DYNAMIC TIME WINDOWS AND GENERALIZED VIRTUAL CLOCK: COMBINED CLOSED-LOOP/OPEN-LOOP CONGESTION CONTROL

Amarnath Mukherjee

Computer and Information Science Department
University of Pennsylvania
Philadelphia, PA 19104

*Lawrence H. Landweber * and Theordore Faber ***

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

Abstract

This paper presents mechanisms for congestion control of data traffic in high speed wide area networks (HSWANs). The network model assumes reservation of resources based on average requirements. The mechanisms address (a) the different time scales for network congestion (short term and medium-term), (b) admission control that allows controlled variance of traffic as a function of medium-term congestion (Dynamic Time Windows), and (c) prioritized switch scheduling, based on a new fairness criterion for short term congestion (Pulse). This latter criterion is proposed as the appropriate fairness measure for HSWANs.

Preliminary performance studies show that when network utilization approaches unity, the Dynamic Time Window admission control can protect the network from buffer overruns and excessive queuing delays, and when network utilization level is smaller, the admission control may be relaxed to allow a controlled amount of variance that attempts to simultaneously meet the performance goals of the end-user and that of the network provider.

The Pulse prioritized switch scheduling algorithm proposed and studied in this paper is a generalization of the Virtual Clock algorithm [Zhang 91].

The study presented investigates:

- a fairness criterion for bursty traffic,
- necessary and sufficient conditions for bursty traffic to meet that fairness criterion,
- simulation and (limited) analytical results for expected queuing times in switches,
- ability to protect against misbehaving users, and
- relationship between end-point admission control and internal scheduling at the switch.

1. Introduction

High Speed Wide Area Networks (HSWANs) must support a wide range of applications with varying bandwidth requirements. While current network components such as lines and switches can generally operate at data rates sufficient to support existing or planned applications, the challenge is to integrate these hardware components and to design protocols that will enable use of available hardware capabilities by applications. This paper addresses the second issue, the design of network protocols which will support a variety of applications. Key to this is the management of resources / control of congestion. Major performance issues which must be addressed include:

- the network must be able to provide its applications (data, motion video, voice, still images, graphics etc.) a desired quality of service,
- the network must be able to cope with the high bandwidth-delay product that makes timely control operations difficult.

This paper addresses congestion control for *data traffic* in an HSWAN. The HSWAN may also support several kinds of real-time services such as voice or video. The specific mechanisms for handling such (non-data) traffic, e.g., by reservation schemes based on average demands, is beyond the scope of this paper.

The workload for data traffic is bursty and unpredictable. Therefore, the congestion control solution that is sought is one that has the elasticity to handle bursty workloads. In the context of data traffic, the goal of flow control is to simultaneously provide service that is efficient, free from loss, and fair. Of course, different applications can accommodate different levels of packet loss, so the second goal is often relaxed.

In traditional data networks, fairness usually implied complete equity among competing users. Scheduling algorithms like fair-queuing [Nag 87,Nag 85] were considered appropriate for guaranteeing fairness. In an HSWAN, however, the environment should facilitate reservation of resources, provision for describing user workloads, and a high degree of heterogeneity in user characteristics. The fairness criterion needs to be redefined in this environment.

The position taken here is that a user should be able to advertise its arrival pattern and receive good service in return. A fair scheduling strategy may then be defined as a function of the actual arrival pattern and the advertised arrival pattern. The Virtual Clock algorithm exhibits this behavior [Zha 89] and is used as the starting point for our Pulse algorithm.

In keeping with the goals above, this paper proposes a set of mechanisms for congestion control and avoidance. These are expected to be especially suitable for a high bandwidth-delay pro-

*This work was supported by the DARPA/NSF Gigabit Testbed Program via CNRI.
**This work was supported by an IBM Graduate Fellowship.

duct environment. The key ideas involve separation of different sources of network congestion, short-term bursts and medium-term load, and using separate mechanisms to address them. Thus, Dynamic Time Windows (DTW) is proposed as a mechanism to limit traffic burstiness from sources as a function of the medium-term load on the system, while Pulse is proposed as a mechanism for dealing with fair scheduling by switches of short-term bursts.

We now describe the intuition behind the DTW admission control and the Pulse scheduling algorithm. A more rigorous treatment follows in Section 3.

Most window-based congestion control strategies limit the number of outstanding packets in the network by placing an explicit limit, the window size, on each user. A user can only transmit more data when it has room in its window, and once the window is full, capacity in it is released by receipt of acknowledgements of packets sent. Congestion control is accomplished by changing the size of these windows. By limiting the amount of data a user can send in a given time, here the round trip time, standard window-based congestion control modulates each user's throughput.

However, it is generally the bursts in traffic that cause congestion. DTW gives network administrators a way to control this traffic characteristic. Under DTW, each user tells the network what average rate it would like to send at, as well as the peak rate it achieves at its maximum burstiness. The interval over which that average is computed, often called I in this paper, can be viewed as a window in time that regulates the burstiness of the source. The larger the time window, the longer the duration and the higher the rate of a burst can be. DTW is based on the notion that by regulating the burstiness of users by changing the size of these time windows, congestion can be avoided without unduly restricting users.

Preliminary performance studies of DTW show that the queue length statistics at switching nodes (mean, variance and maximum) are approximately proportional to the time window size. Further,

- when network utilization approaches unity, the DTW mechanism can protect the network from buffer overruns and excessive queuing delays, and
- when network utilization level is smaller, the time window may be increased to allow a controlled amount of variance that attempts to simultaneously meet the performance goals of the end-user and that of the network administrators.

Pulse is designed to use the additional information provided to the network by DTW to more efficiently schedule bursty traffic at internal switches. The Virtual Clock algorithm is modified to take the burst rate information into account when scheduling packets to be switched. By using the extra information, Pulse tries to clear bursts faster in order to minimize congestion.

The study with respect to prioritized scheduling algorithms, such as Pulse, investigates

- necessary and sufficient conditions for accomplishing the desired fairness in the presence of controlled bursts,
- simulation and (limited) analytical results for expected waiting times,
- ability to protect against misbehaving users, and
- relationship between end-point admission control (DTW) and internal scheduling (Pulse and Virtual Clock).

The rest of this paper is organized as follows. Section 2 presents the model of the network. Section 3 discusses the DTW and Pulse mechanisms in detail. Section 4 presents a detailed analytical and simulation study for static time-windows and various scheduling algorithms. Section 5 discusses preliminary results on DTW. Section 6 discusses related work and Section 7 presents our conclusions and outlines future work.

2. Preliminaries

2.1. Network Model

Our network consists of hosts, internal switches, and point-to-point links interconnecting them. The hosts each serve as computing engines for one or more users. Each entity requesting or receiving a connection is considered a user, and is assumed to be in control of the packets it presents to the network. A user also receives information from the network in the form of control packets, and provides information to the network regarding its anticipated traffic patterns. Internal switches take packets from an incoming link and transmit them on an outgoing link. They are also responsible for queuing packets to be transmitted, and scheduling them for transmission.

The smallest unit switched in the network is the packet. The network (HSWAN) operates at very high speeds over a wide geographical area (e.g., a 1 gigabit/sec network over the continental USA). The high bandwidth-delay product of the network and the various real-time components of traffic require some form of resource reservation to hedge against relatively slow control operations and to ensure that performance guarantees are met.

We restrict ourselves to congestion avoidance on data traffic, where we define data traffic to be traffic that does not require strict guarantees on delay or jitter. The principal difficulty with data is that very little *a priori* knowledge is available about its traffic characteristics, so reserving resources for it is difficult. Depending on the actual environment some average bandwidth may be reserved, but it is expected that the variability in traffic will be considerable.

2.2. Congestion Model and Motivation for a Two-Level Approach

Consider an internal switch in this network. For the purposes of congestion control, we distinguish the queue length fluctuations due to short-term bursts, and medium-term variations in load. The former is caused by bursty traffic flowing through the network, while the latter is due to the ensemble of users active at any time. More precisely, short-term fluctuations are those which have a duration of the order of a round-trip delay or less, while medium-term fluctuations are those which have a larger duration.

A reactive congestion control scheme uses implicit or explicit feedback information from the network to adjust the input transmission rate or window size. The feedback information is virtually guaranteed to be out of date when it arrives at the sender relative to the duration of short-term fluctuations in queue length caused by bursty traffic. To deal with short-term bursts, new *open-loop* protocols have recently been proposed to meter and police the traffic on each connection [BCS 90, Gol 91, Tur 86, SLCG 89, Zha 89]. We agree with these proposals that short term congestion must be addressed without resorting to feedback congestion control.

On the other hand, medium-term congestion is caused by an excess of traffic from some longer-lived factor than a brief variance in traffic. In fact, there may be many different disturbances, each of varying duration involved. In any case, since these are long-lived disturbances they can be detected by and responded to effectively via a feedback or *(closed-loop)* congestion control scheme. It is therefore important that our perception of congestion distinguish between short-term and medium-term congestion.

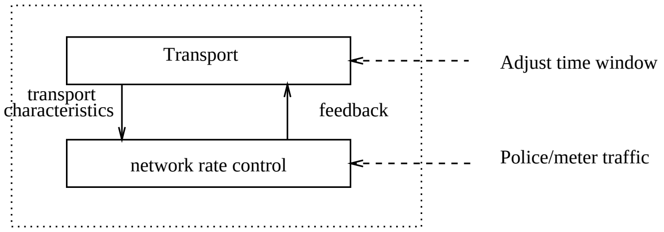

Figure 2.1: Two level control for data traffic.

This distinction suggests a 2-level congestion control strategy as shown in Figure 2.1. The lower level's responsibility is to use open-loop controls which meter and police user traffic, and protect the network from short-term bursts. The upper level's responsibility is to deal with closed-loop control and provide some elasticity to an otherwise restrictive lower level. In particular, the combined solution should accommodate variance because burstiness is an inherent characteristic of data traffic. Further, it should also incorporate priority handling; if a faster transmitter is willing to pay more, there should be mechanisms in place to provide for this service.

It is also important to realize that although there are two levels to the proposed control scheme, they are not to be viewed as strict analogs to the ISO definition of a layer. In particular, our layers are implemented in physically separate entities, and at different layers of the ISO hierarchy. The upper layer, DTW, is an end-to-end, transport level, closed-loop congestion avoidance scheme based on admission control which is implemented between hosts or between switches and hosts. The lower layer, Pulse, is an open-loop congestion control discipline based on prioritized scheduling which is implemented in switches. Although the two layers share parameters and are designed to complement each other, they are independent.

3. Specification of the Systems

This section will describe both DTW and the Pulse scheduling discipline in detail. Although the schemes are independent, they are designed to work together to meet our goals of an effective HSWAN; this section will also address how they complement each other.

3.1. Basic Model and Definitions

Throughout the rest of the paper, we will be using several terms and conventions to describe the behavior of both users and switches. A source is taken to be any source of traffic, either a user or switch attached to the current switch or host. All sources send traffic in packets of some fixed size. The ultimate source of any traffic is a user, and any packet can be uniquely associated with a user, for example by a virtual circuit identifier or a source address. A switch queues packets on a per-user basis, and serves every packet at the deterministic rate μ. Queuing is done on input, and buffer space is taken to be infinite, or large enough not to be an issue.

At reservation time, every user gives the network a specific description of its traffic. The description takes the form of three quantities, an average sending rate in bits/sec, denoted by $\bar{\lambda}$; a burst rate in bits/sec, denoted by $\hat{\lambda}$, that indicates the maximum sending rate it will achieve; and an averaging interval in seconds, denoted by I, indicating the period over which the averaging will be done. These amount to a contract between the user and the network that the user will send at most $\bar{\lambda} I$ packets in any interval I, and that the maximum rate that it will ever send at is $\hat{\lambda}$. The quantity I is the *time window* over which that the user can distribute its burstiness. This admission control mechanism for controlling congestion is called *Dynamic Time Windows (DTW).*

Pulse, a generalized version of Virtual Clock, is a method by which a switch determines the transmission priority of packets received on different connections. Pulse uses the DTW values I, $\bar{\lambda}$ and $\hat{\lambda}$. for a connection. It increments the priority value of successive packets by $1/\hat{\lambda}$. After $\bar{\lambda} I$ packets are received, the next packet's priority is increased by I - ($\bar{\lambda} I / \hat{\lambda}$). Hence pulse assumes the worst case, i.e., that all packets in a burst are transmitted at the beginning of the time window I.

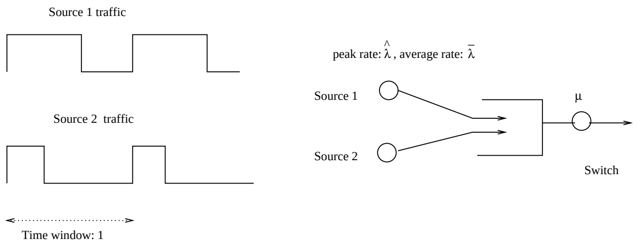

Figure 3.1: Two bursty sources feeding into a switch.

3.2. Dynamic Time Windows

The premise that motivates DTW is that no traffic control scheme to date really provides network designers with a simple way to specify and control the burstiness of sources. The control we propose is to vary the interval I defined above. The larger the I that the network allows the user, the more variation that user's traffic pattern can have. Conversely, the smaller the time window the less burstiness that user's traffic pattern exhibits. By way of example, consider a scenario where a user is allowed to transmit at an average rate of 10,000 packets/second. This average rate can be met by 100 packets over a 10 msec period or 10 packets over a 1 msec period. While the average rates are the same in both cases, the first case, with the larger time window, allows greater variability.

Thus, although the end-point has reserved a fixed bandwidth, it can potentially vary the interval size to adjust the variability of traffic it generates. If this interval size is agreed upon with the network, then a combined solution emerges that simultaneously protects the network from overloads and allows for a controlled variance that a user desires.

To illustrate the effect of variability of arrivals on queue length, consider a G/D/1 queue. An upper bound on the expected queue length is given by [Kle 76]:

$$E[Q] = \sigma_a^2 \overline{\lambda^2} / 2(1-\rho),$$

where $E[Q]$ is the expected queue length, $\bar{\lambda}$ is the mean arrival rate, σ_a^2 is the variance in packet interarrival time and ρ is the traffic intensity ($\bar{\lambda}/\mu$). Clearly, as $\rho \rightarrow 1$, one must make sure that $\sigma_a^2 \rightarrow 0$ for acceptable performance. On the other hand, when ρ is smaller, a higher σ_a^2 may be allowed. We can view σ_a^2 as a measure of the burstiness of the source, and ρ as a measure of the congestion of the switch. Clearly modulation of burstiness of sources is key to reducing queue lengths at a congested switch.

Dynamic adjustment of interval size is motivated by this consideration. As a quantitative example of how the interval size affects queue length, consider a single queue fed from two sources as shown in Figure 3.1. The parameters are taken to be:

Source 1		Source 2	
$\hat{\lambda}_1$	$\bar{\lambda}_1$	$\hat{\lambda}_2$	$\bar{\lambda}_2$
100 Mb/sec	40 Mb/sec	100 Mb/sec	20 Mb/sec

The sources have an infinite number of packets to send, and the switch service rate, μ is 80 Mb/sec. For the parameter values shown in the figure, the traffic intensity, ρ, is 0.75. This configuration was simulated using a worst case traffic pattern from both sources as shown in Figure 3.1.. These simulations show that for I=1 msec, the queue length statistics are: $E[Q]$=6.92, and $var(Q)$=5.51. For I=10 msec, the corresponding numbers are $E[Q]$=63.17 and $var(Q)$=52.49. i.e., there is a factor of 10 increase for a similar increase in interval size. An approximate analysis (Section 4) shows that this indeed should happen, and additional simulations bear it out (Section 4). Specifically, if I is increased to αI and everything else remains the same, then $E[Q]$ increases approximately to $\alpha E[Q]$. Further, the variance and maximum of Q also increase by the same proportion. Not only does the time window concept give designers a way to specify burstiness of traffic, but it turns out to be one with an intuitive meaning.

In order for DTW to be an effective scheme, it requires that two basic requirements be met by users. They have to negotiate the traffic characteristics in good faith and abide by them. Users must also abide by the adjustment policies and change their behavior when the time window is opened or closed.

We close this section by comparing DTW with traditional window-based systems, and traditional rate-based systems.

With traditional packet windows:

- With traditional windows, the user can transmit a certain number of packets before receiving an acknowledgment. This limits the number of packets that can be in transit during one round trip time. With DTW, the user may transmit a certain number of packets in a specified time interval. This interval may be shorter than the round-trip time.
- With traditional windows dynamic window adjustment [Jac 88, MiSe 90, RaJa 88] changes window size. DTW adjusts the interval size. In the first case, the 'interval' (= round-trip time) is held constant while the number of packets in the interval is changed. In the second case, the interval itself is adjusted. In both cases, the number of packets that can arrive at a switch in a short time period is controlled.
- Traditional windows do not address questions of sending rate or degree of burstiness of the source. DTW allows control of these characteristics of the source.

With rate based schemes:

- Zhang [Zha 91] and Golestani [Gol 91] characterize traffic workload as an average rate over a specified interval. They devise scheduling algorithms, given this workload. DTW attempts to adjust the workload itself (traffic shaping), if the network is congested. In conjunction with this, Pulse may be used to facilitate scheduling of bursts over the intervals. Unlike the User Behavior Envelope proposed by Zhang [Zha 91], DTW adapts to network congestion levels, allowing sources greater burstiness when the network load is lighter.
- Ramamurthy and Dighe proposed the Distributed Source Control Protocol [RaDi 91]. DTW, in conjunction with Pulse / Generalized Virtual Clocks (discussed in the next section), extends and generalizes Distributed Source Control to handle multiple time constants in the network.
- Leaky Bucket [BCS 90, SLGC 89, Tur 86] is another technique that has been proposed for admission control. In Leaky Bucket, a fixed sized bucket is associated with each transmitter and it is fed with tokens at some constant rate. If the bucket is full, tokens are dropped. An incoming data packet must grab a token before entering the network. Medium-term load may be incorporated into this scheme in several ways as for instance (a) by changing the token flow rate into the bucket and (b) by changing the bucket size. The latter is similar in spirit to the scheme proposed here, but we do believe that DTW will be more closely matched to traffic characteristics on the one hand and switch scheduling on the other.

3.3. Pulse Scheduling

The proposed solution to priority scheduling, Pulse, extends the Virtual Clock algorithm [Zha 91]. We describe this latter algorithm first, then the proposed extension along with motivation for doing so. A new fairness criterion appropriate for this environment is defined and appropriate properties of a scheduling algorithm with respect to this criterion are identified (Proposition 1).

The Virtual Clock Protocol takes the approach of allocating switch resources to individual users based on their average demands. Let us suppose that a user i is allocated a rate $\bar{\lambda}_i$. Each switch associated with this user (i.e., on the end-end path for this session) maintains a virtual clock v_i which determines user i's priority with respect to other competing users. Initially, v_i is set to the real clock. When a packet belonging to user i arrives, v_i is incremented to **max** [$(v_i+1/\bar{\lambda}_i)$, *realArrivalTime*]. The packet that is dispatched next corresponds to the user with the minimum virtual clock. This is a *dynamic priority* scheduling algorithm since priorities are based on the current behavior of the sources. Notice that if a user were to transmit faster than its specified average rate, it would still get scheduled if the resource is free.

Now, if a user were to provide a more detailed arrival pattern for its packets, the priority assignment curve could be extended to follow such a curve. This generalization is referred to here as the Generalized Virtual Clock algorithm.

As a special case, let us suppose that a user i specifies its peak transmission rate $\hat{\lambda}_i$, and its burst characteristics $t_{1,i}$ and $t_{2,i}$ as shown in Figure 3.2. Notice that given a burstyfactor (which is defined as $\hat{\lambda}_i/\bar{\lambda}_i$) and an interval size, I, the parameters $t_{1,i}$ and $t_{2,i}$ are easily obtained. $\hat{\lambda}_i$ here corresponds to the speed that user i

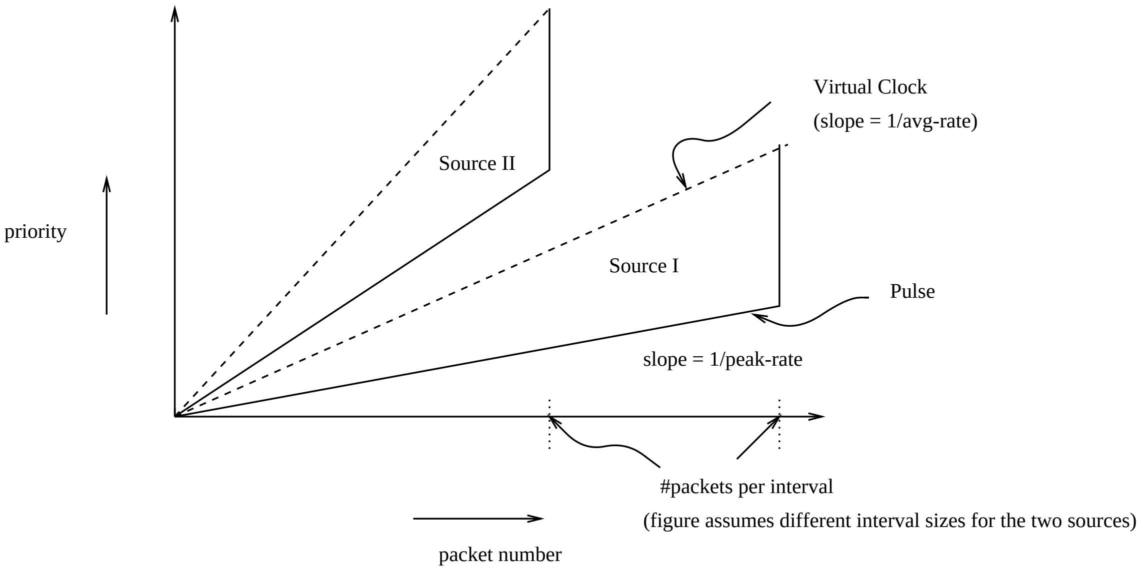

Figure 3.3: Dynamic priority assignment with Virtual Clock and Pulse.

wishes to transmit at. It is the maximum of the speed at which it can transmit and the speed decreed by flow control. A user may of course settle on providing $t_{2,i}$=0 which implies normal Virtual Clock. If it provides the extra information, it will get better service for its bursts.

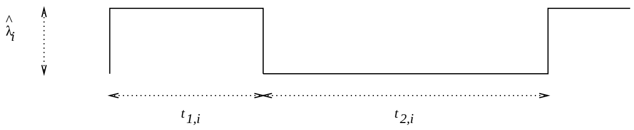

Figure 3.2: User specifications for Pulse.

The priority assignment curve for this special case is shown in Figure 3.3. We call it the Pulse Algorithm and it works as follows. Let $M_i = \hat{\lambda}_i t_{1,i}$/PacketSize. M_i is the average number of packets that user i would transmit in time $t_{1,i}$. Associate a virtual clock v_i and an integer A_i at every switch in the path of user i. Initialize A_i to M_i and v_i to real clock.

```
priority[packet] := v_i
If ( A_i>0) then
        v_i := v_i+1/λ̂_i
        A_i := A_i−1
else
        v_i := v_i+t_2,i
        A_i := M_i
end
{ schedule packet with minimum priority}
```

Algorithm 3.1: Pulse scheduling.

When a packet from user i arrives at the switch, execute Algorithm 3.1.

Assuming appropriate allocation of switch resources, if a user behaves according to its own specifications, this scheduling algorithm guarantees it good service in that its packets will be dispatched without excessive delay. Also, packet loss can be avoided altogether. However, if a user exceeds its specified burst size or transmits faster than that agreed upon then, as in Virtual Clock, its priority is reduced and its performance is affected.

Notice that Pulse is identical to Virtual Clock if $t_{2,i}$=0 (in this case, M_i could be any positive integer). Thus Pulse allows users to specify only their average rates of transmission if they so choose. However, if a user provides more information to the network, it will get better service

The obvious question that arises is, what is an appropriate scheduling discipline. Clearly giving higher priority to someone will hurt someone else (since the total amount of unfinished work is conserved for any work conserving discipline). The fairness criterion that is adopted here is that the number of packets that are dispatched for user i be proportional to its peak rate, provided it has not exceeded its 'quota' for the interval. The motivation for this is that a comparable delay would hurt a faster transmitter more than a slower one. If it is desired that the faster transmitter get a good quality of service (it may be willing to pay more for instance), then it should be scheduled by its peak rate.

Different priority assignment curves, referred to as Generalized Virtual Clocks, may be compared with respect to this fairness criterion. The following Proposition summarizes the results.

Notation: Let

Δt= some time period.

N_i= number of source i packets served at a switch over Δt.

N= total number of packets served at the switch over Δt.

$\hat{\lambda}_i$= advertised peak transmission rate of flow i.

Fairness Criterion:

$$\forall i \quad N_i \propto \hat{\lambda}_i \tag{3.1}$$

Assumptions:

With each flow i, there exist:

(1) an arrival curve $a_i(n)$ which gives the time of arrival of the n^{th} packet.

(2) a priority curve $f_i(n)$ which specifies the relative priority of this packet with respect to the others. This priority determines which packet is to be scheduled next by the switch. A lower value of $f_i(n)$ implies a higher priority.

Proposition1:

(1) If all traffic is 'homogeneous,' then any arbitrary single priority assignment curve will guarantee fairness at the switch as defined by (3.1). By homogeneous traffic we mean that either

(a) $a_i(n)=a_j(n)\ \forall\ i,j,n$ deterministically, or

(b) $a_i(n)$ equals $a_j(n)\ \forall\ i,j,n$ in distribution (i.e., with probability one).

(2) If traffic is non-homogeneous, and there is queuing at the switch, then it is necessary and sufficient for fairness (as stated in (3.1)), that the following hold:

$$\text{if } a_1(n) < a_2(n) \text{ then } f_1(n) < f_2(n)\ \forall n. \tag{3.2}$$

(Aside: If there is no queueing at the switch, then there is no need for a scheduling discipline.)

A consequence of (2) is that if $f_1(n)$ and $f_2(n)$ satisfy (3.2), then $cf_1(n)$ and $cf_2(n)$, for $c>0$, also satisfy (3.2) so $cf_1(n)$ and $cf_2(n)$ are also fair. Hence, there is no unique fair priority assignment curve.

(3) $N_i=\dfrac{\hat{\lambda}_i}{\sum_j \hat{\lambda}_j}N$, where N is conserved, i.e., N is constant across all scheduling disciplines.

Parts 1(a) and 1(b) above contradict a previous finding in [Zha 91]. If the scheduling discipline is Virtual Clock, then the virtual clocks of a set of sources were thought to diverge with respect to each other as time progressed. However, the theory of coupled stochastic processes [Gre 78] suggests that if the sources are statistically homogeneous, then they would in fact converge almost surely, every so often. The proof will be included in a future paper.

Element	Parameter	Value
Source 1	peak rate ($\hat{\lambda}_1$)	100 Mb/sec
	average rate ($\bar{\lambda}_1$)	40 Mb/sec
	time window (I)	1-50 msec
Source 2	peak rate ($\hat{\lambda}_2$)	100 Mb/sec
	average rate ($\bar{\lambda}_2$)	20 Mb/sec
	time window (I)	1-50 msec
Switch	service rate (μ)	61-120 Mb/sec
	packet size	250 Bytes
	buffers (B)	infinite

Table 4.1: Simulation parameters.

Our simulations showed this to indeed hold true. Zhang's suggestion of synchronizing the virtual clocks to the real arrival time is therefore, strictly not necessary, but perhaps desirable nevertheless. An alternative would be to set a bound on the maximum difference between virtual and real clocks. This would allow some flexibility but put a bound on the aggravation that may be caused to well behaved sources due to someone who was idle for a long time.

4. Detailed Study

A simple model was used as the first step towards determining the properties of DTW and Pulse. Two bursty sources were used and their average rates were enforced over a single invariant time-window *I*. The parameters used in this study are summarized in Table 4.1.

4.1. Notation and Source Behavior

$\bar{\lambda}_i$ is the average rate of user *i*.

$\hat{\lambda}_i$ is the peak rate of user *i*.

I is the invariant time window size.

μ is the service rate of the switch

$E[\cdot]$ is the expected value of a given random variable.

Q is the switch queue length, the sum of the packets waiting from both sources.

W_i is the waiting time of a packet from source *i* at the switch.

ρ is the switch utilization, and is defined as $\rho \equiv \sum \bar{\lambda}_i / \mu$

In all simulations, both sources used a simple worst case behavior. They transmitted at their peak rate at the beginning of every interval, *I*, until they had transmitted $I\,\hat{\lambda}_i$ bits. Both time windows were synchronized as well, so both sources began transmitting at the same time every interval. The transmission characteristics are shown in Figure 3.2. All simulations used the network configuration in Figure 3.1. Buffers in the switch were considered infinite, so no loss occurred due to buffer overflow.

All simulations considered only one hop, from source to switch. Future studies will consider multi-hop paths.

The following properties were investigated:

- Overall mean queue length, $E[Q]$, and its variance, as a function of time window (*I*) and switch utilization (ρ).
- The expected waiting times, $E[W_1]$ and $E[W_2]$, as a function of *I* for Pulse, Virtual Clock and *First Come First Served (FCFS)*. With FCFS, packets for all connections are put on a single output queue when they arrive at the switch. The first to arrive are then the first to be transmitted. FCFS was selected for comparison purposes and also to determine the effectiveness of Virtual Clock and Pulse in keeping misbehaving users from hurting the performance of well-behaved ones.
- Effect of mismatched *I* between the transmitter and the switch, i.e., the time-window used by the end-points and that used by the switch were made different. The purpose was to determine the ability of the priority algorithms to withstand transmitter misbehavior
- the effect of a non-synchronized interval size on well behaved transmitters due to possible delays in the *I*-adjustment process.

Discrete event simulation was used as the principal tool in this investigation. The DeNet simulation tool [Livny 87, DeNet 88] was used.

The results obtained from simulations have been supplemented with analytical justification whenever such an analysis has been feasible.

4.2. Expected queue length vs. I and ρ

ρ	E[Q] (packets)		
	I = 1 msec	I = 10 msec	I = 20 msec
0.98	10.54	98.3	195.9
0.92	9.61	89.2	177.7
0.86	8.58	79.3	157.9
0.75	6.92	63.2	125.7
0.50	3.05	25.5	50.5

Table 4.2: $E[Q]$ as a function of ρ and I.

The numbers in table 4.2 suggest that I may be changed to effectively deal with changes in ρ. Notice that as I is increased to αI, the expected queue length increases to $\alpha E[Q]$. Thus we have:

Theorem 1

If I_1 and I_2 are two values of I such that $I_2/I_1=\alpha$, then the corresponding expected queue lengths have the following relationships:

(a) $E[Q]_2=\alpha E[Q]_1$, and

(b) the upper bound $\alpha E[Q]_1$ is a reasonable approximation for $E[Q]_2$.

(c) The same results hold for $var(Q)$ and $Q_{\max}$ as well.

4.3. Expected waiting times vs. I for Pulse, Virtual Clock and FCFS

Tables 4.3 and 4.4 show the expected waiting times for Pulse, Virtual Clock and FCFS for different time window sizes as ρ is varied. In these simulations, all the transmitters were well behaved in that their packet generation pattern matched the pattern described in Section 4.1. As a consequence, the numbers for Pulse and FCFS are similar. The differences arise primarily because of an implementation difference in breaking ties for simultaneous events in the event list: for Pulse, ties were broken using an uniformly distributed random number; for FCFS, a preexisting library module was used, and this ordered simultaneous events by queueing them up in the order the program discovered them.

The simulation scenario was the same as in Table 4.1: there were two bursty sources, the first with a peak rate $\hat{\lambda}_1$, of 100 Mbits/s and an average rate, $\bar{\lambda}_1$=40 Mbits/s, the second with a peak rate $\hat{\lambda}_2$=100 Mbits/s and an average rate, $\bar{\lambda}_2$=20 Mbits/s. Notice that Virtual Clock resulted in better service for the less bursty traffic ($E[W_1]<E[W_2]$) while Pulse resulted in better service for the more bursty one ($E[W_2]<E[W_1]$). This bears out our reasoning. However, providing better service to one hurt the other. In fact, the waiting-time-conservation-law [see Kle 76] holds:

$$\sum_i \bar{\lambda}_i E[W_i] = \text{Constant}$$

irrespective of the scheduling discipline. The important point though is that the network was able to withstand a load as high as 0.98 if the time-window was small.

As an aside, an interesting property was seen in the $E[W_1]/E[W_2]$ rows: the ratios of the waiting times approach a limit as the time-window was increased.

I (msec)	Expected Wait	Pulse (μsec)	VClock (μsec)	FIFO (μsec)
1	$E[W_1]$	405	304	400
	$E[W_2]$	245	446	254
	$\frac{E[W_1]}{E[W_2]}$	0.60	1.47	0.64
10	$E[W_1]$	3768	2931	3764
	$E[W_2]$	2297	3972	2305
	$\frac{E[W_1]}{E[W_2]}$	0.61	1.36	0.61
20	$E[W_1]$	7500	5850	7502
	$E[W_2]$	4588	7886	4584
	$\frac{E[W_1]}{E[W_2]}$	0.61	1.35	0.61

Table 4.3: Expected waiting time vs. time-window size ρ=0.98.

I (msec)	Expected Wait	Pulse (μsec)	VClock (μsec)	FIFO (μsec)
1	$E[W_1]$	112	62	110
	$E[W_2]$	80	181	85
	$\frac{E[W_1]}{E[W_2]}$	0.71	2.92	0.77
10	$E[W_1]$	937	511	935
	$E[W_2]$	681	1532	685
	$\frac{E[W_1]}{E[W_2]}$	0.73	2.99	0.73
20	$E[W_1]$	1851	1012	1852
	$E[W_2]$	1354	3030	1352
	$\frac{E[W_1]}{E[W_2]}$	0.73	2.99	0.73

Table 4.4: Expected waiting time vs. time-window size for ρ=0.50.

An approximation for the average waiting time for each flow i with Virtual Clock scheduling and DTW arrival pattern has been obtained. The assumptions and the main result are as follows.

The sources are assumed to transmit packets at their peak rates; however, they are well behaved in that they maintain their specified average over the specified interval. The switch is assumed to schedule according to the lowest virtual clock.

Let $\bar{\lambda}_i$ and $\hat{\lambda}_i$ be expressed in packets per unit time. Let the total number of active sources be J.

Then the average waiting time for source i under Virtual Clock is given by

$$E[W_i] \approx \frac{1}{2\mu}\sum_i^J(\bar{\lambda}_i I) - \frac{1}{\hat{\lambda}_i}\left\lceil \frac{\bar{\lambda}_i I+1}{2} \right\rceil \qquad (4.1)$$

4.3.1. Experimental Validation of Virtual Clock waiting time

Using the parameters in Table 4.1, and taking $\mu = 80$ Mb/sec, we compare Virtual Clock simulation values to the values predicted by (4.1).

For I=1 msec the waiting time numbers are as follows:

$E[\cdot]$	Analysis	Simulation
$E[W_1]$	177.5 µsec	187 µsec
$E[W_2]$	297.5 µsec	319 µsec

For I=10 msec, the corresponding numbers are

$E[\cdot]$	Analysis	Simulation
$E[W_1]$	1752.5 µsec	1762 µsec
$E[W_2]$	2752.5 µsec	2793 µsec

The simulation and analysis numbers were also found to be close for other parameter values. In a later paper, we expect to report analytical and simulation results for Pulse with DTW.

4.4. Effect of mismatched *I*

Two sets of simulations were run to study the effect of mismatched time-windows. The average and peak rates for the two sources were as in Table 3.1. In each set, one source was well-behaved and had its *I* exactly equal to that used by the switch.

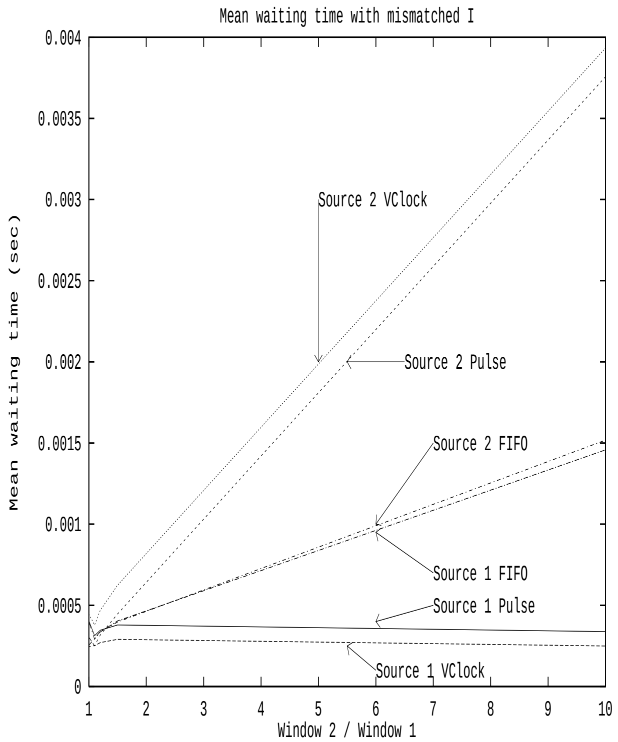

Figure 4.1: Source 2 mis-behaves. I_1=1, µ=61, ρ=0.98.

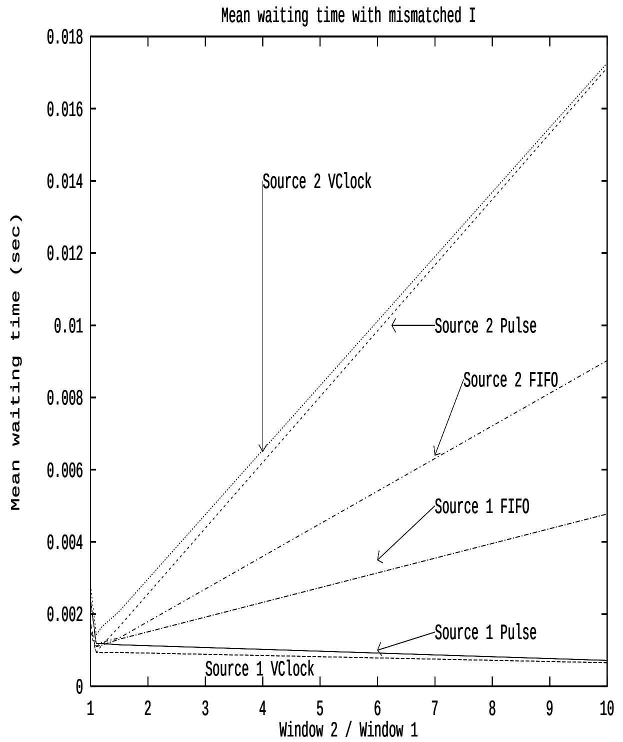

Figure 4.2: Source 2 mis-behaves. I_1=10, µ=80, ρ=0.75.

The time-window of the other source was increased from 1.1 to 10 times that of the well behaved source and consequently that of the switch. In practice this situation may arise due to a miscommunication between host and switch, or a lapse in the adjustment of the host. Of course a user may also be intentionally misbehaving.

Figures 4.1 and 4.2 show the expected waiting times as a function of the extent to which the bad source is misbehaving. In these figures Source 1 is well behaved and Source 2 is not well behaved. The curves are similar over a wide range of I_1, ρ and µ values.

Several things are immediately observable:

- Both Pulse and Virtual Clock provide effective firewalls against misbehaving sources as compared to FCFS.
- The relative ordering of Pulse and Virtual Clock is not preserved across the parameter space, indicating that neither one is necessarily better than the other when it comes to firewalling.
- Performance of the well-behaved source: For smaller time-windows, with either Pulse or Virtual Clock, the well-behaved source gets its due share. For larger time-windows, its performance in fact improves as the ratio I_{bad}/I_{good} increases. For lower values of I_{bad}/I_{good}, there is a slight degradation in its performance.
- The curves for FCFS also show a separation in waiting times (i.e., the well behaved source gets better response). This separation improves with lower utilization and large time-windows. However, the well behaved source is still adversely affected by the misbehaving source when compared to Pulse or Virtual Clock.

-- --

5. Dynamic Time Window Adjustment

This discussion is split into three parts. The first part discusses the objectives of time-window adjustment, the second part discusses the mechanisms for the same and the third part discusses policy issues. Additional results on feebback mechanisms will be forthcoming in a future paper.

5.1. Objectives

Ideally, the switch will not need to put any restriction on transmission. Under low loads (i.e., low ρ) this ideal may be achieved without additional effort. When the load increases however, the switch will get into more congested states: queue lengths will increase, waiting times will increase and buffers may overflow. The objective now would be to regulate the arrivals such that the cumulative arrival rate conforms to the average arrival rate over smaller intervals of time. The interval size will determine the maximum queue lengths possible. Assuming that $\rho < 1$, it will also determine the maximum length of the busy period (because queues are expected to empty out by the end of the interval).

5.2. Mechanisms

The switch will set the interval size it will support at any time. It will have at its disposal multiple levels (or interval sizes). Depending on current traffic intensity, it will set the maximum interval size that it is willing to support.

Each packet (or cell) will have a special field that will be set by the switches to the maximum I that the path can support.

It is a policy issue as to what I should be set to. For instance, I may be adjusted when $\sum_i \bar{\lambda}_i$ changes. When this latter change occurs, every *active* source will detect the change in one round trip time. Non-active sources will find out about the change within one round trip time of their becoming active again.

Each source, i, should ensure that its own interval size I_i is less than or equal to I. In general, if I_i is large, source i may see larger queuing delays (as we saw in Section 4).

5.3. Policies

Several policies are possible for I-adjustment. The choice would depend on the objectives to be met. Two policies are discussed here. This is the subject of continuing investigation.

5.3.1. Policy 1: Prevent Buffer Overflow

One policy would be to size the interval so as to prevent buffer overflows from occurring. An alternative (optimistic policy) would be to consider only the active sources for the above calculation. The average number of active sources may be computed using Mean Value Analysis as done by Hahne et al. [HKM 91].

5.3.2. Policy 2: Follow Iso-E[Q] Curves

An alternative policy is to constrain the expected queue length at a switch to a constant. (Or equivalently, to constrain the average waiting times to a constant.) The expected queue length, $E[Q]$, is a function of ρ, I, and the burstyfactors $(\bar{\lambda}_i/\hat{\lambda}_i)$. Hence, it may be decreed that

$$E[Q] = f(\rho, I, \bar{\lambda}_1/\hat{\lambda}_1, \bar{\lambda}_2/\hat{\lambda}_2, \ldots, \bar{\lambda}_n/\hat{\lambda}_n) = \text{Constant}. \qquad (5.1)$$

Now, if one of the quantities that $E[Q]$ depends upon should change, a dynamic algorithm (the details of which are under investigation) would determine the new value of I such that (5.1) is still valid. (One may also want to constrain higher moments of Q, but we shall limit ourselves to the simple case of the first moment.)

One expression for $E[Q]$ is the following upper bound expression for the G/D/1 Queue [Kle 76]:

$$E[Q] \le \frac{\sigma_a^2 \overline{\lambda^2}}{2(1-\rho)},$$

where $\bar{\lambda}$ is the cumulative mean arrival rate and σ_a^2 is the cumulative variance of interarrival times.

This expression of course ignores the burstyfactors $(\bar{\lambda}_i/\hat{\lambda}_i)$, but it is simple. Experiments however, reveal that it grossly overestimates the actual average queue lengths that are observed in simulations (by an order of magnitude or more). A tighter upper-bound is therefore desirable and is investigated next.

Notation

N = number of active sources;

$k_i = \bar{\lambda}_i/\hat{\lambda}_i \;\; 1 \le i \le N$. Arrange k_i in increasing order and $k_0 = 0$ and $k_{N+1} = \rho$.

The expected queue length is given by

$$E[Q] = \frac{1}{I}\int_I Q(t)\,dt.$$

This integral is equivalent to adding up the areas under the curve $Q(t)$. It can be shown that

$$E[Q] = \frac{1}{I}\sum_{j=1}^{N+1}\left\{\frac{Q(k_j I) + Q(k_{j-1} I)}{2}\right\}(k_j I - k_{j-1} I)$$

where Q(t) is the queue length at time t.

5.4. Signalling Protocols

The switches will choose from a set of intervals such that one of the above policies is satisfied. When new transmitters join or existing ones leave, the I value needs to be renegotiated. The transmitters may be signalled the I changes through a special field in their data packets. This field will be changed/set by the switches.

Constraining the interval size may be achieved in one pass through the switches. Increasing the size would, however, require a second pass because everyone in the path has to agree to such a change.

6. Related Work

NetBlt [CLZ 87], an end-to-end rate based flow control protocol was designed for high performance bulk data transfers. Data is transmitted in NetBlt as a sequence of ‘buffers,’ each consisting of several ‘bursts’ which are themselves decomposed into multiple ‘packets.’ The parameters negotiated between the transport end-points are burst-size, burst-rate, packet-size and buffer-size. The transmission rate per packet is burst-rate/burst-size when the transmission is active. Considerations in choosing the burst-rate are the sender’s timing granularity, the processing speed of the receiver and that of any intermediate gateways. There is provision for re-negotiating the burst-size and burst-rate parameters after each buffer transmission, but no performance studies have been reported on the tradeoffs involved.

There has been a considerable amount of work reported lately on open-loop control. Among them are scheduling strategies like Earliest Deadline [FeVe 90], Fair Queueing [Nag 87, Nag 85], Hierarchical Round-Robin [KKK 90], Stop-and-Go [Gol 91] and Virtual Clock [Zha 89]. These algorithms attempt to schedule traffic inside the network. Parallel work has focussed on admission control such as Distributed Source Control [RaDi 91] and the Leaky Bucket schemes [Tur 86, SLCG 89, BCS 90].

The problem of bandwidth allocation has been addressed in [AMS 82, DeTo 90, DTVV 88, GAN 90, GRF 89, JoVe 89, MGF 90, Sch 88, WoKo 90]. The studies by Anick et al. [AMS 82] and Gueren et al. [GAN 90] focus on equivalent bandwidth for on/off sources such that a given quality of service (Pr { loss $<\varepsilon$ } for some ε) is achieved. The principal difference between these studies and bandwidth allocation in the context of Dynamic Time Windows is the following. In the earlier studies, equivalent bandwidth was computed assuming that transmitters could be in the 'on' state for arbitrarily long periods and this increased the complexity of the analysis. The Dynamic Time Window mechanism forces the 'on' state to be less than or equal to the time window, thereby limiting the maximum length of the busy period at the switch. This simplifies equivalent bandwidth computations.

Considerable research has focussed on closed loop control. It has been studied in the context of window flow control by Jacobson [Jac 88], Ramakrishnan et al. [RaJa 88] and Mitra et al. [MiSe 90]. In these studies, the end-point window size is adjusted dynamically in response to network congestion. Preliminary studies on the dynamics of such feedback control has been reported in [BoSh 90, DKS 89, MuSt 91, Oht 89].

Resource allocation (and packet discarding techniques) for real time traffic in broadband networks is under considerable investigation, see for instance [AnHe 90, CGCS 88, HoSu 90, HlKa 88, Hui 89, MGF 90, OON 88, PeFr 90, ReMu 91, SMRA 89, SKT 90, ShMc 90, VPV 88].

7. Conclusions and Future Work.

We have presented a combined set of mechanisms for open-loop/closed-loop congestion control of data traffic over HSWANs. These mechanisms are intended to address network congestion of varying duration.

Dynamic Time Windows was proposed as a mechanism to provide a controlled load-dependent traffic shaping. A 'correct' end-point time window was found to effectively deal with queue length fluctuations at the switches, even for (medium-term) utilizations as high as ρ=0.98 (see Tables 4.2-4.3). For lower utilization levels, increasing the time window allowed for more more bursty traffic to go through without severely aggravating the congestion level at the switches. Simulations and analyses showed that the queue length statistics (in particular, $E[Q]$, $var(Q)$, and $Q_{\max}$) at the switches scaled with the time-window size. This, in conjunction with an expression for expected queue length, may be used for dynamic adjustment of time windows. We are investigating other possibilities at this time.

The Pulse scheduling algorithm was investigated in detail and its performance relative to Virtual Clock and FCFS was determined. The Pulse algorithm gave preferential service based on advertised peak rates, provided that the average rate was maintained over each time window resulting in good service to controlled bursts. The Virtual Clock protocol provided better service to smoother traffic. Both Pulse and Virtual Clock were found to be resilient to interference from misbehaving users. Therefore, both scheduling disciplines are appropriate for certain network traffic. Since Pulse is a superset of Virtual Clock, if Pulse is supported by the switch, then Virtual Clock can trivially be implemented.

Several analytical results were also obtained, in particular, the expected waiting time for Virtual Clock with the DTW arrival pattern, and necessary and sufficient conditions for a dynamic priority assignment curve to meet a (new) fairness criterion.

One result with respect to relative convergence and divergence of virtual clocks of several sources was found to be in contradiction to a previous finding in [Zha 90].

This study can be extended in several directions. We are currently investigating heuristics for dynamic time-window adjustment and the exact nature of medium-term congestion.

Acknowledgments

We thank Prof. Miron Livny for providing us with the DeNet software and both him and Rajesh Mansharamani for being extraordinarily helpful with DeNet. The simulations reported here were run on the Condor remote job scheduler, designed at Wisconsin.

References

[AMS 82] Anick, D., D. Mitra, M.M. Sondhi, ''Stochastic theory of a data-handling system with multiple sources,'' *Bell Sys. Tech. Journal*, pp. 1871-1894, Oct. 1982.

[AnHe 90] Anderson, D.P. and R. G. Herrwich, ''Resource management for digital audio and video,'' *IEEE Workshop on Real-Time Operating Systems and Software*, pp. 99-103, Charlottesville, May 1990.

[BCS 90] Bala, K., I. Cidon and K. Sohraby, ''Congestion Control in High Speed Packet Switched Networks,'' *Proc. IEEE INFOCOM*, pp. 520-526, June 1990. San Francisco,

[BoSh 90] Bolot, J., and A. U. Shankar, ''Dynamical behavior of rate-based flow control mechanism,'' *ACM SIGCOMM Computer Communication Review, vol. 20, pp. 35-49,* April 1990.

[CGGS 88] Cidon, I., I. Gopal, G. Grover and M. Side, ''Real-time packet switching: a performance analysis,'' *IEEE J. Select. Areas Commun.*, vol. 6, pp. 1576-1586, Dec. 1988.

[CLZ 87] Clark, D.D., M.L. Lambert and L. Zhang, ''NETBLT: A bulk data transfer protocol,'' RFC 998, March 1987.

[DeTo 90] Decina, M., and T. Toniatti, ''On bandwidth allocation to bursty virtual connections in ATM networks,'' *Proc. ICC,* pp. 844-851, 1990.

[DeNet 88] The DeNet User's Guide, M. Livny, 1986-1990, University of Wisconsin, Madison.

[DKS 89] Demers, A., S. Keshav and S. Shenker, ''Analysis and simulation of a fair queueing algorithm,'' *Proc. ACM SIGCOMM, Austin*, pp. 1-12, Sept 1989.

[DTVV 90] Decina, M., T. Toniatti, P. Vaccari and L. Verri, ''Bandwidth aassignment and virtual call blocking in ATM networks,'' *Proc. of the IEEE INFOCOM, San Francisco,* pp. 881-888, June 1990.

[FeVe 90] Ferrari, D., and D. Verma, ''A scheme for real-time channel establishment in wide-area networks,'' *IEEE J. Select. Areas Commun.*, vol. 8, pp. 368-379, April 1990.

[GAN 90] Guerin R., H. Ahmadi and M. Naghshineh, ''Equivalent capacity and its application to bandwidth allocation in high-speed networks,'' *IEEE J. Select. Areas Commun.,* vol 9, pp.

968-981, Sept. 1991.

[Gol 91] Golestani, S. J., "A framing strategy for congestion management," *IEEE J. Select. Areas Commun.,* vol. 9. pp. 1064-1077. Sept. 1991.

[Gre 78] Griffeath, D., ‘‘Coupling methods for Markov processes,’’ *Advances in Mathematics Supplemental Studies,* Vol. 2, Academic Press, pp. 1-43, 1978.

[GRF 89] Gallassi, G., G. Rigolio and L. Fratta, ‘‘ATM: Bandwidth assignment and bandwidth enforcement policies,’’ *Proc. IEEE GLOBECOM, Dallas,* pp. 1788-1793, Nov. 1989.

[HoSu 90] Hong, D.K. and T. Suda, ‘‘Survey of techniques for prevention and control of congestion in an ATM network,’’ *Proc. ICC 1990.*

[HKM 91] Hahne, E.L., C. R. Kalmanek, S. P. Morgan, ‘‘Fairness and congestion control on a large ATM data network with dynamically adjustable windows,’’ Proc. ITC 13, Copenhagen, June 1991.

[HlKa 88] Hluchyj, M.G., and M.J. Karol, ‘‘Queueing in high-performance packet switching,’’ *IEEE J. Select. Areas Commun.,* vol. 6, pp. 1587-1597, Dec. 1988.

[Hui 88] Hui, J.Y., ‘‘Resource allocation for broadband networks,’’ *IEEE J. Select. Areas Commun.,* vol. 6, pp. 1598-1608, Dec. 1988.

[Jac 88] Jacobson, V., ‘‘Congestion avoidance and control,’’ *Proc. ACM SIGCOMM, Stanford,* pp. 314-329, Aug. 1988.

[JoVe 89] Joos P., and W. Verbiest, ‘‘A statistical bandwidth allocation and usage monitoring algorithm for ATM networks,’’ *Proc. ICC,* pp. 415-422, 1989.

[KKK 90] Kalmanek, C.R., H. Kanakia and S. Keshav, "Rate Controlled Servers for Very High Speed Networks," *Proc. IEEE GLOBECOM, San Diego, pp. 12-20, Dec.* 1990.

[Kle 76] Kleinrock, L., *Queueing Systems Vol II: Computer Applications* , John-Wiley and Sons, New York, 1976.

[Livny 87] Livny, M., ‘‘DeLab - A simulation laboratory,’’ *Winter Simulation Conference,* 1987.

[MGF 90] Monteiro, J.A.S., M. Gerla and L. Fratta, ‘‘Statistical multiplexing in ATM networks,’’ *Proc. 4th Int’l Conference on Data Commun. Systems and Their Performance* , Barcelona, June 1990.

[MiSe 90] Mitra, D. and J. Seery, ‘‘Dynamic adaptive windows for high speed data networks: theory and simulations,’’ *Proc. ACM SIGCOMM,* pp. 30-37, Philadelphia, September 1990.

[MuSt 91] Mukherjee, A. and J.C. Strikwerda, ‘‘Analysis of dynamic congestion control protocols --- a Fokker-Planck approximation,’’ *Proc. ACM SIGCOMM, Zurich, pp. 159-169,* Sept. 1991.

[Nag 85] Nagle, J., ‘‘On packet switches with infinite storage,’’ RFC 896, 1985.

[Nag 87] Nagle, J., ‘‘On packet switches with infinite storage,’’ *IEEE Trans. Commun., COM-35,* pp. 435-438, April 1987.

[Oht 89] Ohta, M., ‘‘Engineering of congestion control in a large scale network,’’ *Proc. 4th Int’l Network Planning Symposium* , Palma de Mallorca, Spain, pp. 301-305, Sept. 1989.

[OON 88] Ohnishi, H., T. Okada and K. Noguchi, ‘‘Flow control schemes and delay/loss tradeoff in ATM networks,’’ *IEEE J. Select. Areas Commun.,* vol. 6, pp. 1609-1616, Dec. 1988.

[PeFr90] D. W. Petr and V. S. Frost, ‘‘Optimal packet discarding: an ATM-oriented analysis model and initial results,’’ *Proc. IEEE INFOCOM,* San Francisco, pp. 537-542, June 1990.

[RaJa 88] Ramakrishnan, K.K., and R. Jain, ‘‘A Binary Feedback Scheme for Congestion Avoidance in Computer Networks with a Connectionless Network Layer,’’ *Proc. ACM SIGCOMM, Stanford,* pp. 303-313, Aug. 1988. Also, .ul ACM Trans. Computer Systems, vol. 8, pp. 158-181, May 1990.

[RaDi 91] Ramamurthy, G., and R.S. Dighe, "Distributed source control: a network access control for integrated broadband packet networks," *IEEE J. Select. Areas Commun.,* vol. 9, pp. 990-1002, Sept. 1991.

[ReMu 91] Rehman, J., and A. Mukherjee, ‘‘A simple resource allocation model for variable bit rate video sources,’’ in preparation.

[Sch 88] Schoute, F. C., ‘‘Simple decision rules for acceptance of mixed traffic streams,’’ *Proc. 12th Int’l Teletraffic Congress,* pp. 771-777, 1988.

[SMRA 89] Sen, P., B. Maglaris, N.E. Rikli and D. Anastassiou, ‘‘Models for packet switching of variable-bit-rate video sources,’’ *IEEE J. Select. Areas Commun.,* vol. 7, pp. 865-869, June 1989.

[SKT90] Schulzrinne, H., J. F. Kurose, and D. Towsley, ‘‘Congestion control for real-time traffic in high-speed networks,’’ *Proc. IEEE INFOCOM*, San Francisco, pp. 543-550, June 1990.

[ShMc 90] Shacham, N., and P. McKenney, ‘‘Packet recovery in high-speed networks using coding and buffer management,’’ *Proc. IEEE INFOCOM,* San Francisco, pp. 124-131, June 1990.

[SLCG 89] Sidi, M., W. Liu, I. Cidon and I. Gopal, ‘‘Congestion control through input rate regulation,’’ *Proc. IEEE GLOBECOM, Dallas, pp. 1764-1768,* Nov. 1989.

[Tur 86] Turner, J., ‘‘New directions in communications (or which way to the information age?),’’ *IEEE Comm. Magazine,* vol. 24, pp. 8-15, Oct. 1986.

[VPV 88] Verbiest, W., L. Pinnoo and B. Voeten, ‘‘The impact of the ATM concept on video coding,’’ *IEEE J. Select. Areas Commun.,* vol. 6, pp. 1623-1632, Dec. 1988.

[WoKo 90] Woodruff, G.M. and R. Kositpaiboon, ‘‘Multi-media traffic management principles for guaranteed ATM network performance,’’ *IEEE J. Select. Areas Commun.,* vol. 8, pp. 437-446, April 1990.

[Zha 91] Zhang, L., ‘‘Virtual clock: a new traffic control algorithm for packet switching networks,’’ ACM TOCS, vol. 9, pp. 101-124, Nov. 1991.

Appendix A Properties of Dynamic Priority Assignment Curves

Notation:

Let

Δt= some time period.

N_i= number of source i packets served at a switch over Δt.

N= total number of packets served at the switch over Δt.

$\hat{\lambda}_i$= advertised peak transmission rate of flow i.

Fairness Criterion:

$$N_i \propto \hat{\lambda}_i, \quad \forall i$$

Assumptions:

With each flow i, there exist:

(a) an arrival curve $a_i(n)$ which gives the time of arrival of the *nth* packet.

(b) a priority curve $f_i(n)$ which specifies the relative priority of this packet with respect to the others. This priority determines which packet is to be scheduled next by the switch. A lower value of $f_i(n)$ implies a higher priority.

Proposition:

(a) If all traffic is 'homogeneous,' then any arbitrary single priority assignment curve will guarantee fairness as defined by (A.1). By homogeneous traffic we mean that either

(i) $a_i(n)=a_j(n) \quad \forall\, i,j,n$ deterministically, or

(ii) $a_i(n)$ equals $a_j(n) \quad \forall\, i,j,n$ in distribution.

(b) and there is queueing at the server, then it is necessary and sufficient for fairness (as stated in (A.1)), that the following hold:

if $a_1(n)<a_2(n)$ then $f_1(n)<f_2(n) \quad \forall n.$

(Aside: If there is no queueing at the server, then there is no need for any scheduling discipline anyway.)

(c) $N_i=\dfrac{\hat{\lambda}_i}{\sum_j \hat{\lambda}_j}N$, where N is conserved, i.e., it is constant across all scheduling disciplines.

Proof

($a.i$) When $a_i(n)=a_j(n) \ \forall\, i,j,n.$ The proof is trivial.

($a.ii$) When $a_i(n)$ equals $a_j(n)$ in distribution, $\forall\, i,j,n.$ The proof is in two parts. First consider a finite interval I. It is possible that flows i and j will have different sample paths for their arrivals over I. Denote the sample paths $s_i^{(1)}$ and $s_j^{(1)}$ and let us assume that their probabilities of occurrence are p_0 and p_1 respectively. Let the average waiting times for these sample paths be $w_i^{(1)}$ and $w_j^{(1)}$ respectively. Since $a_i()$ and $a_j()$ are homogeneous, the same two sample paths may be interchanged, and this would give $w_i^{(2)}=w_j^{(1)}$ and $w_j^{(2)}=w_i^{(1)}$ with probabilities p_0 and p_1 interchanged as well. Therefore the average waiting time over the interval I is the same for i and j.

Next, consider a sequence of successive time intervals $I_1, I_2, \cdots$. The flows i and j may have transmitted different number of packets (say i_1 and j_1) in interval I_1. If they both started off with 0 packets transmitted, their statistical behavior (and the service obtained) over the first interval is the same. However, the relative priorities over the next interval would be dependent on the number of packets transmitted over the previous interval if the priority curves are not 'corrected' for unused packets over the interval. Is it possible that the number of packets transmitted by the two flows diverge from each other as the number of intervals increase? The theory of coupled stochastic processes says no.

The theory goes as follows. Consider a sequence of finite intervals $I_1, I_2, ...$, and let the number of packets transmitted in these intervals by flows i and j be $i_1, i_2, \cdots$ and $j_1, j_2, ...$ Then the following 'coupling' result holds.

Theorem [Griffeath 78]

Let $i_1, i_2, \cdots$ and $j_1, j_2, \cdots$ be independent and identically distributed random variables. Let $S_i(k)=i_1+i_2+\cdots+i_k$ and $S_j(k)=j_1+j_2+\cdots+j_k.$ Then $\exists\ N$ such that after some finite number of intervals, the total number of packets transmitted by i and j must be equal with probability one.

As a consequence of this theorem, the two 'homogeneous' flows will converge every so often (with probability one). Over the period in between, the two flows may follow different sample paths and receive different service. However, the point where they meet serves as a renewal epoch and over each such renewal epoch, the distribution of the sample arrival paths is the same. Hence, the flows will receive statistically similar service asymptotically as the number of intervals increase.

($b.i$) Consider an interval of time I. Suppose $a_1(n)$ and $a_2(n)$ are sample paths of flows 1 and 2 over I. Suppose $f_1(n)=a_1(n), f_2(n)=a_2(n)$. Now let n_1 and n_2 be such that $f_1(n_1)=a_1(n_1)=f_2(n_2)=a_2(n_2)$. Without loss of generality, let $\dfrac{da_1(n_1)}{dn}<\dfrac{da_2(n_2)}{dn}$ i.e., $a_1(n_1+1)<a_2(n_2+1)$. Let,

$$a_1(n_1+k)\approx a_2(n_2+1)$$
$$\rightarrow a_1'.k = a_2'.1$$
$$\rightarrow k/\lambda_1 = 1/\lambda_2$$
$$\rightarrow N_1/N_2 = k = \lambda_1/\lambda_2$$

which proves the sufficiency condition.

($b.ii$) We next prove the necessary condition in the proposition. The proof is by contradiction. Assume there are two flows 1 and 2 and there is a single priority assignment curve, $f_1()=f_2()=f()$. Let the two (different) arrival curves be $a_1()$ and $a_2()$. **WLOG,** let $a_1(n)<a_2(n)$ for some interval I.

Assume that at a time t in the interval, there is queueing, i.e.,

--
--

$$\frac{da_1}{dt} + \frac{da_2}{dt} > \mu$$

i.e.,

$$\frac{1}{a_{1'}} + \frac{1}{a_{2'}} > \mu$$

Let $a_1(n_1) = a_2(n_2) = t$. Our assumptions imply that n_1 is greater than n_2 and since $f()$ is a monotonically increasing function, $f(n_2)$ is less than $f(n_1)$. Clearly, the slower traffic gets higher priority in this case, and so (A.1) cannot hold. This proves the proposition in the reverse direction (i.e., the necessary condition).

Corollary to $(b.ii)$

If the priority assignment curves (say $f_1()$ and $f_2()$) are such that (1) is satisfied for a set of arrival curves $a_1()$ and $a_2()$, then for any constant C greater than 0, $Cf_1()$ and $Cf_2()$ will satisfy (A.1) as well.

(c) The proof is trivial.

Appendix B: Waiting Time Analysis of Virtual Clock

The average waiting time for each flow, i, is computed in this subsection. The sources are assumed to transmit packets at their peak rates. However, they are assumed to be well behaved in that they maintain their specified average over the specified interval. The switch is assumed to schedule according to the lowest virtual clock.

Let the number of packets transmitted in an interval by source i be $n_{i.}$ Let $A_{i(k)}$ and $D_{i(k)}$ be the arrival time and departure time, respectively, of the kth packet of flow $i, k=0,1,...,n_{i.}$ Then the average waiting time of flow i packets is

$$\bar{w}_i=\frac{1}{n_i}\sum_{k=1}^{n_i}\left[D_i(k)-A_i(k)\right]$$

$$=\frac{1}{n_i}\sum_{k=1}^{n_i}D_i(k)-\frac{1}{n_i}\sum_{k=1}^{n_i}A_i(k)$$

$$=\bar{D}_i-\bar{A}_i$$

$\bar{A}_i$ may be obtained directly from the I-arrival distribution. Since the packets from flow i are transmitted at peak rate $\hat{\lambda}_{i,}$ the kth packet arrives at $k/lambdahat_i$ from the beginning of the interval (this latter point is used as the reference point for measuring time in this analysis). Therefore

$$\bar{A}_i=\frac{1}{n_i}\sum_{k=1}^{n_i}A_i(k)$$

$$=\frac{1}{n_i}\left[1/\hat{\lambda}_i+2/\hat{\lambda}_i+\cdots+n_i/\hat{\lambda}_i\right]$$

$$=\frac{1}{\hat{\lambda}_i}\left(n_i+\frac{1}{2}\right)$$

The average departure times, $\bar{D}_i$ obey a rather startling conservation law as stated in the following lemma.

Lemma {} If $B(mu)$ denotes the busy period in an interval (as a function of μ), and the virtual clock algorithm is used for scheduling, then

$$\bar{D}_i=B(\mu)/2 \qquad i=1,2,...,J$$

Notice that $\bar{D}_i$ is independent of i. $B(\mu)$ is computed over all active flows $i=1,2,...,J$.

Proof

Define a 'round' such that every flow gets to transmit at least one packet in the round and there exits at least one flow that transmits only one packet in a round. Assume the round size is R 'slots'.

If there are a total of c rounds in the busy period, then

$$B(\mu)=\frac{1}{\mu}\sum_{i=1}^{J}n_i$$

and

$$\sum_k D_i(k)=\frac{1}{\mu}(n_{i,1}\tilde{x}_i+n_{i,2}(R+\tilde{x}_i)+n_{i,3}(2R+\tilde{x}_i)+\cdots n_{i,c}((c-1)R+\tilde{x}_i))$$

where $n_{i,j}$ is the number of packets transmitted by flow i in round j. For simplicity, we have approximated the time used inside of a round by $\tilde{x}_{i.}$ In the subsequent discussion, $xtilde_i$ is assumed to be an i.i.d. random variable with mean $Ex_{i.}$ Now,

$$n_{i,1}=n_{i,2}=\cdots=n_{i,c}=n_{i/c}$$

Therefore,

$$\sum_i D_{i(k)}=\frac{1}{\mu}\cdot\frac{n_i}{c}\left(\tilde{x}_i+R+\tilde{x}_i+2R+\tilde{x}_i+\cdots+(c-1)R+\tilde{x}_i\right)$$

and

$$E\left(\sum_i^{n_i}D_i\frac{(k)}{n_i}\right)=\frac{1}{\mu}\frac{1}{c}\left\{cEx_i+R[1+2+\cdots+(c-1)]\right\}$$

$$=\frac{1}{\mu}\left(Ex_i+R\frac{(c-1)}{2}\right)$$

$$\approx B(\mu)/2$$

$$=\frac{1}{2}\mu\sum_i^J n_i$$

This proves the lemma. The average waiting time is therefore given by

$$\bar{w}_i=\frac{1}{2\mu}\sum_i^J n_i-\frac{1}{\hat{\lambda}_i}\left(\frac{n_i+1}{2}\right)$$