INTERNATIONAL SWITCHING SYMPOSIUM 1992

"Diversification and Integration of
Networks and Switching Technologies
Towards the 21st Century"

Yokohama, Japan
October 25-30, 1992

Proceedings
Vol. 2

Volume 1	Session A1 - C2, P A3 - C4
Volume 2	Session A5 - C10

XIV International Switching Symposium
Yokohama, Japan
October 25-30, 1992

THE SPACING POLICER, AN ALGORITHM FOR EFFICIENT PEAK BIT RATE CONTROL IN ATM NETWORKS

Eugen Wallmeier and Tom Worster

Siemens AG, OEN ZL S, P.O. Box 700073, W-8000 Munich 70, Germany

ABSTRACT

A policing function enforcing the peak cell rate in an ATM network which does not prevent clusters of cells from entering the network cannot protect the network from congestion under all conditions. Proper protection can be achieved by using "cell spacing" where the policing function spaces out cells that arrive too close together, thus ensuring that the interdeparture times of consecutive cells are never in conflict with the negotiated peak cell rate. The paper presents a new algorithm for combined policing and cell spacing. The high performance of the algorithm results from two key principles: 1 - Shared memory is used for the waiting cells of all VC's, keeping memory requirements down; 2 - arriving cells are linked to an appropriate queue without using a search algorithm, thus the dynamic processing requirements are similar to those of simpler non-spacing algorithms. Concerning the inevitable delays due to spacing, the algorithm shows a tendency to prefer users who comply to their negotiated peak bit rate over non complying users.

1. INTRODUCTION

An ATM network's decision whether to accept an incoming connection depends on traffic parameters (e.g. the peak or mean cell rate - or equivalently - bit rate) which are negotiated between user and network at call set up time. A policing function (also known as usage parameter control (UPC) function) has to monitor the compliance of the user to the negotiated parameter values throughout the connection's holding time. This contribution addresses the problem of peak cell rate policing.

In most of the policing functions proposed so far in the literature, excess cells are discarded but the rest of the cell stream passes the policing function unaltered. In [1] such policing functions are named pick-up algorithms. Meanwhile it is recognized (see e.g. [2-4]) that pick-up algorithms cannot protect the network efficiently under all conditions because they do not prevent clusters of cells from entering the network in the presence of delay jitter. The impact of jitter on peak cell rate policing is discussed in more detail in Chapter 2.

The problems caused by clusters of cells belonging to one connection can be avoided if the policing function controlling the peak cell rate not only discards excess cells but also smooths out momentary peaks in the incoming traffic with the aid of a particular kind of traffic shaping known as "cell spacing". Cell spacing (cf. [1]) denotes the technique of delaying certain cells such that the interdepature times from the policing unit between consecutive cells of one connection are never below a minimal value which is allowed according to the negotiated peak cell rate. Chapter 3 describes the logical functionality a cell spacing function has to provide. The problem of cell spacing on a VC basis at the User Network Interface is rather complex because a large number of VCs using the same access line have to be spaced individually but all at the same time.

One way to implement a cell spacing function is described in [5]. Here we present an alternative algorithm (denoted as Spacing Policer) which allows combined cell spacing and policing for an access line used by any large number of ATM virtual connections. The algorithm which is described in Chapter 4 shows that cell spacing on a VC basis is feasible in practice with a moderate increase in the complexity of the UPC function compared with a simple Leaky Bucket. The dynamic processing requirements for the presented Spacing Policer are moderate since arriving cells, which have to be delayed, are linked to an appropriate queue without using a search algorithm. In every cell cycle a small *deterministic amount of work* has to be done. This facilitates an implementation for lines with very high speed (e.g. at the Network Node Interface).

The required memory for the spacing policer, which is shared among all VCs, depends on the jitter which may arise between the terminal equipment and the policing function. Given a bound for the jitter, the cell memory can be dimensioned such that *no cell loss due to lack of memory* can occur. In Section 5 it is shown that a memory for 75 ATM cells will be sufficient even if an access lines connected to a large ATM PABX is policed.

The algorithm shows a tendency to prefer users who comply with their negotiated peak cell rate over users who exceed the negotiated peak cell rate during short bursts. On the average cells of well behaving sources will be less affected by the inevitable multiplexing delay due to spacing than cells of sources which send short bursts violating the allowed peak cell rate. First results concerning the performance of the Spacing Policer are described in Section 6.

The high and low priority cell streams within one virtual connection have to be policed individually. In Section 7 two UPC schemes which can be used for this purpose are described. It is shown how the Spacing Policer can be used in these schemes.

2. THE IMPACT OF JITTER ON PEAK CELL RATE POLICING

In general, the peak cell rate observed by the UPC function will be different from the peak cell rate directly at the source [1)] which is agreed in the contract between user and network. The difference results from jitter which may be introduced by the use of the GFC protocol, the slotted nature of the physical layer of an ATM system and multiplexing arrangements located between source and UPC function. The UPC function has to tolerate this jitter, otherwise it might discard cells of well behaving sources. However it is principally impossible for the policing device to decide whether short bursts that violate the specified peak cell rate are caused by badly behaving sources or by delay jitter. If the policing function lets bursts of both kinds pass transparently through, this weakness may be exploited by "tricky users" who send short bursts at a cell rate exceeding the negotiated peak value (e.g. because they want to send small messages, such as X.25 packets, consisting of some ATM cells at the full cell rate of the User Network Interface). Errant terminal behaviour or intended misbehaviour could also cause bursts violating the specified peak cell rate. The public network cannot simply trust that none of the traffic sources are malicious or tricky users and that sources always conform to their negotiated peak cell rates. The results in [2-4] show that already a relatively small amount of bursty traffic allowed by pick-up policing algorithms (e.g. the Leaky Bucket or the Sliding Window) may introduce severe degradation in network performance.

3. A CONCEPTUAL MODEL FOR A COMBINED POLICING AND SPACING FUNCTION

If an access line being policed is used only by one virtual connection then the incoming traffic can be spaced with the aid of a buffer, which is organized as a FIFO (first-in-first-out) queue and is served at the policed cell rate. Using such a "Leaky Bucket Queue" for a connection with a peak cell rate 10% of the pipe capacity, a cell could enter the network at most every ten cell cycles. Generally, however, an access line may be used by a number of virtual connections (VCs) simultaneously (potentially up to 4000 VCs). In this case the problem of cell spacing at the network interface is more complex because all VCs have to be spaced individually but all at the same time. In this case a spacing device has to provide the logical functionality of the conceptual model of the system shown in Fig. 1 which shows the VCs being demultiplexed on different lines such that VC individual Leaky Bucket Queues can be applied before the VCs are multiplexed again.

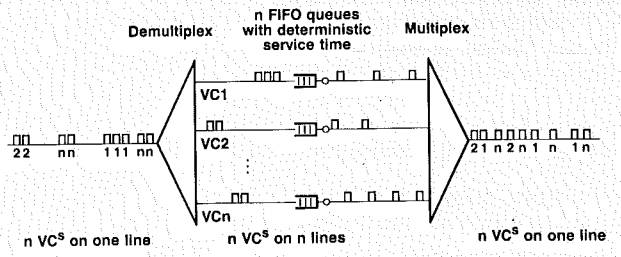


Fig. 1 Conceptual model for a combined Leaky Bucket policing and cell spacing device. (Note that the device would not be implemented as shown here.)

From the implementation point of view a system using VC specific queues is not practical. This contribution presents a combined Leaky Bucket policing and cell spacing device (called Spacing Policer), which emulates the logical behaviour of the system in Fig. 1 without using VC specific queueing.

4. THE DESIGN OF THE SPACING POLICER

In this Chapter we describe the Spacing Policer which monitors the total flow of virtual connections without considering the cell loss priority bit (CLP) in the ATM cell header. In Section 7 it is described how the Spacing Policer can be used in a UPC scheme taking into account the CLP bit.

Fig. 2 shows a block diagram of the Spacing Policer. The input and output lines (providing a slotted system for carrying ATM cells of a possibly large number of VCs) have the same cell rate. One cell can be received per time slot (also denoted as cell cycle). Simultaneously another cell can leave the device. The Spacing Policer consists of two main components, the Leaky Bucket Manager and the Cell Buffer Manager. An incoming cell is forwarded to the Cell Buffer Manager, which is informed by the Leaky Bucket Manager (with the aid of a value D) how long the arriving cell has to be buffered before transmission or whether it should be discarded.

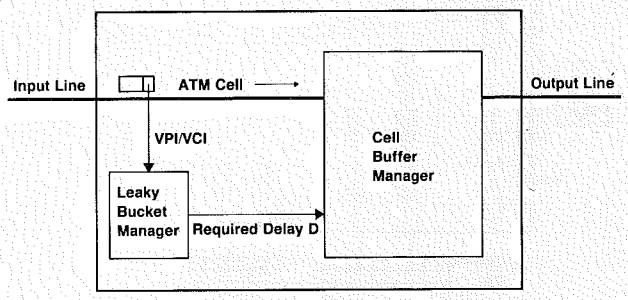


Fig. 2 Block diagram of the Spacing Policer

4.1 The Leaky Bucket Manager

When the header of an arriving cell has been received, the VPI/VCI (virtual path/channel identifier) is given to the Leaky Bucket Manager which holds for every existing VC the variables and parameters of a common Leaky Bucket (in the following also denoted as "Leaky Bucket Counter") which simulates the behaviour of a VC individual Leaky Bucket Queue (i.e. a FIFO Queue with deterministic service time) as shown in the conceptual model.

The value S of the Leaky Bucket Counter can be interpreted as the current content of the Leaky Bucket queue, which has three parameters: the fill value F (at cell arrival the queue length grows by F), the capacity C of the queue and the leak rate L.

The counter S of the Leaky Bucket is initialised to 0. At each of the VC's cell arrivals the counter S is first decremented by $i \cdot L$ (but not beyond 0), where i denotes the elapsed time since the last cell arrival (in cell cycles). If the resulting value S does not exceed C-F, the value D is calculated by

1) CCITT (see [6]) specifies a virtual connection's peak cell rate as the inverse of the minimum inter arrival time of two requests to send an ATM cell. The definition refers to a reference point at the service access point of the ATM layer.

D = ⌊S/L⌋ (⌊x⌋ is the integer part of x) and afterwards S is incremented by F. D represents the number of cell cycles the arriving cell would wait in a Leaky Bucket Queue served at a rate of r_c=L/F cells per cycle. If S exceeds C-F after it has been decremented, then S is not incremented and D is set to a special value (e.g. -1) indicating that the cell should be discarded by the Cell Buffer Manager. In order to facilitate the implementation of the Leaky Bucket Manager the leak rate L should be chosen as $L= 2^m$ for some integer m.

4.2 The Cell Buffer Manager

There are various ways to implement the Cell Buffer Manager. One variant is described in [7]. Here we describe an alternative solution.

Fig. 3 shows the logical components of the Cell Buffer Manager: the Cell Memory (CM), an array denoted as Calendar (CA) and four additional variables specifying special positions in CM.

The Cell Memory is a pool of storage elements, each element capable of storing an ATM cell together with a pointer to another element in the Cell Memory. These pointers allow the elements of CM to be chained together in linked lists.

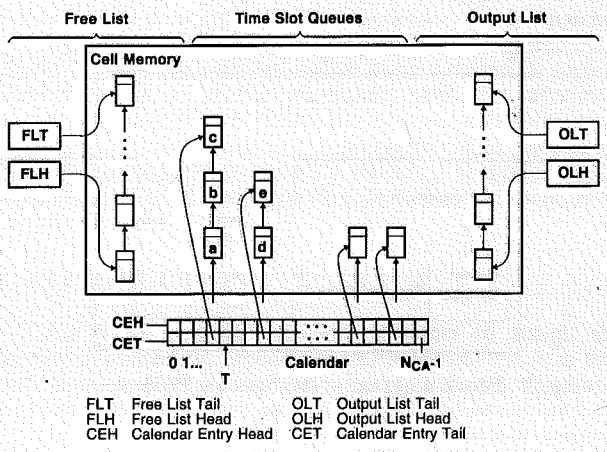


Fig. 3 Structure of the Cell Buffer Manager

All vacant cell spaces are chained together in the Free List. An arriving cell is written into the cell space at the head of the Free List, identified by a variable denoted as Free List Head.

Cell spaces containing ATM-cells which have been delayed for the time D calculated by the Leaky Bucket Manager and wait for transmission are chained together in the Output List. The Output List corresponds to the queue in the output multiplexer of the conceptual model of Fig. 1.

The various VCs' cells that have to be delayed in the Spacing Policer are queued with the aid of the Calendar (length N_{CA}) which has an entry (consisting of two parts, the Calendar Entry Head and the Calendar Entry Tail) for every cell cycle in the (near) future. A variable T contains an index that identifies a special position in CA. T is incremented by one (modulo N_{CA}) at the beginning of every cell cycle and thus provides the present cell cycle number (or time) modulo N_{CA}.

When a cell arrives the cell cycle X when it should leave is determined (namely X = (T+D) mod N_{CA}) and the corresponding entry of the Calendar is marked. If there is only one cell which should leave in cell cycle X, then both parts of the corresponding calendar entry are marked with a pointer to the cell. If more than one cell want to exit the device in cell cycle X then these cells are linked together in a list (called time slot queue) in such a way that the cell which arrived most recently is at the head of the list and identified by the value of the Calendar Entry Head. The cell which has been waiting for the longest time in the time slot queue is identified by the Calendar Entry Tail.

When the time proceeds and T reaches a marked entry of the Calendar, the corresponding time slot queue is transferred to the end of the Output List and the calendar entry is cleared. The time slot queue is linked to the Output List in such a way that the cell at the tail of the time slot queue forms the new tail of the Output List.

In each cell cycle one cell of the Output List can be transmitted. The cells are transmitted in the order they appear in this list (starting from the head of the output list). After the cell at the head of the output list has been selected for transmission the CM element is linked to the tail of the Free List.

All cells that have to wait in the Output List leave later then originally specified. The strategy used for linking arriving cells to a Time Slot Queue implies that cells which should leave at the same time are transmitted in the reverse order of their arrival. (In Fig. 3 the cells a, b, ... e would be read out in alphabetical order.) Thus a delay advantage is given to cells that have to be delayed only little or not at all and thus have a very small D value. On the average, cells of well behaving sources are less affected by the delay in the Output List than cells of bursty sources which send short bursts violating the allowed peak rate.

5. DIMENSIONING OF THE SPACING POLICER

5.1 Dimensioning of the Leaky Bucket Manager

The Leaky Bucket parameters have to be chosen so that cells of a well behaving source are discarded only with a very low probability (e.g. 10^{-10}). The probability that the Leaky Bucket Queue overflows is determined by the ratio C/F which is roughly spoken the capacity of the Leaky Bucket Queue measured in cells. It can be shown (in a similar way as that of inequality (5) in [8] that a cell of a source sending at the policed cell rate r_c = L/F cells per cycle is discarded with a probability not higher than a given value α (e.g. $\alpha=10^{-10}$), when C and F are chosen such that

$$C/F \geq 1+\delta r_c, \qquad (1)$$

where δ denotes the $(1-\alpha)$-Quantile of the variable part of the delay (in cell cycles) experienced between terminal equipment and policing device. Dimensioning examples for the Leaky Bucket algorithm using (1) are given in table 2 of [9].

5.2 Memory Requirements

Memory Requirements for the Cell Buffer Manager depend on the parameters held by the Leaky Bucket Manager. From the Leaky Bucket Counter description in

Section 4.1 we see that the requested delay D, which is calculated by the Leaky Bucket Manager after cell arrival, is always smaller than or equal to D_{max} which is the maximum value of $\lfloor (C-F)/L \rfloor$ over all the VCs. Cells which would require a higher delay are discarded as excess cells. In the Appendix it is shown that no cell loss due to lack of places in Cell Memory will occur even at 100% load if the Cell Memory has

$$N_{CM} = D_{max}+1 \qquad (2)$$

elements. If the parameters of a Leaky Bucket Counter are correctly chosen (i.e. so that, for all Leaky Bucket Counters, the ratio C/F is only a little larger than the lower bound $1+\delta r_c$ given in inequality (1)) then $C/F \approx 1+\delta r_c$ and

$$D_{max} \approx (C-F)/L \approx \frac{(1+\delta \cdot r_c)\cdot F - F}{L} = \delta \cdot r_c \frac{F}{L} = \delta .$$

At the UNI the $(1-10^{-10})$-quantile δ of the queueing delay (and thus also D_{max}) will be below or around 75 cell cycles, assuming that the cells arriving at the UNI have passed at most four queueing stages (e.g. in a large PABX) where each queue has about 80% output utilisation. From this value, which was calculated by autoconvoluting the delay distribution of an M/D/1 queue, we get a necessary cell memory size of

$$N_{CM} = D_{max}+1 \approx 75 \text{ ATM cells.}$$

A Calendar of size

$$N_{CA} \geq D_{max}+1 \qquad (3)$$

guarantees that the cell sequence integrity of a connection is preserved. The proof of cell sequence integrity is similar to the proof given in Appendix B of [7].

6. EFFECT OF THE SPACING POLICER ON A CELL STREAM

The effect of the Spacing Policer on a connection's cell stream should be considered as having three separate components, namely:

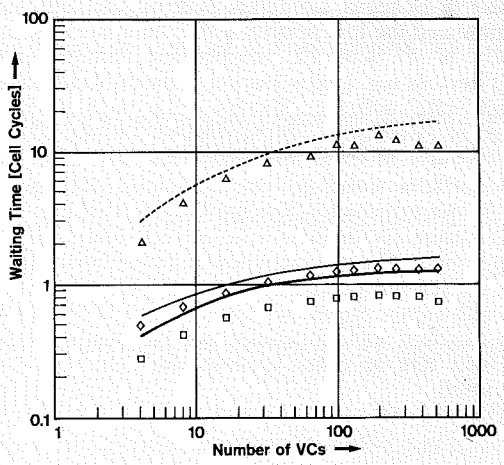
- cell loss due to leaky bucket policing,
- cell delay in the time slot queues (i.e cell delay in the Leaky Bucket Queues of the conceptual model), and
- cell delay in the Output List (i.e. cell delay in the output multiplexer of the conceptual model).

The following remarks refer to these three components:

Leaky Bucket cell loss can be controlled to an arbitrary degree, using inequality (1) above, and therefore needs no further investigation here.

The cell delay experienced by a cell stream due to the spacing effect of the Leaky Bucket Queues is relatively well understood since it is exactly that of the deterministic service time queue (see e.g. [10]). Qualitatively speaking this delay is influenced by two factors: the ratio r/r_c of actual source cell rate to the policing cell rate and the amount of jitter accumulated up to the spacing device. The influence of r/r_c, which is the load on the queue, is that the smaller it is the lower the delay will be. In a CBR connection spaced with $r = r_c$, all cells will be delayed in the Leaky Bucket Queue so that they exit with the same total delay as that cell of the connection which has experienced the highest delay so far. It is therefore always necessary to have a tolerance on the policing cell rate over the negotiated cell rate of a few percent to ensure that the Leaky Bucket Queue always empties reasonably quickly.

While the Leaky Bucket queue does not completely empty the traffic at its output is periodic. Whenever the Leaky Bucket Queue empties then the phase of the periodic stream, once it resumes, will have changed. Thus if a source transmits with a mean cell rate close to the policed cell rate then the traffic after the Leaky Bucket Queue will be a periodic stream with occasional phase shifts, whatever the arrival process before the queue may be. Therefore we postulate that the delay in the Output List (the output multiplexer delay in Fig. 1) is similar to the delay of a $\sum Di/D/1$ queue. However, the more gaps there are in the incoming periodic streams, the smaller the delay will be. Experiments were carried out to verify this hypothesis.



	Mean Waiting Time	Standard Deviation of Waiting Time	10^{-5} Quantile of Waiting Time
nD/D/1	———	———	----------
Spacing Policer Simulation	□	◇	△

Fig. 4 Cell delay in the Output List. Load 0.8 on the inlet line of the Spacing Policer produced by a multiplex of identical Bernoulli sources.

A simulation of the Spacing Policer was fed with a multiplex of n statistically identical Bernoulli sources (with values of n between 2 and 100) producing a total load of 80%. For every n the policing cell rate r_c was adapted to the source cell rate such that $r_c/r \leq 1.02$. The distribution of the delay in the Output List was recorded. As figure 4 shows, the delays experienced are very similar to (slightly better than) the delays of the nD/D/1 queueing model [11]. These results refer to averages taken over all VCs being spaced. In the presence of bursty sources which send short bursts violating the allowed peak rate well behaving sources may experience even smaller delays at the cost of the misbehaving sources due to the strategy used to link the time slot queues to the Output List. (This strategy was described in Section 4).

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.