On CBR Service

January 26, 1995

Abstract

Although CBR service is relatively well understood, there are several open issues that must be resolved before large scale CBR networks can be provisioned and built. In this paper, we investigate the performance of CBR traffic in the context of large-scale networks, where many connections and switches coexist and interact. For this, we develop a framework for simulating such networks, decoupling the influence of breadth and depth. The performance metrics used are the end-to-end delay histogram and metrics derived thereof, such as the 99%-percentile, the mean or the histogram width, and the index of dispersion for intervals (IDI), which we use to assess bunching (or clustering) in superposition streams.

Our results are briefly as follows: we found that CBR traffic can be efficiently transported by the First Come First Served (FCFS) scheduling discipline, which has the least implementation cost. Delays incurred by cross traffic composed of many CBR streams with different bandwidths and phases do not exceed a few cell times even under heavy load, which means that buildout buffers of 10 to 20 cells seem to be sufficient after traversing 20 switches. We also show that the Round Robin (RR) and Weighted Round Robin (WRR) disciplines are ill suited for CBR traffic, both in terms of performance and implementation complexity.

We compare two analytical approximation methods, based respectively on *M/D/1* queues and on the Multiclass Parametric Decomposition Method, with the simulation results and found them to be suitable to estimate delays for the FCFS discipline.

1 Introduction

Our interest in Constant Bit Rate (CBR) traffic has several reasons. It will probably be the first service class offered by the BISDN. It is backwards compatible with circuit-switched networks, which means that a large user base already exists for this type of service: current video and audio codecs mostly produce CBR traffic. Former leased-line users wishing to switch to Virtual Private Networking (VPN) will use CBR service as well, emulating mostly T1 and T3 lines. Finally, CBR is easier to describe, handle and administer for both the user and the network than service types with more degrees of freedom, such as Variable Bit Rate (VBR).

However, there still are open issues that have to be addressed to build a large-scale network offering CBR service. For example, it is not clear yet what the effect of bunching or clustering is on a CBR stream as it travels through many switches, and what buildout buffer sizes at the endpoint are necessary to restore initial cell equispacing. Also, it is not clear what scheduling discipline among the obvious candidates First Come First Served (FCFS), Round Robin (RR) and Weighted Round Robin (WRR) is best suited, especially when implementation complexity is taken into account as well. Finally, good analytical approximations for end-to-end delays and buildout buffer sizes for provisioning and call admission are desirable.

We address all these issues in this paper. The goal is to come up with a set of engineering guidelines that will support the deployment of CBR service. In Section 2 we give a survey of related literature. Section 3 discusses the methodology used, and Section 4 presents results from simulation and analysis. Section 5 summarizes the results, and Section 6 concludes the paper.

2 Related Work

In this section we review past work in the quantitative analysis of CBR traffic. Roberts and Virtamo [17] derive the queue size distribution for superposed CBR streams with identical periods, and upper and lower bounds on this distribution for different periods. However, they do not discuss the properties of the output process of the queue. Matragi, Bisdikian and Sohraby [15, 16] derive analytical approximations for jitter for the single and the multihop case. However, their notion of jitter is based on the difference of the interdeparture process at subsequent switches and does not allow to derive the end-to-end delay histogram, which is necessary to estimate buildout buffer sizes. Gruenenfelder [10] observes that the end-to-end delay of a reference connection going through multiple queues where it is multiplexed with cross traffic depends largely on the autocovariance of the latter. DeSimone [6] studies a network of three queues in tandem with one cross traffic stream and uses squared coefficients of variation as a measure of burstiness. He uses simulation and a simple analytic model to compare cell-level and packet-level FCFS and Round Robin (RR). He has shown that with FCFS scheduling, the squared coefficient of variation of the departure process of a CBR stream increases with load, and that this increase is larger for FCFS scheduling than for RR scheduling. However, he has studied only a single link, and only two interacting streams. Whitt [20] discusses how squared coefficients of variation can be applied to approximate open queueing networks, and also presents results for the case of multi-class networks [21]. Golestani [9] has shown by an example that for FCFS, bursts can be formed simply by superposing CBR streams. He develops a measure for traffic smoothness and a scheduling discipline, called *stop-and-go queueing*, that guarantees that the smoothness criterion is preserved as streams go through switches. Cruz [4, 5] analytically derives hard delay bounds for different scheduling disciplines and leaky-bucket compliant traffic. Unfortunately, it has been observed that these bounds can be fairly loose under practical circumstances [22]. Banerjea and Keshav derive tight queueing delay bounds for two non work-conserving scheduling disciplines, HRR and Stop-and-go queueing [2]. Kurose, instead of finding absolute delay bounds, derives bounds on the distribution of the end-to-end delay [13], which is interesting if statistical QoS guarantees are sufficient.

To our knowledge, several aspects of our work are new. First, we are interested in large-scale networks, where many connections travel through many hops. Some of our experiments involve as many as 600 cross traffic connections interacting with a reference connection. Second, we consider not only scheduling performance, but also attempt to consider the *implementation cost* of scheduling disciplines in relation with their performance.

3 Approach

We present the metrics used to assess delay, clustering (or bunching) and implementation complexity. We also show how the RR scheduler works in detail.

3.1 Delay metrics

The 99%-percentile delay is important as it measures the delay seen after reconstruction of the initial equal cell spacing through buffering (while allowing for 1% of cell loss). It also determines the amount of buffering needed at the end points, which translates directly into a hardware cost for memory. In other words, we consider how much buffering has to be done to recreate a constant-delay connection. Note that for many real time users, it is more important that the delay be as constant as possible (i.e. small delay jitter) than that this delay be very low. For example, a delay difference of 100ms in a video transmission can hardly be perceived. However, 100ms of delay jitter will result in a rather "jerky" image.

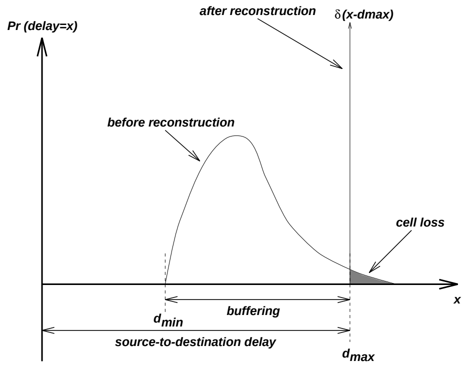

Figure 1: The relationship between the end-to-end delay histogram measured *before* the reconstruction buffer, and how buffering and delay *after* the reconstruction buffer can be determined.

To estimate the buildout buffer size, assume that all cells experience minimum delay d_{min} and have to be buffered during an interval of length $(d_{max} - d_{min})$ in order to reconstruct the nominal delay d_{max}. Furthermore, assume that the cells arrive at their peak rate $c_{max} = 1/T_{min}$. This results in the maximum number of cells P the buildout buffer has to accommodate.

$$P = \lceil \frac{d_{max} - d_{min}}{T_{min}} \rceil = \lceil c_{max}(d_{max} - d_{min}) \rceil \tag{1}$$

Fig. 1 shows how d_{min} and d_{max} are related to the end-to-end delay histogram.

3.2 Clustering metrics

We measure clustering of a cell stream with the *index of dispersion for intervals (IDI)* of its interarrival process, which provides us with the variability of cell arrivals on different time scales. The IDI for an interarrival process $\{X_i\}$ is defined as follows [11, 7]:

$$J_i(n) = \frac{n \cdot \mathrm{var}[\sum_{k=1}^{n} X_{i+k}]}{\mathrm{E}^2[\sum_{k=1}^{n} X_{i+k}]} \tag{2}$$

If the process is assumed to be wide-sense stationary (that is, the coefficient of variation is constant in the observed interval), then $J_i(n) = J(n)$ for all i. The function $J(n)$ describes the variation of arrivals over different time frames. Note that for a Poisson process, $J(n) = 1$ for $n = 1, 2, \ldots$, and for a CBR stream, $J(n) = 0$. Also note that $J(1)$ is the squared coefficient of variation c^2 of the stream. Furthermore, the IDI of point processes with positive correlation coefficients monotonically increase in n [11]. This can be seen if relation (2) is rewritten as

$$J(n) = J(1)\left(1 + 2\sum_{j=1}^{n-1}\left(1 - \frac{j}{n}\right)\rho_j\right) \tag{3}$$

with $\rho_j = \mathrm{cov}[X_i, X_{i+n}]/\mathrm{var}[X_i]$. In other words, if $J(n)$ increases up to a certain $n = n_0$ and then remains constant for $n > n_0$, then we have correlation in the interarrival process up to a lag of n_0. Thus, the IDI is a powerful metric to measure burstiness over different time scales and to detect correlation in arrival processes.

3.3 Complexity metrics

As hardware implementation cost in VLSI directly depends on the required chip area, which in turn is dominated by *memory*, the cost of an ATM switch depends mostly on its state memory size. Thus, it makes sense to consider the amount of state a scheduling discipline requires. As the memory necessary to hold the cells itself is independent of how the cells are scheduled, we do not take it into account.

We assume that pointers consist of $n_p = \lceil \log_2(P) \rceil$ bits, where P is the number of words in the memory that can be addressed. Also, as all disciplines put cells either in an overall queue (FCFS) or in a per-connection queue, there is a pointer of size n_p associated with each cell. This is not counted as state information, either. In addition to per-cell pointers, queues need a head and a tail pointer, in order to know where to add and where to remove cells. Each queue accounts therefore for $2n_p$ bits of state. We call N the number of connections and Q the maximum size of a service quantum, with $n_q = \lceil \log_2(Q) \rceil$. M is the number of levels for HRR and RCSP. F is the maximum frame size for HRR, with $n_f = \lceil \log_2(F) \rceil$.

Scheduling discipline	amount of state [bits]
FCFS	$2n_p$
RR	$N(3n_p + 1) + 2n_p$
WRR	$N(3n_p + n_q + 1) + 2n_p + n_q$
HRR	$N(4n_p + n_q + 1) + M(5n_f + 4n_p)$

3.4 Round Robin scheduler

For the sequel, it is helpful to define the RR/WRR implementation adopted, which is based on *service lists* (cf. Fig. 2). The service list contains connection identifiers and determines the order in which connections are served. Each connection identifier can be contained in the service list at most once. The connection at the head of the service list is the one being serviced. After service has completed for this connection, its identifier is removed from the head of the service list. It is appended to its end if there are still cells queued for this connection (i.e., the connection is still active). If a cell arrives at an inactive connection, then the connection identifier is appended to the *end* of the service list. [1]. If a cell arrives at an active connection, then the service list is left unchanged. For RR,

[1]Note that if they were appended to the head of the service list, it would be possible for malicious users to get more than the fair share of the link bandwidth by sending cells at a rate that would leave the queue empty for a little moment after serving a cell, thus becoming

connections are served one cell at a time. For WRR, they are served up to a maximum number of cells given by the `service_quantum` associated with the connection. When service for a connection begins, a counter `remaining` is initialized to the `service_quantum` of that connection and decremented each time a cell is served. When the counter reaches zero or no more cells are queued for this connection, service terminates.

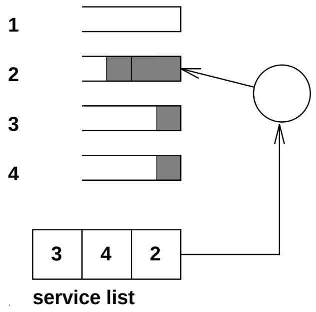

Figure 2: A RR/WRR implementation using service lists is considered. This approach is preferable from a hardware point of view, as no "search" operations need to be performed to find the next connection requiring service.

4 Results

The topology we used for the first set of experiments aims at looking at one single traffic stream (the *tagged connection*) and how its characteristics are altered as it interacts with other traffic streams, called *cross-traffic*. In other words, we want to study the impact of network *depth*. For this purpose, we use the *multihop* topology as depicted in Fig. 3. This model is a fairly general one and has been used in the literature [10, 15, 16]. Another interesting approach is outlined in [22, 14], where the cross-traffic itself interacts with other traffic streams before interacting with the tagged connection. The topology depicted in Fig. 4, on the other hand, allows to study network *breadth*, i.e. the characteristics of a stream that is a superposition of many individual streams.

Due to the deterministic nature of CBR, it is not straightforward to see why its scheduling would need further consideration. It is necessary to differentiate between two cases. In the first case, the interpacket spacing T for all streams is the same. The superposition stream then has periodicity T as well. Burstiness results purely from the phase relationships between the constituent streams. If the phases are arbitrary and independent, then ensemble averages for the delay experienced by such a CBR stream can be derived [18, pages 117-119]. The maximum delay a cell can experience in such as case is T. In the second case, which we are going to consider below, bandwidths *and* phases are going to be arbitrary. We believe that this case has a lot of practical importance and has not received enough attention so far, as it is very probable that in future ATM networks, users will be allowed to choose the bandwidth they desire with a small granularity.

To get an idea of what can result from a superposition of such CBR streams, consider Fig. 5. It is remarkable that the superposition of only four streams can exhibit so much variation.

Note that the superposition process does not necessarily have to be periodic. In fact, the superposition process is non-periodic if and only if two constituent CBR streams exist such that the ratio between their respective interarrival

inactive, and shortly after sending the next cell, which would then get immediate service, and so on.

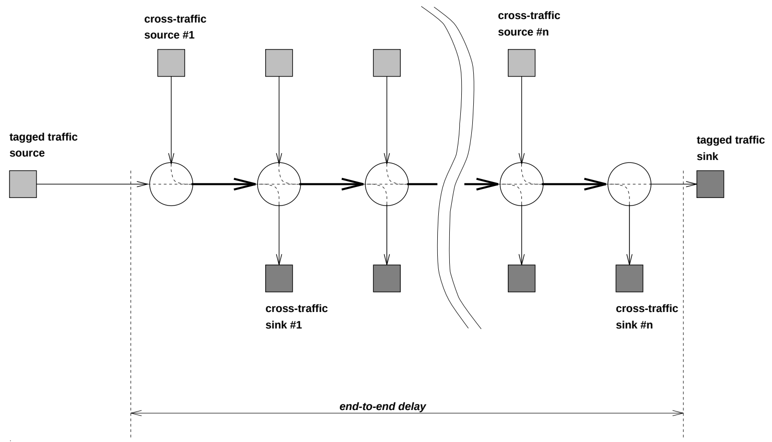

Figure 3: The *multihop* topology used to simulate the interactions with other streams and the resulting impact on the characteristics of a "tagged" connection.

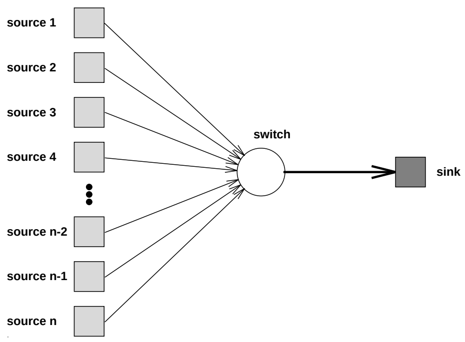

Figure 4: The topology used to study the characteristics of a CBR superposition process.

times is irrational. In practice, users will probably be required to select the bandwidth of a CBR connection as a multiple of some base bandwidth $1/T$. The resulting superposition process will then have a maximum period of T.

As we have already outlined above, we will first study the superposition of CBR streams to study the impact of breadth, and then the impact of depth through the multihop topology. Then we extend the multihop experiment by replacing the CBR cross traffic with superpositions of CBR streams. This will allow us to draw conclusions about large networks, where each connection typically travels through many hops, and we will be able to compare the performance of two scheduling disciplines. We will compare this performance based on the 99%-percentile end-to-end delay (Fig. 1) and the mean end-to-end delay.

In all experiments, we compare the FCFS with the RR scheduling discipline. The main goal is to find out if FCFS performs well enough under the circumstances described above so that the additional implementation complexity of a discipline providing for explicit fairness between connections, such as RR, is not required.

Some simulation parameters that were common to all of the experiments below are given in the following table:

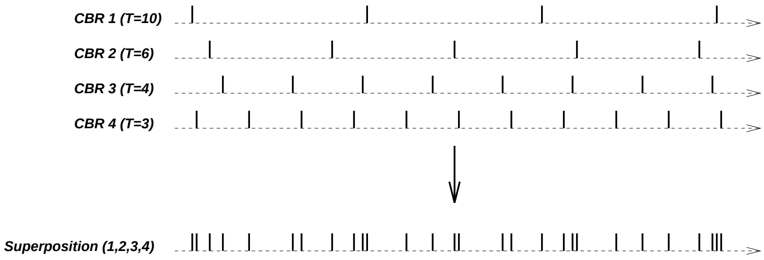

Figure 5: An illustration of the creation of burstiness through superposition of CBR sources: four sources with different bandwidths and phases as well as the resulting aggregate stream are shown. The aggregate stream exhibits burstiness and a "pseudo-stochastic" behavior.

Simulation parameters:			
line propagation delay	switching delay	cell size	switch buffer size
0	0	1000 bit	∞

4.1 Multiplexed CBR

We have simulated CBR streams coming into a switch (cf. Fig. 4) and leaving the switch on a common outgoing link. The total bandwidth of the incoming streams is partitioned in the following way: We call *target bandwidth* the average bandwidth of the superposition stream. The first stream's bandwidth is a uniform random variable on the interval from 0 to half of the target bandwidth. The second stream's bandwidth is chosen in the same way between 0 and the remaining bandwidth, i.e. the target bandwidth minus the first stream's bandwidth, and so on. The last stream gets the remaining bandwidth such that all streams sum up to the target bandwidth. This procedure gives a mix of high and low speed streams. Our results are averaged over several bandwidth partitions.

Simulation parameters:		
incoming line speed	outgoing line speed	target bandwidth
1Mb/s	100 Mb/s	800 kb/s

The fairly large IDI at a time frame of a couple of cells (for comparison, a Poisson process has an IDI of 1) indicates considerable variability on the outgoing link. As the observed interval increases (n large), the arrival variability decreases (which means that the arrival process does not have a positive correlation function). Furthermore, we observe that the IDI function increases over the entire range of n as more connections are multiplexed together.

It is important to note that this phenomenon has nothing to do with the scheduling discipline, and any work-conserving discipline would show the same results. As a matter of fact, the outgoing link can be considered infinitely fast, such that no contention occurs at the multiplexing point. The nonzero index of dispersion of the interarrival process on the outgoing link is a pure result of the *superposition* of the incoming CBR streams.

These results suggest that a superposition of CBR connections carried by one link, for example in a backbone

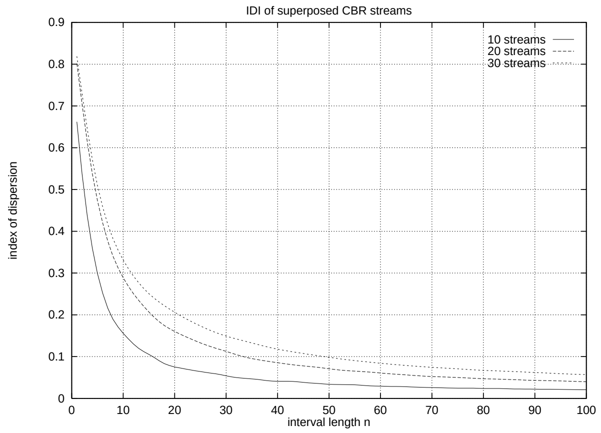

Figure 6: The IDI for the superposition of a number of CBR streams with arbitrary bandwidths.

network, cannot be replaced by an "equivalent" CBR stream with a bandwidth equal to the sum of the bandwidths of the constituent streams when assessing scheduling performance. Furthermore, the order of magnitude of the IDI function for small n suggests the use of a Poisson process as an approximation of the cell arrival for the study of scheduling performance when delays in the order of a few cells are of importance.

4.2 Multihop CBR with CBR cross traffic

In the following experiment, we look at a CBR stream going through a number of switches, sharing each link with another CBR stream. The link utilization was set to a fairly high value ($\rho = 0.8$). The tagged stream's bandwidth was then varied over a range of values (and the cross traffic accordingly, to keep total utilization at ρ). We were interested in determining how the CBR stream changes as it moves through the network[2].

Simulation parameters:				
line speed	traffic intensity ρ	tagged traffic intensity ρ_T	cross traffic intensity ρ_{CT}	transmission delay (end-to-end)
1Mb/s	0.9	$0.10 \ldots 0.80$	$0.80 \ldots 0.10$	22ms

We have simulated a CBR stream (subsequently called *tagged stream*) going through 21 switches. At each switch except the last one, a cross traffic stream enters the network, shares one link with the tagged stream and exits from the network at the next switch (cf. Fig. 3). The cross traffic streams all have the same bandwidth. Bandwidth allocation is usually done in multiples of some minimum bandwidth (the *bandwidth granularity*). If L is the cell size and T the intercell time for this minimum bandwidth, then bandwidth allocation can be done in multiples of L/T. If the tagged traffic has bandwidth $A_T(L/T)$ and the cross traffic $A_{CT}(L/T)$, with $A_T, A_{CT} \in \mathbb{N}$, then the cell delay pattern is periodic with period

$$T_p = \frac{A_T A_{CT}}{\gcd(A_T, A_{CT})} T \tag{4}$$

[2] This transmission delay is composed as follows: 20 cell service times on the links connecting switches, one cell service time on the tagged stream's access link, and one cell service time on the link going into the sink.

For example, if $A_T = 6$ and $A_{CT} = 15$, then the cell arrival process seen on the common link is periodic with period 30 cells. This cell delay pattern is highly dependent on the phases of the two streams. For example, if the two streams have the same bandwidth allocation, then three situations can occur, depending on the phase difference between the two streams: (1) no cells are delayed; (2) every tagged cell is delayed; (3) every cross traffic cell is delayed. As we do not wish to assume anything about the phase relationship between the connections, we would like to measure ensemble averages over all possible phases. We approximate this in our simulation by introducing arbitrary phase changes in intervals of length $\gg T$. Another reason for the interest in ensemble averages is the possibility of clock drifts between the user and the network, which would destroy this periodic delay patterns. Although ensemble averages are often derived analytically, as far as we know, this method of simulating ensemble averages is new.

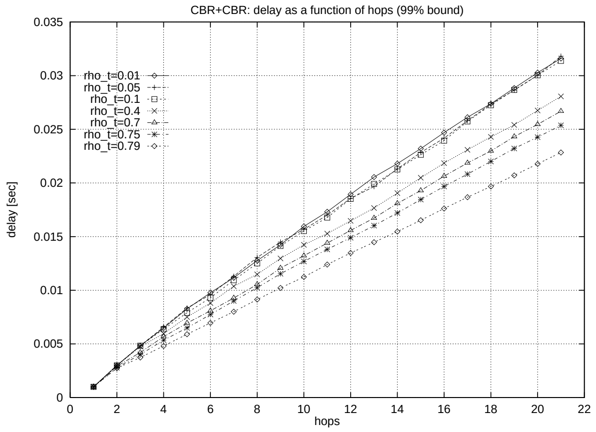

Figure 7: The end-to-end delay in function of the number of hops the cell goes through (FCFS).

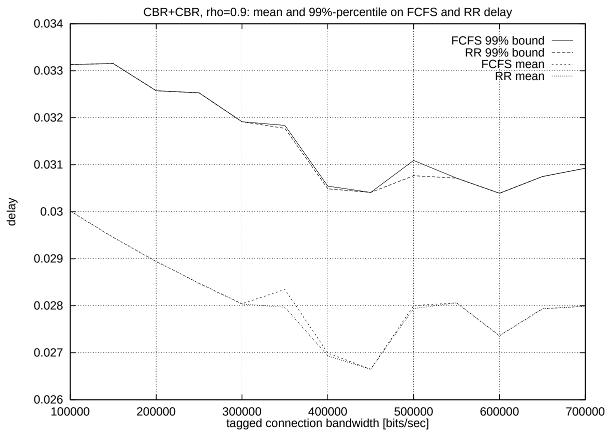

Figure 8: The end-to-end delays (mean and 99%-percentile) for FCFS and RR scheduling for different bandwidth partitions. The total link utilization ρ was kept constant at 0.9. The values shown are ensemble averages over phase combinations.

We can see that the two scheduling disciplines have very comparable performance over the entire range of bandwidth partitions considered (cf. Fig (8)). It is interesting to note that the end-to-end delay of the tagged connection decreases as its share of the link bandwidth increases. This is due to the fact that on average, a cell belonging to the tagged stream encounters a nonempty queue at a switch with a (ensemble) probability ρ_{CT}, the link utilization of the cross traffic. As the cross traffic bandwidth decreases, the tagged stream has a higher chance of finding queues empty and experiences less delay, even as its own bandwidth increases.

As the end-to-end delay increases almost linearly with the hopcount (cf. Fig. 7) means that the bunching of tagged cells seems not to be important. If bunching did occur and accumulate as the tagged connection goes through more and more hops, the delay would not be a linear function in n. We will make this same important observation later with Poisson cross traffic in Section 4.4.

4.3 CBR with aggregated CBR cross traffic

We now go a step further by replacing the cross traffic with a superposition of CBR streams. The bandwidths are chosen as described in the Multiplexed-CBR experiment 4.1 above. In fact, due to simulator limitations as to the number of nodes that the network may contain, we had to use traces of superposition streams that were read out by a special type of source node. We have done this experiment with cross traffic consisting of 10, 20 and 30 superposed CBR streams. The simulation parameters are as follows [3].

Simulation parameters:					
trunk line speed	cross traffic access line speed	traffic intensity ρ	tagged traffic intensity ρ_T	cross traffic intensity ρ_{CT}	transmission delay (end-to-end)
1Mb/s	100Mb/s	0.9	0.1	0.8	22ms

Note that we intentionally did not use the same trace (representing the same bandwidth partition) for each cross traffic, but a different one at each switch.

The difference between the two scheduling disciplines is very small (cf. Fig.10. Note that in this experiment, and unlike in the Poisson cross traffic experiment to be discussed in Section 4.4, the cross traffic is composed of a mix of thick and thin streams. The property of the RR discipline to allocate bandwidth fairly between all streams can therefore result in lower delays for the tagged stream if the cross traffic contains thick streams, as seen in Fig. 10. In the Poisson experiments, we will assume that infinitely thin streams compose the cross traffic.

Again, the delay is almost linear in the number of hops (cf. Fig. 9), and bunching does not adversely affect the delay experienced by tagged cells.

As a rule of thumb we can observe that the mean delay in both cases is approximately one cell service time per switch, which is about the delay that would be seen on average if the cross traffic was purely CBR (the ensemble average over phases) at the same link utilization. The 99%-delay-percentile is about 2 cell service times per switch, which is absolutely tolerable in a high speed cell-based network. For example, in the context of ATM with its 53 bytes cells, two cell service times correspond to $2.74\mu s$ on a 155 Mb/s link.

[3] The *trunk lines* are the lines interconnecting switches; the *cross traffic access lines* are the ones connecting the cross traffic sources to the switches. The cross traffic line speed has been chosen higher in order to emulate multiple incoming lines of the same speed as the trunk lines

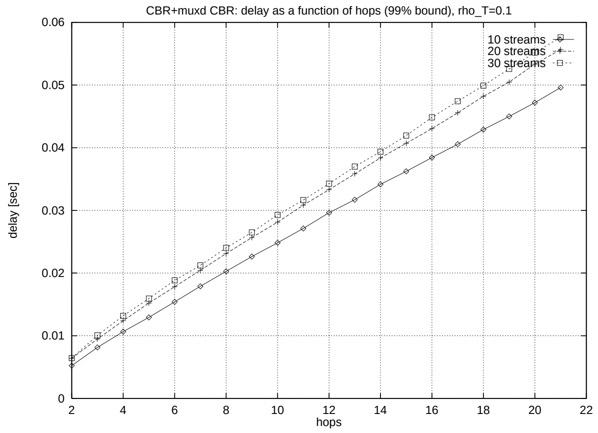

Figure 9: The end-to-end delay as a function of the number of hops (FCFS).

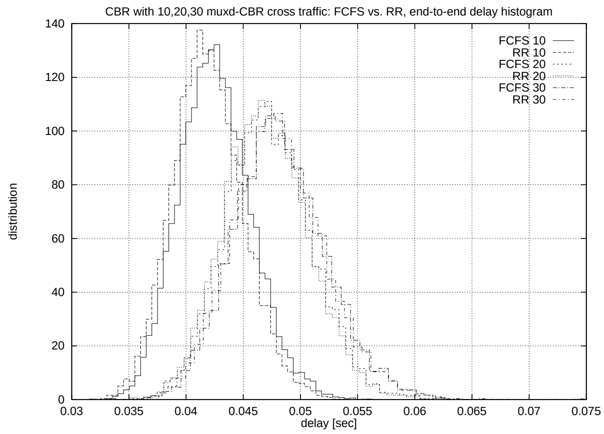

Figure 10: The end-to-end delay histogram with cross traffic consisting of 10, 20 and 30 CBR streams

Both the 99%-percentile and the histogram width of the end-to-end delay increase as the number of multiplexed CBR streams increases. In the next experiment, we try to find an upper bound on the delay by using an conservative approximation for the cross traffic, based on the observations in Section 4.1.

For an analytic discussion of this problem (limited to one queue), the reader is referred to [18, pages 122-129].

4.4 CBR with Poisson cross traffic

In this section, we replace the cross traffic stream in experiment 4.3 with Poisson arrival processes, based on the results of the multiplexing experiment. This experiment must be viewed as a limiting case, when the number of streams constituting the cross traffic tends to infinity [3]. The results found in this section will be *conservative* and have also be used for the analysis of an *M+D/D/1* queue in [18, page 129].

Simulation parameters:				
line speed	traffic intensity ρ	tagged traffic intensity ρ_T	cross traffic intensity ρ_{CT}	transmission delay (end-to-end)
1Mb/s	0.9	$0.01 \ldots 0.1$	$0.89 \ldots 0.8$	22ms

Switches implement one of the FCFS or RR disciplines. In the RR discipline, cells arriving to a currently idle VCI (that is, to an empty queue) are added to the tail of the service list. When emulating a large number of CBR streams with a single Poisson stream, we face the problem that a cell in the Poisson stream may belong to any one of the VCIs emulated by that stream. Since this VCI is not known to the scheduler, it cannot decide which VCI is active and which is inactive. We therefore have to resort to a way of emulating the behavior of the entire set of cross traffic CBR connections without knowing which connections the incoming cross traffic cells belong to.

It is necessary to make the following assumption: *in the RR case, no per-connection queue would ever contain more than one cell.* As the individual CBR streams forming the cross traffic are very thin compared to the total cross traffic, cell arrivals belonging to a same connection are therefore rare compared to overall cell arrivals. To illustrate this, consider the following example. Suppose the cross traffic consists of 1000 CBR streams. Then two cells belonging to a same VCI will be interleaved on the average by about 1000 cells belonging to other VCIs. On the other hand, we can estimate the number of cross traffic cells queued using the following result for the *M/D/1* queue [12, page 188].

$$\overline{q} = \frac{\rho}{1-\rho} - \frac{\rho^2}{2(1-\rho)} \tag{5}$$

For example, for $\rho = 0.95$, the average number of queued cross traffic cells would be ≈ 10. In other words, an incoming cell will hardly ever be queued long enough so that another cell belonging to the same VCI would arrive before this former cell has been serviced. We therefore believe that this assumption does not appreciably alter the results.

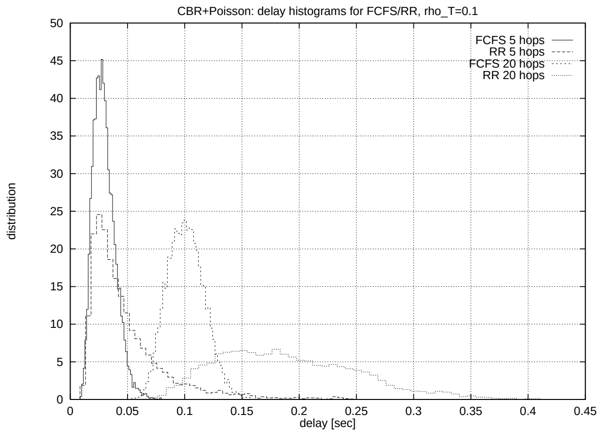

Figure 11: Delay histograms for FCFS and RR for $\rho_T = 0.1$.

Fig. 11 shows the end-to-end delay histogram for FCFS and RR scheduling disciplines after 5, 10 and 20 hops. It is clear from the figure that both the mean delay and the delay jitter are smaller for FCFS than for RR. The 99%-percentile on the delay for different tagged stream bandwidths is depicted in Fig. 12. FCFS and RR have

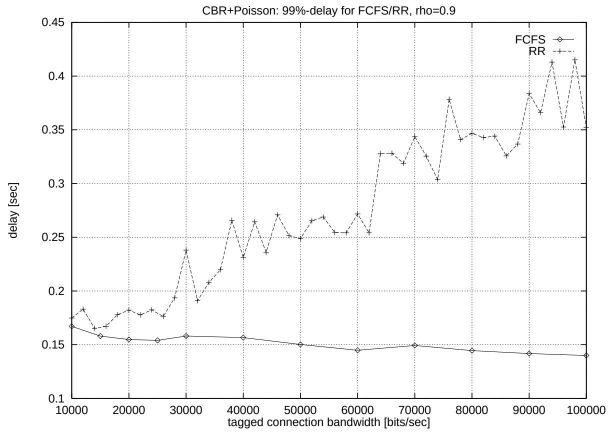

Figure 12: The 99%-percentile of the measured end-to-end delay after going through 21 switches for FCFS and RR.

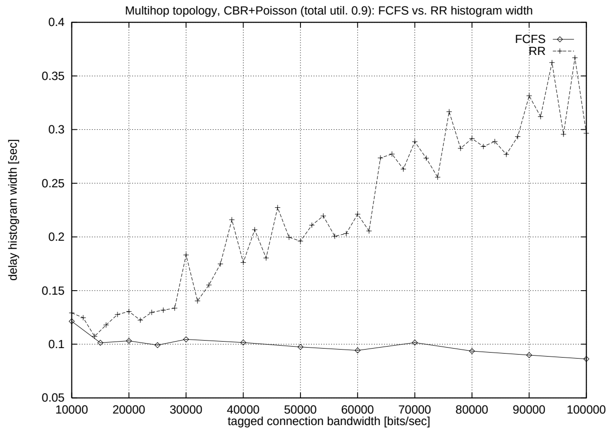

Figure 13: The histogram width of the measured end-to-end delay after going through 21 switches for FCFS and RR.

comparable delay for small tagged stream intensity. As this intensity increases, the RR delay gets worse, while the FCFS delay stays approximately constant. The largest delay is about 170ms for FCFS. Taking into account the 22 ms of end-to-end transmission delay means that FCFS has about (170ms−22ms)/20 $\approx$ 7.4ms or 7.4 cell service times of queueing delay per switch. The size of the buildout buffer can be estimated from Fig. 13: For example, when the tagged connection has 100kb/s bandwidth, we observe a histogram width of about 90ms for FCFS, which corresponds to a buildout buffer size of approximately 9 cells. When the tagged connection has 10kb/s bandwidth, the histogram width is about 130ms, corresponding to 1.3 cells.

From Fig. 14 and Fig. 15, we see that the delay is again approximately linear as a function of the hopcount, as observed previously. Bunching still has no notable effect on the delays.

We now propose and prove a theorem that confirms that the delay of each tagged cell in the RR case in greater or equal to the delay in the FCFS case. This theorem is based on the assumption made earlier, i.e. that the per-connection queues of the cross traffic streams contain at most one cell at any time. The set of cross traffic and tagged cells is called $\mathcal{C}$ and $\mathcal{T}$, respectively, with $\mathcal{I} = \mathcal{C} \cup \mathcal{T}$. The arrival and departure times of a cell i are called $a(i)$ and $d(i)$, respectively. For simplicity, we denote a cell with i' if it belongs to the same connection as a cell i

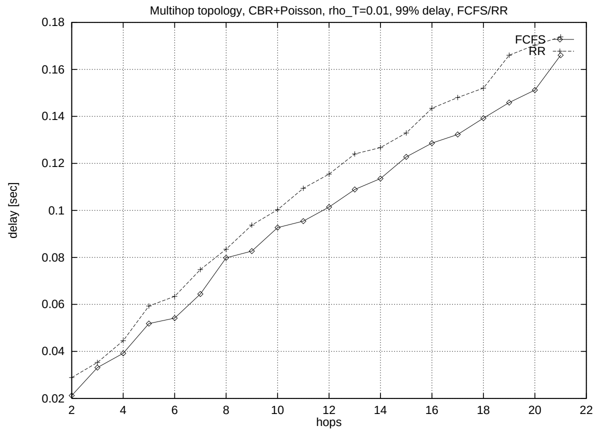

Figure 14: The 99%-delay-percentile as a function of the number of hops the tagged stream goes through, with $\rho_T = 0.01$.

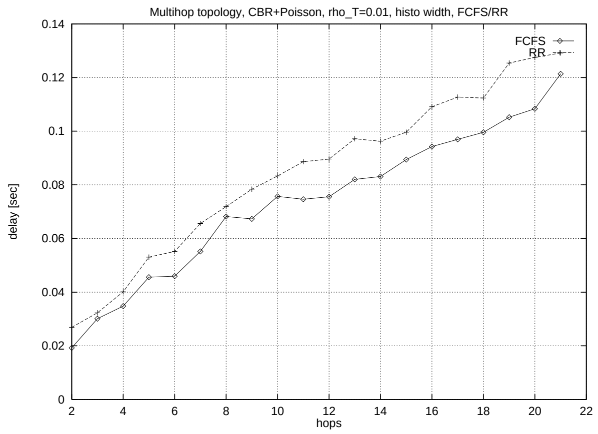

Figure 15: The delay histogram width as a function of the number of hops the tagged stream goes through, with $\rho_T = 0.01$.

and immediately precedes it on this connection.

Theorem 1 *For given arrival process* $\mathcal{A} = \{a(i)|i \in \mathcal{I}\}$ *the queueing delay* $d(i) - a(i)$ *of any tagged cell* i *in the RR case is greater or equal to the delay experienced in the FCFS case.*

Proof: The proof is based on the following two lemmas.

Lemma 1 *For each tagged cell* $i \in \mathcal{T}$*, there exists a set of cross traffic cells* $\mathcal{J}(i) = \{j|a(i) < a(j) < d(i'), \forall j \in \mathcal{C}\}$ *that are served* after i *with FCFS scheduling, but* before i *with RR scheduling.*

Proof: First, note that in the FCFS case, $a(i) < a(j)$ implies by definition that i is served before j for any i, j. In the RR case, $j \in \mathcal{C}$ is served *before* $i \in \mathcal{T}$ although $a(i) < a(j)$ if and only if $a(i) < a(j) < d(i')$, because in this case the tagged connection is appended to the service list at time $d(i')$, while the cross traffic connection to which j belongs is appended at time $a(j)$, as by assumption j arrives into an empty queue. $\Box$

Lemma 2 *The number of cells in the scheduler is equal in both cases at any time* t.

Proof: Both scheduling disciplines are work-conserving. □

It follows from lemma 1 that in both cases, an arriving tagged cell i has to wait until service of all cells present in the scheduler at time $a(i)$ has finished. Furthermore, it follows from lemma 2 that this time is equal for FCFS and for RR. But in addition to this, with RR scheduling, the tagged cell i has to await completion of service of cross traffic cells contained in $\mathcal{J}(i)$. □

Intuitively, RR attempts to share the link equally between all connections. If the tagged connection has a higher bandwidth than the cross traffic streams, then this connection experiences a degradation in performance. Clearly, this is not desirable.

A way of explicitely giving each connection the desired share of bandwidth is by using Weighted Round Robin (WRR). Unfortunately, this discipline has the drawback of implicit bunching. We will discuss this phenomenon in the next section.

4.5 WRR bunching

WRR allocates a certain service quantum to each connection. When this connection's identifier is at the head of the service list, then up to `service_quantum` number of cells are served. As these cells leave the WRR scheduler back-to-back, although they might have arrived with some spacing between them, some bunching is introduced. This bunching is implicit to the WRR scheduling discipline. The following is an attempt to determine the worst-case bunching delay and to show that there is a tradeoff between bandwidth allocation granularity and this delay.

Weighted Round Robin (WRR) is implemented most easily by serving each connection in turn during its entire allocated service quantum. This means that cells arriving spaced out, but being served in the same service quantum will leave the scheduler back-to-back. This leads to cell delays that we call *bunching delays*.

We would like to find out what determines worst-case bunching delays. For this, consider a WRR scheduler serving M connections. Each one of these connections is characterized by the size of its service quantum n_i ($i = 1, \ldots, M$), with $n = \sum_{i=1}^{M} n_i$. Assume that bandwidth is entirely allocated, i.e. all of the available slots N are used: $n = N$. We denote the line speed with C and the cell size with L. The bandwidth granularity is then $c_1 = C/n$, and a connection i has bandwidth allocation $c_i = n_i c_1$.

We want to compare the delay variation (jitter) that can be introduced by the WRR scheduler. We consider a stream of cells that is compliant with the bandwidth allocation for the considered connection, and measure the jitter as the difference in delay these cells experience. The delay of a cell depends on the "phase" of the scheduler at its arrival (cf. Fig. 16). The worst case delay T_{max} is experienced by a cell arriving when service of its connection has just begun and all other connections will take advantage of their full service quantum. This cell has to wait for an entire round until its queue is served (n cells). The best case delay is for a cell that arrives just before service of its connection begins. It only has to await service of queued cells belonging to the same connection ($n_i - 1$ cells).

Therefore, the difference between best and worst case delay is

$$T_{max} - T_{min} = \frac{L}{C}(n - n_i + 1) = \frac{L}{c_1}(1 - \frac{c_i - c_1}{C}) \tag{6}$$

It is interesting to see that the bunching delay varies only little with increasing line speed, and actually gets worse. If one assumes that $c_i \ll C$, then the bunching delay is inversely proportional to the bandwidth granularity

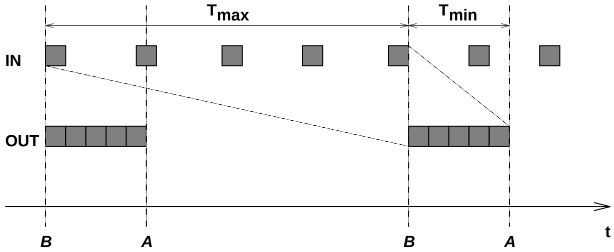

Figure 16: Incoming cells experience different delays, ranging from T_{min} to T_{max}, depending on when they arrive within a round. Service of the considered connection begins at `B`, and service of all other connections begins at `A`.

and is given by

$$T_{max} \approx \frac{L}{c_1} \tag{7}$$

The buildout buffer has to be sized to hold the service quantum of cells. This can become large if the bandwidth granularity is considerably smaller than the bandwidth allocated to a connection.

4.6 Two Analytical Approaches for FCFS delays

In this section, we will present two analytical approaches that can be used to study the delay experienced by the tagged connection in the FCFS case with Poisson cross traffic. We will be interested to see to what extent simulation and analysis agree.

A first approximation stems from the observation that the tagged stream is thin compared to the cross traffic. We can thus view the server as having a capacity that is reduced by the (approximately) deterministic tagged stream and serving the cross traffic alone, which is Markovian:

$$\rho = \frac{\rho_{CT}}{1 - \rho_T} \tag{8}$$

Furthermore, as only the tagged stream travels through multiple queues, we can assume that the queues are independent. Finally, as the service time is constant due to constant cell size, we model the queues as *M/D/1*. The first and second moment of the waiting time of a *M/G/1* queue is given in [12, page 201] (where b_i is the i-th moment of the service time, which in our case is simply τ^i, with τ the service time) with

$$\mathrm{E}[w] = \frac{\lambda b_2}{2(1-\rho)} = \frac{\lambda \tau^2}{2(1-\rho)} \tag{9}$$

$$\mathrm{E}[w^2] = 2\mathrm{E}^2[w] + \frac{\lambda b_3}{3(1-\rho)} \tag{10}$$

From this, we find the variance of the waiting time

$$\mathrm{var}[w] = \mathrm{E}[w^2] - \mathrm{E}^2[w] = \frac{\lambda \tau^3 (4 - \lambda \tau)}{12(1-\rho)^2} \tag{11}$$

Based on the independence assumption, the delay distribution will approach a Gaussian distribution as the number of queues the tagged stream goes through increases. It seems therefore interesting to plot a Gaussian distribution with mean μ and variance σ^2 given as follows, with n denoting the number of switches (not counting the last one, where no new cross traffic is injected).

$$\mu = n\,\mathrm{E}[w] + (n+2)\tau \tag{12}$$

The second term accounts for the service time in each queue as well as for the transmission time on the first and last link (i.e. the link leaving the source and the link entering the sink).

$$\sigma^2 = n\,\mathrm{var}[w] \tag{13}$$

Consider Fig. 17 for a comparison of the *M/D/1* approximation with the end-to-end delay histograms of the tagged stream. Two intensities for the tagged stream are shown, $\rho_T = 0.01$ and $\rho_T = 0.1$. The correspondence between measurement and approximation is good, but the Gaussian approximation should only be applied if the number of traversed switches is quite large. The noisy delay histogram is due to the fact that simulation time gets prohibitive for very small tagged stream intensities. For example, if the tagged stream's intensity is 0.01 and the cross traffic 0.89, then for each tagged cell, on average 89 cross traffic cells are produced *at each switch*. Therefore, in order to get a reasonable number of end-to-end delay samples, the total number of cells simulated becomes extremely large.

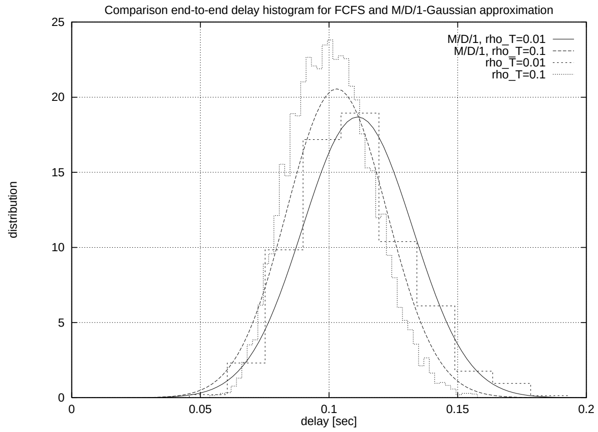

Figure 17: Two end-to-end delay histograms and the delay distribution resulting from the *M/D/1*-Gaussian approximation

We will study a more sophisticated model from papers by Whitt that is supposed to give accurate results even if the deterministic traffic is not negligible and if the number of switches is small [20, 21]. Whitt's methods allow us to derive approximate relationships for the coefficients of variation of the tagged stream at each switch in the network, and to derive the expected wait time at each switch. Each (wide sense stationary) interarrival process $\{X_i\}$ can be associated with its *squared coefficient of variation (SCV)* c_a^2, which is defined as

$$c_a^2 = \frac{\mathrm{var}[X_i]}{\mathrm{E}^2[X_i]} \tag{14}$$

As the interdeparture process ($c_{d,T}^2$) of the tagged stream at switch i is at the same time the interarrival process of the tagged stream at switch $i+1$, we calculate the SCVs for the multihop topology recursively from left to right. The variables used for one segment are depicted in Fig. 18.

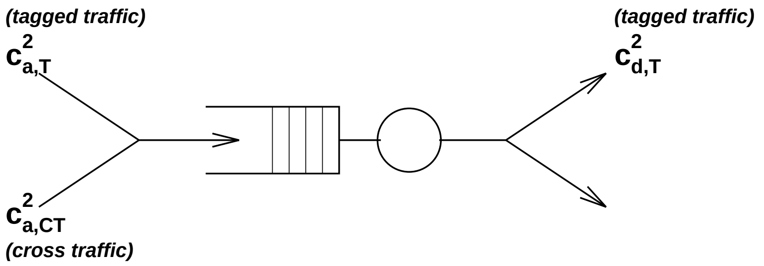

Figure 18: The model used to approximate one segment in the multihop topology.

Equation (11) in [21] (below) allows us to determine the departure SCV of the tagged cells. Note that the service time SCV c_s^2 is zero in our case, as cells are of constant size. ρ is the total traffic intensity, and $\lambda_{T,CT}$ are the intensities of the tagged and the cross traffic streams, respectively. $p_1 = \lambda_T/(\lambda_T + \lambda_{CT})$ is the fraction of cells belonging to the tagged stream. The cross traffic is a Poisson process, and therefore has SCV $c^2_{a,CT} = 1$.

$$c^2_{d,T} = \rho^2 p_T c_s^2 + (2-\rho)p_T(1-p_T)c^2_{a,CT} + [(1-p_T)^2 + (1-\rho^2)p_T^2]c^2_{a,T} \tag{15}$$

If we denote the arrival SCV at switch i with $c^2_{a,T}(i)$, then we obtain the recursion

$$c^2_{a,T}(i+1) = (2-\rho)p_T(1-p_T) + [(1-p_T)^2 + (1-\rho^2)p_T^2]c^2_{a,T}(i) \tag{16}$$

(with $c^2_{a,T}(1) = 0$), which allows us to determine the departure SCV at each switch.

Once the $c^2_{a,T}$ are known at each switch, we need to determine the SCV of the *superposed* interarrival process at each switch. We use (29), (30) and (33) in [20].

$$c_a^2 = \omega\left(\frac{\lambda_T c^2_{a,T} + \lambda_{CT} c^2_{a,T}}{\lambda_T + \lambda_C T}\right) + 1 - \omega \tag{17}$$

$$\omega = \frac{1}{1 + 4(1-\rho)^2(v-1)} \tag{18}$$

$$v = 1 + 2\left(\frac{\lambda_T \lambda_{CT}}{\lambda_T^2 + \lambda_{CT}^2}\right) \tag{19}$$

This allows us to determine the expected wait time at each switch, given by (44) and (45) in [20]. τ is the mean cell service time (1ms in our case).

$$\mathrm{E}[W] = \frac{\tau\rho(c_a^2 + c_s^2)g}{2(1-\rho)} \tag{20}$$

where

$$g(\rho, c_a^2, c_s^2) = \begin{cases} \exp\left[-\frac{2(1-\rho)}{3\rho}\frac{(1-c_a^2)^2}{c_a^2+c_s^2}\right] & c_a^2 < 1 \\ 1 & c_a^2 \geq 1 \end{cases} \tag{21}$$

We see that given the FCFS discipline, the SCV of the CBR traffic increases with the number of hops (cf. Fig. 19). This indicates that with FCFS scheduling, the effect of aggregation in the cross traffic is transferred to equi-spaced input streams.

For small ρ_T, we observe considerable differences between the SCV values predicted by theory and the ones measured. The approximation tends to get better as ρ_T increases. Nevertheless, over the whole range of tagged traffic intensities, the mean delay is predicted very well by Whitt's method.

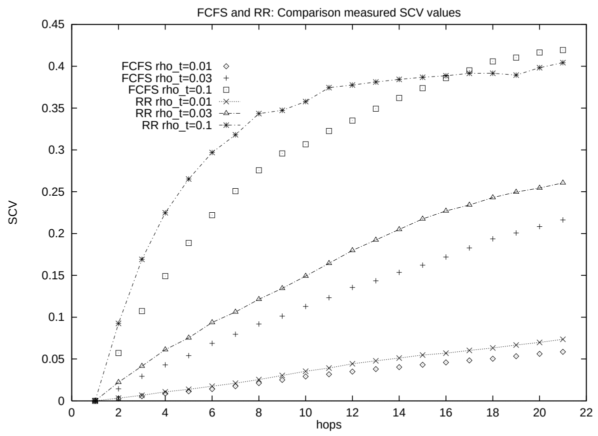

Figure 19: SCV values for FCFS and RR as a function of the number of hops for various tagged traffic intensities

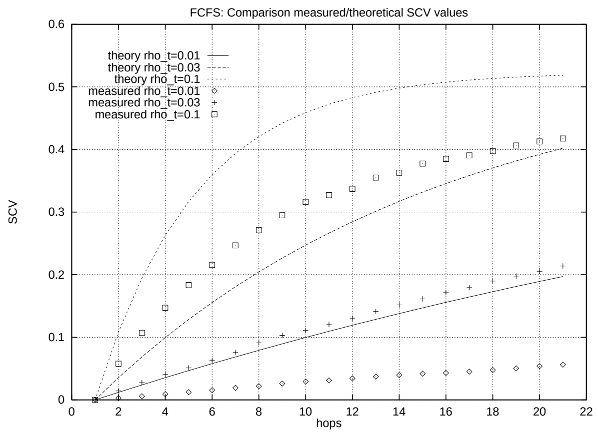

Figure 20: Comparison of analytic and measured SCV as a function of number of hops for different tagged stream intensities ρ_T.

Two facts may help to understand this. First, the definition for the SCV as used in the Parametric Decomposition method is actually not the same as in (14), which only captures the variation of interarrival times between two cells. Rather, in this method, the SCV attempts to incorporate both long-term and short-term behavior of the process under consideration, of the form $\omega J(\infty)+(1-\omega)J(1)$ (see (17)), and then assumes that $J(1) = 1$. Our measurements, on the other hand, strictly follow the definition in (14), which is actually equal to the IDI for intervals of size one, $J(1)$. This is the reason why the SCV is overestimated by the theory in this case. Second, it is intuitively clear that the smaller the tagged traffic's intensity is compared to the cross traffic, the smaller the impact of the tagged traffic on the queue behavior and therefore the delay. This means that for small ρ_T, it matters less how accurate the SCV of the tagged traffic is predicted than for higher ρ_T.

We conclude that the *M/D/1*-Gaussian approximation appears to have its justification for conservative delay distribution estimation when the traffic goes through many queues and the tagged stream is very small compared to the cross traffic, while Whitt's Parametric Decomposition method, at least in our setting, gives excellent, slightly conservative, mean delay estimates under more general circumstances.

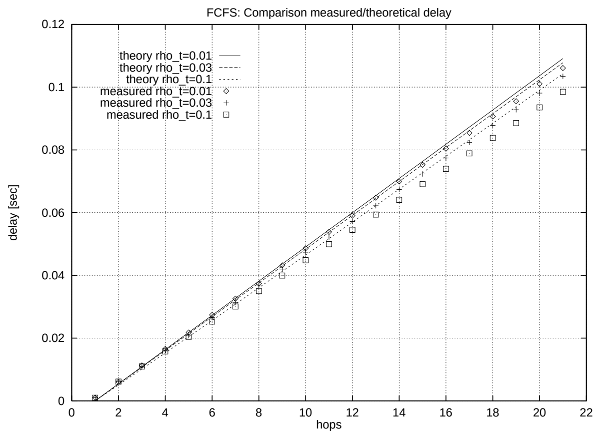

Figure 21: Comparison of analytic and measured delay as a function of number of hops for different tagged stream intensities ρ_T.

5 Discussion

One would like to exploit the deterministic nature of CBR to offer high QoS in terms of delay and cell loss. However, we have shown that the superposition of CBR streams with different bandwidths and phases can introduce burstiness that makes the resulting traffic look like Poisson over short time frames. This has led us to the modeling of cross traffic as a Poisson stream to study how this affects the end-to-end delay of the tagged connection.

We have seen that for Poisson cross traffic, FCFS has a smaller 99%-percentile delay and a smaller end-to-end delay histogram width than RR. The histogram width grows with the number of hops a connection travels through, and the buildout buffer has to be sized accordingly. We have found that with FCFS, very reasonable buildout buffer sizes are sufficient. As a rule of thumb, for a connection using one tenth of the trunk bandwidth, less than a cell of buffer space per hop is required to keep losses below 1%.

In all three multihop experiments, the end-to-end delay grows linearly in function of the number of hops. This suggests that bunching of cells belonging to the tagged stream is not sufficient to have an impact on delays experienced by these cells.

We have used fairly high link utilizations in our experiments. If the link utilization is lower ($\rho \ll 1$) then most of the cells see empty queues, and the difference between FCFS and RR decreases correspondingly.

These observations have to be contrasted with the hardware implementation complexity of the scheduling disciplines, which is dominated by the amount of state that has to be maintained. To get an idea about the relative cost of these implementations, we present the cost given some typical values of the parameters. We assume that the output queue can serve 16K conversations at any time, so that $N = 16K$. Let the largest service quantum be 256 quanta, so that $n_q = 8$. We assume that the queue has 16 MB of memory, which is cell addressable, so that we can store approximately 39570 cells, so that $n_p = 16$ (otherwise, with byte addressable memory, $n_p = 24$). Thus, the state information required for FCFS is $2n_p$ or 32 bits (48 bits with byte addressing). The state information for WRR is 116 Kbytes with cell addressing and 165 Kbytes with byte addressing. The state information with RR is 100 Kbytes with cell addressing and 149 Kbytes with byte addressing. Thus, FCFS not only performs better than RR (and WRR), but requires much less state information.

6 Conclusion

We have carried out a detailed study of the behavior of CBR traffic in the presence of cross traffic, using available analytical techniques as well as exhaustive simulation. Earlier work had indicated [9] that multiplexing of such sources could lead to bunching - in our work, we have characterized the bunching using the IDI and 99% delay metrics. We have also compared simulation results with the results predicted by multiclass parametric decomposition techniques proposed by Whitt. Finally, we compare the behavior of the FCFS and RR disciplines with respect to bunching of CBR traffic.

Our main results are as follows. First, our simulations indicate that when CBR traffic is multiplexed with a small number of up to 30 other CBR streams, the difference in end-to-end delays for FCFS and RR are small (see Sections 4.2 and 4.3), and there is no appreciable bunching, even as the depth in the network increases to 20 hops (cf. Fig. 7 and Fig. 9).

Second, as the number of CBR sources multiplexed at a single link (the *breadth*) increases, the output stream looks like a Poisson stream at smaller time scales (2-10 cell arrival times, cf. Fig. 6). Given this observation, we model the aggregate cross traffic by a Poisson stream, which is admittedly a pessimistic modeling, in order to find conservative delay estimations.

Third, when a CBR source interacts with Poisson cross traffic, the scheduling discipline has an impact on the output stream (cf. Fig. 11 to Fig. 13). When the scheduling discipline is FCFS, the output stream experiences some bunching and delays, which increases as the load or the depth in the network increases. The performance of RR is comparable to FCFS for low intensity streams, but deteriorates as a connection's intensity increases. Bunching again does not have an appreciable impact on delays experienced by a stream going through many hops (cf. Fig. 14). This means that for CBR streams, no special measures have to be taken to avoid bunching, such as per-connection reregulation at intermediate switches. It is sufficient to restore cell-equispacing in a buildout buffer at the connection endpoint.

Fourth, we have found that the expected waiting delays for FCFS scheduling are well predicted by theory (cf. Fig. 21), although the predicted squared coefficient of variation differs substantially from the observed values (cf. Fig. 20). Another approach using *M/D/1*-queues and the law of large numbers provides for good delay distribution estimations if the number of hops is large (cf. Fig. 17).

Fifth, WRR has the undesirable property of implicit clustering, which is inversely proportional to the bandwidth allocation granularity. Also, it needs explicit connection setup, while FCFS and RR do not.

Past work on CBR traffic has included analytic approaches, as in [18] and [6]. However, they treat only single queues, not taking into account the fact that the original CBR stream will not be equispaced any more after leaving the queue. This means that these methods will not be applicable to subsequent queues. Verma [19] has done multihop simulations, but his cross traffic is composed of CBR streams of equal bandwidths (among other cross traffic models). He observes bunching by measuring the change of the minimum inter-cell spacing after a number of hops. He does not discuss, however, the implications for end-to-end delay and the impact of correlated cross traffic. To the best of our knowledge, this is the first comprehensive simulation study that confirms the hypothesis that the FCFS scheduling discipline is sufficient for CBR traffic even in a large-scale network. This has previously been claimed (for example [1]), but not been verified. Based on our work, we recommend that CBR networks be built with FCFS scheduling, with a few cells of buffering per switch, and a buildout buffer proportional to the number of hops in the path.

A limitation of this study is that the 99%-percentile on delay corresponds to a relatively large cell loss rate of 1% in the buildout buffer. Future ATM networks will have to offer considerably smaller error probabilities, e.g. 10^{-6}. In order to do experimental work with loss probabilities in this range, one has to resort to more advanced techniques, e.g. rare event simulation [8]. A second limitation is that we assume that incoming CBR traffic is smooth at the cell level. If this traffic is not smooth at this level, a regulator would have to be introduced at the network entracnce to perform smoothing. We do not consider the smoothing delay at this regulator; but it is easily computed as the worst case buildup over the averaging interval of the CBR source.

References

[1] To preserve anonymous review, this reference has been deleted.

[2] A. Banerjea and S. Keshav. Queueing Delays in Rate Controlled ATM Networks. In *Proc. IEEE INFOCOM '93*, San Francisco, California, April 1993.

[3] E. Cinlar. Superposition of Point Processes. In P.A.W.Lewis, editor, *Stochastic Point Processes: Statistical Analysis, Theory, and Applications*, pages 549–606. Wiley Interscience, 1972.

[4] Rene L. Cruz. A Calculus for Network Delay, Part I: Network Elements in Isolation. *IEEE Transactions on Information Theory*, 37(1):114–131, January 1991.

[5] Rene L. Cruz. A Calculus for Network Delay, Part II: Network Analysis. *IEEE Transactions on Information Theory*, 37(1):132–141, January 1991.

[6] Antonio DeSimone. Generating Burstiness in Networks: A Simulation Study of Correlation Effects in Networks of Queues. In *ACM Computer Communication Review*, pages 24–31, 1991.

[7] K.W. Fendick and W. Whitt. Measurements and Approximations to Describe the Offered Traffic and Predict the Average Workload in a Single-Server Queue. *Proceedings of the IEEE*, 77(1):171–194, January 1989.

[8] Victor S. Frost and Benjamin Melamed. Traffic Modeling For Telecommunications Networks. *IEEE Communications Magazine*, pages 70–81, March 1994.

[9] S. Jamaloddin Golestani. Congestion-Free Transmission of Real-Time Traffic in Packet Networks. In *Proc. IEEE INFOCOM '90*, pages 527–536, San Francisco, California, June 1990.

[10] R. Gruenenfelder. A Correlation Based End-to-End Cell Queueing Delay Characterization in an ATM Network. In *Proc. Thirteenth International Teletraffic Congress (ITC-13)*, volume 15, pages 59–64, Copenhagen, Denmark, June 1991. North-Holland Studies in Telecommunications.

[11] R. Gusella. Characterizing the Variability of Arrival Processes with Indices of Dispersion. *IEEE Journal on Selected Areas in Communications*, 9(2):203–211, February 1991.

[12] Leonard Kleinrock. *Queueing Systems, Volume 1: Theory.* John Wiley & Sons, New York, NY, 1975.

[13] Jim Kurose. On Computing Per-session Performance Bounds in High-Speed Multi-hop Computer Networks. In *Proc. ACM SIGCOMM '92*, 1992.

[14] Jim Kurose. Open Issues and Challenges in Providing Quality of Service Guarantees in High-Speed Networks. In *ACM Computer Communication Review*, volume 23, pages 6–15, January 1993.

[15] Wassim Matragi, Chatschik Bisdikian, and Khosrow Sohraby. Jitter Calculus in ATM networks: Single Node Case. In *IEEE INFOCOM '94*, Toronto, Canada, June 1994.

[16] Wassim Matragi, Chatschik Bisdikian, and Khosrow Sohraby. Jitter Calculus in ATM networks: Multiple Node Case. In *IEEE INFOCOM '94*, Toronto, Canada, June 1994.

[17] James W. Roberts and Jorma T. Virtamo. The Superposition of Periodic Cell Arrival Streams in an ATM Multiplexer. *IEEE Transactions on Communications*, 39(2):298–303, February 1991.

[18] J.W. Roberts, editor. *COST 224: Performance evaluation and design of multiservice networks.* Commission of the European Communities: information technologies and sciences, October 1991.

[19] Dinesh Chandra Verma. *Guaranteed Performance Communication in High Speed Networks.* PhD thesis, University of California, Berkeley, Berkeley, CA 94720, December 1991. Report No. UCB/CSD 91/663.

[20] Ward Whitt. The Queueing Network Analyzer. *The Bell System Technical Journal*, 62(9):2779–2815, November 1983.

[21] Ward Whitt. Towards better multi-class parametric-decomposition approximations for open queueing networks. *Annals of Operations Research*, 48:221–248, 1994.

[22] David Yates, James Kurose, Don Towsley, and Micheal G. Hluchyj. On per-session end-to-end delay distribution and the call admission problem for real-time applications with QOS requirements. In *Proc. ACM SIGCOMM '93*, pages 2–12, San Francisco, CA, September 1993.