INPUT RATE CONTROL FOR ATM NETWORKS*

José Augusto Suruagy Monteiro†, Mario Gerla‡, and Luigi Fratta§

† Dept. de Informática, UFPE, C. Postal 7851, 50739 Recife-PE, BRAZIL
‡ Computer Science Dept., UCLA, 3732 Boelter Hall, Los Angeles, CA 90024, USA
§ Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, ITALY

In broadband ISDN implemented using the Asynchronous Transfer Mode (ATM), a statistical gain is obtained by multiplexing several sessions on the same channel. The knowledge of the bandwidth required by a source can be used by the admission control mechanism in order to decide whether or not a new call can be accepted still guaranteeing the Grade Of Service (GOS). However, this will be achieved only if the sources comply with the parameters specified at call set-up. Input Rate Control (or Policing) mechanisms have the function of assuring that the sources abide by their initial specification. In this paper we use simulation, to compare the effectiveness of several policing mechanisms some of which have been previously proposed and some new ones.

1 Introduction

In a recent paper [1] we studied the bandwidth allocation problem for bursty and variable bit rate traffic using an analytic method.

We say that a source is bursty if it interleaves active periods (in which it transmits at the peak rate) with idle periods (in which it remains silent). Several traffic sources can be characterized as bursty. The best known examples are packet voice (coded with silence detection), and imaging services.

We characterize bursty source by the parameters: \( B_p \) (peak bit rate), \( B_m \) (mean bit rate), and \( T \) (mean burst duration). We define burstiness \( b \) as the peak to mean ratio (i.e., \( b = \frac{B_p}{B_m} \)). We assume that both active and silence periods are exponentially distributed with averages \( T \) and \( T(b - 1) \), respectively. Consistently with ATM standards, we assume that the cell length \( n_{cell} \) is 53 bytes (48 data bytes + 5 header bytes) [2], i.e., 424 bits.

The burst average length (in cells), \( L \), relates to the other parameters through the following equation:

\[
L = \frac{T}{n_{cell}/B_p}.
\] (1)

The knowledge of the bandwidth required by a source can be used by the admission control mechanism in order to decide whether or not a new call can be accepted still guaranteeing the Grade Of Service (GOS). However, this will be achieved only if the sources comply with the parameters specified at call set-up. Input Rate Control (IRC) mechanisms have the function of assuring that the sources abide by their initial specification.

In a previous paper [3] we studied the leaky bucket behavior under bursty traffic and we concluded that it is not adequate for deletion of abusive traffic cells. In this paper two new mechanisms are proposed and their behavior is compared with that of leaky bucket and jumping window.

This paper is organized as follows: Section 2 defines what do we expect from an ideal input rate control mechanism and describes the simulation system that was used to study the conformance of the proposed mechanisms with the ideal one. Section 3 introduces the IRC mechanisms that are studied and compared in Section 4. Finally, we conclude in Section 5 proposing also some directions for further study.

2 The Ideal IRC Mechanism

The ideal input rate control mechanism is one that marks/deletes all and only the violating cells (i.e., cells in excess of the average) [4].

Therefore, an ideal policing mechanism would not mark nor delete any cell from sources with average rates up to the nominal, and would mark a fraction \( \frac{\sigma - 1}{\sigma} \) of the cells, where \( \sigma \) is the average normalized to the nominal rate (\( \sigma \geq 1 \)).

One of the criteria that will be used to measure the performance (quality) of a policing mechanism is its conformance to the ideal cell loss/marking probability behavior (i.e., the ability to mark/delete all and only the cells in excess of the average).

As we will see, not all the proposed input rate control mechanisms are appropriate for deleting abusive cells. They would

---

*This work was supported by the State of California and Pacific Bell under a MICRO Grant, by DARPA under Grant MDA 903-87-0663, by CAPES under Grant 6886/84-4, and by CNR under Grant PFT 1989.
either delete too many normal cells or let go too many abusive ones. In the other hand, due to the multiplexing effect, the multiplexer (and ultimately, the network) might be able to carry a percentage of the abusive cells without deteriorating the traffic of well behaved sources; increasing, therefore, network throughput. This suggests that marking cells instead of deleting them is a better bet for the input rate control. By only marking cells and not deleting them at the entry point, we can afford to mark a greater number of well behaved cells without the risk of deteriorating too much their loss probabilities. Still the ideal behavior would be that of not marking any of the average traffic cells.

However, one of the costs of this scheme is that it requires the implementation of priority handling by all network buffers. This priority scheme works as follows: an unmarked (high priority) cell is blocked only if there are only unmarked cells in the buffer. Furthermore, if there is at least one marked (low priority) cell in the full buffer when an unmarked cell arrives, the last marked cell is removed from the buffer (and lost) and the unmarked cell is accepted. In the other hand, a marked cell that finds the buffer full is lost.

Another way of looking at this performance measure is the effect that abusive traffic has on well behaved ones. In the ideal case there should be no sensible effect. However, with the use of proposed IRCs either a considerable amount of average traffic cells will get marked, which increases their probability of being deleted, or a considerable amount of abusive cells will go unmarked, interfering with the traffic of unmarked cells of well behaved sources.

In order to study the effect of violating traffic sources in the cell loss probability of well behaved ones for the various input rate control mechanisms, the system reported in Figure 1 was simulated. The multiplexer (MUX), which represents one of the buffers in the network, behaves as described above. It has a buffer size of 35 cells, and its service rate is designed in order to achieve a GOS=10^{-4}, for the desired number of sources \(n_{rc}\). The choice of GOS=10^{-4} was made in order to reduce simulation running times. However, \(n_{rc}\) is the number of identical sources that can be multiplexed in a 150 Mbps channel in order to achieve GOS=10^{-9}. For a bursty source with parameters \(B_s=10\) Mbps, \(b=10\), and \(L=100\), \(n_{rc}=34\). Each source has an input rate control (IRC). In each simulation, all IRC’s are assumed to be of the same type. The sources are divided into well-behaved and abusive sources. Well behaved sources have average bit rate \(B_m\), as specified, while abusive sources, have an average bit rate \(B_m = \sigma B_m\).

The modified average bit rate is obtained by modifying the average active and silence periods keeping the average cycle period constant (i.e., \(T + S = T' + S'\)), as follows:

\[
T' = \sigma T \\
S' = (1- \sigma)T + S
\]

Two other measures will be used in the comparison of IRC mechanisms. One of them is the “reaction time”, i.e., time required for the IRC mechanism to react (i.e., drop/mark cells) after the start of the burst. The other one is implementation complexity.

## 3 IRC Mechanisms

In this section we present some of the policing mechanisms that have been proposed in the literature, namely, jumping and moving windows, and leaky bucket, as well as two new mechanisms called Virtual count.

### 3.1 Jumping and Moving Windows

The Jumping Window (JW) mechanism basically consists in counting the number of cells generated during a given interval (the window size) and marking/deleting the ones that are above a desired maximum. At the end of the interval, the counter is resetted and a new window is started. The problem with JW is that with a reset the mechanism looses the past history (even immediate) of the traffic source. However, if the window is large enough, these losses in history should not cause any problem.

In the Moving Window (MW) mechanism the window as the name says moves in time keeping track of how many cells were transmitted during the past interval (the window size). Again cells that are found in excess of the desired maximum are marked/deleted.

Implementation-wise, moving window is costlier because it requires that the mechanism remembers not only the number of cells in the window but also their arrival time, so that the window can be updated correctly. This requirement is specially costly for large windows.

### 3.2 Leaky Bucket

The Leaky Bucket (LB) [5,6] is a well known ATM policing mechanism. The heart of the leaky bucket, the pseudo queue, is a counter which is incremented whenever a cell arrives from the monitored source and is depleted at a constant rate \(B_m\) ("depletion" rate). The counter has a maximum value \(Q\) (the pseudo queue length). The cells that upon arrival find the counter at its maximum value are dropped.

Some variations of this basic approach have also been proposed in the literature. One such approach is the Virtual Leaky Bucket [7] where the extra cells are not dropped but simply marked as they are forwarded into the network.
3.3 Peak Counters

The Peak Counter (PC) input rate control is a mechanism where the decision to mark/delete is based on the period of time that an input source has been operating above its nominal (predefined) average rate.

This mechanism is implemented with two counters. The first one (the average counter) is similar to the LB's pseudo queue and it is used to capture the average behavior of the input source. The second counter (the peak counter) basically keeps track of how long the first counter is kept above its threshold, i.e., how long has the input source been operating above the nominal rate. The range of the second counter is between 0 and a maximum value \( T_{pk} \). Its value is increased at a constant rate while the average counter is above the threshold; otherwise it is decreased at the same constant rate. A cell that upon arrival finds the second counter at its maximum, is marked.

There are two variations of this mechanism. In the first one (PC), the average counter is not limited but marked cells are not counted. In the second one (PCWL), the average counter has a maximum value \( Q \) and marked cells are still counted.

Therefore, we have basically 5 parameters: the maximum value of the average counter \( (Q) \); the threshold of the average counter \( (T_a) \); the depletion rate of the average counter \( (B_a) \); the maximum value of the peak counter \( (T_{pk}) \); and the insertion/depletion rate of the peak counter \( (B_{pk}) \).

\( T_{pk} \) and \( B_{pk} \) are dependent, since an increase in the insertion/depletion rate of the peak counter \( (B_{pk}) \) can be compensated by an increase in the maximum value of the peak counter \( (T_{pk}) \) and vice-versa. Let us call \( D_{pk} \) the marking delay, i.e., the amount of time that the average counter is allowed to stay above its threshold \( (T_a) \) before it starts marking cells. \( D_{pk} \) is given by

\[
D_{pk} = \frac{T_{pk} \cdot n_{cell}}{B_{pk}} \tag{4}
\]

Figures 2 and 3 show the evolution of the state of the average and peak counters for PC and PCWL, respectively. Note that cells get marked only when the peak counter is at its maximum value \( T_{pk} \). Note also that in Figure 2 marked cells are not counted by the average counter.

While LB marks cells only when the pseudo queue overflows, the Peak counter mechanism can monitor how long did the input source generate cells above its nominal rate, and it can mark the cells that exceed the allowed maximum duration. Therefore, we claim that the Peak counter mechanism is more flexible than LB. This flexibility comes from the freedom of choosing the marking delay.

The main difference between the two variations of the Peak counter mechanisms is that while the PC average counter tends to work around its threshold \( (T_a) \) even for abusive traffic cells; in the PCWL case this is no longer true, and abusive traffic will push the average counter to its maximum value \( (Q) \). This effect can be explained by the fact that the former counts only the unmarked cells, while the later counts all cells until the limit \( Q \) is reached.

4 IRC Mechanisms comparison

In this Section we compare the effectiveness of the policing mechanisms presented in the previous Section.

The criteria for our comparison are: conformity to the ideal marking probabilities; effect on well-behaved sources; reaction time; and implementation complexity.

4.1 Conformity to the ideal marking probabilities

Figure 4 compares the marking probabilities of jumping window, leaky bucket, and Peak counter mechanisms with the
marking probability of the ideal IRC mechanism, for our bursty source.

From Figure 4 we can conclude that jumping window and leaky bucket have an equivalent effect on the cell loss probability of well-behaved sources, while PCWL although worse for $1.0 < \sigma < 1.1$, presents even a reduction with respect to the ideal cell loss probabilities for $\sigma > 1.2$.

Therefore, we conclude that the PCWL mechanism is the best among them as far as the effect on well-behaved sources is concerned.

### 4.3 Reaction time

Figure 6 compares the lower bound on reaction times for jumping window, leaky bucket, and Peak counter mechanisms. We define reaction time as the amount of time that the given mechanism requires to go from a predefined state ("empty", or average) to the marking state. In order to obtain a lower bound, we assume that cells arrive at peak bit rate until the marking state is reached.

Reaction time for leaky bucket is linearly dependent on the bucket size. Both queue sizes (i.e., 50K and 100K) present almost the same marking characteristics (see Figure 4), and therefore, we can use the 50K one for the comparison. The same applies to the PC mechanism. Therefore, we can rate the IRC mechanisms according to reaction times as: LB $<$ jumping window $<$ PC $<$ PCWL.

### 4.4 Implementation complexity

In this Section we evaluate the mechanism complexity by the number of "basic" hardware elements they require. The "basic" hardware elements are: counters, comparators, and rate generators.

The counters are registers that hold the counter value, and are updated according to the arrival of a cell, or a given bit rate. The comparator compares counter values to values stored in registers. These values represent thresholds or maximum counter values, and have to be set according to the traffic currently being monitored. The rate generators which may be only programmable divisors, are responsible for generating the counter bit rates.
Here, we are not concerned with the implementation details of these "basic" elements. They could be even implemented just in software, as long as time constraints are not violated.

Table 1 lists the number of such "basic" elements for each of the studied IRC mechanisms. Therefore, we can rate their relative complexity as: Jumping Window = Leaky Bucket < PC < PCWL. The exact order depends on the relative costs of comparators and rate generators.

4.5 Summary

Table 2 summarizes the comparisons made in previous subsections.

Even though leaky bucket gets a good rating in most of the criteria, including implementation complexity, which is crucial in obtaining its cost, we argue that still the most important criteria is the effect on well-behaved sources, as far as the design of broadband networks is concerned. In this case, a mechanism with a marking probability like PCWL is more desirable than leaky bucket.

5 Conclusions

In broadband networks built using ATM (Asynchronous Transfer Mode) technology, we need not only to allocate enough bandwidth to each connection, in order to achieve a desired grade of service (GOS). We need also to ensure that the sources abide by the declared rates or else well-behaved traffic will suffer.

Input Rate Control (IRC) mechanisms were devised to enforce such predefined maximum rates. Even though, the sources themselves can monitor their traffic and deliver only at the predefined rates, the network still must perform a repressive function to protect well-behaved sources.

The most popular input rate control for broadband networks reported in the literature, namely, leaky bucket, was shown in a companion paper [3] not to be adequate for deletion of abusive traffic cells. A "virtual" marking scheme proposed elsewhere was adopted. In this scheme, abusive traffic cells are not deleted but simply marked. In case of congestion, marked (low priority) cells have to defer to unmarked (high priority) ones. The performance of leaky bucket, jumping window, and two new mechanisms (Peak counter with and without limit) were studied and compared for a bursty traffic.

We conclude that the Peak counter with limit mechanism, even though not conforming to the ideal marking probability behavior, nor being the simplest one, has the nice property of "overpenalizing" abusive traffic sources, by marking more than the abusive cells for large offending factors. As a result, the cell loss probability of well-behaved sources may even be reduced below the normally expected levels.

Further work is necessary in the use of analytical models (e.g., Markovian Modulated Poisson Process – MMPP) for the mix of different traffic sources. A particularly interesting case is the mix of well-behaved and offending cells. This is a particularly difficult problem because of the priority handling at the buffers; and the need to distinguish cell loss for each priority class (well-behaved and offending ones).

We have studied the IRC mechanisms behavior under steady offending traffic sources. Even though we derived some bounds for reaction times, it would be of interest to study the behavior of such mechanisms under transient (sporadic surges of) offending cells.

Most of our simulations were performed at the cell level, and consequently, required long simulation times even for moderate cell loss probabilities ($10^{-4}$ to $10^{-5}$). Simulation results are always useful, either because no analytical solution is know, or just to validate analytical methods and approximations. Therefore, we suggest the use of some (new) techniques to reduce simulation times, such as the Extreme

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Counters</th>
<th>Comparators</th>
<th>Rate Generators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jumping Window</td>
<td>1</td>
<td>1 ($N$)</td>
<td>1 ($window$)</td>
</tr>
<tr>
<td>Leaky Bucket</td>
<td>1</td>
<td>1 ($Q$)</td>
<td>1 ($B_e$)</td>
</tr>
<tr>
<td>Peak (w/o limit)</td>
<td>2</td>
<td>2 ($T_s, T_pk$)</td>
<td>2 ($B_e, B_{pk}$)</td>
</tr>
<tr>
<td>Peak (w/ limit)</td>
<td>2</td>
<td>3 ($T_s, Q, T_{pk}$)</td>
<td>2 ($B_e, B_{pk}$)</td>
</tr>
</tbody>
</table>

Table 2: IRC Mechanisms Comparison Summary.

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Conformity</th>
<th>Effect on WBS</th>
<th>Reaction Time</th>
<th>Implement. Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jumping Window</td>
<td>Good</td>
<td>Poor</td>
<td>Good</td>
<td>Excellent</td>
</tr>
<tr>
<td>Leaky Bucket</td>
<td>Excellent</td>
<td>Poor</td>
<td>Good</td>
<td>Excellent</td>
</tr>
<tr>
<td>Peak (w/o limit)</td>
<td>Fair</td>
<td>Poor</td>
<td>Fair</td>
<td>Fair</td>
</tr>
<tr>
<td>Peak (w/ limit)</td>
<td>Poor</td>
<td>Excellent</td>
<td>Poor</td>
<td>Poor</td>
</tr>
</tbody>
</table>
Value Theory [8]; and to simulate as much as possible at higher levels (e.g., burst or frame levels) rather than at the cell level.

As pointed out by one of the reviewers, in this study we assume that sources are fed directly into the IRCs. However, in most of the cases we have to take into account the delay jitter introduced by multiplexers in the private network. The study of such effects is left as future work.

References


