switch, and thus there is no need for a source router to have to wait to see if there are VCIs available at each hop along the path. Analysis and simulation described later show this to be the case for realistic values of VCI space size and VC patterns (i.e., duration and creation rate). We also believe that in the common case other types of resources (e.g., buffers) will also be available, so that again there is no need for the source router to wait to see if that is indeed the case. For the moment, however, we will ignore issues relating to buffer space availability within switches. We discuss issues related to channels requiring guaranteed performance in a later section. Congestion control mechanisms can help manage buffer resources when guaranteed performance is not required.

Assuming our claim is true that in the common case there is no need for switches to perform a time consuming search to find a unique outgoing VCI to represent a newly created virtual channel, another problem must still be solved before fast VC establishment can be performed. Even if the switch can be relieved of locating a

unique outgoing VCI, it still must perform a lookup in a routing table to determine the appropriate outgoing line for the virtual channel. In the following description of our method we assume a static routing algorithm, which solves this problem. Once the source router chooses the VCI for a VC, the route to the destination is essentially fixed.

There are two critical aspects to NoW virtual channel establishment.

1) The identity of the destination router is encoded in each VCI representing a virtual channel. Note that this reduces the total number of bits available to actually represent the VCI.
2) Expectations of VC creation rate and VC duration are calculated for all pairs of routers. That is, each router computes an expectation of how often it will create VCs to every other router and how long each VC will last. The expectations are based on anticipated usage patterns provided by users, as well as observations of past VC patterns.

If we assume a static routing algorithm, then encoding the destination router identity within the VCI eliminates the need for switches to do a routing table lookup at the moment the VC is established. Rather, every outgoing VCI will automatically have a single outgoing line associated with it. Namely, that line which is used to reach the destination encoded within the VCI.

The expectations of VC establishment patterns are used for two purposes. First, to determine the number of VCIs which each router can use to create VCs to each destination. And second, within each switch, to partition the VCI space associated with each destination among all incoming lines which establish VCs to that destination.

The first use of the VC establishment pattern works as follows. Every router has some expectation of how often it will create VCs to every other router, and of how long each VC will last. Three factors help determine how many VCIs each router can use to create VCs to a given destination: 1) the size of the VCI space (excluding bits used to encode destination), 2) the product of the router's expected VC creation rate and VC duration for the given destination, and 3) the sum of the products of creation rate and duration for the given destination over all incoming lines at the point in which the router begins sharing the VCI space with other lines, (these incoming lines are the *Numlines* in the equation below). Figure 2 provides an example. Assume the size of the VCI space (excluding bits used to encode the destination) is 256. Suppose router 0 (R0) expects to establish VCs to R1 at the rate of 1/sec with an expected duration of 4 seconds. In addition to the link from R0, switch 1 (S1) has two other incoming links which expect to establish VCs to R1 at rates of 2/sec and 0.5/sec with durations of 2.5 seconds and 14 seconds. In general, the number of VCIs which a source router R_s can use to establish VCs to a destination router R_d is computed as

$$VCI_{Rs,Rd} = \frac{rate_{Rs,Rd} * duration_{Rs,Rd}}{\sum_{i=1}^{Numlines} rate_{Li,Rd} * duration_{Li,Rd}} * VCI_{totalsize}$$

Thus $VCI_{R0,R1} = \frac{4}{4+5+7} * 256 = 64.$

The reason for limiting the number of VCIs R0 can use to establish VCs to R1 is because, as will be seen shortly, S1 partitions its outgoing VCI space associated with destination R1 based on the same formula given above. Thus, in S1, R0 will only have 64 VCIs to represent its VCs to R1 over link 4 (L4). So there is no advantage in having R0 use more than 64 different VCIs over L1. By limiting R0 to 64 VCIs, however, if a VCI is available within the pool of 64, then R0 knows the VC establishment request will traverse at least through S1 without being rejected. If the destination router is directly connected to S1, R0 knows the VC establishment request *cannot* be rejected. Clearly, if all VCIs in the pool of 64 are in use, a problem has arisen. Such problems, and solutions to the problems, are discussed in the following subsection.

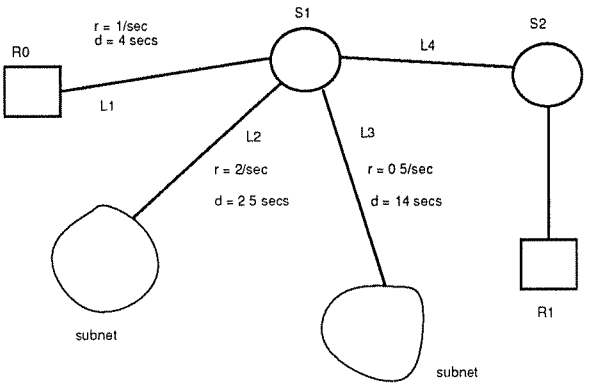


Figure 2

The second use of the VC establishment pattern, to partition the VCI space within each switch among incoming lines, works as follows. Since each router has an expectation of a creation rate and duration for VC establishment to each destination, it is possible to compute the same expectations for all lines entering a switch. Expected creation rates (for a given destination) over links into switches can be computed by summing over all creation rates from routers which utilize the link. Expected VC duration (for a given destination) over a link can be computed by taking the weighted average of all durations from routers utilizing the link. For example, suppose we want to partition the outgoing VCI space within S1 that is used to reach destination R1. Using the creation rates and durations from Figure 2 and a formula essentially the same as the one used above, the outgoing VCI space associated with destination R1 can be partitioned within S1 as follows.

$$VCI_{(S1,L1),R1} = \frac{4}{4+5+7} * 256 = 64,$$
$$VCI_{(S1,L2),R1} = \frac{5}{4+5+7} * 256 = 80,$$
$$VCI_{(S1,L3),R1} = \frac{7}{4+5+7} * 256 = 112$$

Note that this should be a true partition. None of the outgoing VCIs assigned for use by one incoming link can be used by any other incoming links. This is unlike the allocation of VCIs to a router. The router may use any values for the VCIs but should not use more than the allocated amount. The switches are restricted in their choice of outgoing VCIs not only in the number of VCIs they can use but also in the values which may be chosen. This is to insure that no two separate VCs (from different incoming links) will simultaneously use the same outgoing VCI over the same link to the same destination.

One can now see the implication of encoding the destination router ID in the VCI and using the expected VC establishment patterns to restrict routers and switches to using only certain portions of the VCI space for channels to each destination. These two conditions allow a special *initialization* of VCI map tables within all switches and routers. This initialization pre-establishes a large number of virtual channels, thereby eliminating the need to perform any type of computation at setup time, thus making VC establishment very fast. For example, in Figure 2, within S1 all VCI map table entries with *in slot* of 1 and *in VCI* indicating destination R1 would be initialized to contain *out slot* of 4 and *out VCI* equal to one of the 64 VCIs in the partition associated with L1. Thus when R0 wishes to establish a VC to R1, it simply locates a VCI from within the pool of VCIs it has been assigned for communication with R1. At this point R0 knows that appropriate mappings exist in S1 which will route cells originally labeled with any VCI chosen from the pool, to R1. S2 will have performed similar calculations and will also have appropriate mappings. *However, even though the mappings exist in the switches, it may be that some of them are currently in use by other VCs.* That is, the mapping from incoming to outgoing VCIs is not necessarily one-to-one. Two or more incoming VCIs may map to the same outgoing VCI. This is a key problem, but is not the concern of the router. Rather, it is the concern of the switch and is discussed in more detail shortly. As far as R0 is concerned, however, it can immediately begin sending data to R1 using the chosen VCI. The only special setup task that R0 must perform is to place the source and destination host addresses within a special first cell, or the first few bytes of the first data cell. This information is needed by the destination router to fill its VCI map table appropriately.

Another way to think about how the NoW method works is to consider that the initialization of the VCI map tables within the switches and routers essentially establishes many **one hop VCs**. A one hop VC is a VC between two adjacent nodes (e.g., two switches or a router and a switch). In addition, the one hop VC can only be used to reach a single destination. A full VC from source to destination router can then be *patched* together using multiple one hop VCs. Such patching is done automatically due to the special initialization of the VCI map tables. Problems occur when two distinct full VCs wish to use the same one hop VC at the same time. We denote such occurrences as **VCI collisions** and discuss them more fully in the next section.

4.2. Consequences of Partitioning in the NoW Method

The major problem which must be solved when using the NoW VC establishment method is the handling of **VCI collisions**. There are two types of VCI collisions which can occur in both routers and switches, weak VCI collisions, and strong VCI collisions. A **weak VCI collision** occurs during VC establishment when the VCI which is to be allocated to the new VC, by a router or switch, is already allocated to a different VC. A **strong VCI collision** occurs during VC establishment when, after a weak VCI collision, a search of the remaining relevant VCI partition reveals all VCIs are currently in use. The remainder of this subsection describes the details of how VCI collisions may occur in routers and switches and some solutions for avoiding them and detecting and correcting them.

As described earlier, each router has a pool of VCIs associated with each destination with which to establish VCs to that destination. The pool can be logically organized as a circular queue with a pointer indicating which VCI is to be allocated next. The pointer is bumped one entry on each allocation and traverses the queue in a circular manner. A weak VCI collision occurs in the router at VC establishment time when the VCI to be allocated (i.e., currently pointed to) is still in use from the last time it was allocated. A strong VCI collision occurs when all entries are still in use when a new establishment request arrives.

VCI collisions within switches are not as straightforward as those within routers. Consider the topology of Figure 3 and assume a VCI space (excluding bits for destination) of size 32. Suppose R0, R1, and R2 all have the same expected VC creation rate and duration parameters associated with destination R3. Then, under the NoW method, the VCI pool sizes associated with R3 within R0, R1, and R2 will be 16, 16, and 11, respectively. At S2, L3 will be allocated 21 VCIs for VCs going out over L5 and L4 will be allocated 11 VCIs going out over L5. Since there are always a total of 32 possible incoming VCIs from another switch, this means that at S2, 32 incoming VCIs over L3 must map to 21 outgoing VCIs over L5. That is, within S2, the size of the incoming VCI space on L3 is larger than the corresponding outgoing VCI space on L5. Thus multiple incoming VCIs on L3 must map to the same outgoing VCI over L5. In this case R0 and R1 cannot be sure that a VCI is available at S2 just because a VCI is available in its local pool. Routers must rely on the intermediate switches to make adjustments when necessary, since two different incoming VCIs have pre-established mappings to the same outgoing VCI. A weak VCI collision occurs at S2 if the first cell associated with a VC arrives marked with an incoming VCI which maps to an outgoing VCI which is already in use by another channel. The switch must detect such situations and either deny the establishment of the channel, or find a new, currently unused VCI to represent the channel. If there are no other available VCIs within the partition, then a strong VCI collision occurs and the VC establishment request may have to be rejected. Such detection and correction of VCI collisions is one example of special support required of switches in order to support the NoW VC establishment method.

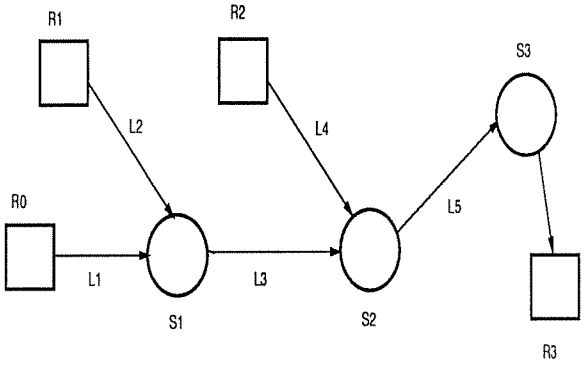


Figure 3

There are two principle methods we propose for handling VCI collisions within routers and switches, avoidance, and detection and correction. We first propose techniques for avoiding VCI collisions. We then discuss the detection and correction of VCI collisions, a task which must occur when the avoidance techniques fail.

4.2.1. Avoidance of VCI Collisions

The size of the VCI partition and the utilization (i.e., creation rate-duration product) of the VCIs within a given partition are two key parameters determining the probability of VCI collisions. The larger the size of the VCI space and the lower the product of creation rate and duration, the smaller the probability of VCI collision. This section describe techniques to keep the usable size of the VCI space as large as possible and the product of VC creation rate and duration as low as possible.

In ATM networks there are a fixed number of bits in the ATM cell header allocated for the VCI field. In the XUNET environment this field length is 16 bits. In the NoW method some number of those 16 bits is used to encode the destination router ID, thereby decreasing the actual number of available, or usable, VCIs. Consider the NSFNET T1 backbone network (see Figure 4) and all *regional networks* (e.g., CICNET, NYSERNET) within the United States as a topology in which the NoW method is to be used. There are approximately 25 regional networks hanging off the NSFNET backbone, with an average of 36 sites per regional network. Thus there are roughly 900 destination sites, implying 10 bits of the VCI field would be required to represent the destination ID and 6 bits would be left for the actual VCI. A single VC would be used for channels between all router pairs. The performance of the NoW method in such an environment is shown in Sections 6 and 7.

However, in reality, it is likely that datapaths spanning large distances will consist of multiple VCs spanning multiple backbones. That is, to send data between regional networks which themselves are backbones, a series of VCs will need to be established. The first VC will extend to the edge of the source backbone. One or more VCs will carry the data through intermediate backbones and the final VC will exist solely within the destination backbone. Such a concatenation of VCs will likely be necessary due to the heterogeneity of regional networks. One consequence of such an environment is that within a single backbone a small number of bits are required to represent destinations. The average number of sites in the regional networks hanging off the NSFNET backbone is 36. The number of nodes in the NSFNET T1 network (Figure 4) is 14. Under the NoW method, (assuming a 16 bit total VCI space), this leaves 10 bits for the actual VCI within regional networks and 12 bits within the NSFNET backbone. This is in comparison to only 6 bits when all 900 destinations must have a unique ID. The advantage of having more bits to represent the actual VCI is that the probability of VCI collision within a single backbone is much lower. A disadvantage of concatenating VCs is that those nodes which transfer data between backbones must perform the tasks associated with establishing a VC. Essentially, such a node must now play the role of a router as well as its traditional role of a switch.

When using the NoW method, the concatenation of VCs has another disadvantage. When establishing a series of VCs, the first VC in the series is addressed to a destination on the same regional network as the source router. This raises the possibility that the total number of simultaneous connections a router can have with non-local destinations is lower than if concatenation of VCs was not done (i.e., each destination on the entire network had a unique ID). The total number of simultaneous connections that a router can have open with non-local destinations is dependent on the size of VCI space within the router's regional network and the number of gateways through which traffic to non-local destinations may flow. If there is only one gateway with a single ID associated with it, then the number of possible simultaneous connections to non-local destinations is lower than if each destination had some number of VCIs associated with it. However, a tradeoff occurs in that the entire pool of VCIs used for the first VC in a series of VCs destined for a non-local destination, is now shared among all these destinations. Thus a single destination is not limited to some fixed number of VCIs. A better utilization of VCIs is likely to occur (and potentially a lower probability of VCI collision) since destinations which see a small number of VCs do not have a dedicated number of VCIs allocated for them. In addition, the smaller number of VCIs available to routers (i.e., smaller number of simultaneous connections to non-local destinations) is not likely to be a problem since the total number is still very large. In the example above the size would be at least 1024 (i.e., 10 bits, one gateway) and even higher if there are more gateways or less than 64 sites in the regional network (so gateways can have multiple IDs).

Within nodes of intermediate backbones, the concatenation of VCs has a similar effect. A pool of VCIs within a node is associated with multiple destinations, rather than each destination having a fixed size number of VCIs associated with it. Again, the advantages of such a shared pool are better utilization of the VCI space and lower probability of VCI collision. The size of this shared pool, relative to the number of VCIs available when concatenation of VCs is not performed, is dependent on the size and topology of the various backbones in the network. If we assume a wide-area backbone with 25 nodes and a regional network back-

bone of size 32 hanging off each wide-area backbone node, then the number of VCIs available at a wide-area node for all destinations within a given regional network is the same whether or not concatenation of VCs is performed. Without concatenation, there are 2048 (64 VCIs/destination * 32 destinations) available VCIs. There are 64 VCIs/destination since 10 bits are required to encode the 800 destinations. With concatenation, there are also 2048 VCIs available for each wide-area backbone destination node (since 5 bits encode the destination).

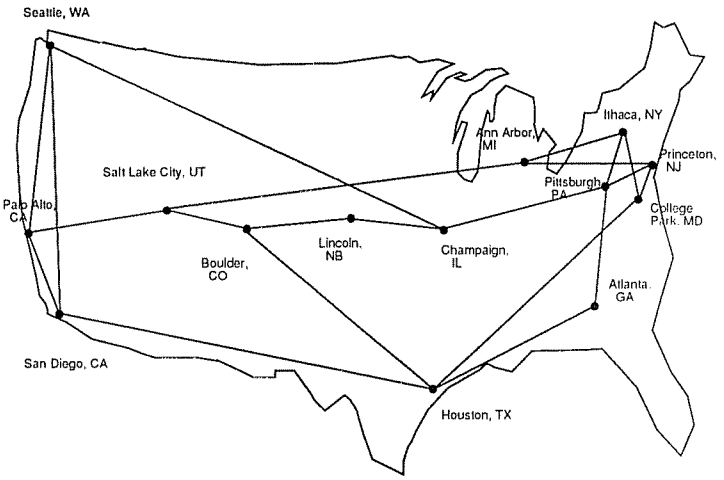


Figure 4. NSFNET T1 Network 1991

Although is it likely the concatenation of VCs will necessarily occur due to the heterogeneity of networks, it is possible to impose a *logical* structure upon a homogeneous network so that such concatenation of VCs must occur. By carefully designing the structure of the network (i.e., partitioning the network into backbones which have a high degree of intra-backbone traffic), the probability of VCI collision can be reduced. As long as the cost of establishing a series of VCs is not too high, such a technique is desirable as a means to avoid VCI collisions. The technique becomes more desirable as the amount of inter-backbone traffic decreases. Careful design of the network structure can help minimize the amount of this traffic.

The technique described above is intended to provide a larger VCI space and thus reduce the probability of VCI collision. There are other factors which effect this probability as well. Not surprisingly, given a fixed size VCI partition, a higher product of the VC creation rate and duration results in higher probability of VCI collision. Thus minimizing that product will help reduce the probability of collision. As network transmission rates increase, the duration of VCs of bulk transfer applications (e.g., file transfer) will decrease, thus decreasing the product of creation rate and duration. This implies a larger creation rate can be tolerated while still keeping the same probability of collision. This suggests that for applications which send periodic bursts of data it may be better to establish a separate VC for each burst so that each VC has short duration, rather than establish a single VC which has a long duration. As line speeds increase, the advantage of establishing many VCs of short duration also increases. If the VCs can be established fast, as when using the NoW method, the establishment of many VCs of short duration becomes more attractive.

The techniques mentioned above are intended to make the probability of VCI collision as low as possible. They cannot, however, eliminate the possibility. However, there is more that can be done to avoid VCI collisions. Switches and routers can monitor the number of VCIs which are currently in use within each partition. Within switches, if one partition has experienced a large number of weak VCI collisions, then a dynamic repartitioning of the VCI space could be performed as long as the partitions which will shrink are not experiencing a similar condition. This technique need not involve any other switches or routers although it may be desirable to notify adjacent nodes when such action occurs so that they may take similar action (e.g., a switch telling a router it may increase its partition size). With this technique, VCI collision is avoided by allocating more resources at times when it appears the current amount is insufficient. The action of repartitioning the VCI space for a given destination will likely not occur often and does not have to be particularly fast since it can be triggered in advance of when total failure (i.e., strong VCI collision) is expected to occur. Thus the mechanism used to implement this technique need not be directly in switch hardware. We intend to implement this technique of collision avoidance within XUNET switches with the help of a special control computer associated with each switch. This control computer is connected to the switch via Ethernet and can thus be used to perform only those tasks which do not require very fast execution.

4.2.2. Detection and Correction of VCI Collisions

When all collision avoidance techniques fail, there is nothing left to do but detect the collision and correct it. Within routers, detection and correction of a weak VCI collision is trivial. It simply involves finding an unused VCI from within the partition. If a strong VCI collision occurs within a router, then a VCI may have to be used which does not contain the properly encoded destination identification. At this point VC establishment must be performed using the general method of waiting one round trip time before sending data.

Detection and correction of VCI collisions within switches is the primary mechanism which must be supported in the switch for the NoW method to operate efficiently. The handling of weak VCI collisions should be such that no other node need be involved. One entry in the VCI map table can simply be rewritten. So that no additional buffering of data is required, detection and correction must be performed extremely quickly. The control computer associated with the XUNET switch will be unable to perform such tasks fast enough. However, due to the high degree of programmability in the XUNET switch hardware, we intend to handle weak VCI collisions directly within the switch in a fast and efficient manner. Strong VCI collisions present a greater problem. Communication with the next node along the path is required since a VCI with the improper destination may have to be used. Such communication takes considerable time, during which extra buffering mechanisms

and/or flow control would be required. Alternatively, the VC establishment request could simply be denied, resulting in the source router attempting a re-establishment using the general method of waiting one round trip time before sending data. Since the probability of strong VCI collision is expected to be very low, however, the poor performance obtained in such cases is considered acceptable. Sections 6 and 7 show the probability of both strong and weak VCI collisions in routers and switches, under various assumptions for the size of the VCI space and VC pattern.

4.3. VC Teardown

Clearly, in a network which uses virtual channels which must be established prior to use, there must be a mechanism by which channels are torn down after use, so the VCIs used to represent the channel can be reused. Since VCs are pre-established in the NoW VC establishment scheme, the setup consists of simply marking the switch VC mapping table entry as in use. Likewise, teardown consists of marking the switch VC mapping table entry as not in use. The XUNET environment will have a mechanism for tearing down virtual channels. Initially, virtual channels established with the NoW scheme will utilize this mechanism. It is not yet clear what the exact mechanism will be. One possible way to tear down a VC is to have the source host explicitly indicate to the source router that the VC is no longer needed. Another possibility is to have the source router detect activity on each VC and tear down those VCs which are inactive for a certain period of time. A third possibility would be for switches to detect inactivity over VCs and initiate the teardown procedure. In all cases a special dedicated channel is likely to be needed to carry the teardown message.

It could be argued that if waiting one round trip time to perform VC setup is undesirable, then waiting one round trip time to perform teardown is also undesirable. It is preferable to free up resources immediately after use, rather than wait one round trip time to teardown a VC and release the resources. We believe this to be true and are considering how to make fast teardown a part of the NoW method.

5. TCP Traffic Measurements

In order to gain intuition as to what types of VC patterns may occur in ATM-based WANs, a traffic study was conducted which investigated the characteristics of TCP connections established between UW-Madison and all non-local sites on the Internet. Of particular interest was the rate of establishment and the duration of such connections. The results of the study were used for two purposes. First, as parameters to analytical and simulated models of routers and switches used to evaluate the performance of the NoW method in a large-scale network. And second, to confirm our belief that VC establishment request interarrival times and VC durations are approximately exponentially distributed.

5.1. Methodology

Raw data for the study was collected via the *tcpdump* utility. This utility allows the collection and timestamp of all TCP headers with the **SYN** or **FIN** flag set which originate from or are destined to a non-local site. We collected such data for three 12 hour periods, 6 a.m. - 6 p.m., August 8, 9, 10, 1991. By matching **SYN** and **FIN** flags we could determine the number of connections within each period, the duration of each one, and the identity of the other end-point of the connection. During the three 12 hour periods we saw a total of 8785 connections established with UW-Madison as the initiator, and 6860 connections established with a non-local site as the initiator.

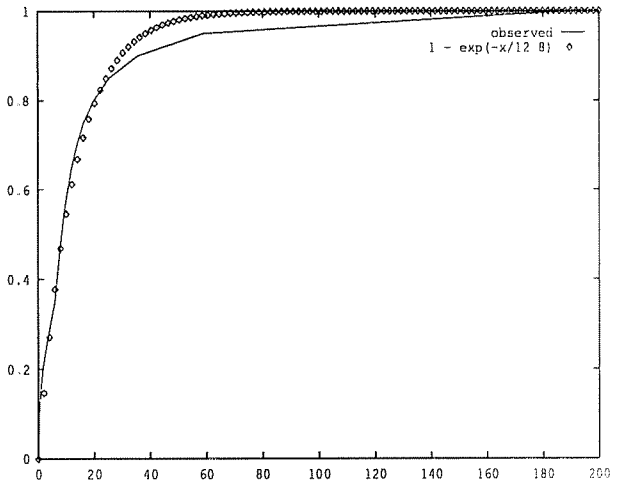
To derive parameters for use in simulation, a number of assumptions were made about the underlying network from which measurements were gathered. We now describe the model of a large-scale WAN which was assumed. We used the NSFNET T1 backbone network (see Figure 4) and all regional networks within the United States as our topology. There are approximately 25 regional networks hanging off the NSFNET backbone, with an average of 36 sites per regional network. Thus the model under consideration has roughly 900 destination sites. In reality, for administrative purposes each site within a regional network is associated with a primary autonomous system (AS). We assumed each autonomous system was associated with a single backbone node and all traffic from the autonomous system to UW-Madison went through the associated backbone node. Traffic between backbone nodes was assumed to be routed on a shortest number of hops basis.

5.2. Results

To determine the performance of the NoW method within a switch (i.e., backbone node) in the model described above, it is necessary to know the arrival rate of connection requests to a particular destination into the switch, and the expected duration of the connections. We computed the arrival rate of connection requests to the computer science department at UW-Madison at each switch by first calculating the average arrival rate from each autonomous system to UW-Madison from the raw data collected. Then, by using our two assumptions (i.e., each AS is associated with a single backbone node and routing between backbone nodes is by shortest number of hops) we were able to sum arrival rates of links which combine to flow over another link. We assumed that the backbone node at Champaign, Illinois was the primary node associated with UW-Madison and that all traffic to UW-Madison flowed through that node. The observed arrival rate (computed as the average of the arrival rates for the three periods) on the link from Seattle to Champaign was 0.0091 connections/second. Arrival rates on the links from Lincoln to Champaign and Pittsburgh to Champaign were 0.005 and 0.0305 connections/second, respectively. In addition, there was an arrival rate of 0.0084 connections/second from sites which were assumed to connect to the backbone directly at Champaign. Thus the total average arrival rate of connection requests to Wisconsin from the rest of the network was .053 connections/second.

We also computed the expected duration of connections to UW-Madison on an AS basis. Due to the variety of applications which utilize TCP (e.g., *finger*, *ftp*, *rlogin*), the observed connection durations varied substantially within a given AS. Across ASs the

average duration also varied greatly, from less than 5 seconds to over 500 seconds. The average duration over all conversations for the three periods was 72.5 seconds. Almost 95% of all connections were less than 200 seconds. The cumulative distribution function characterizing the lowest 95% connection duration times is shown in Figure 5, together with the cumulative distribution function of an exponential distribution with expected value of 12.8. The two functions are very similar, particularly up to a probability of around 0.85. The exponential distribution in Figure 5 is used as an estimate of duration time for the simulations described in Section 7. We feel it is reasonable to exclude the 5% of the connections with the highest durations since in an ATM environment such connections at the application level could easily consist of multiple channels of shorter duration at the ATM level. In applications which have long periods of idle time this does not present any problems. We observed several very long *rlogin* sessions, including some in excess of 8 hours. It is likely there were several periods of inactivity over this period and thus the connection could have been broken into several connections of shorter duration.



Seconds

Figure 5. Distribution of TCP Connection Duration

To determine the performance of the NoW method within routers, we computed the arrival rate of connection requests *initiated* at UW-Madison to all destinations, where destinations are actual sites, not autonomous systems. This is in contrast to the arrival rates described above, which were for connections *destined* for UW-Madison (i.e., initiated outside of UW-Madison). We also computed the connection duration times for such connections. Table 3 gives the average number of connections (in a 12 hour period), the average connection creation rate (per second), and the average connection duration (seconds), for the only three sites which appeared in the top ten for all three observed periods. The University of Illinois was the top network by a significant margin in each period, mainly due to the fact that periodic (every 30 minutes) weather service updates are obtained from that site. The performance of the NoW method under the traffic patterns appearing in Table 3 is shown in Section 6.

Destination	Connections	Duration	Creation Rate
UIUC	349	7.1	.0081
UC-Berkeley	47	44.6	.0011
Rutgers	44	80	.0010

Table 3. Top Sites for UW-Madison Initiated Connections

5.3. Duration and Interarrival Time Distributions

As mentioned earlier, one of the purposes of the traffic study was to determine if the distributions of VC duration and VC establishment request interarrival time (for a fixed source/destination router pair) are approximately exponential. Two tests were performed on the data concerning connections from UW-Madison to UCB and Rutgers. The highest 2% and 6% of the duration times were removed from the Rutgers data and UCB data, respectively, before the tests were performed. This removal was for the same reason stated in the previous section (i.e., connections with long durations are not relevant since they can easily be broken into several connections of shorter duration). We did not test the UIUC data since the large number of connections with the weather server were at regular intervals and of very short duration, characteristics which made connections to that site very unlike those to all other sites. We calculated the coefficient of variance (CV) and the Kolmogorov-Smirnov (KS) statistic for the VC duration and interarrival time data. The coefficient of variance for an exponential distribution is 1. Thus the CV of a data sample is a measure of its deviation from the exponential distribution. The KS statistic is calculated using the Kolmogorov-Smirnov test, a goodness-of-fit test used to determine the likelihood that a data sample has come from a particular distribution. The result of the KS test is the acceptance or rejection of the hypothesis, at some level of significance (i.e., the probability the hypothesis is rejected even though it is true), that the observed data is exponentially distributed. The results of the tests appear in Tables 4 and 5. The first value in the columns labeled Kolmogorov-Smirnov is the mean value which gives the highest significance level for which the hypothesis that the data is exponentially distributed is accepted. That significance level follows the mean. For a comparison to the results, consider Figure 5. The data plotted there appears to be approximately exponentially distributed. That data had a coefficient of variance of 1.68, but the KS statistic led to the rejection of the hypothesis at a 1% level of significance. Thus the results of Tables 4 and 5 suggest the data is approximately exponentially distributed, (at least as much as the data plotted in Figure 5). Based on these results, we feel comfortable that it is reasonable to make the assumptions that VC establishment requests arrive according to a poisson distribution (i.e., interarrival times are exponentially distributed) and VC durations obey an exponential distribution. Indeed, we make such assumptions in the following

two sections.

Destination	Coefficient of Var	Kolmogorov-Smirnov
UC-Berkeley	1.87	5.9/.20
Rutgers	1.6	23.5/.20

Table 4. Duration Distribution

Destination	Coefficient of Var	Kolmogorov-Smirnov
UC-Berkeley	1.71	rejected
Rutgers	1.02	828/.01

Table 5. Interarrival Time Distribution

6. NoW Performance within Routers

This section provides analytical and simulation results which show the likelihood of weak and strong VCI collisions within a router, given assumptions about the size of the VCI space and the expected VC creation rate and duration to some destination router. Recall that within routers, VCI collisions occur within a pool of VCIs associated with a given destination.

6.1. Analysis

6.1.1. Weak VCI Collisions in Routers

The following assumptions apply to the analysis. The previous section addressed assumptions 1 and 2. Assumption 3 simplifies the analysis. However, the results of the analysis (i.e., probability of 0 weak VCI collisions within a partition) are the same whether or not assumption 3 is made.

1) The expected number of VC creation requests made to a router for a given destination in an interval of length r is of the form λr, the expected value of a poisson distribution.

2) The expected duration of each VC, or lifetime of the VCI, to a given destination is $1/\mu$, where μ is the parameter of an exponential distribution.

3) When a weak VCI collision occurs, the VC creation request is denied and a new request is not generated.

Let random variables A and N be as follows:

A = # of VC creation requests for a given destination arriving at the router in an interval of length r

N = # of weak VCI collisions caused by the VC creation requests in an interval of length r

By the law of total probability we know

$$(1)\ P(N = k) = \sum_{n=0}^{\infty} P(N = k \mid A = n) * P(A = n)$$

From the assumption that A is poisson we know

$$(2)\ P(A = n) = e^{-\lambda r} \frac{(\lambda r)^n}{n!}$$

We can think of $P(N = k \mid A = n)$ as n bernoulli trials with success on k of them, where success means a weak VCI collision.

Claim: $(3)\ P(N = k \mid A = n) = \binom{n}{k} p^k (1-p)^{(n-k)}$

where p is the probability of a weak VCI collision on any given arrival. To show this is true (i.e., that $P(N = k \mid A = n)$ is binomially distributed), it must be shown that p is independent of when a particular arrival occurs. That is, p is the same for the ith arrival, the (i + 1)st arrival, the (i + n)th arrival, etc. It turns out p is equal to $(\frac{\lambda}{\lambda+\mu})^C$, where C is the size of the VCI pool associated with the given destination. The proof is found in appendix A. From (2) and (3) we can then evaluate (1) to be

$$(1^*)\ P(N = k) = \sum_{n=0}^{\infty} \binom{n}{k} p^k (1-p)^{(n-k)} * e^{-\lambda r} \frac{(\lambda r)^n}{n!}$$

$$= p^k e^{-\lambda r} \sum_{n=k}^{\infty} \frac{q^{(n-k)} (\lambda r)^n}{k!(n-k)!} \quad \text{where } q=1-p \text{ and the sum can start at k since } \binom{n}{k} = 0 \text{ for } n < k$$

$$= e^{-\lambda r} \frac{(\lambda pr)^k}{k!} \sum_{n=k}^{\infty} \frac{(\lambda qr)^{(n-k)}}{(n-k)!}$$

$$= e^{-\lambda r} \frac{(\lambda pr)^k}{k!} \sum_{m=0}^{\infty} \frac{(\lambda qr)^m}{m!}$$

$$= e^{-\lambda pr} \frac{(\lambda pr)^k}{k!} \text{ since } \sum_{m=0}^{\infty} \frac{\lambda q r^m}{m!} = e^{\lambda qr}$$

Thus the random variable N, which counts the number of weak VCI collisions at a router, in an interval of length r, has a poisson distribution with parameter λpr, where λ is the poisson parameter for the VC establishment arrival distribution and p is the probability of any given arrival causing a collision.

If we define the **wraparound time** of a VCI space (or pool) to be the size of the space divided by the rate at which VCIs are used (i.e., λ), and let r = **wraparound time**, then the poisson parameter for the number of weak VCI collisions in one wraparound time reduces to $(\frac{\lambda}{\lambda+\mu})^C * C$. It is easily shown that when using such a value for r, the probability of weak VCI collision is the same for all (λ_i, μ_i) and (λ_j, μ_j) such that $\frac{\lambda_i}{\mu_i} = \frac{\lambda_j}{\mu_j}$. Appendix A provides this simple proof. This means that what really matters in determining the probability of weak VCI collisions within a meaningful time interval (i.e., the wraparound time), is the product of the creation rate and duration, not the individual values of either parameter. Recall that in the previous section we discussed techniques to keep the value of this product as low as possible. Table 6 shows the probability of having 0 VCI collisions within a wraparound period, for various values of the size of the VCI space, the VC creation rate, and the VC duration. The rows are labeled by the size of the VCI space and the columns are labeled by the probability of 0 weak VCI collisions within a wraparound period. The entries contain the highest value for the product of the creation rate and duration which still results in a probability of 0 collisions equal to that of the column heading. For example, if a router has 256 VCIs in its pool associated with a given destination, and expects to create VCs to that destination at a rate of 4/sec with each VC expected to last 6 seconds, then the probability of having 0 weak VCI collisions in any 64 second (wraparound time) period is equal to 0.99.

To see the implications of these results for the performance of the NoW method in a large-scale network, consider Table 6, where the products of creation rate and duration for connections from UW-Madison to UIUC, UCB, and Rutgers are .058, .049, and .08, respectively. Assuming the large-scale network model described earlier (i.e., 900 destinations implying 10 bits of the VCI field used to encode destination ID), a UW-Madison router would have at most a VCI pool of size 64 (i.e., 6 bits) for any destination. Depending on the number of other routers sharing this pool at the switch where UW-Madison begins sharing the VCI space associated with the given destination, and the relative size of the creation rate-duration product for those routers, the size of the pool would be reduced by some amount. From Table 6, however, we see that a pool size as low as 4 is still sufficient to yield a 0.99 probability of 0 weak VCI collisions for any creation rate-duration product less than 0.2. Thus, at routers, the performance of the NoW method is very good in such a large-scale environment.

Shared VCs	Probability of 0 Weak VCI Collisions						
	1	.99	.9	.8	.5	.2	.01
1	0	0	.05	.25	1	1	1
2	0	.05	.2	.5	1.4	2	2
4	0	.2	.6	.9	1.8	4	4
8	.1	.7	1.3	1.7	2.7	4.5	8
16	.5	1.7	2.7	3.2	4.6	6.4	12.3
32	1	3	5	6	7	10	16
64	2	6	9	10	13	16	23
128	6	13	17	19	24	28	38
256	12	24	32	35	42	50	63
512	24	46	59	65	77	88	108
1024	47	88	111	120	139	158	188

Table 6. Routers: Analysis Weak VCI Collisions

6.1.2. Strong VCI Collisions in Routers

The process of VC establishment requests arriving at a router with a fixed size pool of VCIs to allocate for such requests can be modeled as a special case of a birth-death stochastic process, namely an M/M/m queue. An M/M/m queue has service requests arriving according to a poisson distribution, service time obeying an exponential distribution, and m servers to service the queue. In terms of a router establishing VCs, this translates into VC establishment requests arriving according to a poisson distribution, VCI durations obeying an exponential distribution, and m being the size of the VCI pool. A strong VCI collision corresponds to the birth-death process being in state m or greater when a new request arrives. For a stochastic process to be in some state k means there are k customers (i.e., VCs) currently queued or being serviced. The stochastic process has a long-run steady state probability of being in some state k as long as the process is stable. In terms of a router, this means that the product of the VC establishment rate and VC duration is less than the number of VCIs in the pool. For a stable M/M/m queue, the probability of queueing (i.e., there are m or more customers already in queue upon arrival) is equal to $\frac{(m\rho)^m}{m!}\frac{p_0}{1-\rho}$, where $\rho = \frac{\lambda}{m\mu}$, and p_0, the probability the queue is empty, is equal to $(\sum_{k=0}^{m-1}\frac{(m\rho)^k}{k!}+\frac{(m\rho)^m}{m!}\frac{1}{1-\rho})^{-1}$ [Triv 82]. Table 7 shows the probability of strong VCI collision (i.e., queueing in an M/M/m queue) for various values of m, λ, and μ. The row label indicates the size of the VCI pool. The bottom number in each entry indicates the probability a VC establishment arrival will cause a strong VCI collision, given values for the size of the VCI pool, VC establishment rate, and VC duration. The top number in each entry is the value of the product of the creation rate and duration (i.e., $\frac{\lambda}{\mu}$) which yields the given probability.

Not surprisingly, for a given value of $\frac{\lambda}{\mu}$, the probability of a strong VCI collision is much less than that of a weak VCI collision. As the VCI pool size increases this difference increases. Given a VCI pool size of only 2, a creation rate-duration product value of .125 still results in only a .007 probability of strong VCI collision. Given the product values computed from Table 3, the NoW method performs more than adequately at routers, in terms of probability of strong VCI collision.

Shared VCs	Top Value: Creation Rate * Duration Bottom Value: Probability of Strong Collision					
1	.063 .063	.125 .125	.25 .25	.375 .375	.5 .5	.625 .625
2	.125 .007	.25 .028	.5 .10	.75 .205	1.0 .333	1.25 .48
4	.25 0001	.5 .0018	1.0 .02	1.5 .075	2.0 .174	2.5 .32
8	.5 0	1.0 0	2.0 .0011	3.0 .013	4.0 .06	5.0 .17
16	1.0 0	2.0 0	4.0 0	6.0 .0005	8.0 .009	10.0 .057
32	2.0 0	4.0 0	8.0 0	12.0 0	16.0 .0003	20.0 .0089
64	4.0 0	8.0 0	16.0 0	24.0 0	32.0 0	40.0 .0003
128	8.0 0	16.0 0	32.0 0	48.0 0	64.0 0	80.0 0

Table 7. Routers: Analysis Strong VCI Collisions

6.2. Simulation

Although the analytical results of the previous sections provide insights into the performance of the NoW method within routers, such analysis cannot provide the answers to several interesting questions. For example, if we assume a request experiencing a weak VCI collision is allocated a different, available, VCI, then the probability of weak VCI collisions is different from that found in the analysis of Section 6.1. Similarly, the

analysis of Section 6.2 assumed requests experiencing a strong VCI collision were queued. The probability of strong VCI collision will be different if requests experiencing such a collision are rejected, rather than queued. A simulation was performed to determine the two probabilities not provided by the analysis. VC request arrivals were assumed to be poisson distributed and VC durations were assumed to be exponentially distributed. The amount of time simulated for each run was 12 hours. Table 8 contains the results of simulation runs for various values of VCI partition size and creation rate-duration product. The top number in each entry is the probability a request arrival results in a weak VCI collision and the bottom number is the probability an arrival results in a strong VCI collision. The results of simulation runs with the worst case parameters taken from Table 3 (i.e., Rutgers: duration = 80, creation rate = 0.001) showed 0 probability of both weak and strong VCI collisions for all VCI partition sizes except a size of 1. At this size, the probability of weak and strong VCI collision was the same, and equal to 0.048. Thus again the performance of the NoW method in routers is seen to be very good in our large-scale network model.

	Creation Rate * Duration				
Shared VCs	.5	2	4	8	16
4	.014 .0007	.27 .09	unstable	unstable	unstable
16	0 0	.0008 0	.02 0	.19 0	unstable
32	0 0	0 0	0 0	.02 0	.17 0
48	0 0	0 0	0 0	.003 0	.05 0
64	0 0	0 0	0 0	.0007 0	.01 0

Table 8. Routers: Simulation Weak and Strong VCI Collisions

7. NoW Performance within Switches

This section contains simulation results showing the probability of weak and strong VCI collisions in switches, given various values for the size of the VCI space and the expected VC creation rate and duration associated with a single destination. The probabilities computed are the same as in Table 8. That is, the probability of weak VCI collisions (top value) is for a system in which weak VCI collisions are corrected and the probability of strong VCI collisions (bottom value) is for a system in which strong VCI collisions are rejected. Again, VC request arrivals were assumed to be poisson distributed and VC durations were assumed to be exponentially distributed. The amount of time simulated was 12 hours.

7.1. Weak VCI Collisions in Switches

It should be clear that under the NoW method the probability of a weak VCI collision within a switch, for a VCI partition associated with a link directly connected to a router, is the same as the probability of a weak VCI collision within that router. Thus it would be uninteresting to simulate a switch for which VC request arrivals come only directly from routers. As was seen earlier, the possibility of weak VCI collision arises in switches when a fixed number of incoming VCIs to a switch must map to a smaller number of outgoing VCIs. Our simulation consisted of two switches. The first switch was fed by three different arrival streams assumed to be from routers. Within the first switch the VCI space was partitioned among those streams using the NoW method. The three streams entering the first switch combined to form a single stream which was fed into the second switch. We assumed the large scale network topology described earlier (i.e., 900 destinations, leaving 6 bits for actual VCI). Thus the incoming stream used 64 different VCIs. The number of outgoing VCIs which the 64 incoming VCIs could map to was a parameter to the simulation, (this value indicates the portion of the outgoing VCI space allocated to the incoming line). Because the stream of VCI requests entering the second switch came from another switch, and not a router, there is no reason to expect consecutive request arrivals to have consecutive incoming VCIs. One would expect, then, that the probability of weak VCI collision within such a switch would be higher than within a switch directly connected to only routers. The results of Table 9 show this to be the case. The top value in each entry of Table 9 indicates the probability of weak VCI collision within the second switch. A comparison of Tables 8 and 9 show the difference in weak VCI collision probability for switches connected to routers and switches connected to other switches. In many cases the difference is substantial. This underlines the importance of the collision avoidance and detection/correction techniques described earlier. However, the results of simulation runs with parameters derived from our traffic study show the performance of the NoW method within switches is acceptable in a large-scale environment. Using Figure 4 as our backbone model and the observed TCP connection creation rates and durations for connections to UW-Madison, we simulated the switches at Pittsburgh and Champaign. The three incoming streams at Pittsburgh had creation rates of 0.0036, 0.0202, and 0.0067 connections/second. This led to an incoming arrival stream at Champaign of 0.0305 connections/second. Knowing that all other links into Champaign had a total incoming creation rate of 0.0225, we computed the size of the VCI partition associated with the link from Pittsburgh to Champaign as 36. With an expected VC duration of 12.8 seconds (see Figure 5), the resulting probability of weak VCI collision within that VCI partition was 0.009. The probability of strong VCI collision was 0. It should be noted that these probabilities are for just one partition and one destination. A switch must handle traffic for multiple destinations and thus must manage multiple VCI partitions. However, the switch at Champaign was the most heavily utilized switch for traffic to UW-Madison and the probability of collision was still quite low. If a switch is to handle collisions for all destinations, then clearly the probability of collision for individual destinations must be low. The time to handle a single weak VCI collision will determine how effective a switch executing the NoW

method can serve all destinations. One can now see how special support on the part of switches to handle weak VCI collisions can be very helpful.

7.2. Strong VCI Collisions in Switches

Since the time to recover from a strong VCI collision within a switch is likely to be fairly long, a reasonable policy is that VC establishment requests which experience such collisions will be rejected. The router would then have to attempt to establish a VC in some other manner, or wait and retry later. The probabilities of a strong VCI collision within a switch which does such rejection are shown as the bottom value in the entries of Table 9. In most cases the proability is very low, often 0. For the simulation using the parameters derived from our traffic study the probability of strong VCI collision was 0. Thus we feel the performance of the NoW method in switches of large-scale networks, in terms of request rejection probability, is quite acceptable.

	Creation Rate * Duration				
Shared VCs	.5	2	4	8	16
4	.09 .0007	.42 .10	unstable	unstable	unstable
16	.02 0	.12 0	.31 0	.65 .004	unstable
32	.007 0	.05 0	.14 0	.42 0	.73 0
48	.007 0	.03 0	.07 0	.19 0	.47 0
64	0 0	0 0	.0007 0	.005 0	.15 0

Table 9. Switches: Simulation Weak and Strong VCI Collisions

8. Related Work

Recent work at UC-Berkeley has addressed the issue of establishing connections for real-time applications (e.g., voice, video) which require performance guarantees. These connections are called *real-time channels* [FeVe 90]. To determine whether a real-time channel can be established, various tests are performed, in turn, at each node along the path. The tests determine whether the performance required of the new channel can be guaranteed while still guaranteeing the performance of all previously established channels. The channel establishment method described in [FeVe 90] takes in excess of one round trip time to complete.

[DaVe 89] describes a method to establish *real-time channels* in less than one round trip time. In this method a connection identifier consists of a source ID, destination ID, and timestamp indicating the time the connection request was generated. This is in contrast to an ATM network where the connection identifier is a VCI. In order to guarantee correct performance (i.e., the channel is in fact established before data flows over it), the method described in [DaVe 89] requires a period of inactivity on the channel so that data packets do not *catch up* to the setup packet which instigates the execution of the tests to determine if the channel can be established. The period of inactivity must be long enough such that all establishment tests can be completed at each node along the path. [FeVe 90] states the time for establishment tests on a VAX-8600 is about 4ms. If a large number of nodes must perform such tests, then the inactive period becomes quite high and the benefits of fast establishment are lowered. In the NoW method a similar period of inactivity is required when a weak VCI collision within a switch is being corrected. Clearly the length of this period is dependent on the time it takes to detect and correct a VCI collision within the switch. The better support the switch has for the NoW method, the less time such correction will take. Note, however, that a weak VCI collision is most likely to occur under a heavier traffic load. At such times, an inactive period on each VC is introduced simply due to the servicing of other VCs (assuming VCs are serviced in a round robin fashion). Under light loads, weak VCI collisions are less likely to occur. If a collision does not occur, then there is no need for an inactive period.

The real-time channel establishment protocol of [FeVe 90] could be combined with the NoW method to provide the establishment of virtual channels which guarantee a certain level of performance. The fast establishment of such channels will be the topic of future work. We believe the principle of designing for the common case can be applied to such work. That is, in most cases it is likely there are sufficient resources available so that the desired performance of the VC can be guaranteed. An algorithm which takes advantage of this fact can likely perform better than the algorithm described in [DaVe 89].

The notion of a *virtual path (VP)* as described in [BuDo 91] and [SOT 90] also relates to the concept of fast VC establishment. Virtual paths are essentially a bundle of virtual circuits (VCs). Virtual paths allow easy management of a group of VCs. A single VP can be established between two endpoints and a certain capacity, or performance guarantee, can be associated with that VP. Subsequent VCs which wish to use the already established VP need not go through the entire establishment phase since the route for the VC is already determined by the VP. Virtual paths which are heavily utilized are advantageous. Virtual paths which reserve resources which are not utilized are wasteful. It is possible to dynamically adjust the capacities assigned to VPs to avoid such inefficiencies. Such dynamic adjustment is similar to our proposed method of dynamically adjusting VCI partitions within switches to avoid weak VCI collisions.

It should be noted that the standard ATM cell header actually contains an 8-bit virtual path identifier field. The cells to be carried in XUNET do not have such a field defined.

9. Conclusions and Future Work

The design philosophy of the NoW virtual channel establishment method was based on the tenet of designing for the common case. We propose a method by which virtual channels can be esta-

blished such that the source does not have to wait one round trip time for confirmation that the channel has been established before data can be sent over the channel. Instead, the source can send data immediately after choosing an identifier to represent the channel. We believe that in the common case the task of the router and the switch is not complicated by using NoW VC establishment. Analysis and simulation have shown the percentage of VC establishments which fall under the category of the common case, given assumptions about the parameters characterizing the network. We have shown the NoW method to perform very well in a large-scale network with realistic parameter values. In the uncommon case, when VCI collisions occur, special support is required on the part of the switches to correctly establish a VC. We have described methods and techniques to help avoid such VCI collisions and help the NoW method perform well as network size grows.

Our current work is concentrated on implementing the basic NoW method within the XUNET environment. This includes the implementation of a VCI collision avoidance mechanism which dynamically adjusts the VCI partitions within switches when there is danger of a collision occurring. It also includes the implementation of weak VCI collision detection and correction within switches. Not only will such an implementation provide us with an actual environment in which to experiment with our method, but we also feel that such an exercise will teach us much about how to design new hardware capable of supporting the NoW method even better than the XUNET hardware.

Future work will address several enhancements to the NoW method. The ability to perform dynamic routing would provide another mechanism by which VCI collisions could be avoided. We would also like to support the fast establishment of channels which guarantee particular types of service. Finally, we would like to combine the NoW method for VC establishment with some type of congestion control mechanism to provide a complete resource management scheme for data traffic in ATM-like networks.

Acknowledgements

We thank Brian Morgan and Jim Pruyne for their work in collecting and processing TCP traffic data. We also thank Brian Morgan for his comments which helped us improve the clarity and readability of the paper.

References

[BiNe 84] Birrell, A.D., and B.J. Nelson, "Implementing Remote Procedure Calls", *ACM Transactions on Computer Systems*, vol. 2, pp. 39-59, February 1984.

[BuDo 91] Burgin, J., and D. Dorman, "Broadband ISDN Resource Management: The Role of Virtual Paths," *IEEE Communications Magazine*, pp. 44-48, Sept. 1991.

[DaVe 89] Damaskos S., and D.C. Verma, "Fast Establishment of Real-Time Channels", Technical Report TR-89-022, International Computer Science Institute, Berkeley, May 1989.

[FeVe 90] Ferrari, D., and D.C. Verma, "A Scheme for Real-Time Channel Establishment in Wide-Area Networks", *IEEE Journal on Selected Areas in Communication*, vol. 8, pp. 368-379, April 1990.

[Fras 91] Fraser, A.G., C.R. Kalmanek, A.E. Kaplan, W.T. Marshall, R.C., Restrick, "Xunet 2: A Nationwide Testbed in High-Speed Networking", AT&T Technical Report, June 1991.

[SOT 90] Sato, K., S. Ohta, I. Tokizawa, "Broad-Band ATM Network Architecture Based on Virtual Paths," *IEEE Transactions on Communications*, vol. 38, pp. 1212-1222, August 1990.

[Triv 82] Trivedi, K.S., *Probability & Statistics with Reliability, Queueing, and Computer Science Applications*, Prentice-Hall, Englewood Cliffs, N.J., 1982.

Appendix A

Proof of the claim that each VC establishment request arrival to a router has the same probability of resulting in a weak VCI collision.

- let p_i = probability ith arrival results in a collision
- let d_i = time difference between the ith arrival and the previous arrival which used the same VCI which the ith arrival wishes to use

$$P(d_i > x) = e^{-\lambda x} * \sum_{k=0}^{C-1} \frac{(\lambda x)^k}{k!} \text{ for } 0 < x < \infty$$

This is a Cth order erlang distribution where C is the size of the VCI space and λ is the poisson parameter for the arrival distribution. d_i has an erlang distribution due to the assumption that the distribution of VC establishment request arrivals is poisson. With poisson arrivals, the time until the Cth arrival is a Cth order erlang distribution.

- let t_i = lifetime of the previous arrival using the VCI which the ith arrival wishes to use

From the assumption that VC lifetimes are exponentially distributed, we know

$$P(t_i > y) = e^{(-\mu y)} \text{ for } 0 < y < \infty$$

Thus we can see that p_i, the probability of the ith arrival causing a collision, is simply the probability of the previous lifetime of the VCI to be allocated, being greater than the time since the VCI was last allocated, or

$$p_i = P(t_i > d_i \mid A = n)$$

Applying the continuous version of the law of total probability, we get

$$p_i = \int_0^\infty P(t_i > x \mid A = n, d_i = x) * f_{d_i}(x)\, dx \quad \text{where } f_{d_i}(x) \text{ is the density of } d_i.$$

$$= \int_0^\infty e^{(-\mu x)} * \frac{\lambda^C x^{C-1} e^{-\lambda x}}{(C-1)!}\, dx.$$

$$= \frac{\lambda^C}{(C-1)!} * \int_0^\infty e^{-xk} x^{(C-1)}\, dx \quad \text{where } k = \mu + \lambda$$

But $\int_0^\infty e^{-xk} x^{(C-1)}\, dx$ is a special form of the gamma function and is equal to $\frac{\gamma(C)}{k^C}$, where $\gamma(C) = (C-1)!$

Thus $p_i = \frac{\lambda^C}{(C-1)!} * \frac{(C-1)!}{k^C}$

$$= \frac{\lambda^C}{k^C}$$

$$= \left(\frac{\lambda}{\lambda+\mu}\right)^C$$

Thus p_i is dependent on μ, λ, and C, but not on i.

Thus the claim that each arrival has the same probability of result-

ing in in a weak VCI collision is true. □

Proof that the probability of weak VCI collision during a **wraparound** period is dependent only on the product of the creation rate and duration, and not on the individual values. Consider λ_1, μ_1, and λ_2, μ_2, such that $\frac{\lambda_1}{\mu_1} = \frac{\lambda_2}{\mu_2}$.

Then $\frac{\lambda_1}{\lambda_1+\mu_1} = \frac{\lambda_1}{\lambda_1} * [\frac{1}{1+\frac{\mu_1}{\lambda_1}}] = \frac{1}{1+\frac{\mu_1}{\lambda_1}}$

$= \frac{1}{1+\frac{\mu_2}{\lambda_2}}$

$= \frac{\lambda_2}{\lambda_2+\mu_2}$

Thus the poisson parameter for the distribution giving the probability of weak VCI collision in one **wraparound** period is the same whenever $\frac{\lambda_1}{\mu_1} = \frac{\lambda_2}{\mu_2}$.